MONITOREO DE ANOMALIAS EN MAQUINAS ROTATIVAS CON AGENTES INTELIGENTES JADE Y ARDUINO

PROYECTO DE GRADO PRESENTADO EN CUMPLIMIENTO DE LOS REQUISITOS PARA EL GRADO DE INGENIERO DE SISTEMAS DE LA INSTITUCION UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO

ABEL FELIPE CHAUX GUTIERREZ

OCTUBRE 2017

MONITOREO DE ANOMALIAS EN MAQUINAS ROTATIVAS CON AGENTES INTELIGENTES JADE Y ARDUINO

ABEL FELIPE CHAUX GUTIERREZ

DIRECTOR
MsC. ALEXIS ROJAS CORDERO

CODIRECTOR
MsC. RICARDO GOMEZ VARGAS

INSTITUCION UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
BOGOTÁ
2017

Certifico que he leído este proyecto de grado y que, en mi opinión, es totalmente adecuado en alcance y calidad como un proyecto para el grado de ingeniero de sistemas.

Resumen

La computación basada en agentes representa una síntesis tanto para la Inteligencia Artificial (IA) como más en general, para la Informática. Tiene el potencial de mejorar significativamente la teoría y la práctica de modelar, diseñar e implementar sistemas informáticos para resolver problemas complejos y reales.

En el presente trabajo se pretende mostrar al lector cómo pueden ser usados los agentes inteligentes actuando con Arduino, para poder monitorizar señales de vibración provenientes de máquinas rotativas, con el fin de detectar anomalías para el conocimiento de estas tendencias, implementación que puede ser usada para el propósito de dar mantenimiento a tiempo, sin tener que incurrir en gastos adicionales por culpa del deterioro de dichas máquinas generado por el cuidado tardío de estas. El sistema multi-agente será útil para la toma decisiones basadas en el aprendizaje.

En este trabajo se muestra cómo puede usarse JADE en composición con Arduino y haciendo uso de otras tecnologías, para lograr la detección de anomalías de señales (vibraciones) provenientes de una máquina rotativa.

Agradecimientos

Desarrollar un trabajo tan extenso es un gran mérito personal para mí, debido al nivel de compromiso, responsabilidad, disciplina, sacrificio e investigación que se requirió. Por esto de una manera especial realizo un completo agradecimiento y sentimiento de aprecio a todos y cada uno de los profesores que formaron parte mi crecimiento académico durante el transcurso de toda la carrera, ya que gracias a ellos obtuve considerables herramientas para culminar con éxito mis estudios y el presente proyecto.

De igual manera le agradezco enormemente a mi director de proyecto: el profesor Alexis Rojas Cordero, quien, con su constante acompañamiento desde el inicio, me ofreció una guía y apoyo que fueron muy significativos para el planteamiento del proyecto.

A los profesores Ricardo Gómez y Wilmar Jaimes quienes me apoyaron y aportaron importantes ideas para el proyecto.

A mis padres, ya que siempre estuvieron presentes para apoyarme en lo que fuera necesario.

A todos ellos dedico este reconocimiento, pues es gracias a ellos, que el presente proyecto tiene un resultado final exitoso.

Contenido

Tabla de contenido

Lista de tablas

Lista de figuras

Lista de Diagramas

Glosario

Agente software: Son entidades software, que construyen un modelo propio de su entorno y a partir de él toman decisiones y realizan acciones.

Bluetooth: Es una tecnológica que permite la trasmisión de datos de manera inalámbrica.

FIPA: Organización que desarrolla estándares para agentes software.

JADE: Es un entorno de desarrollo software implementado en el lenguaje JAVA. Entorno que facilita el desarrollo de sistemas multi-agente.

Inteligencia artificial: Termino asociado a los algoritmos que se plasman en programas informáticos que, a su vez, buscan conseguir la imitación del modo de funcionamiento del cerebro humano.

Middleware: Es software intermedio, entre el sistema operativo de un ordenador y las aplicaciones que se ejecutan en él.

Puerto Serial: Termino asociado a la comunicación de datos digitales por medio de una interfaz.

Acrónimos

JDK: Java Development Toolkit

ACL: Agent Communication Language

AID: Agent Identifier

AMS: Agent Management System

AP: Agent Platform

DB: Data Base

DF: Directory Facilitator

FIPA: Foundation for Intelligent Physical Agents

KQML: Knowledge Query Manipulation Language

KIF:

SQL: Structured Query Language (database query language)

HSQLDB: (HyperSQL DataBase)

MIT: Massachusetts Institute of Technology

SPP: (Protocolo de puerto serie)

GUI: Graphical user interface

USB: Universal serial bus

AI: Artificial intelligence

1 Introducción

Actualmente, la necesidad de las compañías de mantener el estable funcionamiento de los dispositivos de producción, asimismo como conseguir de estos una amplia disponibilidad, ha ocasionado un progreso considerable en el mantenimiento industrial de los últimos tiempos.

Atravesando por métodos meramente estancados (a la espera de algún daño) a técnicas más elaboradas (seguimiento eficaz y control) desarrolladas con el propósito de identificar las averías en una etapa inicial e inclusive llegar a establecer el motivo del problema y, seguidamente, tratar de corregirlo [1].

Uno de estos procesos es el monitoreo de vibraciones provenientes de máquinas rotativas, con el fin de detectar anomalías para el conocimiento de estas tendencias, implementación que puede ser usada para el propósito de dar mantenimiento a tiempo, para no ocasionar gastos innecesarios por culpa del deterioro de dichas máquinas generado por el cuidado tardío de estas.

El presente proyecto surge por la necesidad de realizar seguimiento y detección de anomalías en vibraciones, para esto se hace uso de los siguientes materiales:

- Placa Arduino: Esta placa se utilizará como plataforma de adquisición, la cual será monitorizada por los agentes inteligentes.
- JADE: Es el entorno de desarrollo que se usara para la creación del sistema multi-agente para el monitoreo de las señales provenientes de la placa Arduino.
- HSQLDB: Es un gestor de base de datos portable, que se usara como base de conocimiento para los agentes.
- Biblioteca RXTX: Librería necesaria para para la comunicación con la placa Arduino mediante el uso del puerto serial.
- MIT App Inventor: es un entorno de programación web que facilitara el desarrollo de la aplicación Android que se encargara de enviar las respectivas señales de vibración a la placa Arduino.

En generalidades se hace una revisión sobre el impacto de la inteligencia artificial en las industrias y de cuáles son los instrumentos o maneras que se han venido utilizando para el monitoreo de vibraciones.

En el marco conceptual se hace toda una revisión de la literatura para poder comprender el contexto del proyecto.

En materiales y técnicas se presenta el costo del proyecto así como también la metodología a usar para el desarrollo del sistema multi-agente, el cual integrara dos agentes, uno reactivo y uno deliberativo, los mismos que se delegarán de la siguiente manera: uno será el agente receptor del Arduino (reactivo) ; el cual informara las señales de vibración obtenidas por la placa Arduino , y el otro será el agente control (deliberativo) , cuyos objetivos estarán ligados a ejecutar las acciones necesarias tales como manipular las señales de las vibraciones en base de datos y notificar el respectivo seguimiento de estas , ambos agentes establecerán comunicación constante para lograr cumplir con sus objetivos.

En el desarrollo del proyecto se especifican las diferentes fases que se emplearon para la construcción del proyecto, se presenta el desarrollo de cada una de las etapas que se planearon, así como de la metodología *Prometheus* y los diagramas de componentes que se usaron para la descripción de la integración de cada una de las partes que conformarán el sistema.

Se presentan las conclusiones acordes a lo realizado. Y finalmente, en trabajo futuro se hacen sugerencias para implementar el presente proyecto con base en otros propósitos, pero que involucren características semejantes.

2 Generalidades

2.1 Antecedentes

Si damos un vistazo a la actualidad comparada con el pasado, no hay considerable diferencia en cuanto a funcionalidades ofrecidas por herramientas tecnológicas, ya que hace algunos años se monitoreaban las vibraciones con ayuda de equipos analizadores de vibraciones y programas informáticos que facilitaban el estudio de vibraciones (Ver Figura 1), los cuales entregaban

reportes gráficos de las señales, basados en el tiempo o en la frecuencia para que se lograra realizar una interpretación y así poder establecer un diagnóstico apropiado [2] sobre el estado de la máquina, equipos que incluso hoy en día se siguen usando para tal fin, pero con un par de nuevas funcionalidades debido al avance tecnológico hasta hoy (Ver Figura 2):

Figura 1 Analizador de vibraciones convencional -año:2010 fuente [2]

Figura 2 Instrumento lenova para analizar la condición de maquinaria diseñado para ambientes agresivos – año :2017 fuente [3]

Actualmente la mayoría de estos equipos tienen un valor muy elevado en el mercado, por lo que requieren de un capital considerable por parte de las empresas interesadas en adquirirlos.

Un estudio realizado por la compañía Accenture sobre el impacto de la Inteligencia Artificial en 12 economías desarrolladas, revela que la inteligencia artificial podría llegar a duplicar las tasas anuales de crecimiento económico en 2035, cambiando la naturaleza del trabajo y estableciendo una nueva relación entre el hombre y la máquina. Se prevé que el impacto de la IA en los negocios aumentará la productividad del trabajo hasta en un 40% y permitirá a las personas hacer un uso más eficiente de su tiempo, tomado de [4].

Esto permite inferir que la inteligencia artificial ha venido creciendo considerablemente, y seguirá avanzando cada vez más con el pasar del tiempo gracias a los avances tecnológicos (Ver Figura 3) , adicional a ello puede evidenciarse un gran potencial de crecimiento para las compañías , esta es una de las principales motivaciones por las que se quiere incursionar en el mundo de los agentes inteligentes, pues se encuentra una oportunidad considerable, para desarrollar un sistema multi-agente que realice seguimiento y detección de anomalías, que podría ayudar a reducir gastos ya sea en cuanto al mantenimiento o a la obtención de un sistema equivalente pero de difícil acceso debido al capital de las compañías.

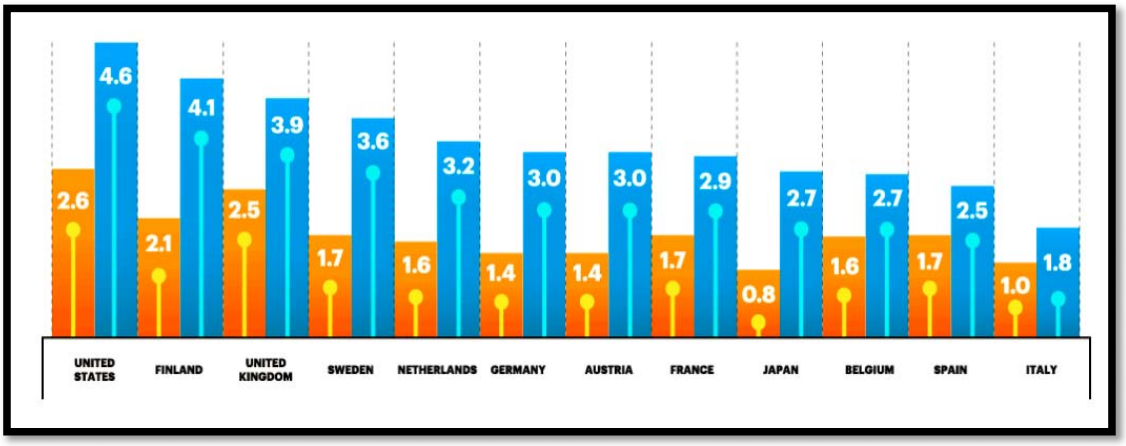

Figura 3 Crecimiento de la inteligencia artificial en la economía para el año 2035. Fuente [4]

De los estudios que se han venido desarrollando por parte de la inteligencia artificial y como parte del crecimiento de la ingeniería de software, surgió un nuevo campo para la investigación y desarrollo denominado los agentes. Los agentes han sido uno de los temas que involucran intensas investigaciones durante años, formando uno de los rumbos computacionales más significativos para abordar cuestiones que necesitan de un modelado flexible, abstracto y un razonamiento elevado [5].

El modelo de agente ha sido aceptado en distintas áreas como las tecnologías de la información, incluyendo las telecomunicaciones, programación orientada a objetos, Ingeniería de Software, Inteligencia Artificial, entre otras.

Pero para hablar de agentes inteligentes no sería posible si no existiera la inteligencia artificial y sus bases importantes:

Redes neuronales: Según [6]**,** Las redes neuronales son una manera de imitar algunas peculiaridades de los humanos, como la capacidad de aprender sobre algo. Si se inspeccionan con cuidado inconvenientes que no pueden enunciarse a través de un algoritmo, se puede decir que estos tienen una peculiaridad que comparten: la experiencia.

Una red neuronal en cortas palabras es “un sistema para la manipulación de información, cuya unidad básica de procesamiento es inspirada en la neurona del ser humano”.

***Machine learning*:** Según la plataforma virtual Coursera [7], el aprendizaje automático o *machine learning* es el saber de lograr que maquinas informáticas operen sin antes haber sido programados para tal fin.

En los últimos años, con el *machine learning* se han desarrollado tecnologías novedosas como los autos autónomos, reconocimiento experto de voz, consulta efectiva en la web, y se ha encontrado una significante comprensión mejorada del genoma humano. Actualmente, el uso del *machine learning* es muy frecuente. Variedad de científicos e investigadores consideran que es la mejor manera de progresar hacia una inteligencia artificial a nivel humano.

2.2 Planteamiento del problema

La industria pesada como petroleras, plantas eléctricas , motores embebidos, motores de difícil acceso como por ejemplo un motor de un avión, motores de naves espaciales y en general, aquellos motores que están en entornos de difícil acceso deben ser monitorizados para de esta manera emitir diagnósticos, realizar mantenimiento, detenerlos o acelerarlos de una forma que no es manual y es aquí donde los agentes inteligentes ofrecen su gran aporte para revisar el entorno con el objetivo de detectar anomalías, analizando y ejecutando acciones inteligentes basadas en el aprendizaje de las señales, cuando estas tienen un comportamiento normal y cuando no.

- ¿Existe algún sistema que pueda detectar y dar seguimiento a anomalías en máquinas rotativas?
- ¿Existen opciones que puedan ser implementadas en el Politécnico Grancolombiano con un alto beneficio?

La implementación de un sistema multi-agente en las plantas eléctricas, para motores que se aceleran o generan alguna vibración.

- ¿Es posible proyectar la implementación de alguna solución como algo escalable y que no se quede obsoleto con el pasar del tiempo?

2.3 Objetivos

2.3.1 Objetivo General

Desarrollar un sistema de agentes inteligentes para el monitoreo de anomalías en máquinas rotativas.

2.3.2 Objetivos específicos

- Diseñar y desarrollar un prototipo para un sistema de agentes inteligentes para el tratamiento de los datos recibidos por la plataforma de adquisición: Arduino.
- Diseñar y desarrollar una aplicación móvil Android para enviar vía bluetooth a la placa Arduino la lectura del sensor acelerómetro del dispositivo.
- Implementar un motor de base de datos, para el almacenamiento de las señales de vibraciones tratadas en Arduino.
- Hacer el test de funcionamiento del prototipo entre Arduino y agentes inteligentes.

2.4 Justificación

Este proyecto es muy significativo en el politécnico Grancolombiano, debido a que se efectuara una contribución a la investigación de la universidad, dejando un camino abierto para que otros estudiantes tengan herramientas y fundamentos de lo que se puede lograr con el área estudiada, para que de esa manera puedan iniciar una idea de proyecto.

Además, será de gran valor para la formación académica, ya que se efectuará la aplicación de conocimientos adquiridos en la carrera como: desarrollo de software, gerencia de proyectos, desarrollo en hardware (Arduino), bases de datos, sistemas operacionales y sistemas distribuidos.

2.5 Delimitación

2.5.1 Tiempo

Fase	Mes 1	Mes 2	Mes 3	Mes 4
Elaboración del Documento		X	X	X
Consulta Bibliográfica	X	X	X	X
Desarrollo	X	X	X	
Pruebas	x	x	X	X
Preparación de Sustentación				X

Tabla 1. Distribución de Tiempo. Fuente: Felipe Chaux (2017)

2.5.2 Alcance

Este proyecto está pensado en general para motores de máquinas rotativas, tales como los de los autos, cohetes o motores no alcanzables por el humano como las naves espaciales, que necesiten de un monitoreo para detectar y hacer seguimiento de anomalías. Para el prototipo a desarrollar se hará uso de un electrodoméstico muy común en los hogares; la licuadora.

Debido a la magnitud del proyecto, se realizará únicamente el análisis y diseño de un prototipo para el desarrollo de un sistema empleando las tecnologías de agentes inteligentes (JADE) y Arduino.

3 Marco Conceptual

3.1 Vibración

Una vibración es una forma de movimiento periódico que tiene un objeto en torno a un lugar de equilibrio. Este lugar será al que llegue dicho objeto, cuando la fuerza que ejerce sobre él sea nula, esta clase de vibración según [8], puede llamarse vibración cuerpo entero.

En un cuerpo entero, el movimiento vibratorio puede ser definido completamente como una composición de movimientos individuales alrededor de los ejes x, y z.

Es de resaltar que, mediante un monitoreo de vibración, pueden identificarse fuerzas ejerciendo sobre una máquina, pero estas fuerzas dependerán de cómo se encuentre la máquina en dicho momento. Gracias a la comprensión de sus peculiaridades e información histórica podría determinarse un fallo en ella.

3.1.1 Análisis de frecuencias

Según [8], la práctica más común para la medición de vibraciones es efectuar un estudio de frecuencias, denominado también estudio de espectro de la señal de vibración. Este estudio es equivalente a transformar las señales ligadas al tiempo, en señales guiadas por la frecuencia. Las siguientes son formulas válidas para representar la relación entre tiempo y frecuencia (ver Figura 4).

$$Tiempo = \frac{1}{Frecuencia}$$

$$Frecuencia = \frac{1}{Tiempo}$$

Figura 4 relación tiempo – frecuencia fuente [8]

3.2 Arduino

Según la página oficial [9], Arduino es una plataforma de electrónica de código abierto basada en hardware y software sencillo de utilizar. La placa Arduino puede leer entradas (luz en un sensor, la pulsación de un botón) y convertirlo en una salida. Para hacerlo, utiliza el lenguaje de programación Arduino y el entorno de desarrollo Arduino (IDE).

3.2.1 Placa que se empleara

La placa Arduino que se va a emplear es Arduino Leonardo (Ver Figura 5), esta placa es suficiente para el desarrollo del prototipo del proyecto, ya que, provee pines digitales necesarios para la recepción de la señal de vibración. la placa puede ser alimentada por puerto USB o con una fuente externa de poder con un voltaje de 5V, además se puede programar de una manera sencilla utilizando el lenguaje propio de Arduino que es muy similar a JAVA, junto con el entorno de integrado Arduino IDE de tal forma que será fácil manipular esta placa, y adicionalmente es de agregar que es de fácil adquirirla debido a su precio económico.

Figura 5 Placa Arduino Leonardo fuente [10]

3.2.2 Alimentación Arduino Leonardo

La placa Arduino Leonardo, según [11], puede alimentarse a través de conexión micro USB o fuente externa de poder. Para usar esta fuente externa,

se puede usar un adaptador de batería. El adaptador, puede conectarse mediante un conector de audio de 2.1mm de centro positivo a la alimentación de la placa.

Al alimentar la placa con batería externa, preferiblemente el voltaje debe estar en el rango de 7 a 12 Voltios. Arduino Leonardo contiene algunos pines para la alimentación de la placa a parte del adaptador para la alimentación:

- VIN: A través de este pin es posible proporcionar alimentación.
- 5V: Se alcanza un voltaje de 5 Voltios como fuente de alimentación regulada desde este pin.
- 3.3V: Se puede obtener un voltaje de 3.3 Voltios y una corriente máxima de 50 miliamperios.
- GND. EL ground (0 Voltios) de la placa (o tierra)

Arduino puede ser programado de una manera muy fácil utilizando el lenguaje propio de Arduino (similar a JAVA) junto con Arduino IDE.

3.3 Conceptos relacionados con JADE

3.3.1 JADE

Según [12], Jade es un entorno de desarrollo software completamente implementado en JAVA. Su entorno facilita el desarrollo de sistemas multi-agente a través de un software que cumple con las especificaciones FIPA y mediante una serie de herramientas gráficas para la monitorización y depuración de agentes.

3.3.1.1 Características generales

las siguientes características son algunas de las más importantes que proporciona JADE:

- Cumple con el estándar FIPA:

El entorno, facilita el desarrollo de sistemas multi-agente gracias a un middleware que cumple con las especificaciones de FIPA.

- Da soporte a sistemas multi-agente:
  Gracias a una serie de utilidades, ofrece seguimiento a fases como depuración y explotación.
- Es distribuido y multiplataforma:
  Un sistema basado en JADE se puede distribuir entre maquinas (no hace falta que compartan el mismo sistema operativo) y la configuración se puede controlar a través de una interfaz gráfica de usuario (GUI) remota. la configuración puede incluso cambiarse en tiempo de ejecución moviendo agentes de una maquina a otra, cuando sea necesario.

El requisito mínimo del sistema es la versión 5 o superior de JAVA (el entorno en tiempo de ejecución o el JDK.

3.3.1.2 Afinidad al estándar

Implementa completamente el modelo de comunicación FIPA:

- Envoltorio
- Protocolos de interacción
- Lenguajes de contenido
- Ontologías
- Esquemas de codificación
- ACL
- Protocolos de transporte

3.3.2 FIPA (*Foundation for Intelligent Physical Agents*)

Según [13], es una organización que se encarga de trabajar en estándares para software de agentes, para así promover la interoperabilidad de sus

estándares con otras tecnologías, en la (Figura 6) se aprecia lo que propone FIPA para el desarrollo de plataformas de agentes.

FIPA, la organización de estándares para agentes y sistemas multi-agente, fue oficialmente aceptada por el IEEE como su undécimo comité de estándares el 8 de junio de 2005.

FIPA se formó originalmente como una organización suiza en 1996 para producir especificaciones de estándares de software para agentes heterogéneos e interactivos y sistemas basados en agentes. Desde sus inicios, FIPA ha jugado un papel crucial en el desarrollo de estándares de agentes y ha promovido una serie de iniciativas y eventos que contribuyeron al desarrollo y la aceptación de la tecnología de los agentes. Además, muchas de las ideas originadas y desarrolladas en FIPA ahora se están enfocando claramente en las nuevas generaciones de tecnología Web / Internet y especificaciones relacionadas.

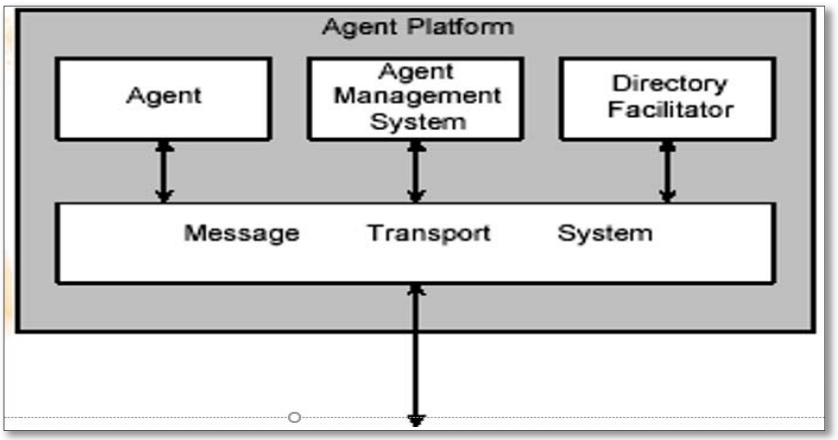

Figura 6 Plataforma de agente. Fuente [14]

3.3.1 ACL (*Agent Communiction Language*)

Es un lenguaje que admite la interacción entre agentes. Según [15] se basa en la teoría del acto de habla, que establece que los mensajes representan acciones, o actos comunicativos, también se conocen como actos de habla o performativos. Cada acto se describe utilizando tanto una forma narrativa como

una semántica formal basada en la lógica modal, que especifica los efectos de enviar el mensaje a las actitudes mentales de los agentes emisores y receptores. Esta forma de lógica es consistente con el BDI o (creencias, deseos e intenciones) (ver Figura 10).

Algunos de los actos más comúnmente utilizados en ACL son informar, solicitar, aceptar, no entender y rechazar. Estos encierran la esencia de la mayoría de las formas de comunicación básica, ACL tiene los siguientes tres elementos:

- Vocabulario
- KIF (*Knowledge Interchange Format*)
- KQML (*Knowledge Query Manipulation Language*)

3.3.2 Plataforma de agente (AP)

La plataforma de agente [15], es un aspecto fundamental de los sistemas de agentes abordados por las primeras especificaciones FIPA, ya que en ella se realiza toda gestión de agentes: un marco normativo dentro del cual FIPA cumple con lo siguiente:

- los agentes pueden existir, operar y administrarse.
- Establecimiento del modelo de referencia lógica para la creación, registro, ubicación, comunicación, migración y operación de agentes.

En la (Figura 7) se hace referencia a los componentes para representar al modelo en cuestión

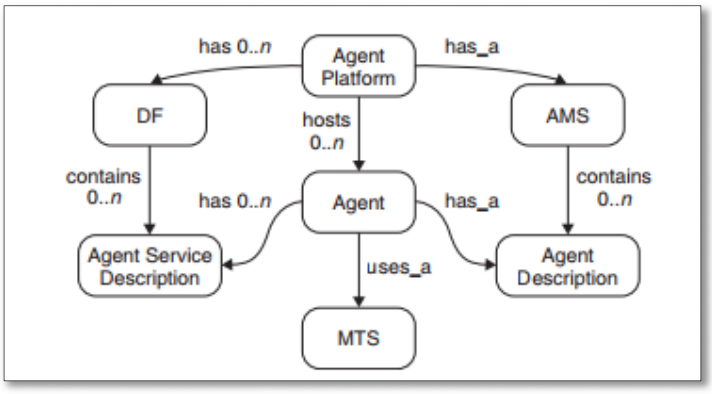

Figura 7 Representación de la ontología de gestión de agentes. Fuente [15]

3.3.3 Directorio facilitador (DF)

El DF [15] es un componente opcional de un AP que proporciona páginas amarillas servicios a otros agentes. Mantiene una lista precisa, completa y oportuna de agentes y debe proporcionar la información más actualizada sobre los agentes en su directorio de forma no discriminatoria para todos los agentes autorizados.

Un AP puede admitir cualquier cantidad de DF que se puedan registrar entre sí para formar federaciones. Todo agente que desee publicitar sus servicios a otros agentes debe encontrar un DF apropiado y debe solicitar el registro de la descripción de su agente. No hay un compromiso futuro intencionado u obligación por parte del agente registrador implícito en el acto de registrarse. Los agentes pueden subsecuentemente solicitar la eliminación del registro de una descripción en cuyo momento ya no existe un compromiso en nombre del DF para intermediar información relacionada con ese agente.

En cualquier momento, y por cualquier motivo, un agente puede solicitar al DF que modifique la descripción de su agente. Además, un agente puede emitir una búsqueda de un DF para descubrir descripciones que coincidan con los criterios de búsqueda suministrados. El DF no garantiza la validez de la información proporcionada en respuesta a una solicitud de búsqueda. Sin embargo, el DF puede restringir el acceso a la información en su directorio y verificará todos los permisos de acceso para agentes que intentan informarle de los cambios de estado del agente.

3.3.4 Ontología JADE

Según [16] , la ontología en JADE es una ejemplificación de la clase **jade. content.onto.Ontology.** Donde se establecen los esquemas, representados por un conjunto de elementos que construyen la distribución de los predicados, las acciones que ejecutan los agentes y conceptos orientados al dominio. A nivel descriptivo estos elementos se definen así:

- **Predicados**: Son expresiones relacionadas con base al estado del entorno.
- **Acciones de los agentes**: Son expresiones que revelan posibles acciones que pueden ejecutar los agentes.
- **Conceptos**: Son expresiones que representan objetos.

3.3.5 Sistema de gestión del agente (AMS)

El AMS [15] es un componente obligatorio de un AP y es responsable para gestionar el funcionamiento de un AP, como la creación y eliminación de agentes, y la supervisión de la migración de agentes hacia y desde el AP. Cada agente se registra con un AMS para obtener una AYUDA que luego es retenida por el como un directorio de todos los agentes presentes dentro de la AP.

3.4 Conceptos relacionados con agentes inteligentes

3.4.1 Agentes

Un agente, es un sistema computacional que tiene la capacidad de tomar decisiones o ejecutar acciones en un entorno, para así cumplir con sus objetivos. los agentes están muy ligados a su entorno o medio y con base en esto ellos pueden modificar este para de igual manera cumplir con los objetivos, en ocasiones los agentes pueden tomar decisiones con base en

criterios estructurados como información historia o también misma experiencia del agente con el entorno.

Según [17] Los agentes son entidades de software, que tienen la capacidad de construir un modelo propio de su entorno y a partir de él pueden tomar decisiones y realizar acciones. estas pueden dirigirse a la obtención y/o elaboración de informaciones.

Las características de los agentes pueden apreciarse en la (Figura 8)

Figura 8 Características de los agentes software. Fuente [17]

Las que podrían resaltarse son las siguientes [18]:

- **Reactividad:** Actúa como un observador del entorno en el que se encuentra y se manifiesta oportunamente a cambios en el.
- **Proactividad**: Tiene un carácter ambicioso y toma la iniciativa propia para actuar, basándose en los objetivos que debe cumplir.

Y otra como la sociabilidad, que no es mencionada en la (Figura 8), la cual define la capacidad de interactuar con otros agentes (incluso humanos) utilizando un lenguaje de comunicación de agentes.

3.4.1.1 Arquitecturas de agentes

Las diferentes arquitecturas que pueden ser implementadas para agentes son las siguientes [17]:

- Deliberativas o simbólicas
    - Intencionales (razonan sobre sí mismos)
    - Sociales (tienen una imagen del mundo)

- Reactivas
    - Pautas empleadas
    - Arquitectura de autómatas finitos
    - Labores competitivas
    - Redes neuronales

- Híbridas

3.4.1.2 Lenguajes de agentes

Los siguientes son tipos de lenguajes que pueden aparecer al emplearse agentes inteligentes [17]:

- Lenguajes de agentes específicos (*Telescript*, *Agent-Tcl*)
- Lenguajes de agentes de propósito general
- Lenguajes de programación de la estructura del agente
- Lenguajes de programación del comportamiento del agente
- Lenguajes de comunicación de agentes

3.4.1.3 Metodologías orientadas a agentes

Existen variedad de metodologías para la construcción de agentes inteligentes, más sin embargo no se hará énfasis en cada una de ellas sino concretamente en la metodología que se utilizara para el desarrollo del presente proyecto, en la (Figura 9) se pueden observar algunas de las metodologías existentes:

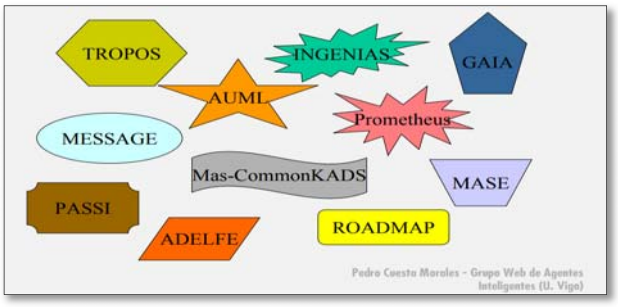

Figura 9 Metodologías de agentes. Fuente [18]

3.4.2 Ontología de agente inteligente

Según [16], una Ontología es utilizada para “definir la especificación de una conceptualización", lo que permite realizar una representación de relaciones y conceptos que pueden hacer parte de la comprensión de un agente o comunicad de agentes (multi-agentes). Las ontologías hacen uso de terminología formal y una serie de definiciones.

Las ontologías juegan un papel importante para los agentes, debido a que los agentes se comprometen con las mismas, Esto puede denominarse como compromiso ontológico, el cual se entiende como un acuerdo para poder utilizar una determinada terminología en común con la que se puede representar el conocimiento compartido. Así pues, se diseñan ontologías con las que los agentes pueden compartir conocimiento y se construyen agentes comprometidos con ontologías.

3.4.3 Agentes deliberativos BDI (*Believe, Desire, Intention*)

Su estructura interna y su capacidad de elección se basan en aptitudes mentales: creencias, deseos e intenciones (ver Figura 10), tomado de [17].

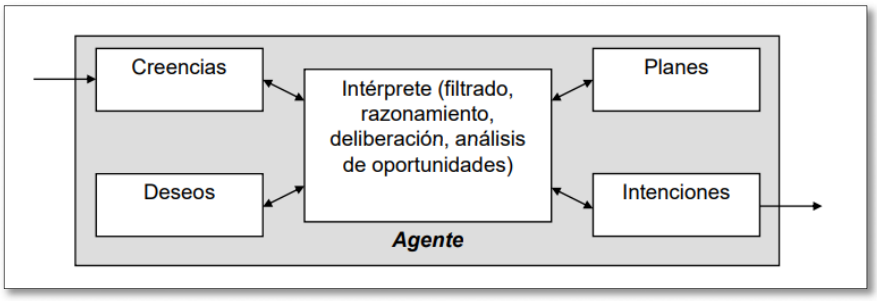

Figura 10 Agente deliberativo. Fuente [17]

3.5 Sensores

Según [19], es un término que hace referencia a distintas clases de sensores. Por esta palabra se concibe tanto las unidades que produce una señal analógica, como las unidades de una señal binaria (nivel alto o nivel bajo). Ellos convierten una magnitud física en una magnitud eléctrica.

3.5.1 Sensores para medir vibraciones - transductores

3.5.1.1 Sensor de proximidad

Según [8] ,el Sensor de proximidad o "Sensor de Corriente de Remolino" es un aparato con ensamble fijo, y requiere una especie de amplificador que establece la señal para la generación de un voltaje de salida (ver Figura 11), que sea ajustado a la distancia entre el transductor y la extremidad de la flecha. Su operación está basada en un principio magnético.

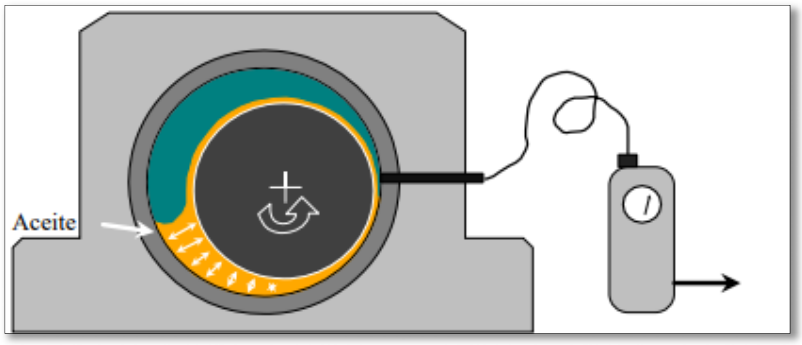

Figura 11 Sensor de proximidad. Fuente [8]

3.5.1.2 Sensor de velocidad

Según [8], es uno de los primeros sensores de vibración en ser creados. Consiste en una bobina de alambre y un imán (ver Figura 12). El movimiento que se genera entre el campo magnético y la bobina provoca una corriente ajustada a la velocidad del movimiento. De esta manera, el sensor hace que se produzca en él una señal directamente conforme a la velocidad de la vibración.

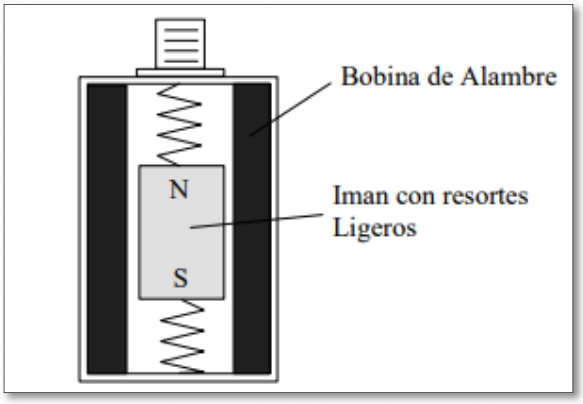

Figura 12 Sensor de velocidad. Fuente [8]

3.5.1.3 Acelerómetro

Según [8], el acelerómetro piezoeléctrico es el sensor predeterminado para la medida de vibración en máquinas, en la (Figura 13) se describe el principio de la operación de dicho sensor.

Estos sensores son consistentes cuando se habla de amplitud, tienen una gran capacidad para detectar niveles bajos de aceleración, los cuales son identificados por medio del ruido electrónico que se genere.

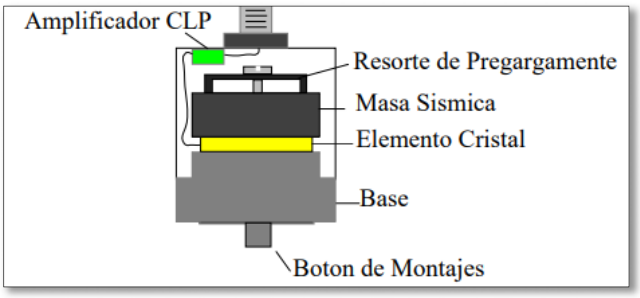

Figura 13 Acelerómetro Piezoeléctrico. Fuente [8]

3.5.2 Sensor que se usara para medir la vibración

Por conveniencia, se utiliza un sensor acelerómetro que viene integrado en un dispositivo móvil Android, en la (Figura 14) se ilustra este sensor. Para poder receptar las señales generadas de este, se desarrolla una aplicación móvil capas de registrar y enviar la señal por bluetooth (ver Figura 18) a través del módulo HC-05 (ver Figura 15).

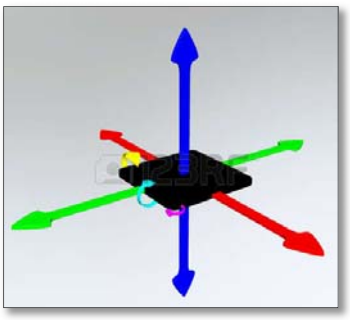

Figura 14 Sensor acelerómetro. Fuente: [20]

3.6 HSQL

Según el sitio oficial [21], HSQLDB (*HyperSQL DataBase*) es el software líder de bases de datos relacionales de SQL escrito en Java. Ofrece un motor de base de datos pequeño y rápido, multiproceso y transaccional con tablas en memoria y basadas en disco, y admite modos incrustados y de servidor. Incluye una poderosa herramienta SQL de línea de comandos y herramientas simples de consulta GUI.

3.7 RXTX

Según el sitio web [22], RXTX es una biblioteca java que utiliza una implementación nativa (a través de JNI) y proporciona comunicación en serie y paralela para *Java Development Toolkit* (JDK). Que se basa en la especificación de la API de comunicaciones Java de Sun.

3.8 MIT APP Inventor

Según el sitio oficial [23], MIT App Inventor es un entorno de programación intuitiva y visual que permite a todos, incluso a niños, desarrollar aplicaciones completamente prácticas para teléfonos inteligentes y tabletas. Los nuevos en MIT App Inventor pueden tener una primera aplicación simple en funcionamiento en menos de 30 minutos. Y, lo que, es más, la herramienta basada en bloques facilita la creación de aplicaciones complejas de alto impacto en mucho menos tiempo que los entornos de programación tradicionales.

El proyecto MIT App Inventor busca democratizar el desarrollo de software al capacitar a todas las personas, especialmente los jóvenes, para pasar del consumo de tecnología a la creación de tecnología.

3.9 Módulo HC-05

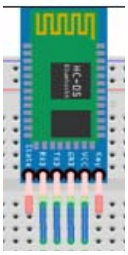

Figura 15 Modulo HC-05. Fuente Felipe Chaux (2017)

El módulo HC-05 (ver Figura 15) es un módulo de bluetooth SPP (protocolo de puerto serie) que puede manipularse muy fácilmente, y su diseño es especialmente para la configuración inalámbrica en serie. El módulo se puede utilizar en una configuración maestra o esclava, lo que le convierte en una buena solución para la comunicaron inalámbrica.

4 Materiales

4.1 Materiales físicos

Descripción	Cantidad	Valor/unitario	Valor Total
Laptop	1	900.000	900.000
Placa Arduino	1	50.000	50.000
Dispositivo móvil Android	1	200.000	200.000
Módulo HC-05	1	23.000	23.000
Cables	4	1500	600
Protoboard	1	6.000	6.000
Total			1.179.600

Tabla 2 Materiales físicos. Fuente Felipe Chaux (2017)

4.2 Metodología que se empleara

4.2.1 Prometheus

Se utiliza esta metodología, ya que según [18], en ella se define un proceso de manera detallada para la especificación, diseño e implementación y prueba de sistemas software orientados a agentes, una de sus características principales son el uso de metas planes y eventos además de las BDI (creencias deseos e intenciones) que se manejan dentro de los agentes.

Es de mencionar que la mayor parte del desarrollo de código de los agentes y de la ontología, se basa en [15].

4.2.2 Fases de la Metodología *Prometheus*

Especificación del sistema, donde se identifican las funcionalidades básicas del sistema. En esta fase hay que tener en cuenta lo siguiente:

- ¿Cómo interactúa el sistema con el entorno?
- Entrada de información desde el entorno (percepciones).
- Mecanismos para actuar sobre el entorno (acciones).
- Eventos que se generan por las percepciones del entorno, provocando reacciones en los agentes.

4.2.2.1 Diseño arquitectónico.

Determinar qué agentes contendrá el sistema y cómo interactúan. Las siguientes son características para tener en cuenta.

- ¿Qué agentes deben existir?
- Asignar funcionalidades a los agentes.
- Agrupación de funcionalidades, si las funcionalidades usan los mismos datos.

4.2.2.2 Diseño detallado.

Desarrollo la estructura interna de cada agente. Se debe especificar el uso de:

- Planes definidos
- Disparadores para metas y eventos
- Implementación de sistemas BDI
- Habilidades (módulos o comportamientos)
- Estructuras de datos

4.2.2.3 Habilidades o comportamientos

- Las habilidades implementarán las funcionalidades que se han descrito anteriormente.
- Para cada agente habrá un conjunto de habilidades que lo definan.
- Una habilidad puede estar formada a su vez por otras habilidades.
- En último nivel las habilidades están formadas por planes, eventos y datos.
- Una misma habilidad puede ser usada por otros comportamientos.

5 Desarrollo del proyecto

5.1 Diseño del sistema

5.1.1 Especificación del sistema

5.1.1.1 Diagrama de componentes

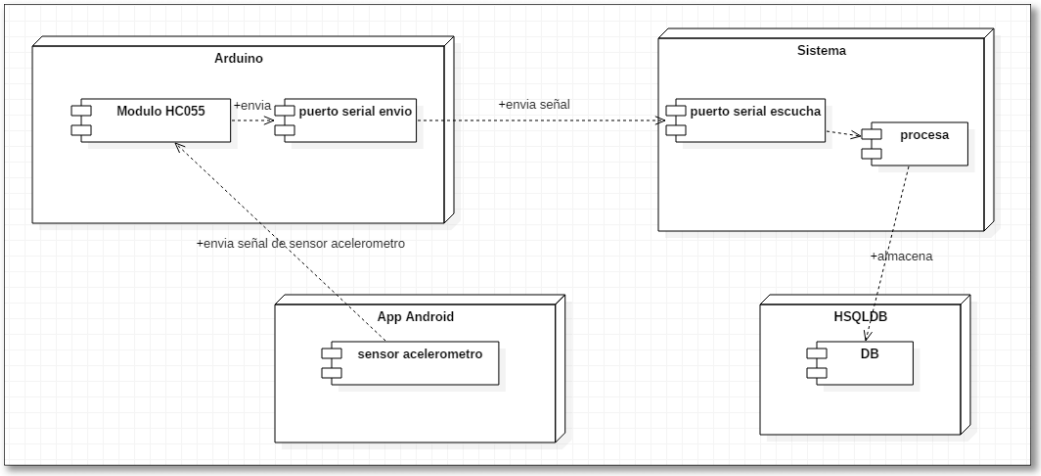

Diagrama 1 Diagrama de componentes del sistema. Fuente Felipe Chaux (2017)

5.1.1.2 Esquema del sistema

En la (Figura 16) se aprecia la manera en la que interactúa ilustrativamente el entorno con el sistema, mediante la identificación de los agentes que deberían existir, la entrada de información desde el entorno (percepciones), y mecanismos de acciones.

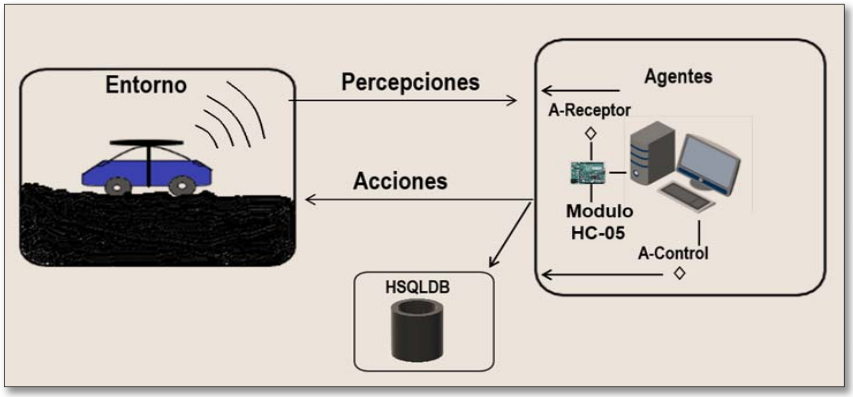

Figura 16 Interacción del entorno con el sistema. Fuente (Felipe Chaux)

5.1.1.3 Diseño de arquitectura del sistema

Como se evidencia en el (Diagrama 2), el actor App Android es el que se encarga de receptar las señales de entrada al sistema, ya que estas son enviadas al Arduino y seguidamente al agente reactivo.

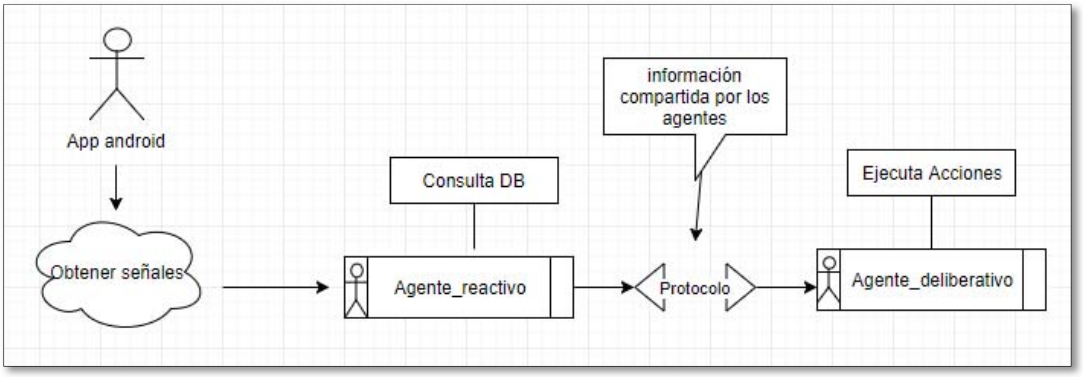

Diagrama 2 Diseño de arquitectura del sistema

La detección de anomalías en las vibraciones se efectuará de acuerdo con las reglas que se ajustan en las creencias del agente deliberativo, de él depende identificar señales normales y anormales, ya que, este agente es el encargado de deliberar con base a las señales obtenidas por el agente reactivo a través del actor App Android.

5.1.1.4 Casos de uso

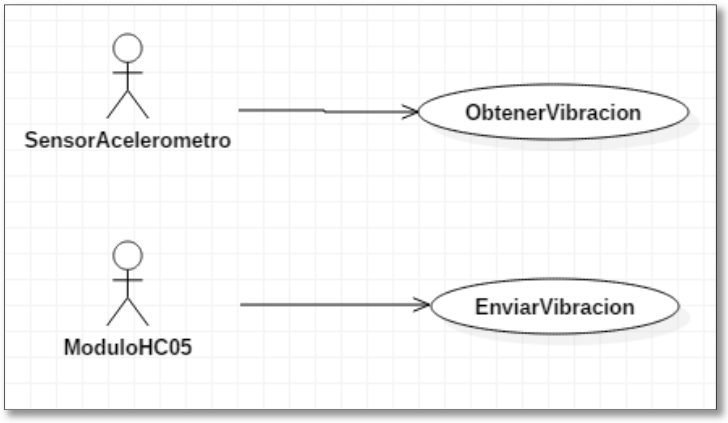

Diagrama 3 Casos de uso de actuadores. Fuente Felipe Chaux (2017)

5.1.1.5 Especificaciones de casos de uso

Caso de uso	ObtenerVibracion	Actor:	SensorAcelerometro
Resumen:	El sensor acelerometro del dispositivo Android obtendra la aceleracion que se genere en el , esta informacion se receptara mediante una APP Android .		
Objetivos:	* Obtener señal de vibracion * Enviar la informacion a la APP Android * Actualizar datos		
Precondiciones:	* Aplicación android ejecutada * Placa Arduino conectada		
Poscondiciones:	* Disponibilidad de datos		

Tabla 3 Descripción de caso de uso actuadores- obtener vibración. Fuente: Felipe Chaux (2017)

Caso de uso	EnviarVibracion	Actor:	ModuloHC05
Resumen:	El modulo de bluetooh HC05 se encargara de enviar via bluetooh la informacion obtenida de la APP Android hacia el agente reactivo receptor del arduino , este agente establecera comunicacion con el agente deliberativo control a traves de mensajes .		
Objetivos:	* Enviar la informacion hacia el agente receptor del arduino		
Precondiciones:	* Aplicación android ejecutada y vinculada a bluetooh * Placa Arduino conectada		
Poscondiciones:	* Disponibilidad de datos		

Tabla 4 Descripción de caso de uso actuadores- enviar vibración. Fuente: Felipe Chaux (2017)

5.1.1.6 Esquema de la ontología que manejaran los agentes

En la (Figura 17) se representa la ontología desarrollada para el entorno de los agentes, se muestran cada uno de los elementos: conceptos, predicados y acciones.

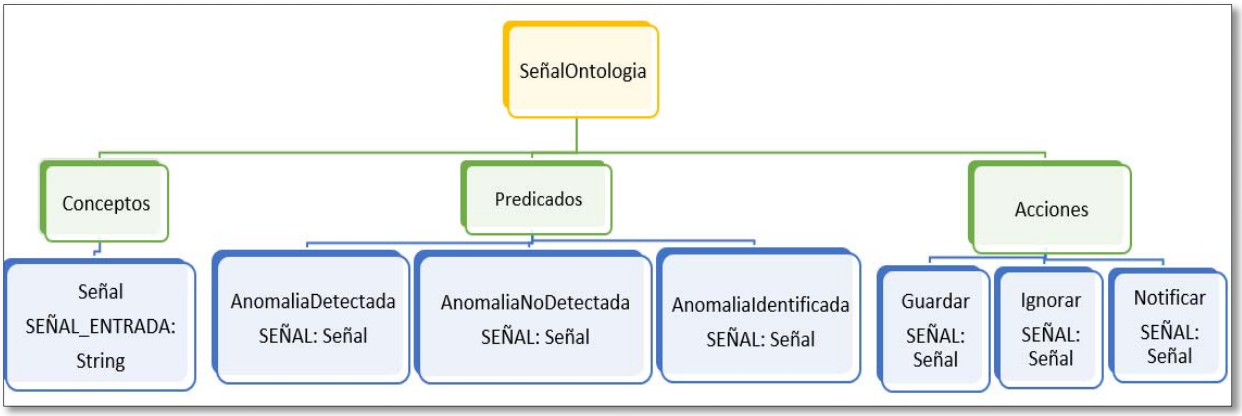

Figura 17 Esquema de ontología. Fuente Felipe Chaux (2017)

5.2 Diseño de aplicación Android y circuito

5.2.1 Diseño de aplicación Android con APP INENTOR

En la (Figura 18) se aprecia la programación por bloques, de la aplicación Android que se encargara de capturar y enviar vía bluetooth, la información que registrara el sensor acelerómetro del dispositivo.

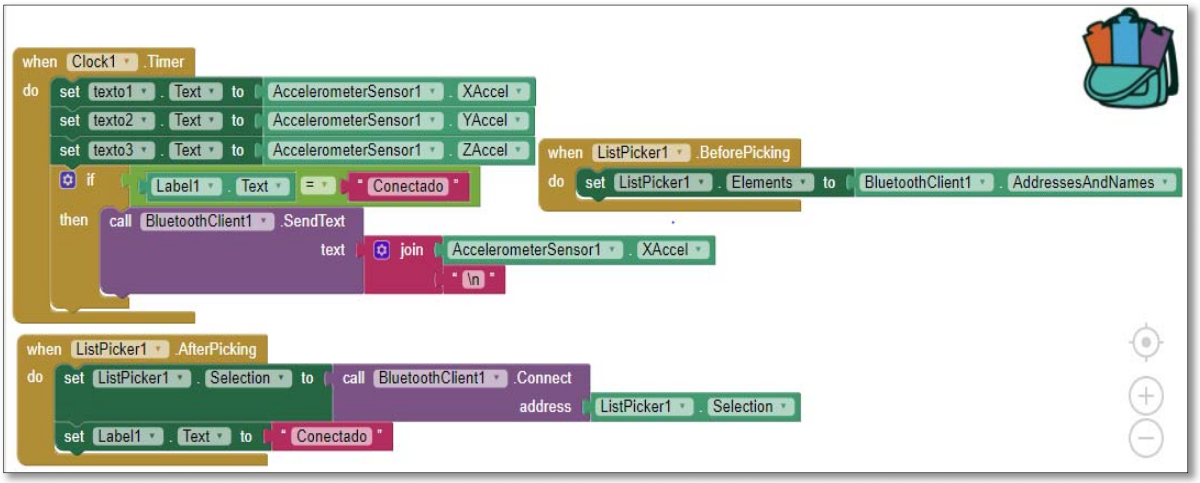

Figura 18 Programación y diseño de aplicación. Fuente: Felipe Chaux (2017)

La interfaz gráfica de usuario (GUI) de la aplicación se observa en la (Figura 19), hay un botón de conexión llamado “conectar” para listar los dispositivos bluetooth disponibles, al tener conectada la placa Arduino con el módulo hc-05 este puede ser visible en la aplicación , pero antes de vincularse por la aplicación debe establecerse un emparejamiento con anterioridad desde el dispositivo móvil .

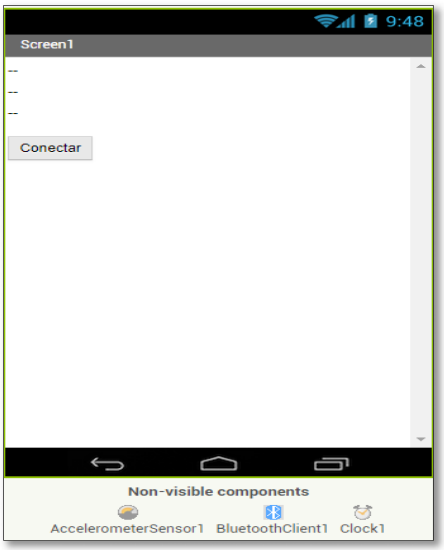

Figura 19 Interfaz gráfica de aplicación. Fuente: Felipe Chaux (2017)

5.2.2 Diseño del circuito Arduino - recepción de datos del acelerómetro del dispositivo con módulo bluetooth HC-05

En la (Figura 20) se evidencia la conexión realizada entre la placa Arduino Leonardo y el módulo bluetooth HC-05, como se ve en la conexión se utilizarán los pines digitales 11, 10 de la placa con los pines RXD y TXD del módulo, más los pines a tierra y de alimentación 5V, GND y VCC respectivamente.

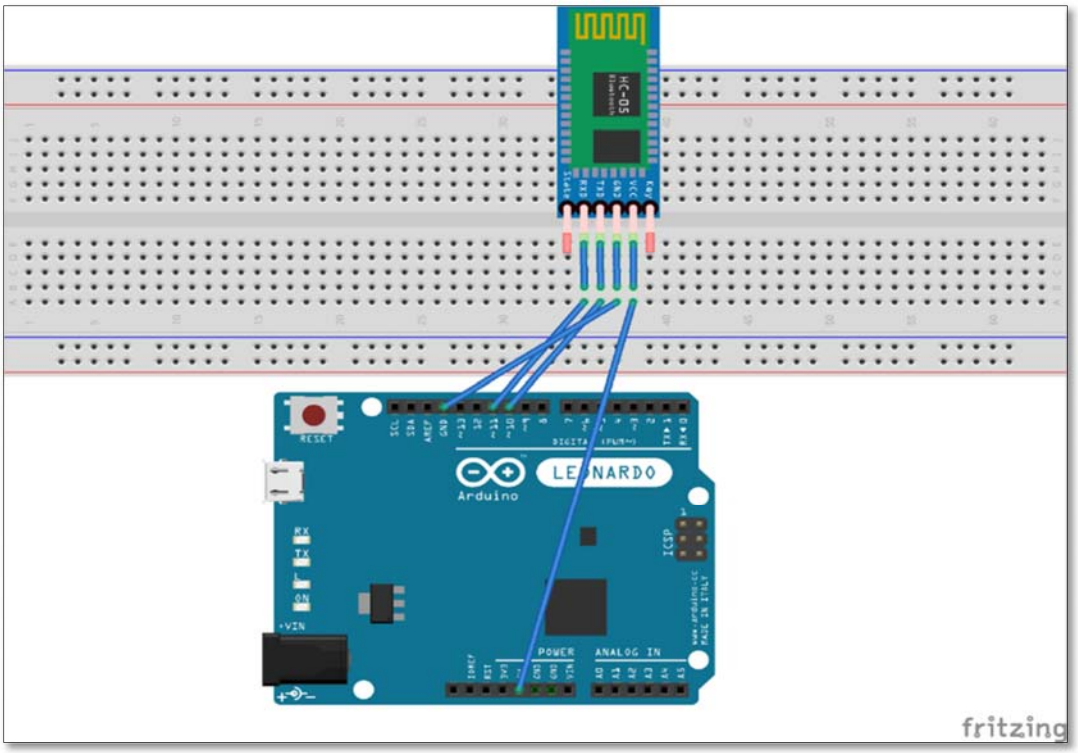

Figura 20 Circuito Conexión Arduino-Modulo HC-05. Fuente: Felipe Chaux (2017)

5.3 Desarrollo de código

En la (Figura 21) se aprecia la inicialización de los agentes que constituirán el sistema multi-agente, básicamente es una clase principal que se encarga de hacer la invocación por medio de argumentos que JADE reconoce.

```
public class Main {
     public static void main(String[] args) throws InterruptedException {
       String[]args1={"-gui",""};
       String[]args2={"-container","AgenteReceptorArduino:co.agentes.AgenteReceptorArdui
       jade.Boot.main(args1);
       jade.Boot.main(args2);

     }

}
```

Figura 21 Creación de los agentes. Fuente: Felipe Chaux (2017)

En las figuras (Figura 22 y Figura 23) se muestran los agentes desarrollados, los cuales extienden de *Agent* (concretamente de una clase de JADE que se encuentra en jade.core), el agente reactivo implementara

SerialPortEventListener gracias a la biblioteca RXTX , para poder recibir por medio de puerto serie, las señales provenientes de la placa Arduino.

```
public class AgenteReceptorArduino extends Agent implements SerialPortEventListener {

    private Codec codec = new SLCodec();
    private Ontology ontologia = SeñalOntologia.getInstance();
    static int cont = 0;

    SerialPort serialPort;
    public BufferedReader input;
    public OutputStream output;
    public static final int TIME_OUT = 0;
    public final String PUERTO = "COM3";
    public static final int DATA_RATE = 9600;
    String inputLine;

    public void inicializarConexionArduino() {
```

Figura 22 Agente reactivo. Fuente: Felipe Chaux (2017)

```
public class AgenteControl extends Agent {

    private Codec codec = new SLCodec();
    private Ontology ontologia = SeñalOntologia.getInstance();

    // Clase que describe el comportamiento que permite recibir un mensaje
    // y contestarlo
    class RespuestaComportamiento extends SimpleBehaviour {

        private boolean finished = false;

        public RespuestaComportamiento(Agent a) {
            super(a);
        }

        public void action() {

            System.out.println("\nEsperando señal arduino  del ReceptorArduino...");
```

Figura 23 Agente deliberativo. Fuente: Felipe Chaux (2017)

Para la establecer comunicación entre la placa Arduino y el entorno de desarrollo integrado NetBeans se utiliza la librería RXTX, en la (Figura 24) se aprecia la forma de establecer el puerto de comunicación y como este se abre para poder utilizarse mediante la declaración de parámetros necesarios como los mostrados en la (Figura 25).

```
public void inicializarConexionArduino() {

    CommPortIdentifier portId = null;
    Enumeration portEnum = CommPortIdentifier.getPortIdentifiers();

     //busqueda de una instancia de puerto serie como se establece en PORT_NAMES
    while (portEnum.hasMoreElements()) {
        CommPortIdentifier actualPuertoID = (CommPortIdentifier) portEnum.nextEle
        if (PUERTO.equals(actualPuertoID.getName())) {
            portId = actualPuertoID;
            break;
        }
    }
    if (portId == null) {
       // System.out.println("No se pudo conectar al puerto COM");
        return;
    }

    try {

        //se abre el puerto serie y se usa el nombre de clase para el appName.
        serialPort = (SerialPort) portId.open(this.getClass().getName(),
                TIME_OUT);

     //establecer parametros de puerto
        serialPort.setSerialPortParams(DATA_RATE,
```

Figura 24 Comunicación con placa Arduino. Fuente: Felipe Chaux (2017)

```
     //establecer parametros de puerto
        serialPort.setSerialPortParams(DATA_RATE,
                SerialPort.DATABITS_8,
                SerialPort.STOPBITS_1,
                SerialPort.PARITY_NONE);

        // se abren las transmisiones
        input = new BufferedReader(new InputStreamRead
        //output = serialPort.getOutputStream();
        // se agregan los eventos que escucharam
        serialPort.addEventListener((SerialPortEventLi
        serialPort.notifyOnDataAvailable(true);

    } catch (Exception e) {
        System.err.println(e.toString());
    }
```

Figura 25 Parámetros de puerto. Fuente: Felipe Chaux (2017)

En la (Figura 26) se puede observar la definición de la ontología, se define el nombre, luego el vocabulario que manejaran los agentes para la comunicación. Seguidamente, se concretan y se agregan estos a la ontología (ver Figura 27).

Los atributos del esquema son de tipo predefinido, así que en la definición de la ontología se usan los tipos primitivos STRING e INTEGER.

```
 */
public class SeñalOntologia extends Ontology {

    // Nombre de la ontología
    public static final String ONTOLOGY_NAME = "ontología de señales";
    // Vocabulario de la ontología que van a manejar los agentes
    public static final String SEÑAL = "Señal";

    public static final String SEÑAL_ENTRADA = "entrada";

    public static final String ANOMALIA_DETECTADA = "Anomalia";
    public static final String ANOMALIA_DETECTADA_SEÑAL = "señal";
    public static final String ANOMALIA_NO_DETECTADA = "SinAnomalia";
    public static final String ANOMALIA_NO_DETECTADA_SEÑAL = "señal";

    public static final String GUARDAR = "Guardar";
    public static final String GUARDAR_SEÑAL = "señal";
    public static final String IGNORAR = "Ignorar";
    public static final String IGNORAR_SEÑAL = "señal";

    // Instancia de la ontología (que será única)
    private static Ontology instancia = new SeñalOntologia();

    // Método para acceder a la instancia de la ontología
```

Figura 26 Definición de elementos de la ontología Fuente. Felipe Chaux (2017)

```
public static Ontology getInstance() {
    return instancia;
}

private SeñalOntologia() {
    // AnomaliaOntologia extiende la ontología básica
    super(ONTOLOGY_NAME, BasicOntology.getInstance());

    try {
        // Adicion de los elementos
        add(new ConceptSchema(SEÑAL), Señal.class);
        add(new PredicateSchema(ANOMALIA_DETECTADA), AnomaliaDetectada.class);
        add(new PredicateSchema(ANOMALIA_NO_DETECTADA), AnomaliaNoDetectada.class);
        add(new AgentActionSchema(GUARDAR), Guardar.class);
        add(new AgentActionSchema(IGNORAR),IgnorarAnomalia.class);
```

Figura 27 Adición de elementos de la ontología. Felipe Chaux (2017)

Para cada uno de los elementos de la ontología (concepto, acción y predicado) se hace la definición de la estructura respectiva como la mostrada en la (Figura 28), dicha estructura se añade mediante la obtención de esquema para la ontología.

```
    // Estructura del esquema para el concepto SEÑAL
    ConceptSchema cs = (ConceptSchema) getSchema(SEÑAL);
    cs.add(SEÑAL_ENTRADA, (PrimitiveSchema) getSchema(BasicOntology.STRING));

    // Estructura del esquema para el predicado ANOMALIA DETECTADA
    PredicateSchema ps = (PredicateSchema) getSchema(ANOMALIA_DETECTADA);
    ps.add(ANOMALIA_DETECTADA_SEÑAL, (ConceptSchema) getSchema(SEÑAL));
    // Estructura del esquema para el predicado ANOMALIA NO DETECTADA
    PredicateSchema ps1 = (PredicateSchema) getSchema(ANOMALIA_NO_DETECTADA);
    ps1.add(ANOMALIA_NO_DETECTADA_SEÑAL, (ConceptSchema) getSchema(SEÑAL));

    // Estructura del esquema para la acción GUARDAR
    AgentActionSchema as = (AgentActionSchema) getSchema(GUARDAR);
    as.add(GUARDAR_SEÑAL, (ConceptSchema) getSchema(SEÑAL));
     // Estructura del esquema para la acción IGNORAR
    AgentActionSchema as1 = (AgentActionSchema) getSchema(IGNORAR);
    as1.add(IGNORAR_SEÑAL, (ConceptSchema) getSchema(SEÑAL));

} catch (OntologyException e) {
    e.printStackTrace();
}
```

Figura 28 Estructuras de esquemas para la ontología. Fuente: Felipe Chaux (2017)

Para los elementos de la ontología desarrollada, se crea una clase relacionada de JAVA, un ejemplo es el concepto señal (ver Figura 29).

```
/**
 *
 * @author Felipe Chaux
 */
public class Señal implements Concept {

    private String entrada;

    public String getEntrada() {
        return entrada;
    }

    public void setEntrada(String entrada) {
        this.entrada = entrada;
    }
```

Figura 29 Concepto seña para la ontología. Fuente: Felipe Chaux (2017)

Finalmente, se desarrolla el código para la recepción de las señales por medio del módulo HC-05 en Arduino (ver Figura 30). Se establecen los pines a usar del arduino : 10 y 11, para los respectivos pines del modulo: RX y TX, si hay disponibilidad de datos, se escribiran en el puerto serie.

blueto

```
#include <SoftwareSerial.h>   // librería  SoftwareSerial
SoftwareSerial BT(10,11);    // Definicion de  pines RX y TX del Arduino conectados al Bluetooth
 char datos=0;

void setup()
{
  BT.begin(9600);       // Inicializar el puerto serie BT
  Serial.begin(9600);   // Inicializar el puerto serie
}

void loop()
{
  // Si llega un dato por el puerto BT se envía al monitor serial
  if(BT.available())
  {

    datos =BT.read();
   Serial.write(datos);

  }

}
```

Figura 30 Desarrollo de código en Arduino IDE. Fuente: Felipe Chaux (2017)

5.4 Pruebas

5.4.1 Montaje

En la (Figura 31) se muestra el montaje del prototipo, se evidencia el electrodoméstico a usar, el dispositivo móvil que registrara las vibraciones por medio de la aplicación Android, y la conexión de la placa Arduino para recibir estas vibraciones y enviarlas por puerto serie al ordenador.

Figura 31 Montaje de prototipo. Fuente: Felipe Chaux (2017)

5.4.2 Despliegue de agentes

En la (Figura 32) se muestra el despliegue exitoso de los agentes, cada agente empezara a poner en marcha sus respectivos comportamientos, el agente reactivo: realizara él envió de las señales provenientes de la placa Arduino hacia el agente deliberativo, para que este ejecute las acciones basadas en sus objetivos.

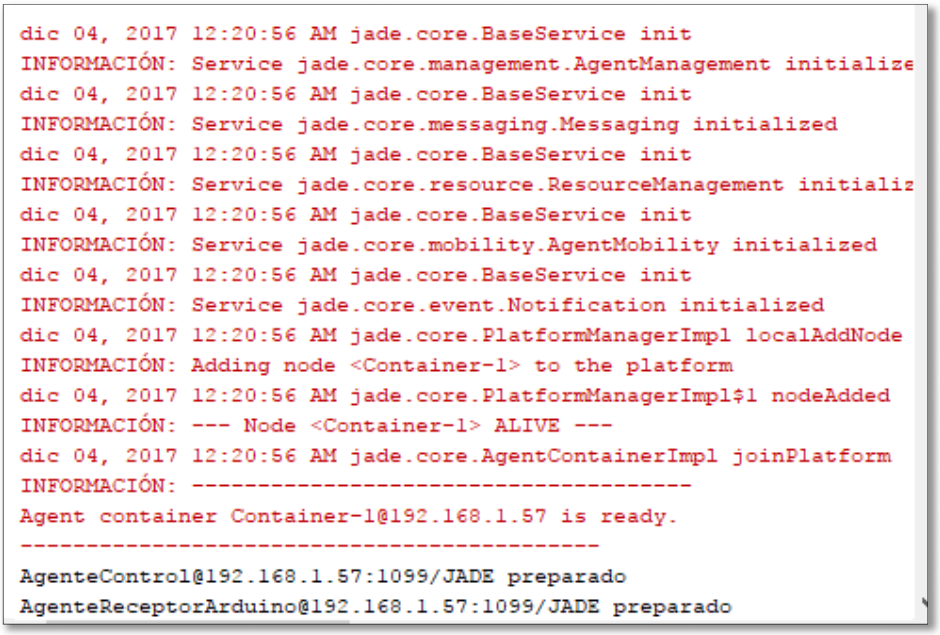

Figura 32 Despliegue de agentes. Fuente: Felipe Chaux (2017)

5.4.1 Ejecución de Aplicación Android

Se debe habilitar el bluetooth en el dispositivo móvil y seguidamente vincular con módulo HC-05, en este caso el nombre configurado es “felipe” (ver Figura 33)

Figura 33 Vinculación bluetooth dispositivo móvil. Fuente Felipe Chaux (2017)

Seguidamente se inicia la aplicación, y se selecciona el botón de conectar, para vincular la aplicación con el módulo bluetooth ver figuras (Figura 34 y Figura 35).

Figura 34 Ejecución de aplicación Android. Fuente: Felipe Chaux (2017)

En la (Figura 35) se muestra la lista de dispositivos bluetooth disponibles, se procede a elegir el nombre del módulo que se ha configurado previamente como “felipe”.

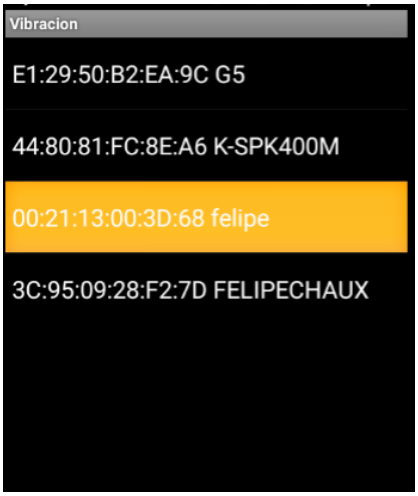

Figura 35 Vinculación de aplicación Android con modulo bluetooth. Fuente: Felipe Chaux (2017)

5.4.2 Comunicación entre agentes

En la (Figura 36) se evidencia el envío de mensajes entre los agentes, mediante un entorno de interfaz gráfica (GUI) que ofrece el agente *Sniffer* de JADE, donde se visualiza un diagrama de interacción, en el que se aprecia la comunicación entre el agente reactivo y al agente deliberativo.

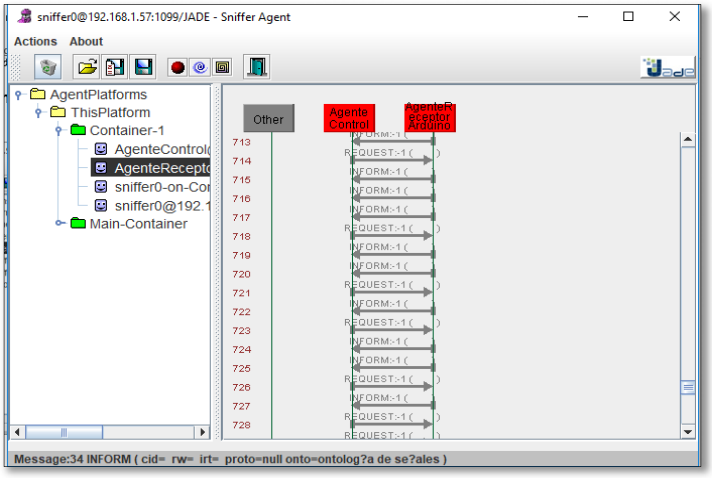

Figura 36 Comunicación de agentes. Fuente: Felipe Chaux (2017)

Los agentes son creados dentro de un contenedor (Container-1).

5.4.3 Recolección de datos

En la (Figura 37) se evidencia el comportamiento del agente deliberativo, y de cómo este empieza a recibir las señales de vibración ver (Figura 38), señales que estarán siendo enviadas por el agente reactivo.

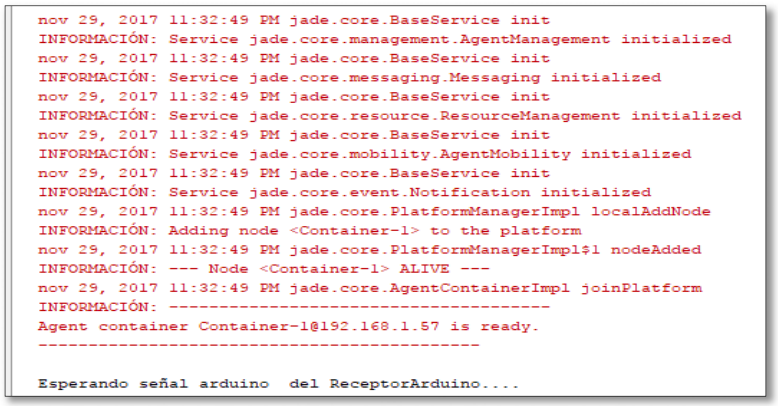

Figura 37 Adquisición de datos. Fuente: Felipe Chaux (2017)

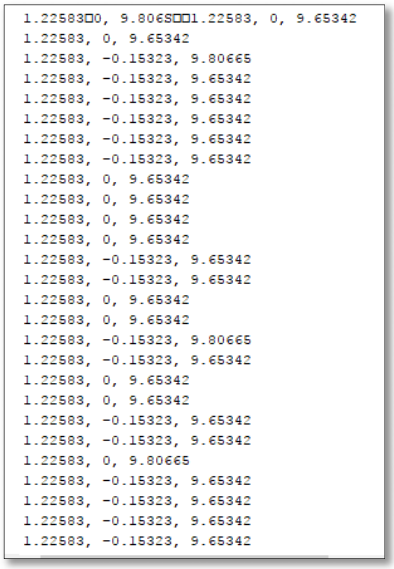

Figura 38 Agente reactivo- Recepción de información. Fuente: Felipe Chaux (2017)

5.4.4 Procesamiento de datos del sistema

En la (Figura 39) se aprecia la información obtenida del sensor acelerómetro del dispositivo móvil, en los ejes (x, y, z). Se hacen mediciones con frecuencias de 100 ms, 500 ms y 1000 ms para la identificación del mejor tiempo. Este cambio de frecuencias se hace desde la aplicación Android con el fin de hacer pruebas.

Potencia1			Potencia3			Potencia5		
x	y	z	x	y	z	x	y	z
367.749	0.30646	58.227	398.395	-0.45969	1.057.279	5.2.45166	-0.15323	658.884
398.395	0.15323	398.395	367.749	-0.61292	995.988	352.426	-0.76614	536.301
337.104	-199.198	995.988	398.395	-0.61292	1.011.311	321.781	-0.61292	735.499
321.781	-21.452	1.041.957	260.489	-0.61292	1.011.311	337.104	-0.45969	536.301
383.072	-0.45969	858.082	245.166	-0.15323	965.342	291.135	-0.61292	367.749
536.301	0.61292	413.718	413.718	-0.91937	658.884	275.812	-0.61292	735.499
58.227	0.30646	520.978	337.104	-10.726	980.665	367.749	-0.30646	597.593
643.561	-0.15323	827.436	275.812	-0.45969	1.057.279	337.104	-0.30646	505.655
321.781	-199.198	1.072.602	10.726	-0.61292	444.364	398.395	-0.45969	79.679
291.135	-183.875	116.454	153.229	-0.30646	919.373	459.687	-0.30646	812.113
352.426	-0.76614	1.225.831	245.166	-0.76614	827.436	291.135	-0.30646	68.953
47.501	0.30646	398.395	260.489	-0.76614	628.239	459.687	-0.91937	505.655
229.843	-183.875	1.057.279	245.166	0	934.696	337.104	-0.45969	68.953
275.812	-168.552	1.087.925	413.718	-0.91937	643.561	321.781	-0.45969	735.499
352.426	-122.583	1.210.508	291.135	0	858.082	306.458	-0.30646	750.822
383.072	0.45969	383.072	275.812	-0.45969	1.011.311	321.781	-0.30646	643.561
398.395	0.61292	306.458	137.906	-0.61292	566.947	275.812	-0.45969	704.853
505.655	0.61292	398.395	137.906	-0.30646	965.342	459.687	-0.61292	827.436
520.978	-122.583	934.696	291.135	-0.91937	950.019	413.718	-0.61292	750.822
612.916	-0.30646	827.436	337.104	-0.91937	720.176	260.489	-0.45969	429.041
245.166	-199.198	1.057.279	168.552	-0.45969	58.227	337.104	-0.76614	275.812

100 ms | 500 ms | 1000 ms | Sin vibracion

Figura 39 Estudio manual de datos. Fuente Felpe Chaux (2017)

Este estudio se realiza, para analizar de manera manual las señales que procesaran en ultimas los agentes, para que no existan inconvenientes en la etapa de aprendizaje.

6 Resultados

6.1 Detección de anomalías

En las figuras (Figura 40, Figura 41 y Figura 42) se evidencia la detección de anomalías, mediante la comunicación del predicado “identificación de anomalía” y la acción “notificar” respectivamente, usados por los agentes.

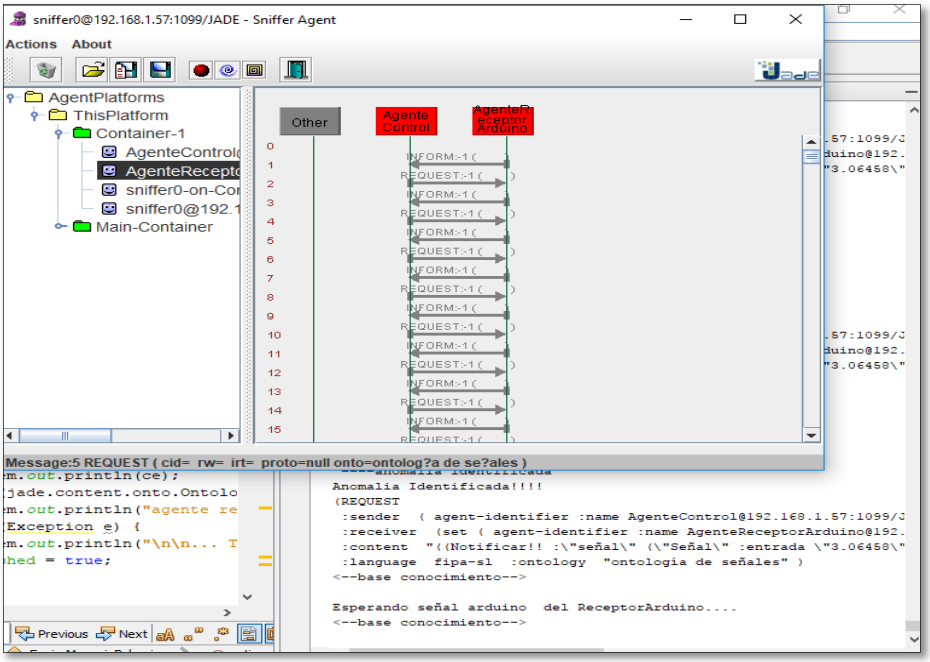

Figura 40 Identificación de anomalías. Fuente Felipe Chaux (2017)

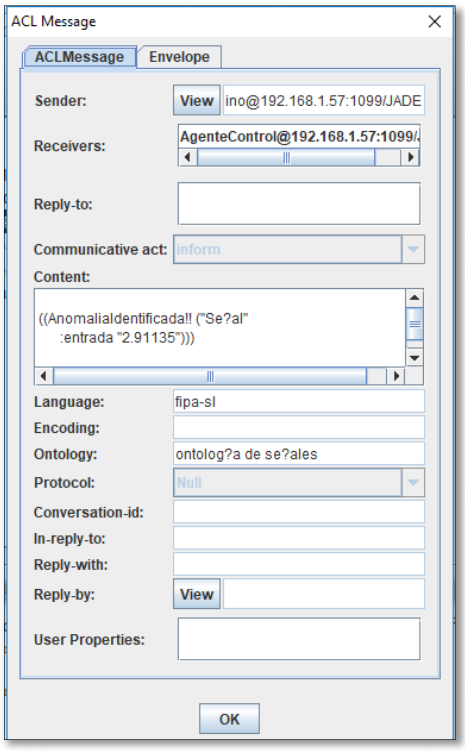

Figura 41 Predicado: anomalía identificada, enviado por el agente reactivo. Fuente Felipe Chaux (2017)

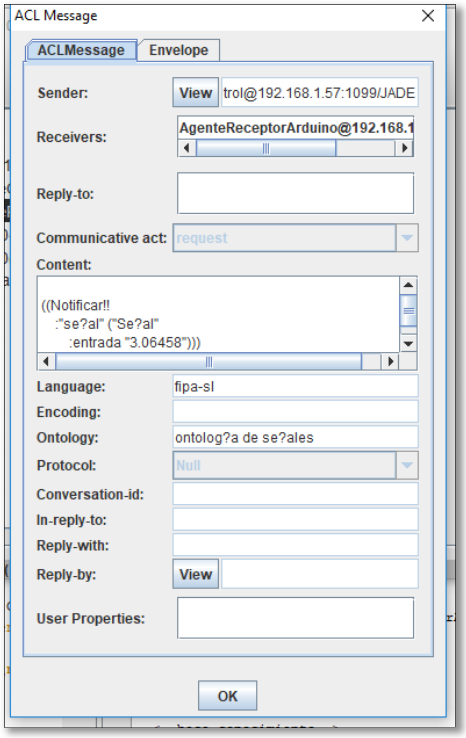

Figura 42 Acción: notificar, ejecutada por el agente deliberativo. Fuente Felipe Chaux (2017)

El sistema multi-agente reconoce anomalías que ya han sido procesadas por los agentes, para llegar a ello, el agente reactivo le comunica al agente deliberativo la existencia de alguna anomalía identificada, enseguida el agente deliberativo ejecuta la acción de notificar, y antes de que pase todo esto, los agentes deberán ir guardando conocimiento de las señales que son anomalías y las que no lo son, basándose el primer comportamiento que ejecuta el agente reactivo, el cual guarda las primeras percepciones de las señales.

7 Conclusiones

- Se evidenció que el *framework* JADE fue una gran herramienta para el desarrollo del sistema multi-agente, ya que el lenguaje de programación que se manipula es JAVA, este lenguaje es ampliamente utilizado por desarrolladores, y se encuentra bastante documentación en internet sobre él, así que por esto fue sencillo el desarrollo de los agentes. Adicionalmente JADE es muy intuitivo, ya que integra una interfaz gráfica de usuario (GUI) que hace fácil la gestión de los agentes.

- Se comprobó el correcto funcionamiento de la aplicación Android desarrollada para él envió de la señal de vibración hacia Arduino mediante el módulo de bluetooth HC-05.

- El sensor acelerómetro del dispositivo móvil Android, registró datos consistentes de vibración a la hora de hacer pruebas de mediciones en la licuadora a diferentes potencias.

- Las soluciones de software automatizadas pueden ser reemplazadas por agentes inteligentes, ya que ellos tienen características destacables como la capacidad de aprendizaje y construcción de percepciones de un entorno para la toma de decisiones.

- Se evidenció la integración, entre el sistema-multiagente y la placa Arduino para poder manipular las señales de vibración mediante los actuadores (Aplicación Androd y Modulo HC-05).

8 Trabajo Futuro

- A partir de lo realizado en el proyecto, se sugiere trabajar a futuro en la implementación de prototipos similares, para el manejo de temperaturas en plantas eléctricas, cuando estas superan sus horas de trabajo.
- Aplicación del sistema al monitoreo de signos vitales.
- Aplicación del sistema también en motores de autos y otras máquinas que generen vibraciones.

9 Referencias

[1] F. C. G. de León, «Tecnología del mantenimiento industrial,» 1998. [En línea]. Available: https://books.google.es/books?hl=es&lr=&id=bOrFC3532MEC&oi=fnd&pg=PA21&dq=automatizacion+industrial+mantenimiento&ots=6N91JGOnOO&sig=I25HCZlyVSsn-rk3jhInI6_EUbw#v=onepage&q=automatizacion%20industrial%20mantenimiento&f=false . [Último acceso: 19 10 2017].

[2] U. T. d. Pereira, «Análisis de vibraciones: una herramienta clave en el mantenimiento predictivo,» 08 2010. [En línea]. [Último acceso: 12 10 2017].

[3] E. d. m. predictivo. [En línea]. Available: http://www.superdirectorios.com/emp/?gclid=CNulg4Pa7NYCFYVBhgod5SYDTQ. [Último acceso: 12 10 2017].

[4] Accenture, 2017. [En línea]. Available: https://www.accenture.com/co-es/insight-artificial-intelligence-future-growth. [Último acceso: 11 10 2017].

[5] M. Wooldridge, «An introduction to multiagent systems,» 10 10 2009. [En línea]. Available: https://books.google.es/books?hl=es&lr=&id=X3ZQ7yeDn2IC&oi=fnd&pg=PR13&dq=An+Introduction+to+MultiAgent+Systems,+2nd+Edition&ots=WGhltu9v80&sig=L_lib-KDx3h6w2YhJI9jAmnZ-WU#v=onepage&q=An%20Introduction%20to%20MultiAgent%20Systems%2C%202nd%20Edition&f=fals.

[6] «wikybayes.wikidot,» [En línea]. Available: http://wikybayes.wikidot.com/redes-neuronales. [Último acceso: 02 12 2017].

[7] https://www.coursera.org, «Coursera,» [En línea]. Available: https://www.coursera.org/learn/machine-learning/lecture/Ujm7v/what-is-machine-learning. [Último acceso: 24 11 2017].

[8] G. White, «Introducción al análisis de vibraciones,» 2010. [En línea]. Available: https://s3.amazonaws.com/academia.edu.documents/40509240/290210174-Glen-White-Analisis-de-Vibraciones.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1509237159&Signature=mXXQNeaRtxooezbVGKRjPOPxgJ8%3D&response-content-disposition=inline%3B%20filename%3D2. [Último acceso: 28 10 2017].

[9] «Arduino,» [En línea]. Available: http://arduino.cl/que-es-arduino/. [Último acceso: 12 10 2017].

[10] «Arduino store,» [En línea]. Available: https://store.arduino.cc/usa/arduino-leonardo-with-headers. [Último acceso: 28 10 2017].

[11] «Arduino leonardo alimentacion,» [En línea]. Available: https://geekytheory.com/arduino-leonardo. [Último acceso: 04 11 2017].

[12] Jade, «Marco de desarrollo de JAVA Agent,» [En línea]. Available: http://jade.tilab.com/. [Último acceso: 04 11 2017].

[13] F. ORG, «fipa,» [En línea]. Available: http://www.fipa.org/. [Último acceso: 12 11 2017].

[14] «JADE PROGRAMMER'S GUIDE,» [En línea]. Available: http://jade.tilab.com/doc/programmersguide.pdf. [Último acceso: 12 11 2017].

[15] M. Wooldridge, «Developing-Multi-Agent-Systems-with-JADE».

[16] programacionjade, «programacionjade,» [En línea]. Available: https://programacionjade.wikispaces.com/Ontolog%C3%ADas. [Último acceso: 24 11 2017].

[17] U. p. d. salamanca, «Sistemas de Agentes y Multiagentes,» Madrid, 2007.

[18] P. C. Morales, «grupo web de agentes inteligentes,» Vigo.

[19] «Sensores,» [En línea]. Available: http://www.pce-iberica.es/instrumentos-de-medida/sistemas/sensores.htm. [Último acceso: 11 04 2017].

[20] «123rf,» [En línea]. Available: https://es.123rf.com/imagenes-de-archivo/acelerometro_movil.html?imgtype=1&sti=m47penjlo2s3x2q5zj|&mediapopup=41372023. [Último acceso: 29 11 2017].

[21] «hsqldb.org,» [En línea]. Available: http://hsqldb.org/. [Último acceso: 24 11 2017].

[22] «http://rxtx.qbang.org/,» [En línea]. Available: http://rxtx.qbang.org/wiki/index.php/FAQ. [Último acceso: 23 11 2017].

[23] «App Inventor WEB,» [En línea]. Available: http://appinventor.mit.edu/explore/about-us.html. [Último acceso: 23 11 2017].

[24] D. J. Matich, «Redes Neuronales: Conceptos básicos y aplicaciones. Cátedra de Informática Aplicada a la Ingeniería de Procesos–Orientación,» 2001. [En línea]. [Último acceso: 19 10 2017].

[25] «Arduino leonardo,» [En línea]. Available: http://arduino.cl/arduino-leonardo/. [Último acceso: 04 11 2017].

[26] J. M. C. Javier Bajo, «Agentes y Sistemas Multiagente - curso de doctorado,» salamanca, 2008.

10 Anexos

10.1 Repositorios de código fuente

10.1.1 One Drive

https://poligran-my.sharepoint.com/personal/abchauxgutierre_poligran_edu_co/Documents/codigo-agentes/AGENTS_IA?csf=1&e=20b37c928a8f488d8164522b3c8dd05d

10.1.2 GitHub

https://github.com/felipechaux/MonitoreoAnomaliasAgentesJade