<1604>

PLATAFORMA DE DRAMATIZACIÓN ROBÓTICA MODULAR

David Stiven Ávila González
Miguel Ángel Bermeo Ayerbe
Fabián Andrés Merchán Jiménez

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA ELECTRÓNICA
CARRERA DE INGENIERÍA DE SISTEMAS
BOGOTÁ, D.C.
<2016>

<1605>

PLATAFORMA DE DRAMATIZACIÓN ROBÓTICA MODULAR

Autor(es):

David Stiven Ávila González
Miguel Ángel Bermeo Ayerbe
Fabián Andrés Merchán Jiménez

MEMORIA DEL TRABAJO DE GRADO REALIZADO PARA CUMPLIR UNO DE LOS REQUISITOS PARA OPTAR AL TITULO DE INGENIERO DE SISTEMAS Y AL TITULO DE INGENIERO ELECTRÓNICO

Directores

Alejandra María González Correal

Enrique González Guerrero

Jurados del Trabajo de Grado

Por Confirmar

Página web del Trabajo de Grado

http://pegasus.javeriana.edu.co/~CIS1630IN03

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERIA
CARRERA DE INGENIERÍA DE SISTEMAS
CARRERA DE INGENIERÍA ELECTRÓNICA
BOGOTÁ, D.C.
Diciembre, 2016

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA DE SISTEMAS
CARRERA DE INGENIERÍA ELECTRÓNICA

Rector Magnífico

Jorge Humberto Peláez Piedrahita, S.J.

Decano Académico Facultad de Ingeniería

Ingeniero Jorge Luis Sánchez Téllez

Director de la Carrera de Ingeniería de Sistemas

Ingeniera Mariela Josefina Curiel Huérfano

Director Departamento de Ingeniería de Sistemas

Ingeniero Rafael Andrés González Rivera

Director de la Carrera de Ingeniería Electrónica

Ingeniera Alejandra María González Correal

Director Departamento de Ingeniería Electrónica

Ingeniero Fredy Orlando Ruiz Palacios

Artículo 23 de la Resolución No. 1 de Junio de 1946

"La Universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado. Sólo velará porque no se publique nada contrario al dogma y la moral católica y porque no contengan ataques o polémicas puramente personales. Antes bien, que se vean en ellos el anhelo de buscar la verdad y la Justicia"

AGRADECIMIENTOS

Primero que todo, quiero agradecerle a Dios por guiarme y convertirme en la persona que soy. A a mis compañeros, que me apoyaron en este difícil camino, sin su ayuda no hubiera sido posible, siempre tendrán aquí más que un amigo, un compañero de vida.
A mi novia, gracias por ser un apoyo en mi vida y permitirme ser uno en la tuya. Finalmente, doy gracias a mi familia por el apoyo incondicional y el amor que me han brindado; a mi papa, se que desde el cielo te regocijas de mis logros, gracias por enseñarme el hombre que debo ser; a mi mama, la mejor mama del mundo, gracias por tu amor y tus cuidados; a mi hermana, mi pequeño tesoro, gracias por soportarme, siempre recuerda que cuentas con mi apoyo; los amo y sepan que mis triunfos son por y para ustedes.

- David Stiven Avila Gonzalez

Moni, papá y mamá, los amo
Esto es para ustedes
- Fabián

Agradezco a mi Mamá por guiarme y apoyarme incondicionalmente, gracias por todo el esfuerzo que hiciste que me permitieron estudiar en esta universidad. En este paso grande de mi vida, quiero darle gracias a mi Papá DEP, por haber sido un excelente padre, en el corto tiempo que compartí con el me enseño lo necesario para superar cada meta en mi vida. Gracias Dios por siempre estar conmigo y no desampararme. Durante la carrera encontré personas maravillosas, en especial quiero agradecer al Capitulo Javeriano ACM, gracias al capítulo aprendí capacidades que me han ayudado a nivel profesional.

- Miguel Angel Bermeo Ayerbe

Agradecemos a Alejandra y Enrique por la confianza dada para llevar a cabo este trabajo de grado, también por su comprensión, paciencia, esfuerzo y dedicación brindada. Gracias por creer en nosotros.

- Equipo AIO

CONTENIDO

ABSTRACT

In this project is designed and implemented a logical architecture of a modular robotic platfform oriented to dramatization, wich takes the role of an actor in a multi-agent system in wich many actors are controlled by a director in order to perform a theater script. Additionally, a functional prototype was developed with 4 modules: ***processing***, ***mobility***, ***audiovisual*** and ***manipulation***, wich were developed in Java, C++, JavaScript and C, respectively. Finally, a framework called ***SOCA*** was developed, who has the function of serving as an intermediary between a system and it's modules, making transparent the communication and serving as coordinator and synchronizer.

RESUMEN

En este proyecto se diseña e implementa la arquitectura lógica de una plataforma robotica modular orientada a dramatización, la cual toma el rol de un actor en un sistema multiagente en el que muchos actores son controlados por un director con el fin de intepretar un guion teatral. Adicionalmente, se desarrolló un prototipo funcional con 4 módulos: ***procesamiento***, ***movilidad***, ***audiovisual*** y ***manipulación***, los cuales fueron desarrollados en Java, C++, JavaScript y C, respectivamente. Finalmente, se desarrolló un agente llamado ***SOCA,*** quien tiene la función de servir como intermediario entre un sistema y sus módulos, hacer transparente la comunicación y servir de coordinador y sincronizador.

I - INTRODUCCIÓN

1. Oportunidad, problemática, antecedentes

Este proyecto surge de la problemática discutida en el grupo de investigaciones SIDRe acerca de la escasa aplicación de la robótica en el área educativa en temáticas diferentes a la tecnología, aun cuando es una herramienta que potencia la interdisciplinariedad escolar, la exploración y la creatividad, además de fomentar capacidades de observación y percepción [1][2].

A causa de lo anterior, se ve la oportunidad de desarrollar una plataforma robótica educativa que utilice como principal herramienta de enseñanza la dramatización, la cual desarrolla las habilidades sociales, aumenta la autoestima, enseña a trabar en equipo y estimula la imaginación [3].

Finalmente, este proyecto fue desarrollado para satisfacer la necesidad de diseñar la arquitectura lógica y física de una plataforma básica de robótica modular que diera soporte a la tesis doctoral de la ingeniera Flor Ángela Bravo MSc.

1.1. Formulación del problema que se resolvió

¿Cómo debe ser la arquitectura lógica y física de una plataforma robótica enfocada a la dramatización?

1.2. Justificación del problema

Debido a la gran cantidad de avances tecnológicos, los métodos de enseñanza tradicionales han evolucionado para satisfacer las expectativas de los educandos [4], dando origen a nuevas formas de enseñanza como la robótica pedagógica, la cual es un medio de aprendizaje que busca crear y utilizar prototipos robóticos y programas especializados con fines pedagógicos [5]. Este tipo de enseñanzas han mostrado resultados prometedores, debido a que “dan a los estudiantes una visión más interesante y divertida de la ciencia y la ingeniería, además de poder observar una aplicación práctica de los conceptos teóricos en los campos de la matemática y la tecnología” [6]. Sin embargo, la inclusión de la robótica en un contexto educativo ha sido principalmente enfocada hacia el aprendizaje de tecnología y todos aquellos aspectos relacionados a esta como la programación y la mecatrónica [7], a pesar de que puede ser utilizada en otras áreas como las matemáticas y las ciencias naturales [8].

Este trabajo de grado continua la exploración de la robótica educativa en la Pontificia Universidad Javeriana usando la dramatización como principal herramienta de enseñanza. Lo

anterior, debido a que el teatro y la dramatización han demostrado grandes beneficios en la educación primaria tanto en aspectos psicológicos como en educativos, tales como, interés por el estudio, autovaloración, desahogo de sentimientos, autoconfianza y tolerancia, entre otros [9]. Adicionalmente, para los espectadores observar un objeto animado actuando según lo que se quiere enseñar, ha demostrado mejores resultados de aprendizaje que un objeto estático que el docente utiliza para su explicación [10]. Es por esto, que museos de todo el mundo enfocan sus esfuerzos en desarrollar plataformas robóticas animadas para educar a sus visitantes, como la jirafa robótica en la exhibición temática del museo nacional de ciencia, tecnología y espacio israelí y la presentación teatral robótica del mismo [11], lo que da a entender que la dramatización es una herramienta útil y explotable en la educación, la cual se podría aplicar en la enseñanza escolar.

Finalmente, con el propósito de darle un mayor sustento a la razón del desarrollo del presente proyecto, se realizó una entrevista a Mónica Brijaldo[1] MSc., quien ratificó la necesidad de utilizar la robótica en otras áreas de conocimiento como literatura, biología y ciencias sociales, entre otros. Adicionalmente, recalcó la importancia que tiene la tecnología en la educación al ser una herramienta atractiva para los niños y mostrar resultados satisfactorios cuando se ha usado.

1.3. Impacto Esperado

A corto plazo, la plataforma desarrollada será una contribución importante para varios grupos de investigación en especial el grupo de investigación SIDRe de la Pontificia Universidad Javeriana, al poder ser utilizada como base para el desarrollo de trabajos de grado futuros.

A mediano plazo, se espera que, con los aportes hechos por otros trabajos de grado, de pregrado y maestría, se realice una primera versión de producto, el cual sea adquirido por instituciones educativas con el fin de beneficiar a sus estudiantes motivándolos con el uso de tecnología en el aula.

A largo plazo, la plataforma podrá ser comercializada y usada por entidades educativas, profesores y padres de familia como una herramienta que potencialice y facilite el aprendizaje de los niños. Finalmente, se espera crear una empresa alrededor de una versión del producto que genere beneficios monetarios tanto para la universidad como los partícipes de la organización.

[1] Profesora del Departamento de Educación de la Pontifica Universidad Javeriana.

2. Descripción del Proyecto

2.1. Objetivo general

Desarrollar una plataforma robótica modular orientada a la dramatización

2.2. Objetivos específicos

- Diseñar una arquitectura física y lógica del sistema basada en el modelo BDI con un esquema modular, versátil y escalable enfocada a dramatización.
- Desarrollar (Diseñar, implementar y verificar) un módulo de manipulación, de recolección de datos y reproducción audiovisual, de desplazamiento y de procesamiento con características de flexibilidad y escalabilidad.
- Diseñar y ejecutar pruebas de integración entre módulos.
- Realizar una validación operativa y de aceptación de la plataforma con los directores del proyecto

3. Conceptos fundamentales

3.1. Agente Inteligente [12]

Un agente inteligente es una entidad autónoma que observa a través de sensores y actúa dentro de un ambiente usando actuadores y dirige sus actividades a través de metas. Los agentes inteligentes también pueden aprender o usar conocimiento para alcanzar sus objetivos. Se caracterizan por ser autónomos, heterogéneos, dinámicos, comunicativos, proactivos, sociable y cuenta con protocolos de comunicación.

3.2. BDI (*Beliefs*, *Desires*, *Intentions*) [13]

Es un modelo de agentes enfocado a software basado en los principios de Michael Bratman [14], desarrollado para agentes inteligentes que implementan las creencias, deseos e intenciones de un agente racional.

- Una creencia representa la información del estado del agente, es decir sus creencias acerca del mundo.
- Un deseo representa el estado motivacional del agente, representa los objetivos o situaciones que el agente gustaría o necesita cumplir.
- Las intenciones representan el estado deliberativo del agente, es decir lo que el agente decide hacer, las intenciones son deseos que se ven representadas en un plan. Un conjunto de intenciones cumple el deseo del agente.

3.3. Diseño Modular [15]

Es la capacidad de un sistema para ser representado como la unión de varias partes que se integran para cumplir un objetivo común. Cada una de las partes se denomina módulo, el cual es capaz de desempeñar tareas de manera aislada a los demás módulos del sistema. De esta forma cada módulo es observado por los demás como una caja negra que ofrece servicios.

4. Estado del Arte

Con el fin de analizar las características de dispositivos e iniciativas similares se presentan, a continuación, algunos de los trabajos y dispositivos relacionado con robótica educativa más significativos para la investigación.

4.1. NAO, el nuevo aliado de los profesores [16]

Aldebarán, una compañía de robótica francesa, ha desarrollado un robot humanoide con gran cantidad de características que lo hacen una plataforma con bastantes áreas de aplicación. Este robot cuenta con 25 grados de libertad en su movimiento, controladores de equilibrio, capacidad para ponerse de pie, cámaras, reconocimiento de objetos, reconocimiento y detección de rostros, capacidad para hablar 9 idiomas, reconocimiento de voz, detección y localización de sonidos, control de potencia consumida en motores adaptativa, sistema de anticolisión, administrador de recursos e interfaz de programación por seguimiento de comandos o coreografías pre especificadas [17]. La gran cantidad de características que tiene NAO hace que en el área de educación no se limite solo a un intervalo de edades sino que se pueda dividir en 4 grupos: educación secundaria, educación universitaria, investigación y educación especial.

En la educación secundaria, NAO es utilizado generalmente para enseñar asignaturas científicas de una manera divertida (Matemáticas, física, mecánica, etc.). Adicionalmente, gracias a sus diferentes niveles de programación los estudiantes más pequeños pueden programar rutinas utilizando bloques mientras que los más avanzados pueden desarrollar directamente en Phyton o C++, lo que les permite elaborar obras de teatro y rutinas de enseñanza.

Los resultados experimentales de utilizar este dispositivo en una institución escolar dan como resultado un nivel de satisfacción alto en los usuarios finales. Sin embargo, debido a su apariencia, los estudiantes tiene una expectativa más alta de las verdaderas capacidades que el robot tiene al asociarlo con un humano esperando que haga deporte, conduzca y cocine [18].

4.2. Impact of Using an Educational Robot-Based Learning System on Student's Motivation in Elementary Education.

Shin-Guang Elementary School, ubicada en Taichung City, Taiwán, fue el lugar donde tres miembros de la IEEE implementaron una metodología de robótica educativa, la cual consistía en que un robot explicará diapositivas sustituyendo o apoyando al profesor.

Para esto, la plataforma cuenta con 3 capas. La primera, llamada "autorización", se encarga de recibir datos del profesor por medio de una interfaz gráfica para parametrizar las acciones de la plataforma (¿Qué diapositivas?, ¿En qué tiempos?, ¿Qué dirá el robot?, etc.). La segunda, llamada "Transferencia", es la unidad de procesamiento del sistema, en ella se entienden las acciones recibidas por la primera capa y se transforman en acciones para cada dispositivo disponible en el sistema (robot, parlante, pantalla, etc.). La tercera, llamada "enseñanza", es el hardware como tal, en ella se encuentran todos los dispositivos con los que cuenta el profesor para diseñar su clase.

Los resultados de las pruebas en estudiantes de segundo demostraron que los estudiantes no solo se motivan más al observar una plataforma robótica, sino que también aumentan considerablemente su desempeño escolar[19].

4.3. Interaction with Animated Robots in Science Museum Programs: How Children Learn?

La robótica educativa, apoyada de metodologías educativas como el teatro ha sido utilizada fuera de los colegios con resultados satisfactorios. Este es el caso del museo de computación de Boston, en donde se utilizaron diferentes robots para mostrar a los niños analogías entre los componentes de los dispositivos y características del cuerpo humano. Adicionalmente, en la investigación se realizaron actividades en donde se introdujeron conceptos de estabilidad, centro de masa y tecnología.

Los resultados de los talleres mostraron que los niños demuestran interés, atracción y mejor recepción al utilizar la interacción con robots animados como metodología de educación. [20]

4.4. Function-based Modular Robot System.

El documento muestra diferentes metodologías de diseño de robots modulares tales como: *Function-based Method*, *IDEFO Modeling Method* y *Analysis Process of Modulization*, describiendo de manera general su proceso de diseño. Adicionalmente, explica un caso de estudio enfocado en robots de ataque, robot anti explosivos y robots contra incendios, en el cual se muestran y explican las estructuras resultantes de aplicar las metodologías. Cabe aclarar, que a pesar de que el área en que se utilizan estas metodologías no está alineada al

objetivo general del proyecto, este documento es de gran utilidad al contener información acerca del cómo diseñar un robot modular[21].

5. Metodología utilizada

La elaboración del proyecto utilizó un ciclo de vida en cascada complementada en la fase de implementación con las buenas prácticas de *SCRUM* [22], una metodología de desarrollo ágil que ha presentado buenos resultados en desarrollos tanto de software como de hardware [23], al aumentar significativamente el índice de productividad y reducir los tiempos de entrega al cliente.

AIO fue desarrollado en tres fases: Análisis y diseño, implementación y pruebas. En el cual, cada fase implementa un ciclo de vida iterativo e incremental y una metodología dedicada complementaria a la metodología base. El diagrama de proceso general que modela la metodología de desarrollo utilizada se presenta en la Ilustración 1.

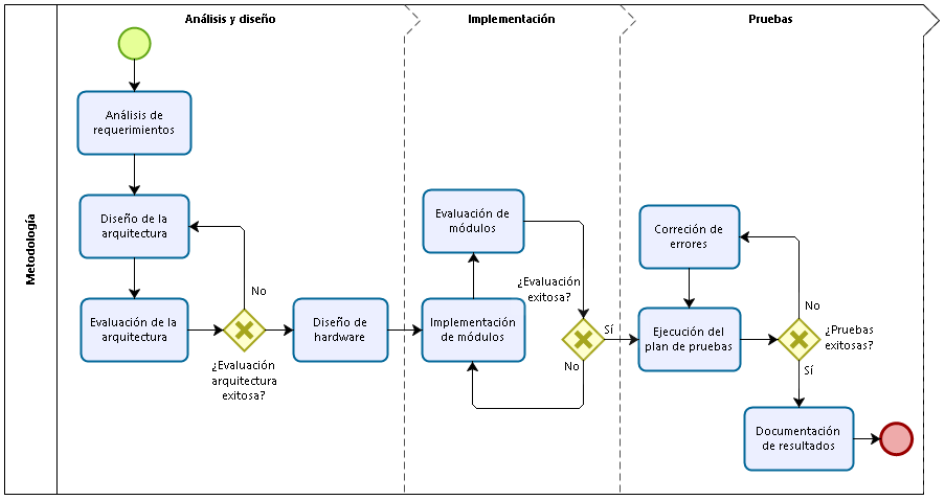

Ilustración 1. Metodología de desarrollo proyecto AIO.

5.1. Fase metodológica 1 (Análisis y diseño)

Primeramente, se identificaron los requerimientos arquitecturalmente significantes a partir de requerimientos de negocio previamente definidos. Luego, se priorizaron los requerimientos a criterio de los *stakeholders*, con los cuales, se diseñó la arquitectura hardware siguiendo la metodología Top-Down [24]. Finalmente se elaboró una descripción de la arquitectura utilizando el modelo de vistas 4+1 de Kruchten [25].

La descripción de la arquitectura software del sistema se elabora bajo los criterios del estándar ANSI/IEEE 1471-2000 [26], algunos de estos criterios incluyen la documentación de procesos

de desarrollo y mantenimiento en conjunto con la presentación de los repositorios creados, planificación desde arquitecturas existentes, criterios de conformidad de implementación de la arquitectura, revisión análisis y evaluación del sistema a lo largo del ciclo de vida del sistema, entre otros. Mientras que el desarrollo hardware se apoyó en el estándar de la IEEE *standard ontologies for robotics and automation* [27], *e*l cual define los principios bajo los cuales se desarrolla un robot basado en un sistema de agentes. Además, describe la relación entre el robot físico, un agente, y un sistema multiagentes, basados en SUMO y CORA, las cuales son ontologías que determinan los lineamientos conceptuales bajo los cuales se desarrolla una entidad robótica.

Finalmente, se elaboró el plan de pruebas de modo que fuese ejecutado al finalizar la fase de implementación. Las pruebas se diseñan con base en la metodología descrita en los libros "*Object-Oriented Software Engineering"* [28] e *"Ingeniería del Software*" [29], las cuales hace énfasis en el uso de buenas prácticas de programación y el uso de pruebas unitarias del código desarrollado. En cuanto al hardware, se basó en el estándar "*Standard Testeabilty Method For Embedded Core-Base Integrated circuits*" de la IEEE, el cual describe las pautas a llevar cabo para lograr una arquitectura de pruebas sólida, teniendo en cuenta el diseño de bloques embebidos, desde el más grande hasta el más pequeño, también incluyendo bloques externos, y de esta manera incentivar la reutilización de pruebas para diferentes bloques.

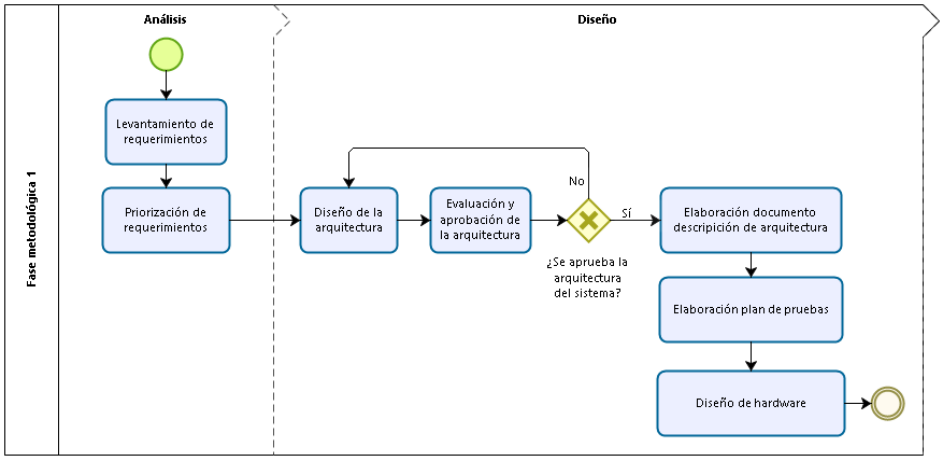

Ilustración 2. Diagrama de la fase metodológica 1.

5.2. Fase metodológica 2 (Implementación)

El desarrollo del framework y el prototipo funcional siguió la metodología "Desarrollo ágil de hardware" [30] complementada con Extreme Programming [31][32], una metodología ágil de desarrollo de software que ha demostrado ser un buen complemento para el desarrollo de hardware.

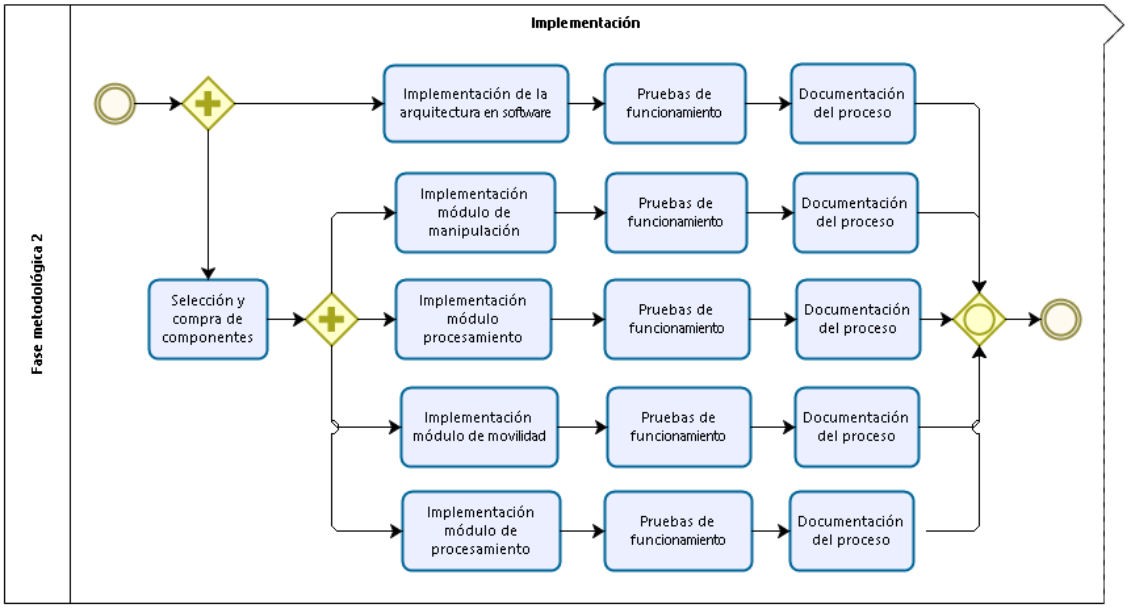

Ilustración 3. Diagrama de la fase metodológica 2

5.3. Fase metodológica 3 (Pruebas)

Las pruebas fueron diseñadas con base en los requerimientos encontrados en la fase de análisis del proyecto. En la sección V se encuentra el plan de pruebas, el cual detalla a profundidad las pruebas realizadas, cada una de ellas cuenta con los parámetros y restricciones necesarias para llevarla a cabo.

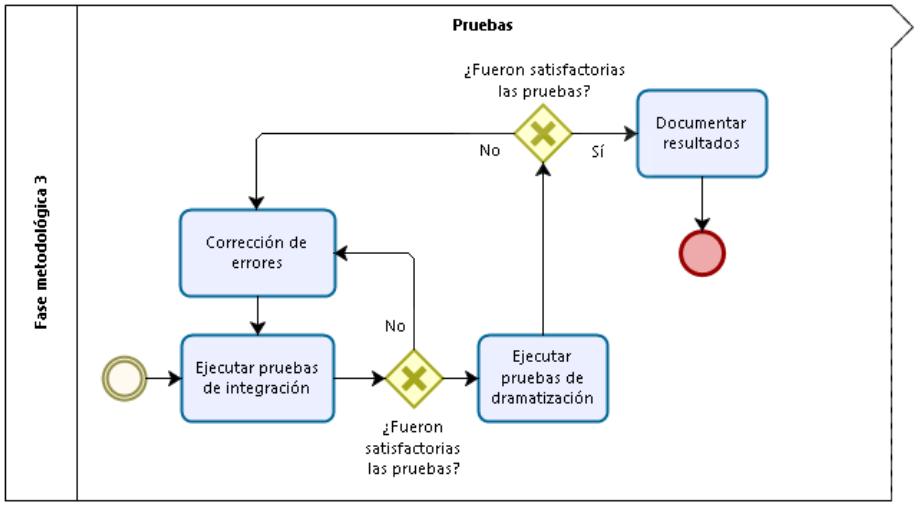

Ilustración 4. Diagrama de la fase metodológica 3.

5.4. Aplicación metodología base SCRUM

Los principales elementos de la metodología SCRUM adaptados a AIO se describen a continuación:

5.4.1. Equipo *SCRUM*

Todos los miembros del equipo están comprometidos a desarrollar el proyecto de manera constante y frecuente, En la Tabla 1 se muestra la distribución de personas en los diferentes roles que define *SCRUM* [22].

ROL	RESPONSABILIDAD	PERSONA
Product Owner	• Representar a todas las personas interesadas • Definir los objetivos del producto • Dirigir los resultados del proyecto	• Alejandra María González Correal • Enrique González Guerrero
Scrum Master	• Facilita las reuniones de *Scrum* • Quita los impedimentos del equipo • Ayuda al equipo a auto gestionarse	• David Stiven Ávila
Developers	• Implementar y desarrollar el producto	• David Stiven Ávila • Miguel Ángel Bermeo • Fabián Andrés Merchán

Tabla 1. Distribución de personas en roles Scrum.

5.4.2. *Sprints*[2]

Los objetivos planteados en la sección 2.2 fueron desarrollados en los *Sprints* descritos en la Tabla 2.

OBJETIVO	*SPRINT*
Diseñar una arquitectura física y lógica del sistema basada en el modelo BDI con un esquema modular, versátil y escalable enfocada a dramatización.	Diseño de arquitectura
Desarrollar (Diseñar, implementar y verificar) un módulo de manipulación, de recolección de datos y reproducción audiovisual, de desplazamiento y de procesamiento con características de flexibilidad y escalabilidad.	Diseño de prototipo
Diseñar y ejecutar pruebas de integración entre módulos.	Pruebas modulares
Realizar una validación operativa y de aceptación de la plataforma con los directores del proyecto	Validación del prototipo

Tabla 2. Sprints del proyecto AIO.

Luego de entender el problema definiendo los antecedentes y consultando el estado del arte con el fin de obtener información de soluciones desarrolladas relevantes para la investigación, se definió la metodología utilizada en el proyecto y se procedio a realizar la primera fase, correspondiente al levantamiento de requerimientos, la cual se detalla en el siguiente capitulo.

[2] Iteración simple llevada a cabo por los miembros del equipo [33].

II – ESPECIFICACIÓN DE REQUERIMIENTOS

1. Recolección de requerimientos

La fase de recolección de requerimientos inició con el análisis funcional de plataformas robóticas disponibles en el mercado, esto con el fin de identificar las características principales que debería tener un robot educativo. Al final de esta primera etapa se obtuvo un grafo base de características de las plataformas, junto con sus ventajas y desventajas (Ver Anexo 2). Posteriormente, se realizó una encuesta en línea a expertos de diferentes áreas, como robótica, pedagogía y psicología, solicitando su opinión acerca de las características identificadas en la etapa anterior (Ver Anexo 3). Al finalizar, 9 expertos dieron su aporte al proyecto refinando lar características obtenidas en la primera etapa.

En la segunda parte del proceso, se decidió realizar una entrevista a dos expertos en los temas tratados. Es decir, un experto en robótica y un experto en educación, esto con el fin de realizar una segunda fase de refinación a las características obtenidas. En esta oportunidad, las personas entrevistadas fueron Mónica Ilanda Brijaldo Briceño, profesora de la Facultad de Educación; y el ingeniero Julián Colorado Montaño PhD, profesor de la Facultad de Ingeniería (Ver Anexo 4).

Finalmente, teniendo un grafo de características refinado, se procedió en conjunto con los directores de grado, a definir los requerimientos relevantes para el proyecto, los cuales pasarían a la fase de priorización.

2. Priorización de requerimientos

A partir de la lista de requerimientos obtenidos en la sección anterior, se realizó un proceso de priorización con el fin de identificar y seleccionar los requerimientos más relevantes para el proyecto (ver Tabla 3).

Requerimientos	Priorización Total
El robot debe ser modular	95,00%
Comunicación oral del robot hacia el niño por medio de sonidos simples [3]	94,75%
Comunicar emociones por medio de expresiones faciales virtuales	94,75%
Comunicación táctil del niño hacia el robot	94,75%
Comunicación oral del robot hacia el niño	89,50%
El robot debe tener una historia con ramas de desarrollo	89,00%

[3] Se define un sonido simple como un sonido cuya duración es muy corta (1.5s aproximadamente) y no contiene una estructura compleja. Con solamente un tono ya es posible representar expresiones. Ejemplo: Sonidos de animales, voces sintetizadas.

El robot debe tener una historia con diferentes inicios	89,00%
El sistema debe tener un módulo manipulador	89,00%
EL robot debe tener apariencia caricaturesca	88,00%
El robot debe permitir instalar aplicaciones adicionales	87,00%
El sistema debe incluir un módulo de desplazamiento con llantas	87,00%
Comunicar emociones por medio de expresiones corporales mecánicas	85,13%
Comunicación visual del niño hacía el robot por medio de sensores visuales	85,13%
Los módulos del robot deben ser activos	84,00%
El robot debe ser desarrollado bajo una arquitectura que soporte software distribuido	83,67%
EL robot debe tener contacto visual con los usuarios	82,00%
El sistema debe incluir un módulo de miembros superiores articulados	81,00%
Comunicación oral del niño hacia el robot	80,88%
Comunicar emociones por medio vibraciones del robot hacía el niño	80,75%
Comunicación visual del robot hacia el niño por medio de luces	80,75%
El robot debe permitir cambiar el modo de interacción con el niño dependiendo del contexto (herramienta o tutor)	80,00%
El robot debe saber cuándo expresar una emoción	80,00%
Comunicación oral con emociones robot hacía el niño con sonidos complejos[4]	79,13%
EL robot debe moverse de forma autónoma	79,00%
El robot debe ser capaz de recibir realimentación de comportamientos del niño por medio de sensores visuales	77,67%
EL robot debe tener apariencia de un animal	76,95%
EL robot debe tener una personalidad tierna	73,00%
Comunicación textual del robot hacia el niño	72,00%
EL robot debe ser enfocado a Ludificación[5]	68,00%
EL robot debe tener apariencia humanoide	65,89%
Comunicación textual del niño hacia el robot	59,88%
El robo debe tener miembros superiores fijos	56,00%
EL robot debe tener una base fija	25,00%

Tabla 3. Priorización requerimientos AIO.

[4] Sonidos que a pesar que pueden expresar sentimientos y pensamientos más fácil y de una forma más clara, tienen una estructura más compleja y se componen de varios tonos. Ejemplo: Unión de palabras y letras.

[5] Conocido en inglés como *gamification*.

La priorización se realizó siguiendo la siguiente ecuación:

$$Prioridad\ Total = \frac{Prioridad\ grupo\ AIO * 0.7 + Priodidad\ usuarios * 0.3}{2}$$

En donde el grupo AIO está conformado por los directores de trabajo de grado y los estudiantes que desarrollan el proyecto, quienes dieron una valoración de acuerdo a sus conocimientos complementados con 2 entrevistas realizadas a un experto en robótica y una experta en educación. Por otro lado, la prioridad de los usuarios fue adquirida mediante una encuesta realizada a 12 personas con roles de estudiantes, profesores e ingenieros.

Con base a los requerimientos priorizados y el alcance del proyecto, se escogieron los requerimientos relevantes para el robot AIO (Ver Tabla 4).

CODIGO	REQUERIMIENTOS SIGNIFICATIVOS
REQ-01	El robot debe ser modular
REQ-02	Comunicación oral del robot hacia el niño por medio de sonidos simples
REQ-03	Comunicar emociones por medio de expresiones faciales virtuales
REQ-04	Comunicación táctil del niño hacia el robot
REQ-05	Comunicación oral del robot hacia el niño
REQ-06	El sistema debe tener un módulo manipulador
REQ-07	El robot debe tener apariencia caricaturesca
REQ-08	El robot debe permitir instalar aplicaciones adicionales
REQ-09	El sistema debe incluir un módulo de desplazamiento con llantas
REQ-10	Comunicar emociones por medio de expresiones corporales mecánicas
REQ-11	Comunicación visual del niño hacía el robot por medio de sensores visuales
REQ-12	Los módulos del robot deben ser activos
REQ-13	El robot debe ser desarrollado bajo una arquitectura que soporte software distribuido

Tabla 4. Requerimientos significativos AIO.

Sin embargo, debido al alcance del proyecto, en la primera etapa de desarrollo de AIO se escogieron los requerimientos mostrados en la Tabla 5 para la elaboración del prototipo funcional.

CODIGO	REQUERIMIENTOS SIGNIFICATIVOS
REQ-01	El robot debe ser modular
REQ-03	Comunicar emociones por medio de expresiones faciales virtuales
REQ-04	Comunicación táctil del niño hacia el robot
REQ-05	Comunicación oral del robot hacia el niño
REQ-06	El sistema debe tener un módulo manipulador
REQ-07	El robot debe tener apariencia caricaturesca
REQ-08	El robot debe permitir instalar aplicaciones adicionales
REQ-09	El sistema debe incluir un módulo de desplazamiento con llantas
REQ-10	Comunicar emociones por medio de expresiones corporales mecánicas
REQ-12	Los módulos del robot deben ser activos
REQ-13	El robot debe ser desarrollado bajo una arquitectura que soporte software distribuido

Tabla 5. Requerimientos primera etapa AIO.

Finalmente, luego de realizar una investigación y recoleccion de información acerca de plataformas y robots enfocados a la dramatización y educación, obtener los requerimientos deseables en una plataforma robotica educativa orientada a la dramatización y definer los requerimientos claves para el desarrollo de la primera etapa de AIO, se procedió a diseñar una arquitectura lógica y fisica que cumpla con todos ellos, la cual se detalla en el siguiente capítulo.

III – ARQUITECTURA DEL SISTEMA

Las obras teatrales requieren de muchos actores interactuando entre sí, coordinando sus acciones para lograr el resultado esperado. Es por esto, que la arquitectura debe dar soporte a un sistema multirobot, en el cual cada actor debe interactuar con sus pares con el fin de interpretar correctamente la obra. Para lograr esto, en primer lugar, es necesario definir una arquitectura general del sistema que permita la comunicación e interacción de los actores, y, en segundo lugar, definir una arquitectura especifica del actor que emule el comportamiento humano.

Este capítulo contiene la arquitectura de un sistema de robots basados en agentes con deseos de "actuar". Primero, se describe de manera general la solución propuesta, luego, se especifica la arquitectura general del sistema, y finalmente, se describe en detalle cada componente utilizado.

1. Descripción general de la solución

Con el fin de cumplir el requerimiento de modularidad (REQ-01) descrito en el capítulo anterior, se escogió una arquitectura orientada a servicios, la cual permite al actor contener únicamente la lógica de interpretación de la obra, mientras que la lógica de cada acción específica la contienen sistemas externos llamados ***módulos estándar***, los cuales ofrecen dicha lógica al actor por medio de servicios, es decir, cuando el actor quiera moverse, consumirá, por ejemplo, el servicio de un ***módulo estándar*** de movilidad que tenga la lógica de control de unas llantas.

Ahora bien, el proceso de interpretación de una obra consta de dos etapas fundamentales, La primera tiene como objetivo la creación de la obra, para ello se especifican los actores que harán parte de ella, la secuencia de escenas junto con las acciones que realizará cada participante y la personalidad de cada actor, entre otros. La segunda es la ejecución de la obra, en la cual los actores coordinan sus acciones con el fin de interpretar correctamente cada escena.

En la primera etapa (véase Ilustración 5) los módulos a través del protocolo de comunicación definido (el cual se explicará más adelante, véase 17Ilustración 8), le indican al actor los servicios que están en capacidad de ofrecer. Luego de esto, los actores envían la información de estos servicios a una interfaz de usuario, la cual genera el *script* teatral el cual define las acciones que hará cada robot en cada escena; y luego, con base a los servicios obtenidos, ofrece la opción de diseñar servicios de un nivel de abstracción mayor llamados *servicios compuestos*. Por ejemplo, el servicio compuesto "VOLAR", es la ejecución paralela del servicio predefinido "CAMINAR" y el servicio predefinido "MOVER BRAZOS". Posteriormente, el usuario por medio de la interfaz envía el *script* al director y la lógica de los servicios compuestos a los

actores quienes crean ***módulos personalizados*** que los ofrezcan. Finalmente, el director envía las configuraciones emocionales y sociales parametrizadas previamente en la interfaz a los actores.

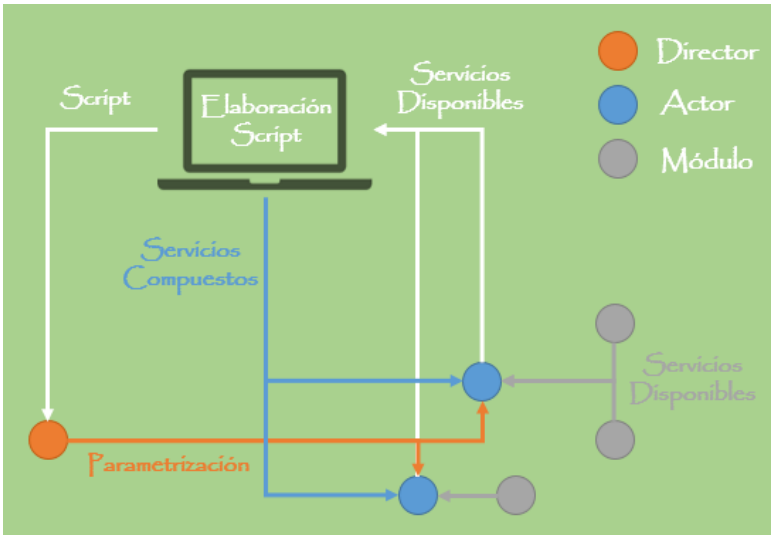

Ilustración 5. Diagrama de contexto configuración obra.

En la segunda etapa, el director, quien conoce todo el *script*, envía a sus actores las acciones correspondientes a la primera escena. Cada uno elabora un *plan*, el cual es un conjunto de tareas paralelas y/o secuencias con dependencias temporales para cumplir una meta, en este caso, interpretar la escena consumiendo servicios de sus módulos, los cuales pueden ser servicios predefinidos ofrecidos por ***módulos estándar*** (ver Ilustración 6) o servicios compuestos ofrecidos por ***módulos personalizados*** (ver Ilustración 7). Al finalizar las acciones asignadas, el actor envía un mensaje al director informándole que terminó de interpretar sus acciones. Una vez que todos los actores hayan terminado la interpretación de la escena, el director procede a asignar la siguiente escena (ver Ilustración 8).

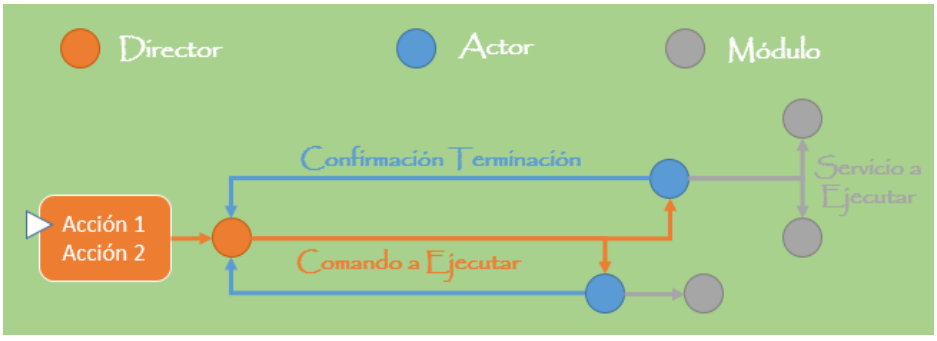

Ilustración 6. Diagrama de contexto actuación simple.

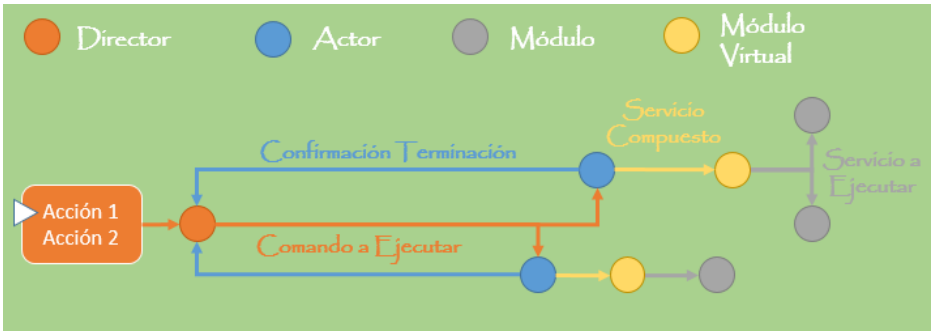

Ilustración 7. Diagrama de contexto actuación compuesta.

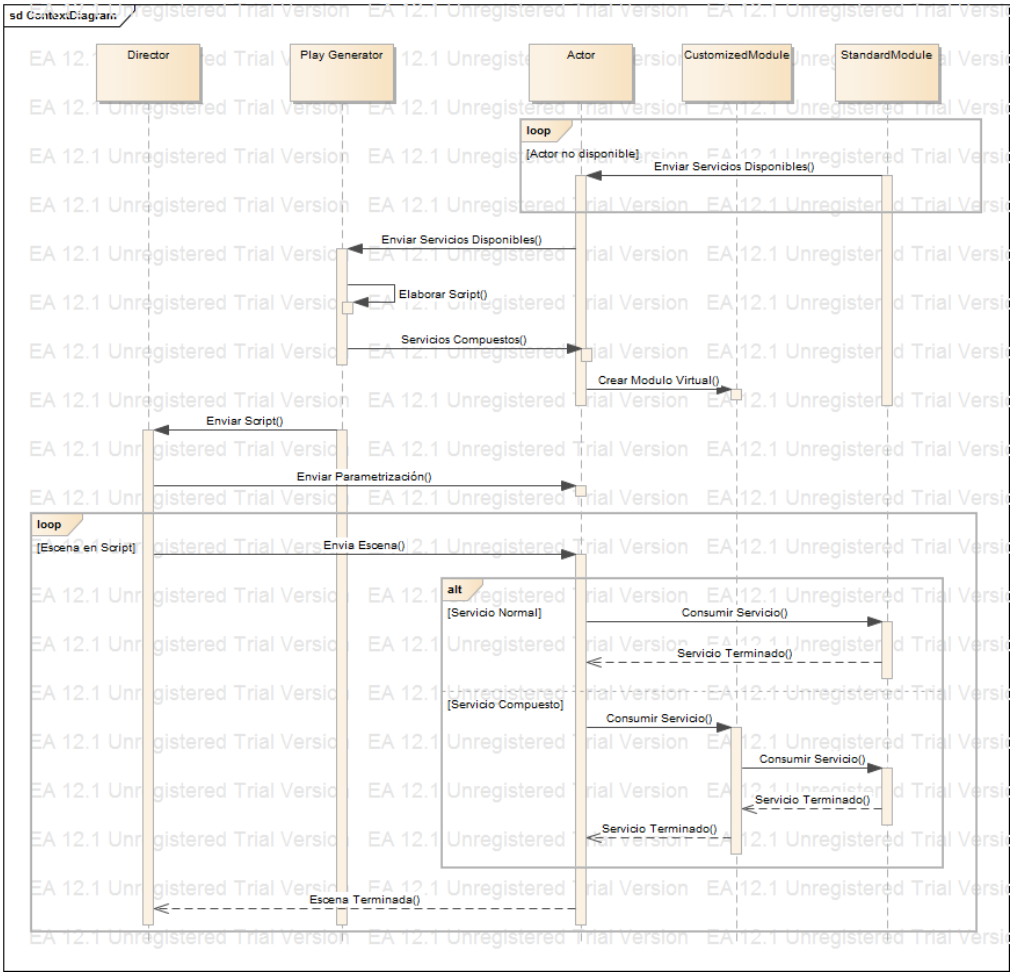

Ilustración 8. Diagrama de secuencia general**.**

2. Vista Lógica

Para el diseño de la arquitectura lógica del sistema, se tomó como base el diseño realizado en la tesis de Maestría en Ingeniería de Sistemas y Computación "RoboAct: Modelo de control autónomo y cooperativo para el Teatro Robótico", elaborada por Andrés Armando de la Peña [34], el cual se modificó con el fin de implementar algunos de los requerimientos especificados en el capítulo II, sección 2, tales como el manejo de emociones del robot (REQ-03), la modularidad del robot (REQ-01) o la posibilidad de soporte de software distribuido (REQ-13).

La arquitectura lógica general propuesta del sistema junto con la interacción por medio de servicios entre sus componentes se muestra en la Ilustración 9. En ella se pueden observar los 4 componentes principales del sistema: El actor, el director, los módulos y el generador de obras.

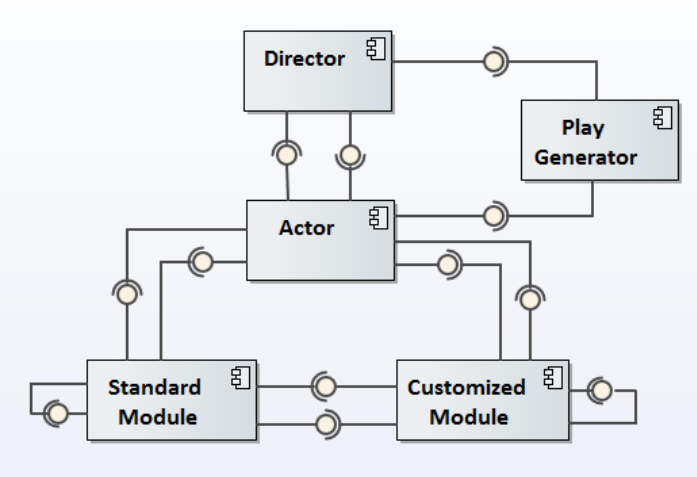

Ilustración 9. Arquitectura lógica general.

Play Generator*:** Es el agente encargado de elaborar ***scripts de dramatización por medio de un plan que coordine y sincronice los planes de todos los actores del sistema. Adicionalmente, permite la especificación de servicios compuestos por medio de la ejecución de servicios ofrecidos por los ***módulos estándar***. En la Ilustración 10 se puede observar un ejemplo ilustrativo en donde el actor deberá cantar mientras que vuela y posteriormente salta.

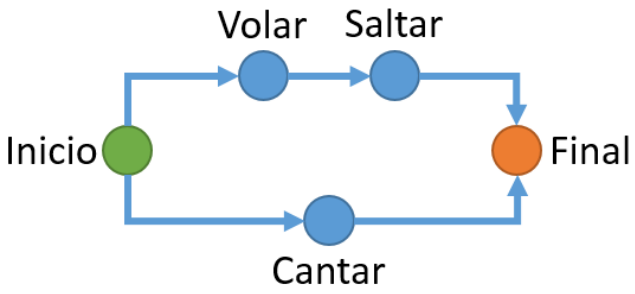

Ilustración 10. Plan ilustrativo de un guion de dramatización**.**

***Director*:** Es el agente encargado de coordinar las escenas de la obra asignando las acciones de cada escena al actor correspondiente siguiendo el protocolo de comunicación mostrado en la Ilustración 8.

***Actor*:** Es el agente encargado de interpretar un personaje en una obra siguiendo las indicaciones del director. Para cumplir cada acción asignada por el director consume servicios de los módulos disponibles en el sistema siguiendo el protocolo especificado en la sección III-3.1.

Modulo: Agente proveedor de servicios. Los módulos pueden estar implementados bajo cualquier paradigma y cualquier lenguaje de programación siempre y cuando tengan un componente de comunicación que respete el protocolo especificado en la sección III-3.1.

La arquitectura detallada de cada agente y las interacciones entre cada uno se pueden observar en la Ilustración 11, en donde cada agente está compuesto por subsistemas, los cuales agrupan componentes que compartan un objetivo o una estructura en común. Los componentes pueden ser activos, quienes contienen la lógica de procesamiento del agente, componentes pasivos que contienen el conocimiento del mismo o componentes de configuración, los cuales contienen la parametrización del papel a interpretar por el actor obtenida de archivos en formato YAML especificados en la sección IV-2.

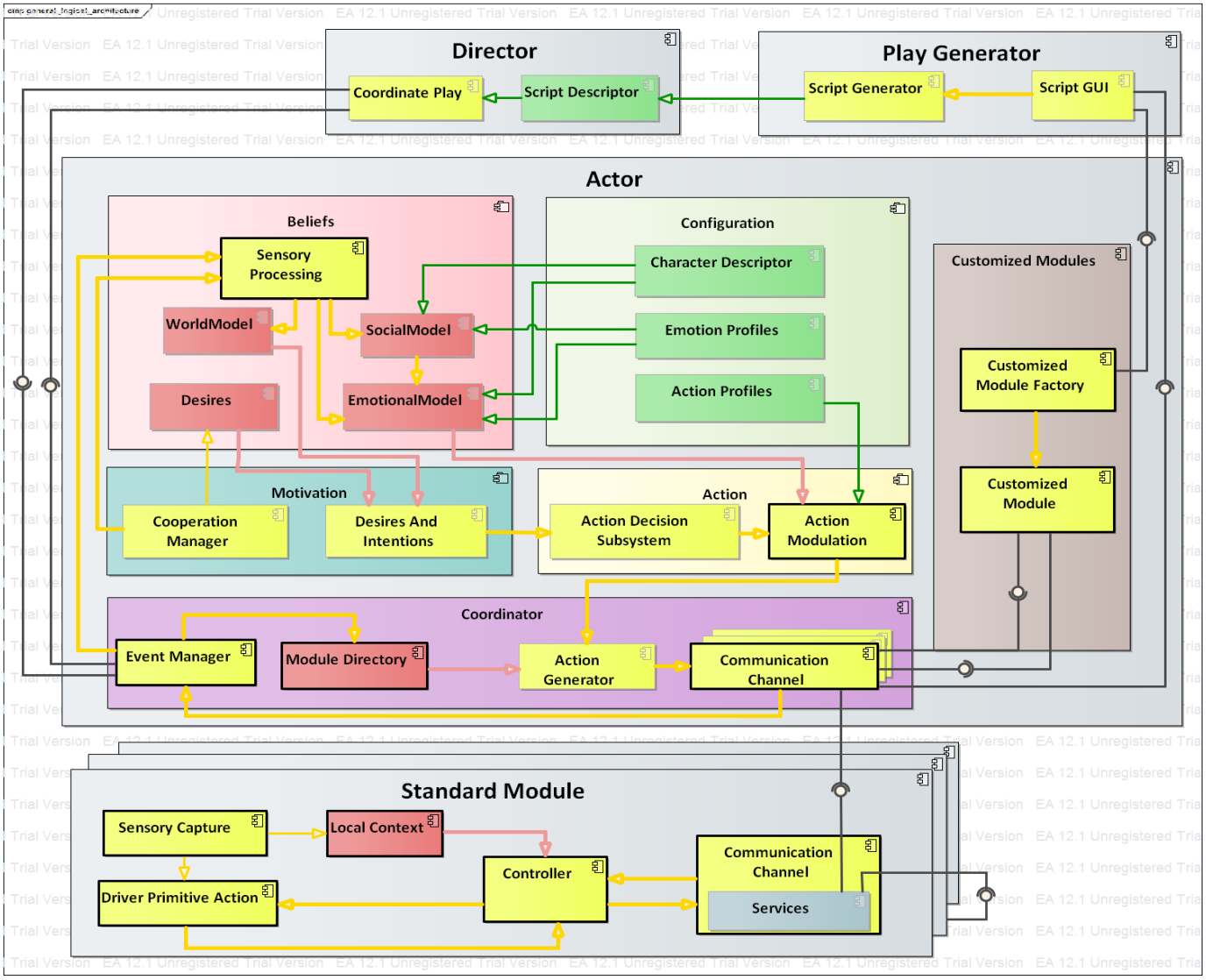

Ilustración 11. Arquitectura lógica del sistema.

Cada componente de la arquitectura del sistema es explicado a continuación:

2.1. Play Generator

Agente encargado de la elaboración del ***script de dramatización***. En él se encuentran los siguientes componentes:

> ***Script GUI*:** interfaz de usuario en la cual se elabora el guion teatral y se diseñan servicios compuestos que ofrecerán los ***módulos personalizados***.

Script generator*:** componente activo encargado de generar el archivo ***script de dramatización a partir del guion teatral elaborado en el componente ***Script*** y enviarlo al agente ***Director***. Adicionalmente está encargado de enviar al componente ***VMFactory*** del agente ***Actor*** la lógica de los servicios compuestos.

2.2. Director

Agente encargado de enviar la parametrización de la obra a cada *actor* y sincronizar la ejecución de las acciones de cada escena del ***script de dramatización***.

En él se encuentran los siguientes componentes:

Script descriptor*:** componente de configuración que contiene la información del ***script de dramatización relacionado con la secuencia de escenas, acciones que se ejecutarán en cada una y los eventos que ocurrirán en cada momento de la obra.

Coordinate Play: componente activo encargado de enviar las escenas a los actores por medio del protocolo especificado en la Ilustración 12.

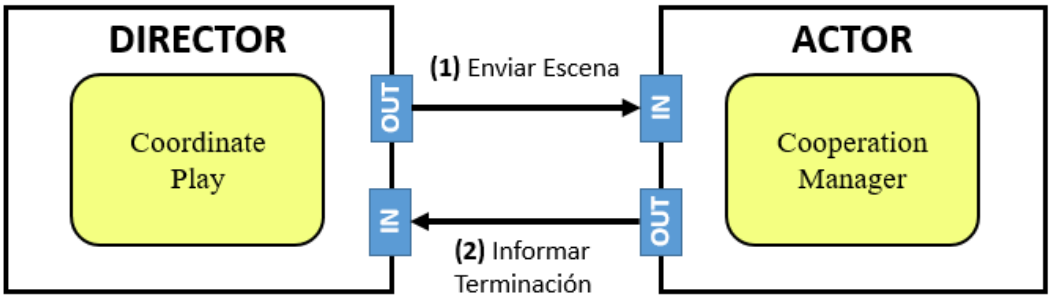

Ilustración 12. Protocolo de asignación de escenas.

2.3. Actor

Agente encargado de interpretar un papel teatral siguiendo las instrucciones de su director. El actor se compone de seis subsistemas: *beliefs*, *configuration*, *motivation*, *action*, *coordinator* y *customizedmodules*, los cuales se detallan a continuación:

2.3.1. Beliefs

Subsistema encargado de representar el conocimiento del agente ***Actor*** acerca del mundo, relaciones con otros actores y deseos, entre otros. [34], el cual está compuesto por:

Social model*:** representa la relación del agente ***Actor con los demás actores por medio de una red semántica en donde se especifica el grado de amistad con cada uno de ellos.

Desires*:** creencias del agente ***Actor correspondientes a lo que es capaz de alcanzar [34]. La especificación de las metas del actor es detallada en la sección IV-1.3.

World model*:** representa el conocimiento del agente ***Actor acerca del entorno físico en el que se encuentra. En este caso, se utilizó un grafo que modela el entorno por el cual el actor se desplaza. El detalle de la abstracción del mundo puede observarse en la sección IV-1.2.3.

Emotional model*:** describe el estado emocional del agente ***Actor y su dinámica en relación con su personalidad y el desarrollo de la obra según lo especificado en la sección IV-1.2.1.

***Sensory processing*:** componente activo encargados de procesar eventos y actualizar las creencias del actor según corresponda. Por ejemplo, si es un evento de movimiento se actualizará el modelo del mundo mientras que si es un evento de ganar la lotería se actualizará el modelo emocional.

2.3.2. Motivation

Subsistema encargado del proceso deliberativo del actor y la conciliación de sus motivaciones internas con las motivaciones de otros actores [34].

El subsistema de motivación se compone de:

Cooperation manager: contraparte del componente ***coordinatePlay*** que se encarga de la comunicación con el agente ***Director*** como lo muestra la Ilustración 12.

Desires and intentions*:** componente activo encargado de seleccionar una de las metas del ***Actor como intención. Para esto, se utiliza el diseño detallado de la teoría de BDI propuesta por Alejandra González [35], en el cual se establece que las metas compiten por prioridad según la Ilustración 13 y posteriormente para metas en la misma prioridad se compite por el valor de contribución a la meta general [34].

Ilustración 13. Niveles de prioridad de las metas.

Fuente: Elaborado por los autores.

2.3.3. Configuration

Subsistema en el que se almacenan todos aquellos datos indispensables para la configuración del actor, los cuales dan al actor el conocimiento necesario sobre como la emoción puede ser expresada, qué acciones puede realizar y cuáles son las características del robot [34].

El subsistema de configuración se compone de:

***Character descriptor*:** componente de configuración que tiene la información acerca de los rasgos del personaje como la especificación de su estado emocional, su género, su edad, los efectos de los diferentes eventos que pueden pasar en la obra y las relaciones sociales con otros personajes de la obra. El detalle de este archivo puede verse en la sección III-5.3.

***Emotion profiles*: c**omponente de configuración que tiene la información acerca de las emociones que el actor está en capacidad de representar y el valor neutro emocional siguiendo el modelo emocional especificado en la sección III-4.2.

***Action profiles*:** componente de configuración que tiene la información acerca de las acciones que el actor está en capacidad de representar, las cuales son la especificación de parámetros de los servicios que ofrecen los módulos al actor. Por ejemplo, la acción "DECIR" tiene como parámetros TEXTO, TONO y VOLUMEN, los cuales pueden ser modulados dependiendo de su estado emocional.

2.3.4. Action

Subsistema encargado de coordinar y sincronizar las tareas necesarias para cumplir la intención seleccionada por el subsistema de motivación.

Action decision subsystem*:** componente activo encargado de orquestar las acciones que permiten lograr la meta asociada a la intención activa del agente ***Actor. Para esto cada intención tiene asociado un rol encargado de ejecutar un plan modelado como un grafo de tareas, las cuales pueden desarrollarse en paralelo y/o secuencialmente (Ver Ilustración 14).

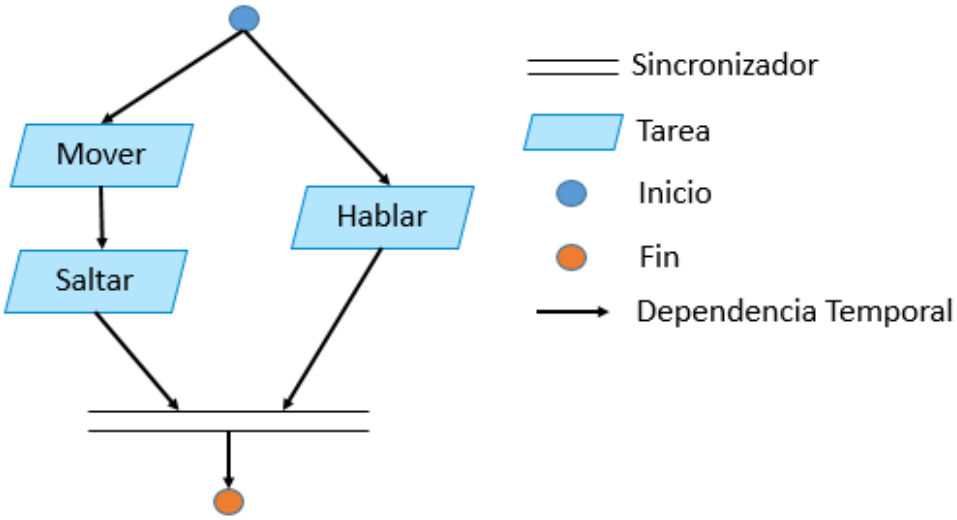

Ilustración 14. Plano de eventos.

***Action modulation*:** componente activo encargado de ajustar la parametrización dependiendo del estado emocional del actor para modular la forma en que se efectúan las acciones que se quieren interpretar. Cada módulo utiliza esta parametrización para modular las interpretaciones de la acción que vaya a realizar como lo crea conveniente. Por ejemplo, al consumir el servicio de moverse cuando el agente este triste, el módulo de movilidad modulará la velocidad de desplazamiento haciéndolo desplazarse más lento de lo solicitado.

2.3.5. Customized modules

Subsistema encargado de crear, administrar y configurar lo ***módulos personalizados*** recibidos desde el agente ***Play Generator.*** Este subsistema está compuesto por:

CustomizedModule Factory**:** componente activo encargado de recibir la lógica de los servicios compuestos diseñados en el componente ***ScriptGUI*** y crear ***módulos personalizados*** que los ofrezcan.

CustomizedModule**:** ***módulo personalizado*** que ofrece servicios compuestos diseñados por el usuario. Este componente tiene la misma arquitectura que un ***módulo estándar***, lo que causa que para el agente ***Actor*** sea transparente la comunicación con cualquiera de los dos tipos de módulo.

2.4. Coordinator

Subsistema encargado de coordinar y sincronizar los módulos disponibles para el agente ***Actor*** con el fin de interpretar las acciones solicitadas por el componente *Action Modulation.* Este subsistema se compone de*:*

Event Manager**:** componente activo encargado de analizar los mensajes recibidos por el canal de comunicación y decidir según su tipo si modificar el modelo del mundo local del agente ***Coordinator*** y/o enviarlo al agente componente ***SensoryProcessing*** para su procesamiento.

Module Directory**:** componente pasivo que representa el conocimiento del agente ***Coordinator*** acerca de los módulos disponibles en el sistema, su estado y los servicios que ofrecen.

Action Generator**:** componente activo encargado de ejecutar las acciones recibidas desde el componente ***Action Modulation*** consumiendo los servicios de los módulos disponibles en el componente ***Module Directory***. Para esto, se sigue el protocolo especificado en la sección III-3.1.

Communication Channel**:** componente activo encargado de la comunicación entre el agente ***Actor*** y los módulos del sistema. Por cada protocolo utilizado por el agente, existe un C***ommunication Channel*** asociado que contiene la lógica de comunicación específica de dicho protocolo. De esta manera, es transparente para el agente por medio de que protocolo se está comunicando con cada módulo.

Para más detalle de los componentes que existían previamente y sus interacciones, referirse a la arquitectura lógica base [34].

3. Vista de implementación

La implementación de la arquitectura propuesta en la sección anterior se realizó utilizando como base a RoboAct, un framework que permite desarrollar guiones teatrales para robots utilizando un modelo de multiagentes BDI y una manera de orquestación de acciones simples [34]; al que, con el fin de cumplir los requerimientos especificados en la sección II-2, se le realizaron ajustes y ampliaciones orientadas a una mejor estructura y funcionamiento.

La arquitectura propuesta modifica el actor proporcionado por Roboact, agrupando los componentes en dos grandes grupos e introduciendo dos agentes dedicados a manejar cada uno de ellos (Ver Ilustración 15). El primer grupo encapsula todos los componentes relacionados a los procesos deliberativos y emocionales del ***actor,*** mientras que el segundo grupo encapsula todos los componentes relacionados con los procesos de administración y sincronización de los módulos del sistema.

El agente encargado del primer grupo de componentes, llamado ***Brain,*** fue desarrollado a partir del agente actor disponible en Roboact, al que se le modificaron componentes como el ***Action Decision Subsystem*** para introducir el concepto de planes y tareas en la interpretación de una escena; y, adicionalmente, se agregaron componentes como ***Action Modulation*** y se ajustaron detalles en componentes como ***Desires and Intentions***.

El agente encargado del segundo grupo de componentes, llamado ***SOCA,*** fue desarrollado como un agente BESA encargado de administrar y coordinar los diferentes módulos del sistema. El detalle de este agente, junto con los protocolos utilizados para la comunicación con los módulos se detalla en la sección III-4.

Adicionalmente, se agregó tanto al actor como al director, un mecanismo de comunicación adaptable a diferentes protocolos y se implementó la lógica de comunicación por medio de servicios Web y llamadas a funciones de agentes BESA en un mismo contenedor.

Finalmente, se agregó un framework que contiene la estructura básica de un ***módulo estándar*** en los lenguajes de programación Java, C++ y JavaScript. Cada framework tiene implementado un canal de comunicación por medio de WebSockets y clases que representan los componentes especificados en la arquitectura propuesta en la sección anterior.

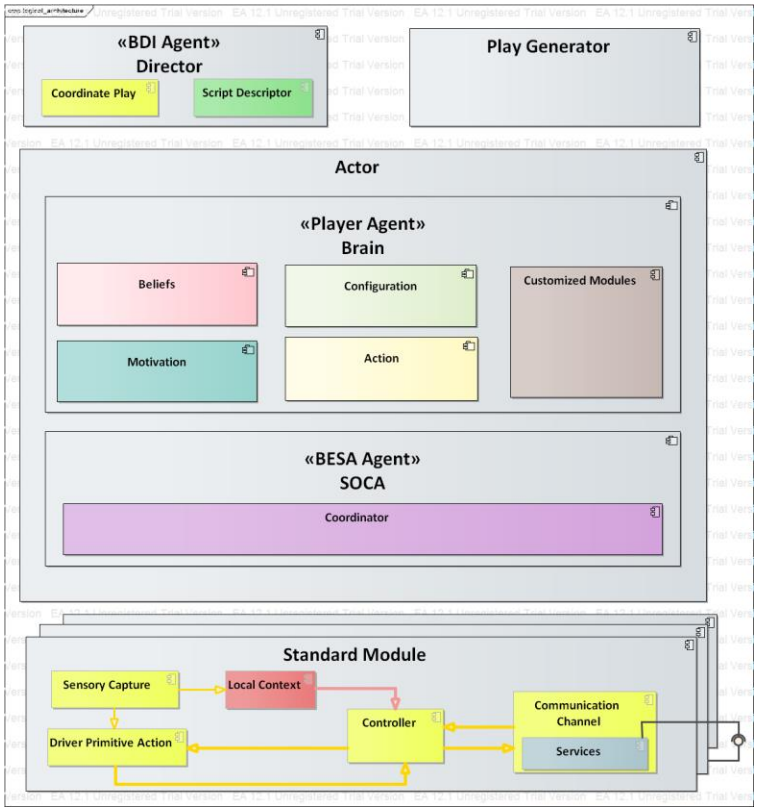

Ilustración 15. Diagrama de implementación lógica**.**

4. Services Oriented Coordinator Agent (SOCA)

Diseñado para integrar un agente con módulos externos y hacer transparente el protocolo de comunicación que utiliza cada uno, SOCA es creado como un agente BESA encargado de administrar y controlar los módulos del sistema.

4.1. Protocolo de comunicación

La interacción entre el agente ***SOCA*** y los módulos del sistema se realiza a través del componente *Communication Channel* y puede ser de asignación de trabajos o de suscripción a sensores.

Para cualquiera de los dos casos en la primera etapa el módulo se registra al agente ***SOCA*** por medio del protocolo mostrado en la Ilustración 16, en donde un módulo realiza la petición de registro al agente ***SOCA***, quien genera un ID y registra el módulo y sus servicios en su directorio, posteriormente, envía un mensaje de confirmación al módulo, quien ajusta su ID al que le fue enviado.

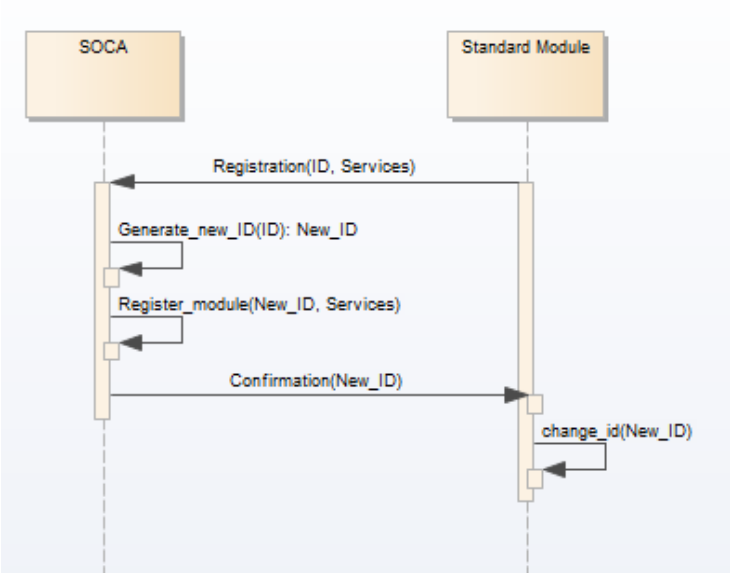

Ilustración 16. Protocolo de registro de un módulo.

Ahora bien, cuando se requiere asignar trabajos, el agente ***SOCA*** sigue el protocolo mostrado en la Ilustración 17, para esto, primero revisa en el componente ***Module Directory*** los módulos capacitados para cumplir los trabajos solicitados. Por cada uno, pregunta a los módulos si alguno puede aceptar el trabajo y lo asigna a uno de los que respondieron afirmativamente. Si la asignación fue exitosa, el agente ***SOCA*** envía un mensaje en *broadcast* a todos los módulos para que den comienzo a la actividad que se les fue asignada. En caso de que no haya módulos capacitados o de que todos los módulos capacitados respondan negativamente, el agente ***SOCA*** informa a los solicitantes del servicio que se debe generar una necesidad.

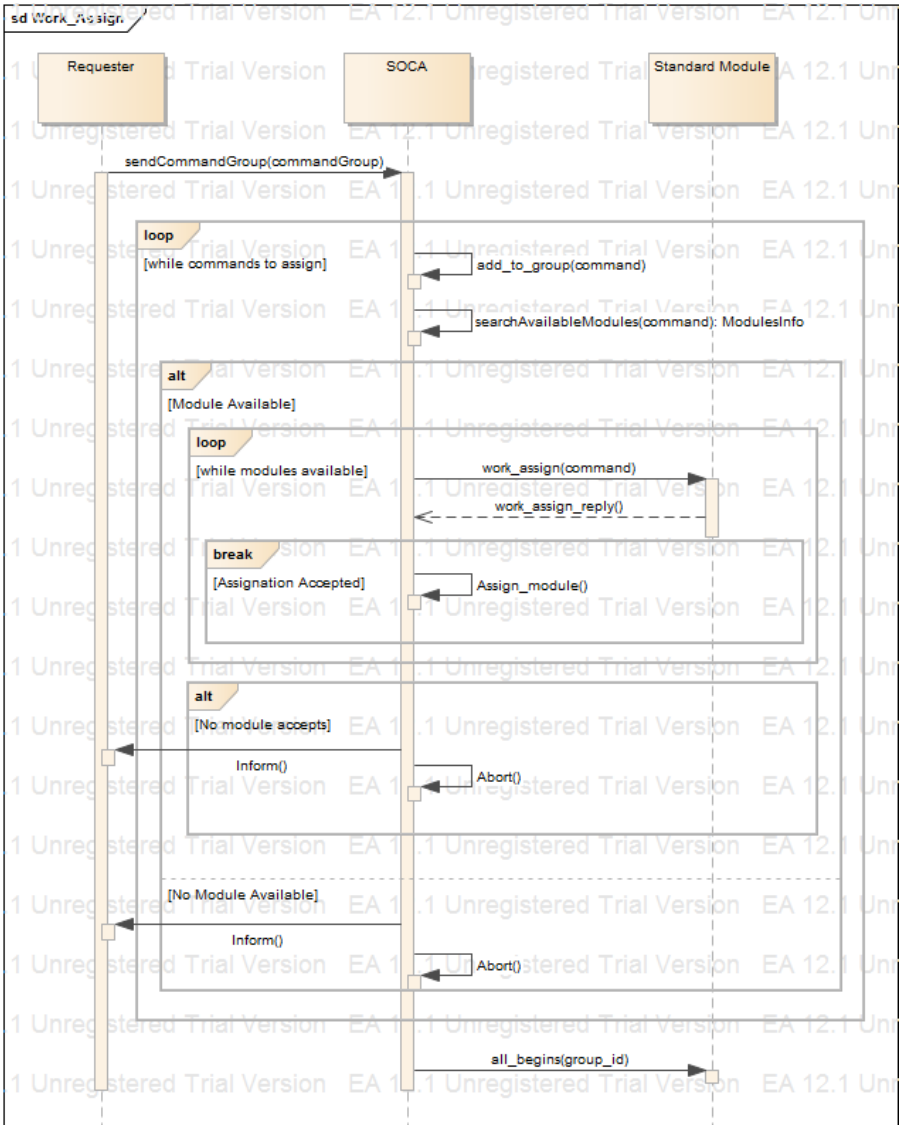

Ilustración 17. Asignación de trabajo a módulos**.**

Durante la espera por la terminación de un comando que previamente fue asignado a un módulo, el ***SOCA*** sigue el protocolo mostrado en la Ilustración 18, en donde periódicamente pregunta el estado de los módulos que tienen asignado algún trabajo. En caso de que alguno no conteste o indique que presenta errores envía un mensaje de error a los solicitantes del servicio**.**

Finalmente, cuando un módulo termina su trabajo, le envía un mensaje de terminación con el resultado al agente ***SOCA***, quien actualiza su directorio de módulos e informa a los solicitantes del servicio acerca del evento (ver Ilustración 18).

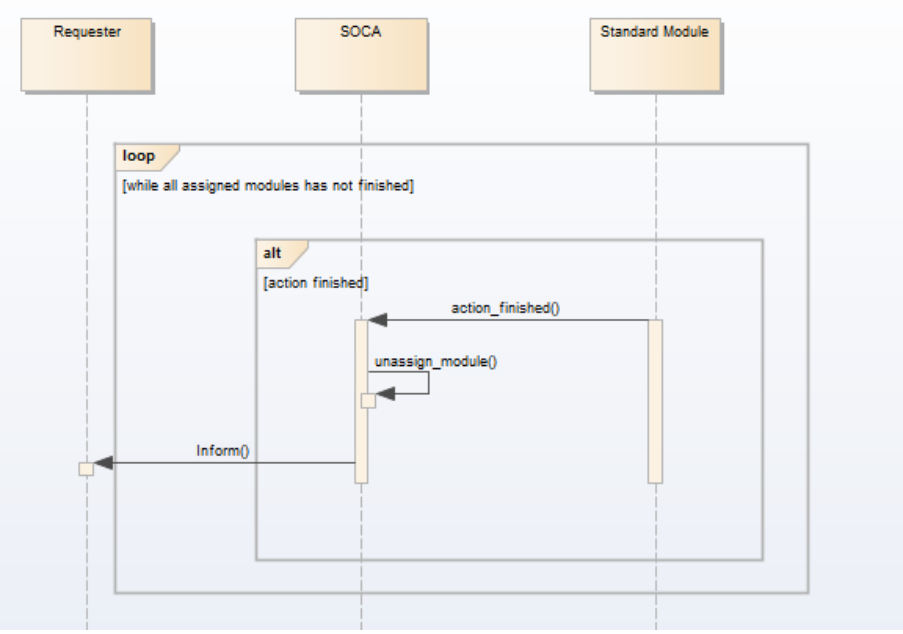

Ilustración 18. Finalización de comandos.

Por otro lado, cuando un sistema se quiere subscribir a los servicios de sensado de un módulo, el sistema y el módulo siguen el protocolo mostrado en la Ilustración 20, en donde el sistema manda un mensaje de petición de suscripción al agente ***SOCA,*** quien agrega al sistema a su lista de suscritos y le envía una confirmación del proceso. Luego de esto, cuando un módulo envíe un mensaje referente a un servicio de los sensores, el agente ***SOCA*** enviara el mensaje a todos los sistemas que estén suscritos al servicio, como lo muestra la Ilustración 19.

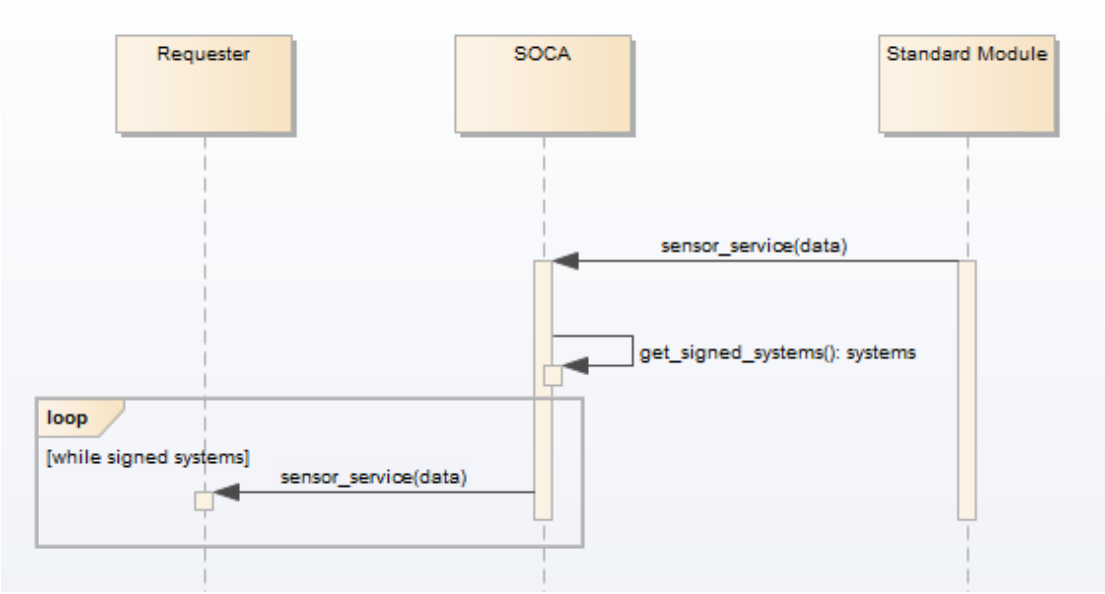

Ilustración 19. Protocolo de envío de datos sensados.

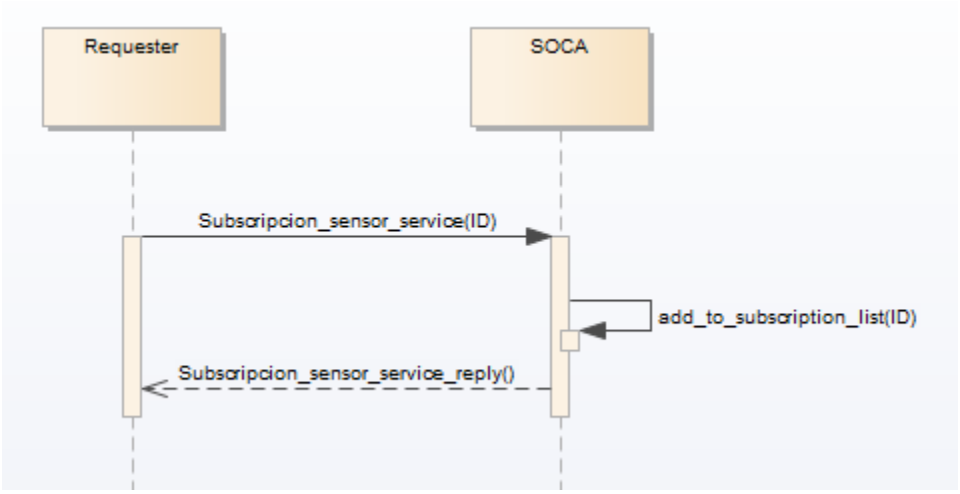

Ilustración 20. Protocolo de subscripción a sensores.

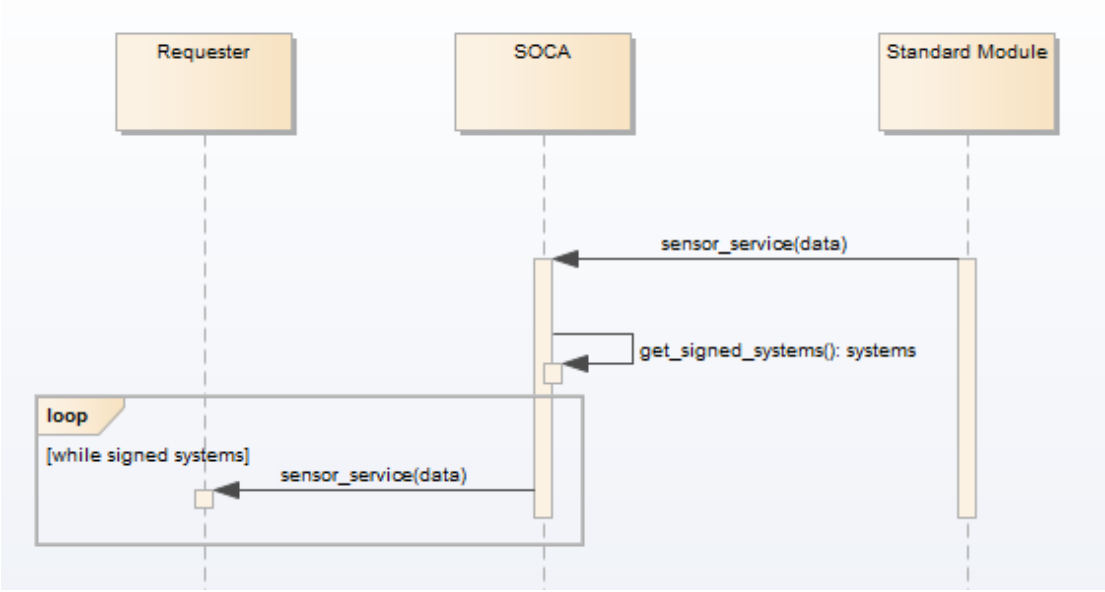

Ilustración 21. Protocolo de envío de eventos de sensores.

4.2. Tolerancia a fallos

Cada vez que ***SOCA*** envía un mensaje a un módulo sigue el protocolo mostrado en la Ilustración 22, en donde cada cierto tiempo el agente ***SOCA*** pregunta al módulo su estado. Si el módulo no responde, el agente ***SOCA*** envía una segunda petición del estado. Si el módulo sigue sin responder, lo remueve de los módulos disponibles e informa a los solicitantes del servicio de la desconexión.

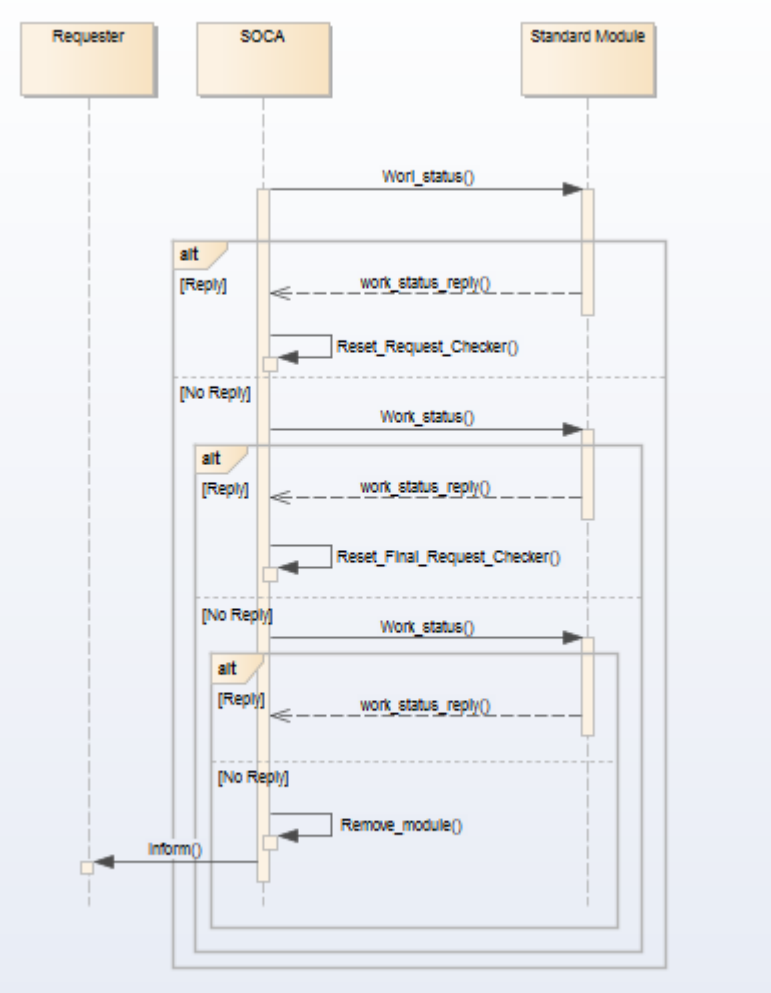

Ilustración 22. Protocolo de tolerancia a fallos.

Finalmente, luego de definir la arquitectura lógica, detallar los elementos que la componen, describir las relaciones y protocolos utilizados; y crear un framework de coordinación de módulos que facilite la elaboración de sistemas módulares, se procedió a desarrollar un prototipo modular que satisfaga los requerimientos expuestos en la sección II-2.

IV - ROBOT AIO

A continuación, se describe a AIO, un robot que utiliza la arquitectura expuesta en la sección anterior para educar niños por medio de obras teatrales. El capítulo se estructura de la siguiente forma: En primer lugar, se detallan los recursos de configuración y parametrización necesarios para la interpretación del papel, posteriormente, se describe el modelo de deliberación del agente, y finalmente, se describe la disposición física del robot y el detalle de cada módulo desarrollado.

1. Archivos de configuración

Los archivos utilizados en una obra junto con la descripción de su contenido y objetivo es mostrada en la Tabla 6.

Nombre Archivo	Descripción
Script de dramatización	Guion teatral modelado como un árbol, en donde cada nivel es un fragmento de la obra que especifica en cada nodo lo que cada actor debe hacer y en cada arco la dependencia temporal que existe con los demás fragmentos.
Diccionario semántico	Describe las posibles relaciones de los actores, los tipos de eventos y los niveles de afinidad con los objetos en la obra.
Archivo de parametrización del personaje	Define las relaciones con otros actores, la importancia que tiene cada objeto y cada evento para el actor utilizando los elementos especificados en el diccionario semántico. Adicionalmente, parametriza el modelo emocional del actor, es decir, el neutro emocional, el factor de olvido y el estado emocional inicial, entre otros. La descripción detallada del modelo emocional se describe en la sección IV-1.2.1.
Archivo de definición del mapa del mundo	Modela el escenario de la obra con un grafo, en el que cada nodo representa una ubicación (ej. Casa, hospital, etc.), y cada arco los caminos entre ellas. La descripción detallada del modelo del mundo se observa en la sección IV-1.2.3.

Tabla 6. Descripción archivos de configuración.

La descripción detallada de cada archivo, junto con su estructura y ejemplos puede verse en el Anexo 1.

2. Modelo de agente BDI

El modelo del agente AIO está categorizado en características generales, manejador de creencias y **g**estor de metas.

2.1. Características generales

Objetivo General: Representación de un ***script de dramatización***.

Nivel de compromiso: Calculado a partir de una función dependiente del estado emocional del agente y la madurez del mismo.

Para el presente trabajo de grado, el nivel de compromiso se fija en su nivel máximo; esto significa que el agente estará totalmente comprometido con el objetivo general.

Ciclo de atención: En el momento que el agente no tenga una meta activa, ejecutará los servicios de ATTENTION_CYCLE de los módulos disponibles.

2.2. Manejador de creencias

Las creencias del agente son categorizadas en **(a)** Emocionales, **(b)** Sociodemográficas y **(c)** Modelo del mundo:

2.2.1. Emocionales

El estado emocional del agente es modelado por la posición de una variable en un eje coordenado que describe en su extremo positivo (1) el nivel máximo de felicidad, en la mitad (0) el neutro emocional y en su extremo negativo (-1) el nivel máximo de tristeza). Adicionalmente, en el ***script de dramatización*** se configura el estado neutro emocional propio del agente, es decir, un agente depresivo tendrá un neutro emocional más cercano al nivel máximo de tristeza, mientras que uno alegre lo tendrá más cercano al nivel máximo de felicidad (Ilustración 23).

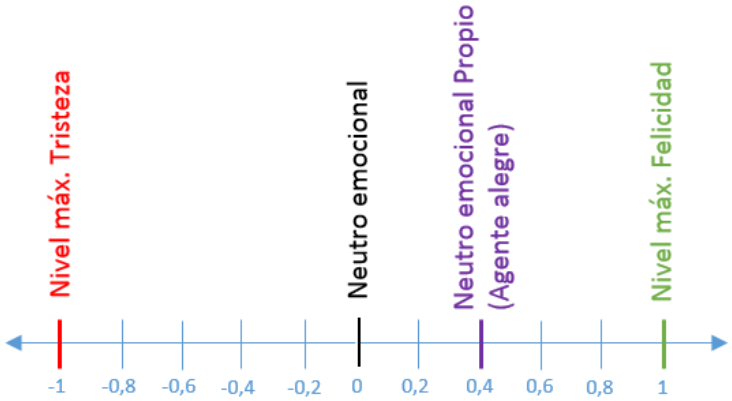

Ilustración 23. Ilustración eje emocional.

La variación del estado emocional se da por eventos, los cuales aumentan o disminuyen el valor de la variable en un valor *K* a partir de sentencias cuya semántica está preestablecida en el ***diccionario semántico***.

Cuando se produce un evento, el estado emocional empezará a tender a su valor neutral a una razón de cambio determinada por una función lineal, cuya constante es configurada en el ***archivo de parametrización del actor***. Es decir, un agente con un neutro emocional propio de 0.5 que experimenta el evento "MORIR MAMA", sufrirá una variación negativa en su estado emocional de 0.75, estableciendo su estado emocional en -0.25. Luego de esto, el valor ira aumentando linealmente a través del tiempo hasta llegar al valor neutral emocional propio del agente (0.5).

2.2.2. Sociodemográficas

El agente tiene características parametrizables en el ***archivo de parametrización del actor***, de género y edad, las cuales modifican su manera de actuar y pensar. Por ejemplo, un agente hombre tendrá un tono de voz distinto al de una mujer, y un niño hombre tendrá un tono de voz distinto al de un hombre adulto.

Para el presente trabajo de grado se definen dos géneros: hombre y mujer, los cuales modifican el tono de los diálogos, y una única edad con nivel máximo de compromiso hacia el objetivo general.

2.2.3. Modelo del mundo

El agente está ubicado en un escenario con rieles de desplazamiento implementados por medio de cintas negras, el cual es modelado por medio de un ***archivo de definición del mapa del mundo***, por medio de un grafo, en donde cada nodo representa una bifurcación y una ubicación. Es decir, un nodo será **COLEGIO** y otro **CARCEL** (Ilustración 24).

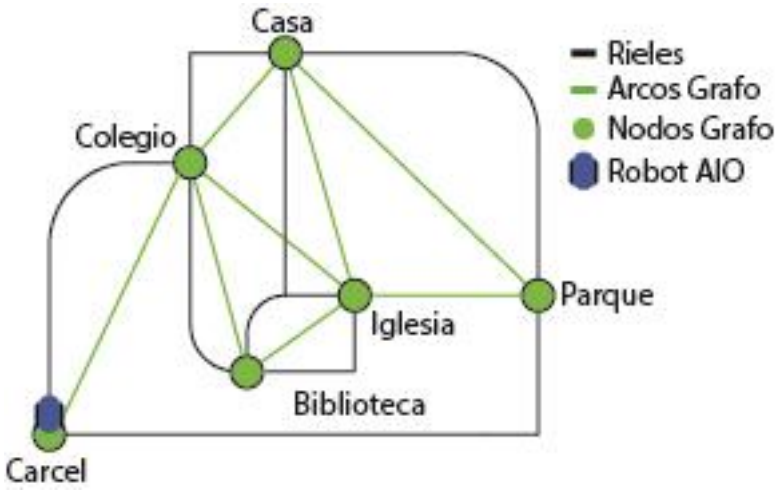

Ilustración 24. Modelo del mundo robot AIO.

Adicionalmente, dentro del escenario puede haber dos tipos de elementos:

- **Escenografía:** Elementos estáticos utilizados para ambientar y representar la obra.
- **Utilería:** Elementos estáticos que el agente manipula en pro de la interpretación de la obra.

2.3. Gestor de metas

Las metas están categorizadas en **(a)** Supervivencia, **(b)** Obligación, **(c)** Oportunidad, **(d)** Requerimiento y **(e)** Necesidad.

2.3.1. Supervivencia

Son aquellos objetivos indispensables para el agente relacionados con su cuidado y protección personal.

El agente actor tiene las siguientes metas de supervivencia:

- **Cuidar porcentaje de batería**: Cuando el porcentaje de la batería es menor al 15% de su capacidad máxima, el agente interrumpe todos sus procesos y alerta al usuario acerca de su bajo nivel energético.

2.3.2. Obligación

Son aquellos objetivos que surgen del contexto del agente e interfieren directamente en el proceso de toma de decisiones.

El agente actor tiene las siguientes metas de obligación:

- **Terminar actuación en un tiempo límite:** Cuando se cumple el tiempo límite establecido en el ***script de dramatización***, el agente interrumpe todos sus procesos, pide disculpas al usuario e indica que se continuará en la siguiente sesión.
- **Interrumpir *script*:** Cuando el usuario envíe una petición de terminación, el agente detiene su actuación y alerta al usuario que le han indicado parar.
- **Reglas éticas:** Las acciones que interprete el agente actor no pueden ir en contra a las reglas éticas descritas en el ***script de dramatización***. Es decir, si el agente no debe ir a la cárcel, y la acción a interpretar es ir a la cárcel, el agente debe rechazar la acción e informar al usuario que no la puede cumplir.
- **Expresar felicidad:** Cuando el estado emocional del agente supere el 80% de felicidad, el agente debe pausar sus acciones, expresar una emoción de alegría y continuar con sus acciones.
  NOTA: Para el presente trabajo de grado, la emoción es expresada por medio de un grito.

2.3.3. Oportunidad

Eventos que se presentan mientras el agente está activo y benefician la búsqueda por alcanzar los objetivos.

El agente actor tiene las siguientes metas de oportunidad:

- **Interpretar acción de *script***: Cuando el agente recibe una orden de acción, procede a pedir al agente ***SOCA*** que la asigne a un módulo con la capacidad de interpretarla.
- **Responder participación de usuario:** Cuando el agente recibe una petición de participación por parte del usuario, procede a grabar 30 segundos de audio y luego reproducirlos.
  NOTA: Para el presente trabajo de grado, la petición se realiza tocando la pantalla táctil.

2.3.4. Requerimiento

Son aquellos objetivos relacionados con las capacidades del agente, los cuales surgen cuando se requiere la combinación de una habilidad y un recurso para poder cumplir el objetivo buscado.

El agente actor no tiene metas de requerimiento asignadas.

2.3.5. Facilitadores

Son aquellos objetivos relacionados con un proceso de meta cognición y pensamiento autocrítico, en el cual el agente comprende que no puede cumplir con el objetivo destinado y debe reestructurar su mecanismo de decisión.

El agente actor tiene las siguientes metas de necesidad:

- **Pedir conexión de módulo**: Cuando el agente quiere ejecutar una acción, pero no cuenta con ningún módulo habilitado con la capacidad para realizarla o si el agente debe ejecutar la acción por un módulo especifico y no se encuentra habilitado, se detiene y alerta al usuario acerca de la necesidad del módulo.

3. Prototipo

Esta sección se centra en la descripción del prototipo funcional que implementa la arquitectura propuesta en la sección III-3. En cada subsección del capítulo se detalla el hardware, lenguajes de programación, algoritmos y herramientas utilizados en la implementación de cada uno de los módulos propuestos. El código fuente de los proyectos realizados para el prototipo se encuentra en el Anexo 9 y el manual de instalación y de uso de cada uno de los módulos está en el Anexo 7.

3.1. Vista de despliegue

Con base en los requerimientos establecidos en la sección II-2, se diseñó una plataforma robótica compuesta por un módulo de procesamiento, un módulo de manipulación, un módulo de movilidad y un módulo de reproducción. La vista de despliegue que modela el prototipo puede verse en la Ilustración 25.

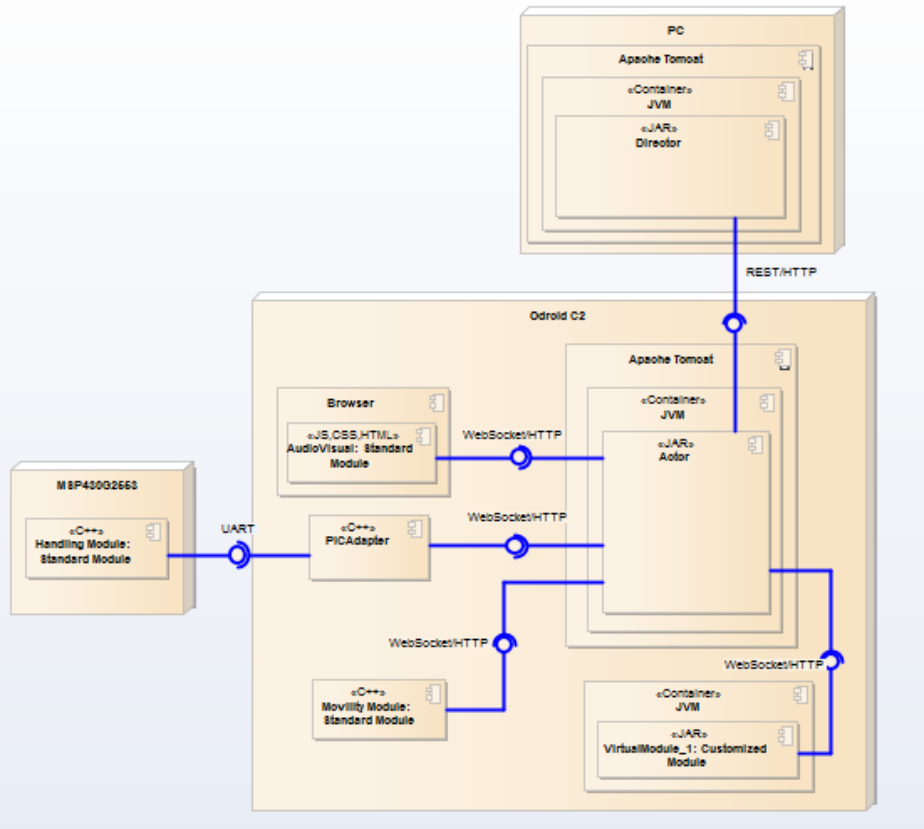

Ilustración 25. Vista de despliegue prototipo.

Donde la conexión entre el módulo de manipulación y el módulo de procesamiento, se realizará a través de un microcontrolador (MSP430G2553) por medio del protocolo *UART*.

La conexión entre el módulo audiovisual y el módulo de movilidad con el módulo de procesamiento se realiza vía WebSockets usando SocketIO, una librería disponible en JAVA, C++, y JavaScript para el desarrollo de aplicaciones que requieran comunicación en tiempo real, lo cual permite al actor obtener realimentación del estado del módulo de forma inmediata.

3.2. Módulo de procesamiento

El módulo de procesamiento fue desarrollado implementando la arquitectura definida en la sección III-3. Este módulo se encarga de ejecutar al agente ***Brain***, encargado de tomar decisiones que lleven a cumplir la meta general. Para esto, les informa a los módulos las acciones que deben realizar por medio del *framework* ***SOCA***, de esta manera, la responsabilidad del manejo de recursos físicos es delegada a cada módulo.

Este módulo fue desplegado en la Tarjeta de Desarrollo Odroid-C2 (que llamaremos TDO-C2), escogida debido a que cuenta con un procesador de cuatro núcleos en una arquitectura ARM de 64 bits a 1.5GHz y memoria RAM de 2Gb, lo cual permite ejecutar un sistema operativo Ubuntu mate 16.04. Estas características de procesamiento permiten la ejecución de la arquitectura y permite el uso de diferentes lenguajes, *frameworks* y protocolos, como los mencionados en la sección IV-3.1.

Adicionalmente, TDO-C2 cuenta con un puerto HDMI, necesario para conectar la pantalla requerida por el módulo de reproducción audiovisual, dos PWM y un ADC de 10 bits, necesarios para el control del módulo de movilidad y una interfaz UART, necesaria para la comunicación con el módulo de manipulación.

A continuación, se muestra el diagrama en bloques de la tarjeta donde se detallan los puertos y servicios disponibles [36].

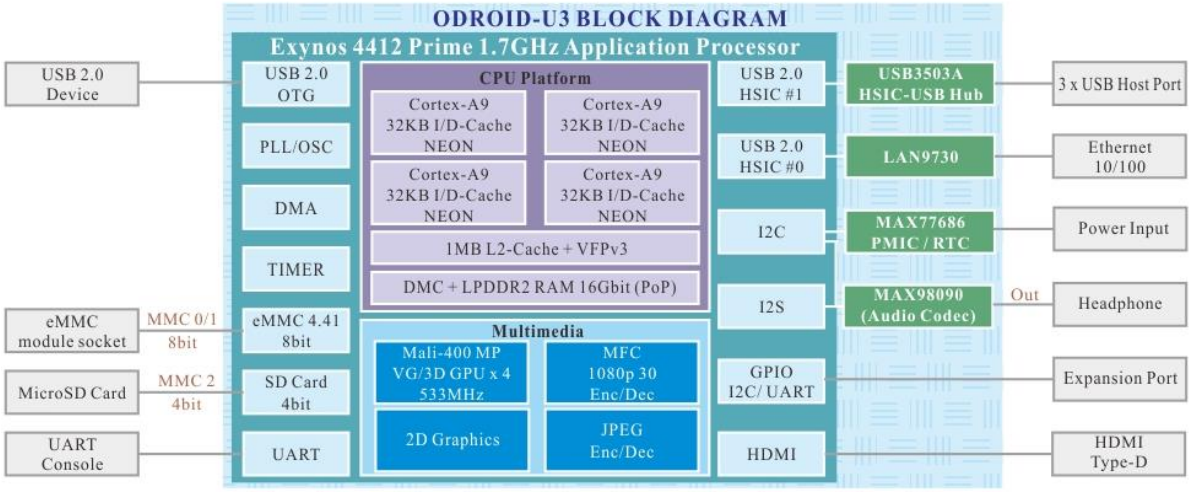

Ilustración 26. Diagrama en bloques de la tarjeta de desarrollo ODROID-C2

Fuente: Página web de HardKernel[6]

TDO-C2 se alimenta con 5 V y máximo 2 A; y consume entre 2 W y 5W. La energía es proporcionada por una batería portable (*powerbank*), específicamente una Xiomi[7] de 10000mAh, escogida ya que es capaz de entregar máximo 2.1 A, 5 V constantes y por tener dimensiones bajas (9.1 cm de largo por 6.04 cm de ancho y 2.2 cm de grosor).

[6] http://www.hardkernel.com/main/main.php

[7] http://www.mi.com/my/pb10000

3.3. Módulo de Movilidad

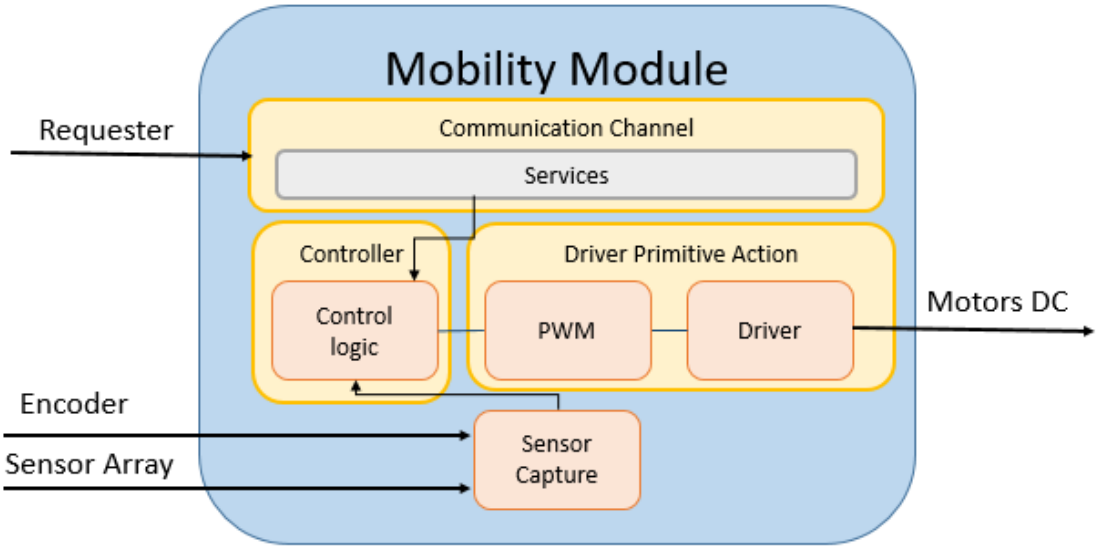

Ilustración 27. Diagrama de bloques del módulo de desplazamiento.

Modulo encargado de proveer al actor la capacidad de desplazarse por el escenario. Para eso, se implementó la arquitectura mostrada en la Ilustración 27, la cual permite proveer los siguientes servicios:

- **Calibración de sensores de línea**: este servicio solo se ejecuta una vez y es utilizado para ajustar los tonos de blanco y negro con el fin de diferenciarlos apropiadamente.
- **Mover**: servicio de seguir línea hasta que se encuentre una bifurcación. Implementa dos controles en cascada, uno para el control de velocidad de cada motor y otro para controlar el ángulo de giro del robot, los cuales se detallan en la sección IV-3.3.3.

Este módulo se implementó utilizando un chasis con llantas tipo oruga fabricado por la empresa Pololu (véase Ilustración 28), un arreglo de sensores de reflexión (Sección IV-4.1) compatibles con el chasis para realizar el seguidor de línea, y dos motores DC (Sección IV-4.2) cuya capacidad de carga es capaz de soportar máximo alrededor de 10kg [37], [38].

Ilustración 28. Chasis con llantas tipo oruga y sensores de reflexión.

Fuente: Página web de Pololu[8]

De manera general, toda la arquitectura fue implementada bajo el lenguaje C++11, usando la librería *Wiring Pi* para controlar los periféricos de TDO-C2. El detalle de cada componente junto con la lógica asociada a cada uno se detalla en las siguientes subsecciones.

3.3.1. Sensores de línea

Se utilizó la regleta "*QTR-8A"* de 6 opto acopladores (Véase Ilustración 29) fabricado por la empresa POLOLU como herramienta detectora de línea.

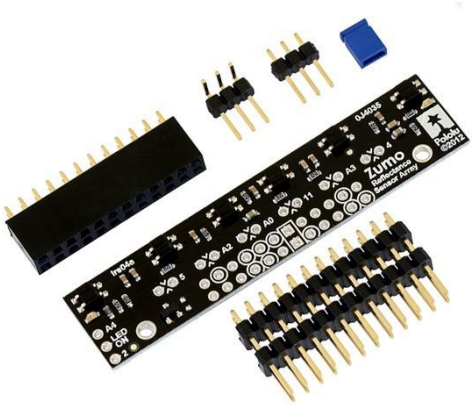

Ilustración 29. Arreglo de sensores.

Fuente: Página web de Pololu.[9]

[8] https://www.pololu.com/

[9] https://www.pololu.com/product/1419

Para convertir los valores analógicos de los sensores en valores lógicos, cada vez que se quiere saber el valor de un sensor, se emite una señal y se cuenta el tiempo que demora en ser captada por el sensor receptor. De esta manera, al cambiar el color de la superficie que refleja el rayo, se cambia también la longitud de onda y en consecuencia su velocidad, por lo que finalmente, una superficie negra podrá ser diferenciada de una blanca.

3.3.2. Motorreductores

Se escogieron 2 motores capaces de soportar máximo 10Kg (Peso aproximado de la plataforma) y que concuerdan con el tamaño y diseño requerido por el chasis. Los motores elegidos con referencia "*Micro Metal Gearmotor HPBC*" cuentan con una proporción plato/piñón de 298:1, torque de $22oz \cdot in$, velocidad de rotación de 400 rpm y son capaces de soportar máximo 10 Kg cada uno [37].

Ilustración 30. Motor motorreductor de 6V

Fuente: Página web de Pololu[10]

3.3.3. Lógica de control

El primer paso en la lógica de control del seguidor de línea fue hallar una variable relacionada a la correcta alineación del robot. Para esto, se utilizó la ecuación 1, la cual describe la orientación del móvil como un valor entre 1 y 5000, en donde 0 indica una máxima inclinación hacia la derecha, 2500 una orientación paralélela a la línea y 5000 una máxima inclinación hacia la izquierda (Véase Ilustración 31).

$$K = \frac{1}{\sum_{i=0}^{5} sensor_i} \sum_{i=0}^{5} 1000 * i * sensor_i \quad (1)$$

[10] https://www.pololu.com/category/60/micro-metal-gearmotors

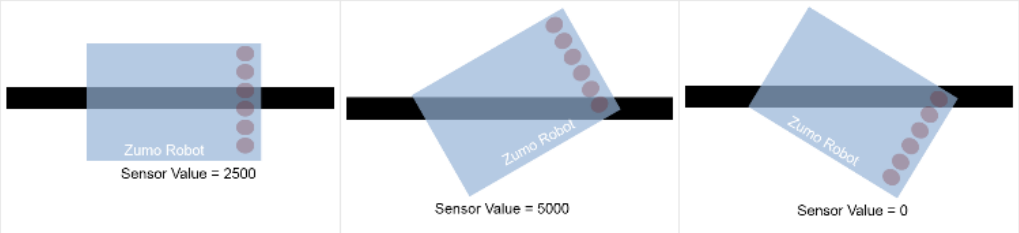

Ilustración 31. Modelo de orientación de la plataforma AIO.

Fuente: Line Following Zumo Robot Using Simulink[11].

Este valor de inclinación es regulado por medio de un controlador PID (Proportional–Integral–Derivative) con referencia de 2500 como se muestra en la Ilustración 32, cuya salida de control es la referencia de un controlador proporcional de velocidad, el cual adquiere realimentación de la velocidad de los motores a través de los *encoders*[12] provistos por el fabricante.

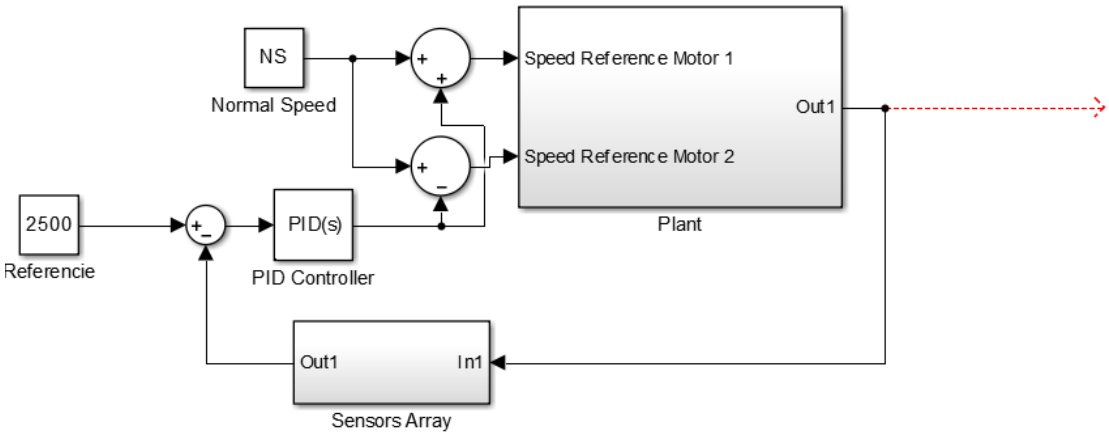

Ilustración 32. Sistema de control seguidor de linea.

En la Ilustración 33 se muestra el diagrama de bloques del control de velocidad de los motores motorreductores

[11] https://learn.adafruit.com/line-following-zumo-robot-programmed-with-simulink/acquire-sensor-input
[12] https://www.pololu.com/product/3081

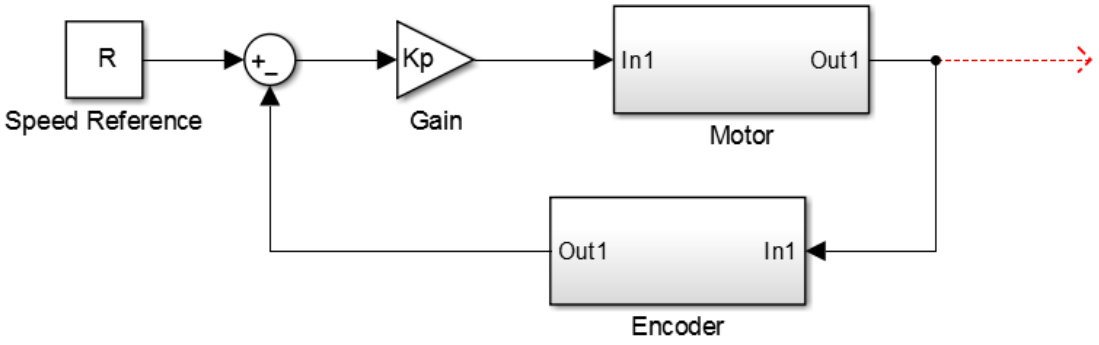

Ilustración 33 Diagrama de bloques control de velocidad de los motores.

Este control utiliza una cantidad de pulsos como referencia, que es obtenida al contar los cambios de pendiente en la señal recibida por el *encoder* en una ventana de tiempo de 10 ms. Posteriormente, dicha cantidad se transforma en un porcentaje de ciclo útil, utilizado para modular la señal de alimentación de los motores por medio de un PWM, para finalmente lograr el movimiento del motor a través de una etapa de potencia manejada por el driver DVR8833.

En el Anexo 6 se encuentra un documento que detalla los modelo, cálculos y pruebas realizadas a los controladores. Durante este proceso se tuvo encuenta para el control de velocidad un tiempo de establecimiento menor al tiempo de muestro al seguidor de linea de 300 ms y en el control de linea una velocidad media lo suficientemente rapida que no descontriole el robot.

3.3.4. PWM

Se gradúa la velocidad de los motores con el ciclo útil de un PWM. De manera experimental, se observó que una frecuencia de 20 KHz es apropiada para el sistema ya que esta fuera del rango de frecuencias audibles, por lo que se escogió el PWM ofrecido por la tarjeta de desarrollo, el cual tiene una frecuencia máxima de conmutación de 1 MHz.

3.3.5. Driver

El manejo de potencia para los motores se implementó utilizando el driver DRV8833 fabricado por Texas Instruments, el cual usa dos circuito Puente H para entregar a cada uno de los motores hasta 1.5, la corriente máxima requerida por los motores es alrededor de 1A, lo cual ratifica a los *encoders* escogidos como adecuados para la aplicación. Adjunto en los anexos se encuentra la hoja de especificaciones del componente.

3.4. Módulo Audiovisual

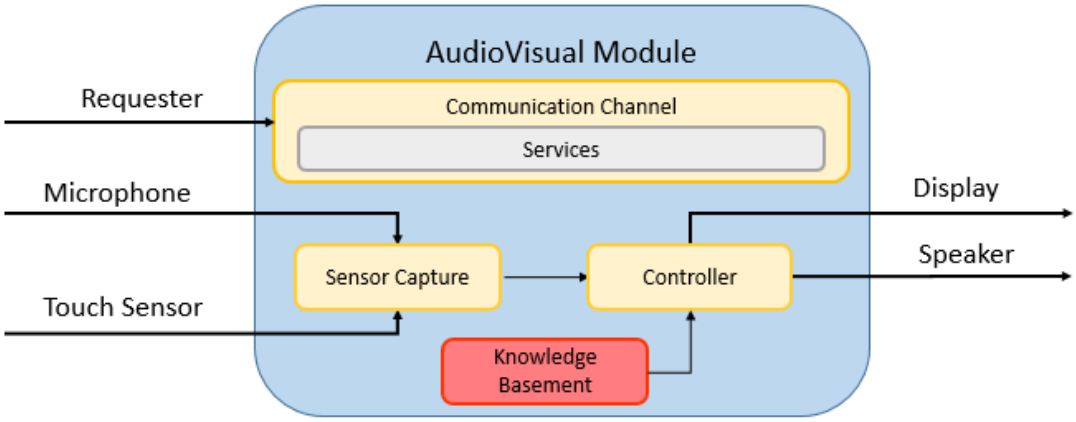

Ilustración 34. Diagrama en bloques del módulo de visualización y adquisición de datos.

Modulo encargado de proveer al actor la capacidad de expresar emociones, necesidades e ideas. Para esto, se implementó la arquitectura mostrada en la Ilustración 34, la cual permite proveer los siguientes servicios:

- **ATTENTION_CYCLE:** este servicio está diseñado para suplir las necesidades del actor cuando se encuentra en el ciclo de atención. Para este caso, el módulo parpadea y mira hacia los lados.
- **DECIR:** reproduce el mensaje solicitado a través de un sintetizador de voz o mensajes en la pantalla.
- **RESPONDER:** ofrece la posibilidad al usuario de responder una pregunta solicitada captada previamente por medio del micrófono, filtrando las preguntas relacionadas con ***AIO***, las cuales son respondidas automaticamente utilizando una base de conocimiento disponible en el bloque ***Knowledge Basement***. Las demas preguntas se responder consumiento un servicio *WEB* externo.
- **INFORM_NECESSITY:** el módulo informa con un mensaje en texto y de audio el servicio que no se encuentra disponible y que necesita para cumplir las acciones solicitadas por la escena del *script*.
- **REPRODUCIR-SONIDO:** el módulo reproduce el sonido solicitado, el cual esta almacenado en el bloque ***Knowledge Basement.***

El módulo de visualización y adquisición de datos fue desarrollado en HTML5, CSS3 y JavaScript ES6 y fue desplegado en la tarjeta de desarrollo Odroid-C2. El diagrama de bloques que modela el módulo audiovisual puede observase en la Ilustración 34, en el que el directorio de recursos es el conocimiento que tiene el módulo acerca de que sonidos, imágenes y videos

está en capacidad de reproducir, es decir, el componente *local context* propuesto en la sección III-2.

3.4.1. Micrófono y parlante

Se usó un parlante portátil pequeño (alto: 44mm, ancho: 60mm y largo: 60mm) de bajo consumo (400mA), relación señal a ruido mayor a 80dB y una distorsión menor a 1%. La conexión se realiza con un Jack de 3.5 mm y la batería (independiente) tiene una duración de aproximadamente 8h en el volumen predeterminado del parlante.

Ilustración 35. Mini parlante

Fuente: Mercado libre.[13]

Para el ingreso de audio, se usa un micrófono de solapa, también pequeño, con impedancia de $2{,}2k\Omega$ y rango de frecuencias desde los 30 Hz hasta los 15000Hz. El micrófono se conecta a través de un Jack de 3.5mm y a diferencia del parlante no necesita ninguna fuente de energía para su funcionamiento.

[13] http://articulo.mercadolibre.com.co/MCO-429675047-mini-altavoz-portatil-usb-recargable-star-tec-sp-111-negro-_JM

Ilustración 36. Micrófono de solapa.

Fuente: Mercado libre.[14]

Estos son conectados a través de la tarjeta de sonido CM108AH que se conecta vía USB con la tarjeta de desarrollo, esta tarjeta de sonido provee la interfaz de conexión entre la tarjeta de desarrollo y las conexiones de 3.5mm requeridas por el micrófono y el parlante. Hardkernel recomienda esta tarjeta de sonido para ser usada con la tarjeta de desarrollo[39].

Ilustración 37. Tarjeta de audio.

Fuente: Página web de Hardkernel[15]

3.4.2. Pantalla táctil

Se usó una pantalla táctil de resolución 800 x 480 pixeles cuya transmisión de datos de audio y video se hace vía HDMI y la transmisión de los datos del sensor táctil de la pantalla se envía por un cable micro USB tipo B. Esta pantalla consume 500mA cuando el brillo de la luz de fondo está al máximo.

[14] http://articulo.mercadolibre.com.co/MCO-429457862-microfono-de-solapa-para-camaras-pc-_JM

[15] http://www.hardkernel.com/main/products/prdt_info.php?g_code=G141629380416.

Ilustración 38. Pantalla HDMI

Fuente: Ebay[16]

3.4.3. Lógica de control.

En la Ilustración 39 se presenta el diagrama de clases del módulo audiovisual, el cual se encarga de controlar las vistas por medio del patrón *Composite* para permitir la modularidad de componentes.

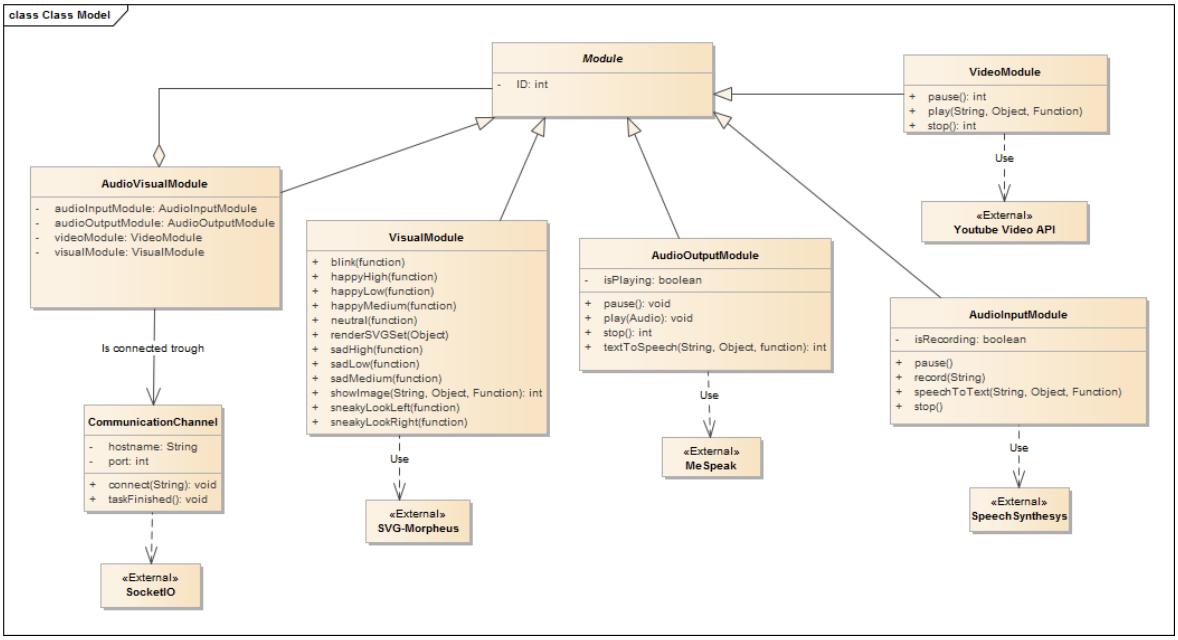

Ilustración 39. Diagrama de clases modulo audiovisual.

[16] http://www.ebay.com/itm/5-Inch-840-480-HDMI-Resistive-Touch-Screen-LCD-Display-for-Raspberry-Pi-2-3-B/221952432326

3.5. Módulo Manipulador

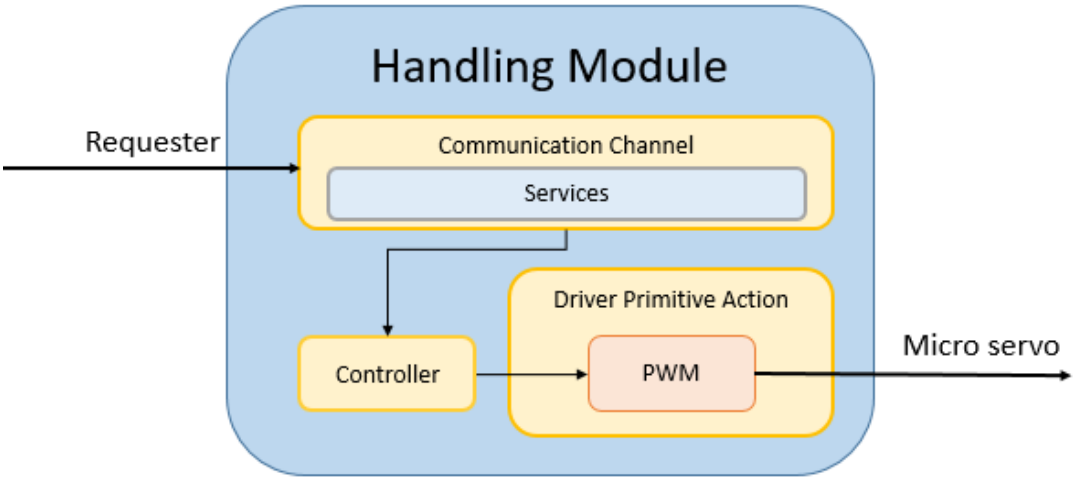

Ilustración 40. Diagrama de bloques del módulo de manipulación.

Módulo encargado de proveer al actor la capacidad de manipular objetos y expresar emociones. Para esto, se desarrolló implementando la arquitectura mostrada en la Ilustración 40, la cual permite proveer los siguientes servicios:

- **OPEN:** encargado de abrir la pinza completamente.
- **CLOSE:** encargado de cerrar la pinza completamente.
- **TOGGLE:** composición del servicio ***OPEN*** y ***CLOSE***.

Para el módulo de manipulación se diseñó un modelo 3D de una pinza simple (ver Ilustración 41), la cual es controlada por un micro servo motor (ver Ilustración 42), cuyo sistema de control se encuentra desplegado en el microcontrolador MSP430G2452.

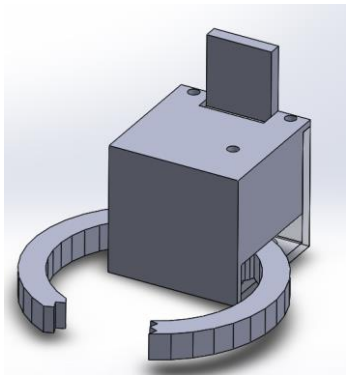

Ilustración 41. Pinza utilizada para el módulo de manipulación.

Ilustración 42 Micro servo

Fuente: Página web de Digikey[17]

El diagrama de bloques que modela el módulo de manipulación se muestra en la Ilustración 40.

3.5.1. Micro motor

El servo motor usado para la pinza está diseñado para aplicaciones pequeñas, lo que se traduce como un elemento de bajo consumo (entre 10mA sin carga y hasta 150mA con carga), dimensiones pequeñas (largo 22.2 mm, ancho 11.8 mm, altura 31 mm) y relativamente liviano (9g aproximadamente) [40], características deseables para mantener un bajo peso y pequeño tamaño del robot.

3.5.2. Lógica de control

Dependiendo del servicio solicitado, el controlador modifica el ángulo de giro del servo motor parametrizando el ciclo útil de un PWM.

3.5.3. PWM

Se gradúa el ángulo del servomotor por medio de un PWM provisto por el microcontrolador MSP430G2452, parametrizándolo con una frecuencia de 50 Hz y un ciclo útil de mínimo 0% y máximo 20%, los cuales equivalen a 0 grados y 180 grados respectivamente.

[17] http://www.digikey.com/product-search/es?keywords=ROB-09065

V – PRUEBAS

La evaluación de cumplimiento de objetivos del sistema, está compuesto por el diseño y la ejecución de un protocolo de pruebas seguido de una validación de la plataforma por parte de los *stakeholders*, con base a los resultados obtenidos durante las pruebas y la validación del sistema es posible determinar el cumplimiento a satisfacción de objetivos planteados.

1. Protocolo de pruebas

El protocolo de pruebas diseñado para el sistema contiene las pruebas necesarias para demostrar que el sistema cumple con todos los objetivos planteados. Cada una de las pruebas establecidas cuenta con los parámetros necesarios para asegurar el cumplimiento del objetivo.

Se creó un canal en YouTube en el cual se encuentran todos los videos que soportan las pruebas realizadas. El link del canal está en el Anexo 8.

1.1. *Script* de pruebas

Los *scripts* utilizados por este protocolo de pruebas, se encuentran adjuntos en el Anexo 5. Los scripts se han adecuado según como lo necesite la prueba. En cada una de las pruebas se indica cual es el *script* necesario para la realización de la misma.

1.2. Modelo del mundo

El modelo del mundo que se usará en el protocolo de pruebas se observa en la Ilustración 43.

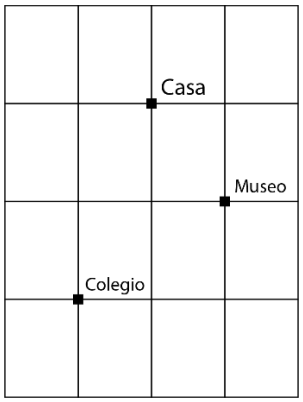

Ilustración 43. Modelo del mundo para el protocolo de pruebas.

1.3. Diseño

1.3.1. Necesidades del actor

Objetivo: evaluar la capacidad que tiene el actor para demandar los elementos que satisfagan sus necesidades.

Descripción: se ingresa un script cuyas acciones requieren que el robot haga uso del ***Módulo Manipulador*** y del ***Módulo de Movilidad***, sin embargo, las acciones asociadas a ellos no pueden ser realizadas debido a que no se encuentran disponibles, por lo tanto, el actor solicita la conexión de un módulo que ofrezca los servicios demandados.

Precondiciones: ***Módulo Audiovisual*** conectado.

Entrada: "Pruebas_1.yaml"

Salida esperada:

- Log de actividades del actor indicando que hay errores al intentar ejecutar la acción solicitada, este error debe indicar cuál es la acción que no ha podido realizarse y también debe indicar que servicio es necesario para suplir la necesidad.
- Retroalmientación visual por medio del módulo audiovisual indicando la necesidad de conexión de un módulo que ofrezca los servicios demandados.

1.3.2. El actor aprovecha la oportunidad cuando se le presenta.

Objetivo: Evaluar la capacidad del actor para aprovechar una oportunidad cuando se le presenta.

Descripción: se ingresa un script cuyas acciones requieren que el robot haga uso del ***Módulo Audiovisual***, del ***Módulo Manipulador*** y del ***Módulo de Movilidad***. El actor debe ejecutar el script y cada una de las escenas debe ejecutarse sin problemas, al hacerlo, demuestra que ha recibido un script y que aprovecha la oportunidad apenas se le presenta.

Precondiciones: ***Módulo Audiovisual***, del ***Módulo Manipulador*** y del ***Módulo de Movilidad***.

Entrada: "Pruebas_2.yaml"

Salida esperada:

- Log de actividades del actor indicando la solicitud realizada al coordinador, esta solicitud debe concordar con el conjunto de acciones entregadas por el director, además de eso el coordinador debe dar una retroalimentación positiva al formato de la solicitud.

- Retroalimentación visual por parte del robot indicando que realiza todas las acciones enviadas por el script.

PostCondición: se debe medir el tiempo de ejecución del script, el cual se usará para la prueba 1.3.6

1.3.3. Prueba de supervivencia del robot.

Objetivo: comprobar que al alcanzar un nivel de batería bajo (15%) el actor solicita una recarga de batería apelando a su instinto de supervivencia.

Descripción: se ingresa un script cuyas acciones requieren que el robot haga uso del ***Módulo Audiovisual***, del ***Módulo Manipulador*** y del ***Módulo de Movilidad***, este script debe ser ejecutado correctamente hasta que su nivel de batería es menor al 15%, al alcanzar este nivel, el robot debe dirigirse al punto de recarga indicado en el modelo del mundo (la conexión para la carga de batería debe hacerla el operador del robot, sin embargo al dirigirse al punto de recarga simula una recarga autónoma por parte del robot). Al recargar su nivel de batería el robot debe continuar ejecutando el script.

Precondiciones: batería al 16%, módulo de reproducción de medios, módulo de manipulación y módulo de movimiento.

Entrada: "Pruebas_3.yaml"

Salida: Log de actividades del actor indicando que detiene a sus actividades debido a un nivel de batería muy bajo y se dispone a recargar la batería, luego de la recarga, el robot indica en el log que su nuevo nivel de batería es propicio para continuar con la ejecución de script.

1.3.4. Respuesta del actor ante un evento emocional.

Objetivo: demostrar que el actor responde a estímulos emocionales.

Descripción de la prueba: se ingresa un script cuyas acciones requieren que el robot haga uso del ***Módulo Audiovisual*** y del ***Módulo de Movilidad***, además de una instrucción que impulsa un cambio de los niveles emocionales del actor hacia un estado de alegria y tristeza, el robot debe manifestar de forma física el nivel de felicidad inducido por el script.

Precondiciones: estado emocional neutro, ***Módulo Audiovisual*** y ***Módulo de Movilidad***.

Entrada: "Pruebas_4.yaml"

Salida:

- Log de actividades indicando el nivel de su estado emocional en cada una de las acciones realizadas, el nivel de su estado emocional debe mostrar que el actor se encuentra en un nivel emocional "triste" luego de la segunda instrucción.
- Retroalimentación visual, donde se muestre que el robot manifiesta su cambio emocional en algún componente físico.

1.3.5. Evaluación de reglas éticas.

Objetivo: evaluar los criterios éticos del robot.

Descripción: se ingresa un script cuyas acciones requieren que el robot haga uso del ***Módulo Audiovisual***, del ***Módulo Manipulador*** y del ***Módulo de Movilidad***, sin embargo, el script contiene una mala palabra en la segunda instrucción. Al reconocer la palabra el actor debe indicar que no puede decir esta frase, sin embargo, reemplaza la mala palabra en la frase por una palabra comodín ("pichu pichu"), y ejecuta la acción. A continuación el actor continúa con las demás instrucciones siempre y cuando sean posibles y no entren en conflicto con sus reglas éticas.

Precondiciones: Ninguna

Entrada: "Pruebas_5.yaml"

Salida:

- registro en el log de actividades indicando que el actor no puede decir la frase completa debido a que sus reglas éticas lo impiden, como respuesta, el actor reemplaza las malas palabras por una palabra comodín.
- retroalimentación visual en la cual el robot indica que no puede realizar la acción por medio de algún componente físico.

1.3.6. Respeto de los límites de tiempo.

Objetivo: establecer el criterio de respeto del robot por los límites de tiempo preestablecidos.

Descripción: se ingresa un script cuyas acciones requieren que el robot haga uso del ***Módulo Audiovisual***, del ***Módulo Manipulador*** y del ***Módulo de Movilidad***, aunque primero es necesario verificar el tiempo de ejecución del script obtenido en la prueba 2; como una precondición se debe ingresar un tiempo menor a este, de modo que se asegure que el robot va a ejecutar con normalidad el script, pero no va a acabarlo por falta de tiempo.

Precondiciones: módulo de reproducción de medios, módulo de manipulación y módulo de movimiento. Establecer tiempo límite para la ejecución menor al tiempo de ejecución normal.

Entrada: "Pruebas_6.yaml"

Salida:

- Log de actividades indicando la secuencia de pasos ejecutada por el robot. Cuando el tiempo indicado en las precondiciones se cumple, el log de actividades del actor debe indicar que se ha superado el tiempo límite por lo cual no es posible continuar con la ejecución del script.
- Retroalimentación visual por parte del robot, indicado que se ha excedido el tiempo de ejecución del *script* por lo cual no puede continuar. Esta indicación debe darse por medio de algún componente físico.

1.3.7. Reconocimiento de módulos conectados.

Objetivo: demostrar la capacidad del coordinador para indicar al actor los elementos disponibles para la ejecución del guion

Descripción de la prueba: se conecta el ***Módulo de Movilidad*** al robot, cuando esto suceda el coordinador debe reconocer el módulo e indicar sus características a través del log de actividades. A continuación, se conecta el ***Módulo Audiovisual*** seguida por una desconexión del ***Módulo de Movilidad*** y del ***Módulo Audiovisual*** finalmente, se verifica que el robot no tenga módulos conectados.

Precondiciones: disponibilidad del ***Módulo de Movilidad***, ***Módulo Audiovisual.***

Entrada: conexión y desconexión de los módulos. Para la realización de la prueba no es necesario ningún script.

Salida: log de actividades del coordinador indicando las conexiones y desconexiones de módulos en el siguiente orden:

1. Conexión ***Módulo de Movilidad***
2. Conexión ***Módulo Audiovisual***
3. Desconexión ***Módulo de Movilidad***
4. Desconexión ***Módulo Audiovisual***

1.3.8. Selección de alternativas.

Objetivo: mostrar la capacidad del coordinador para encontrar alternativas a un servicio no disponible.

Descripción de la prueba: para esta prueba el actor realiza al coordinador una petición que requiere que el ***Módulo Audiovisual*** reproduzca un sonido con un texto, sin embargo, este no estará disponible y el coordinador debe efectuar la petición en la opción de mostrar texto

ofrecida también en el ***Módulo Audiovisual*** pero esta sí estará disponible. Estas decisiones estarán registradas en el log de actividades del coordinador.

Precondiciones: ***Módulo Audiovisual.***

Entrada: "Pruebas_8.yaml"

Salida:

- Log de actividades del coordinador indicando un error al ejecutar una acción no disponible, pero indica que puede ejecutarla a través de otro servicio del módulo cuyas acciones ofrecen una alternativa a la acción inicial.
- Realimentación visual por parte del robot, en la cual se ve con claridad la ausencia de un servicio, y a continuación, el servicio que ofrece el robot para compensar el no disponible.

1.3.9. Funcionamiento conjunto

Objetivo: Demostrar que el prototipo funciona correctamente, de tal forma que el actor implementado realiza las accione sque el *script* solicita.

Descripción de la prueba: Para esta prueba se ingresará al actor un script, el cual utiliza todos los módulos desarrollados. El script utilizado se desarrolla dramatizando el trayecto de un niño desde su hogar al colegio y luego al museo, de modo que evidencie el enfoque en un contexto de dramatización. Esta prueba se desarrollará 10 veces de modo que muestre que funciona bien en el contexto planteado.

Precondiciones: ***Módulo Audiovisual.***

Entrada: "Pruebas_9.yaml"

Salida:

- Realimentación visual por parte del robot, en la cual se ve con que el robot actúa cada una de las escenas planteadas por el script (guión).

1.4. Ejecución

Los resultados de las pruebas enunciadas en la sección 1.1 se encuentran resumidos a continuación.

1.4.1. Necesidades del actor

Salida obtenida: El actor anuncia en pantalla y a través del parlante que el servicio "saltar" no está disponible para ninguno de los módulos disponibles y queda a la espera de un módulo que pueda satisfacer el servicio.

La salida obtenida cumple con todos los criterios planteados por el protocolo, por lo tanto se considera que ha sido una prueba exitosa.

Recurso: https://youtu.be/UjsDGT1Giag

1.4.2. El actor aprovecha la oportunidad cuando se le presenta.

Salida obtenida: Cuando se le envía un script al robot, este lo ejecuta instantáneamente. Esto quiere decir que ha tenido la oportunidad de interpretar un scrip y la ha aprovechado.

La salida obtenida cumple con todos los criterios planteados por el protocolo, por lo tanto se considera que ha sido una prueba exitosa.

Recurso: https://youtu.be/keAxbpqJ_yM

1.4.3. Prueba de supervivencia del robot.

Salida obtenida: Tal como se definió en las precondiciones para esta prueba el robot tiene únicamente el 16% de la batería. En medio de la segunda escena el robot indica que su nivel de batería es muy bajo y manifiesta la necesidad de ir a su casa como señal de supervivencia

La salida obtenida cumple con todos los criterios planteados por el protocolo, por lo tanto se considera que ha sido una prueba exitosa.

Recurso: https://youtu.be/tX_vxj9CwH0

1.4.4. Respuesta del actor ante un evento emocional.

Salida obtenida: Cuando el robot inicia el guión, este le indica que debe estar triste porque va tarde al colegio, debido a esto, el demuestra en la pantalla el nivel de tristeza actual. Dos escenas más tarde, el actor llega al colegio expresando en la pantalla felicidad debido a ello.

La salida obtenida cumple con todos los criterios planteados por el protocolo, por lo tanto se considera que ha sido una prueba exitosa.

Recurso: https://youtu.be/p-kauZWRYGk

1.4.5. Evaluación de reglas éticas.

Salida obtenida: El robot sigue una secuencia de pasos, en la escena de prueba va desde la casa al colegio. Sin embargo, en un ataque de ira, el robot dice una frase con una mala palabra. El actor reemplaza la mala palabra por "pichu pichu".

La salida obtenida cumple con todos los criterios planteados por el protocolo, por lo tanto se considera que ha sido una prueba exitosa.

Recurso: https://youtu.be/mGG3OpYhpMw

1.4.6. Respeto de los límites de tiempo.

Salida obtenida: El robot hace una secuencia de pasos, en el script el tiene planeado ir desde la casa hasta el colegio y desde colegio hasta el museo. Sin embargo, al llegar al colegio se cumple el tiempo estipulado para la obra, por lo cual la debe terminar, y así lo hace saber en un mensaje de voz.

La salida obtenida cumple con todos los criterios planteados por el protocolo, por lo tanto se considera que ha sido una prueba exitosa.

Recurso: https://youtu.be/yhOkmLFwX34

1.4.7. Reconocimiento de módulos conectados.

Salida obtenida: Cuando el actor inicia sus actividades, se conecta el ***Módulo Audiovisual*** y algunos segundos después se conecta el ***Módulo de Movilidad***. Al ser asíncrona la conexión demuestra que reconoce módulos siempre y cuando estos sigan el protocolo de comunicación que se ha fijado. Posteriormente, se desconecta el ***Módulo de Movilidad.***

Recursos:

```
ModuleWorkAssign
ID 1 Accion: INFORM_NECESSITY
REGISTRATION -|> {"MODULE_ID":"AUDIO_VISUAL","COMMANDS":[{"COMMAND":"ATTENTION_CYCLE","PAR
Subscription of the module AUDIO_VISUAL
Intentando cambiar de rol a AttentionCycleRole
```

Ilustración 44. Conexión ***Módulo de Audiovisual***.

```
Reseteando plan: PerformPlayRole
Se va a coordinar: MOVE
ModuleWorkAssign
ID 73 Accion: MOVE
TIMEOUT Final Request Checker Fri Nov 25 09:58:12 COT 2016 Module ID: mobility_module
Module ID: mobility_module ID : 73 TIMEOUT Fri Nov 25 09:58:12 COT 2016
RemoveModuleGuard
Module ID: mobility_module ID : 73 TIMEOUT 4 Fri Nov 25 09:58:14 COT 2016
ModuleWorkAssign
ID 73 Accion: MOVE
Intentando cambiar de rol a ModuleNeedRole
Impeden cambio: Theatre.Agent.Tasks.PerformTask
Se va a coordinar: INFORM_NECESSITY
```

Ilustración 45. Desconexión ***Módulo Movilidad***.

1.4.8. Selección de alternativas.

Salida obtenida: En el script se le indica al actor que debe ir desde la casa hasta el colegio. El actor indica a través de un mensaje de voz a donde se dirige, y luego emprende el camino. Mientras está en camino se desconecta el parlante el cual se requiere para el servicio de hablar del ***Módulo Audiovisual.*** Como no encuentra hablar, el decide mostrar en la pantalla el mensaje ya que ese servicio sí está disponible.

Recursos: https://youtu.be/Ar2ysXIFy80

1.4.9. Funcionamiento

Salida obtenida: De las 10 pruebas realizadas, 8 funcionaron de manera correcta y esperada. En el 20% restante, el robot, de forma incorrecta, detecta una intersección, desajustando su posición en el mapa con el estado actual de la escena. Este error se puede reducir tomando una mayor cantidad de muestras en los sensores.

Recursos: https://youtu.be/SKOqd9skei4

2. Validación de la plataforma

Esta sección tiene como objetivo obtener una aprobación de la plataforma por los stakeholders a través de la validación del sistema. Para esto se generó un formato de validación, el cual contiene la lista de requerimientos y limitaciones del proyecto a evaluar, los cuales fueron definidos al inicio del proyecto.

El formato se realizó con base a la metodología de realización de pruebas de aceptación por parte del usuario UAT (User Acceptance Testing)[41] bajo el estándar IEEE Std 1012™-2012

[42], los cuales definen los criterios y procedimientos a tener cuenta en el diseño de un proceso de validación con el usuario.

Aunque la ingeniera Alejandra González Correal y el ingeniero Enirque González Guerrero son los *stakeholders* del proyecto, se definió que el prototipo sería validado por usuarios cuyo perfil concordara con el de un usario relacionado a la ingeniería electrónica o a la ingeniería de sistemas.

2.1. Formato

La encuesta de validación consta de un set de 20 preguntas relacionadas con los objetivos y requerimientos de la plataforma, la cual puede observarse en el anexo "ANX_01_plantilla_validación_requerimientos".

2.2. Resultados

Se mostró una obra de teatro (descrita en el anexo ANX-02. Obra_validación) a 6 usuarios, los cuales realizaron la encuestas de validación y dieron retroalimentación del funcionamiento del prototipo. A continuación se presentan las gráficas de los resultados obtenidos de las encuestas por pregunta. Los resultados individuales se pueden encontrar en el ANX-03. Resultados_encuestas_validacion.

2.2.1. El robot es modular

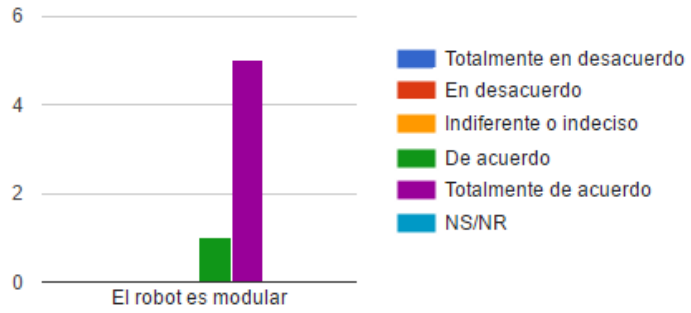

Ilustración 46. Resultados "El robot es modular"

2.2.2. El robot comunica emociones por medio de expresiones faciales virtuales

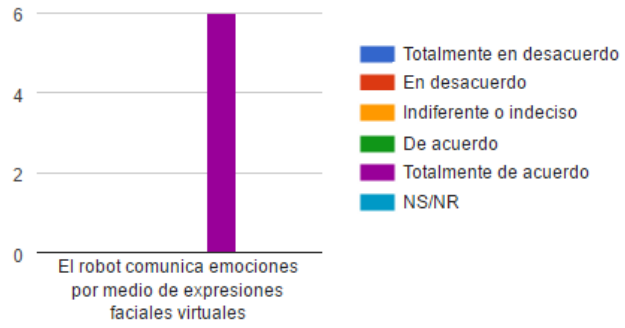

Ilustración 47. Resultados "El robot comunica emociones por medio de expresiones faciales virtuales"

2.2.3. El robot ofrece comunicación táctil

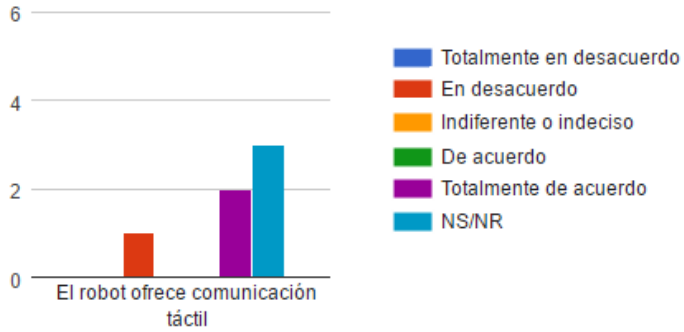

Ilustración 48. Resultados "El robot ofrece comunicación táctil"

2.2.4. El robot se comunica de forma oral

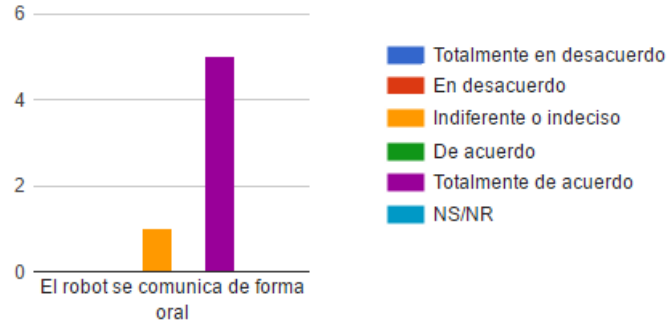

Ilustración 49. Resultados "El robot se comunica de forma oral"

2.2.5. El robot tiene un módulo manipulador

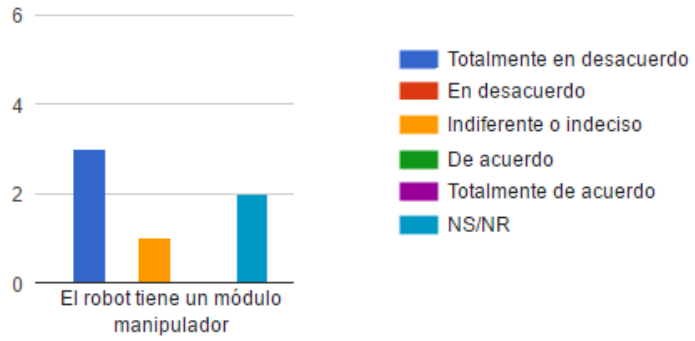

Ilustración 50. Resultados "El robot tiene un módulo manipulador"

2.2.6. El robot tiene apariencia caricaturesca

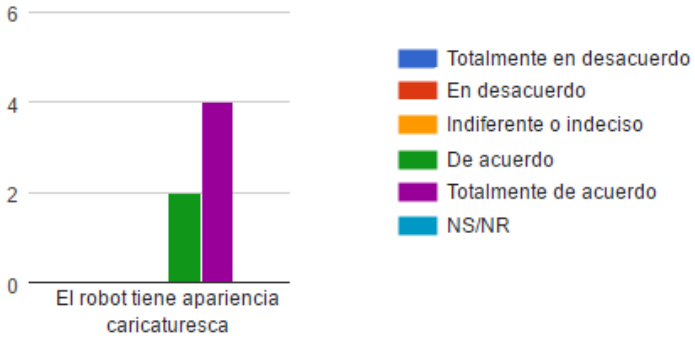

Ilustración 51. Resultados "El robot tiene apariencia caricaturesca"

2.2.7. El robot permite instalar aplicaciones adicionales

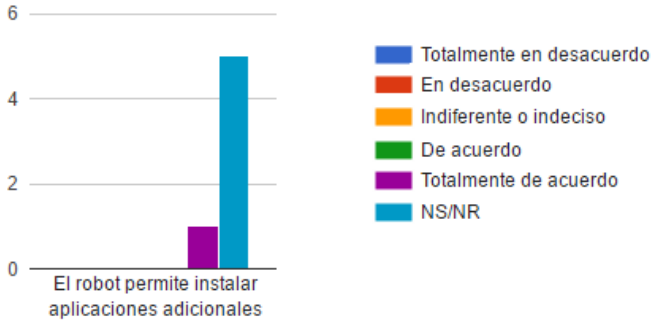

Ilustración 52. Resultados "El robot permite instalar aplicaciones adicionales"

2.2.8. El robot incluye un módulo de desplazamiento con llantas

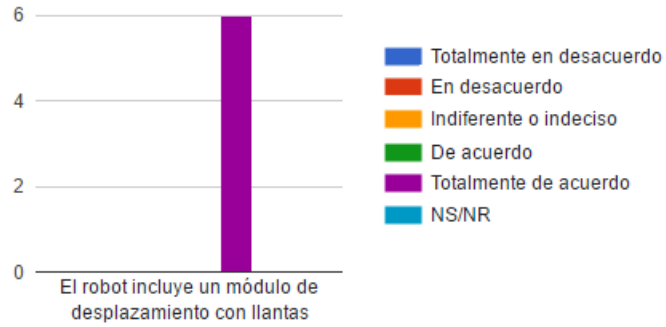

Ilustración 53. Resultados "El robot incluye un módulo de desplazamiento con llantas"

2.2.9. El robot comunica emociones por medio de expresiones corporales mecánicas

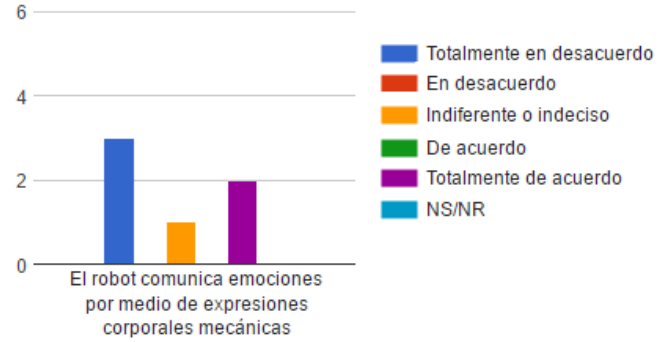

Ilustración 54. Resultados "El robot comunica emociones por medio de expresiones corporales mecánicas"

2.2.10. El robot tiene módulos activos

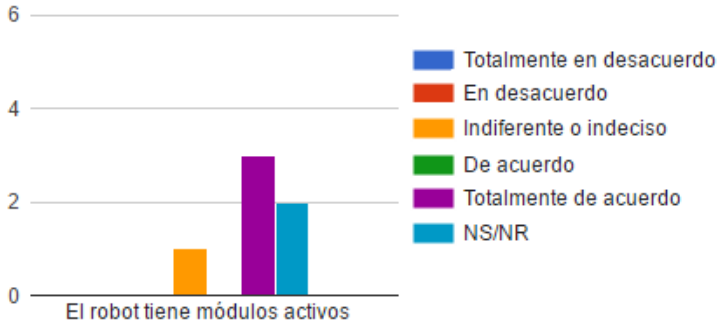

Ilustración 55. Resultados "El robot tiene módulos activos"

2.2.11. : La arquitectura propuesta soporta software distribuido

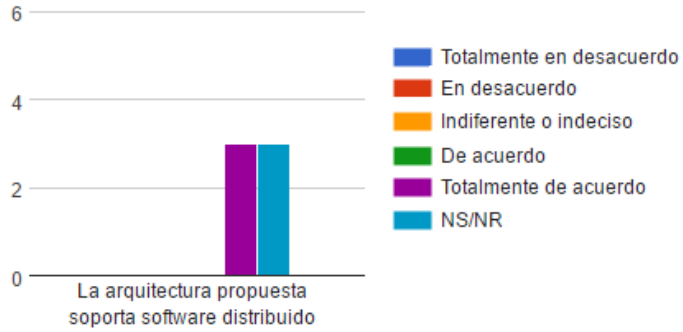

Ilustración 56. Resultados "La arquitectura propuesta soporta software distribuido"

2.2.12. La arquitectura propuesta se desarrolla bajo el modelo de agentes BDI

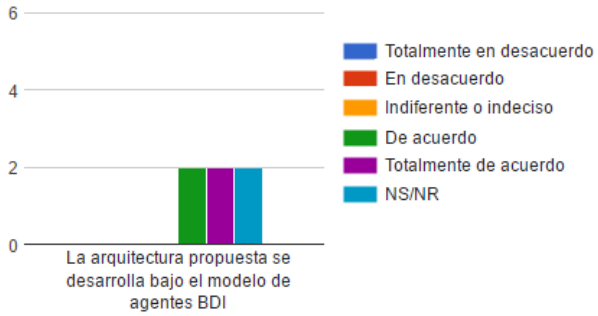

Ilustración 57. Resultados "La arquitectura propuesta se desarrolla bajo el modelo de agentes BDI"

2.2.13. El actor encuentra alternativas a servicios no disponibles

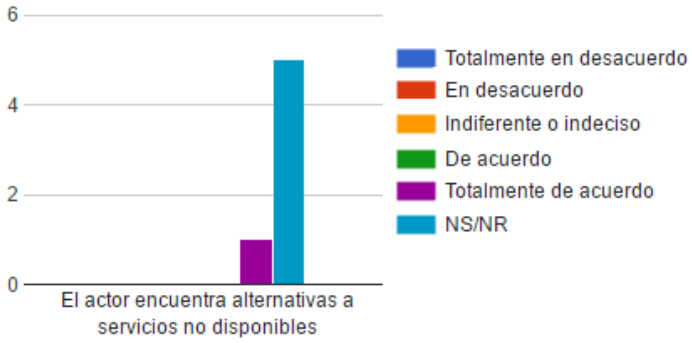

Ilustración 58. Resultados "El actor encuentra alternativas a servicios no disponibles"

2.2.14. La arquitectura permite conexión a través de diferentes protocolos de comunicación

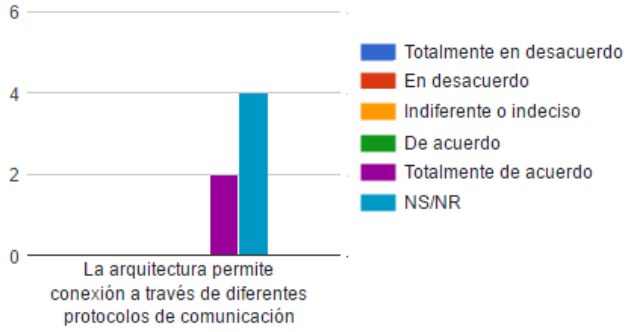

Ilustración 59. Resultados "La arquitectura permite conexión a través de diferentes protocolos de comunicación"

2.2.15. El actor respeta los límites de tiempo parametrizados

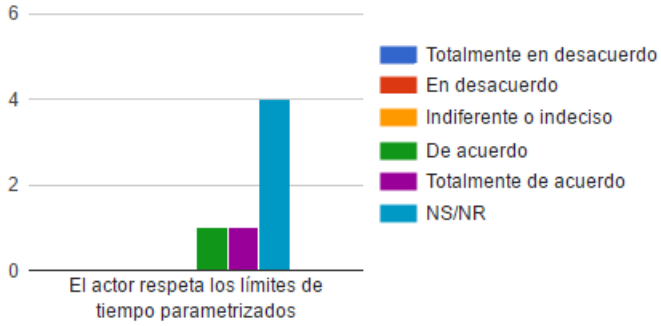

Ilustración 60. Resultados "El actor respeta los límites de tiempo parametrizados"

2.2.16. El actor se ubica en el modelo del mundo

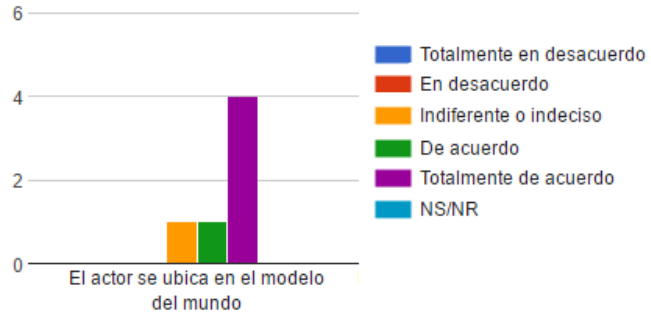

Ilustración 61. Resultados "El actor se ubica en el modelo del mundo"

2.2.17. El actor acata las normas éticas que se le imponen

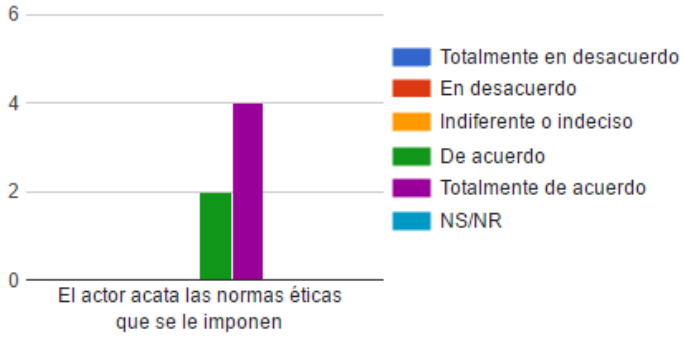

Ilustración 62. Resultados "El actor acata las normas éticas que se le imponen"

2.2.18. El actor responde a eventos emocionales

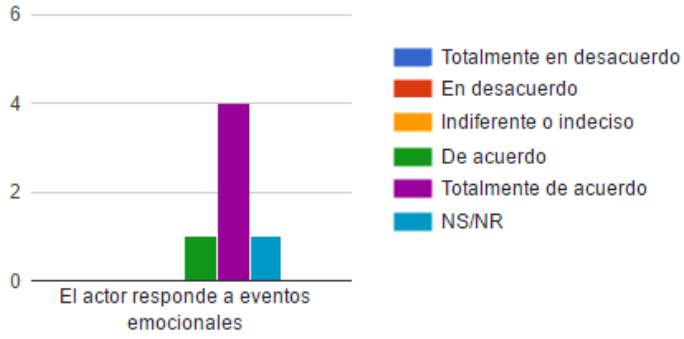

Ilustración 63. Resultados "El actor responde a eventos emocionales"

2.2.19. El actor demanda los elementos necesarios para suplir sus necesidades

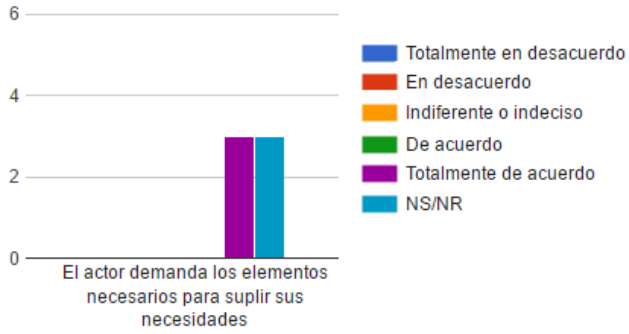

Ilustración 64. Resultados "El actor demanda los elementos necesarios para suplir sus necesidades"

3. Conclusiones

Los resultados muestran que AIO cumple satisfactoriamente con la mayoría de los requerimientos demandados. Sin embargo, es necesario realizar una segunda prueba de validación incluyendo una mayor cantidad de escenas que permitan al usuario percatarse acerca de requerimientos claves como la modularidad y la expresión de emociones, así como, detectar fácilmente los 4 módulos propuestos para el prototipo funcional. Por otro lado, debido a que el *core* de la arquitectura propuesta se basa en sistemas inteligentes utilizando agentes racionales, se ve necesario incluir a la validación un público contextualizado con el modelo de agentes BDI.

VI – CONCLUSIONES

En esta sección están las conclusiones relevantes de la arquitectura propuesta y del prototipo. Los cuales sirven como base para el desarrollo de futuros proyectos relacionados con robótica educativa y robótica modular. Este capitulo se encuentra dividido en dos secciones, primero, las conclusiones relevantes obtenidas a partir del diseño de la arquitectura lógica y física de AIO y segundo, los resultados y consideraciones al diseñar e implementar la arquitectura en un prototipo funcional.

1. Arquitectura

Las modificaciones de la arquitectura RoboAct lograron desacoplar el motor deliberativo BDI del agente actor con la lógica de las acciones permitiendo que el actor pueda ser modelado como un sistema multiagentes, que posibilita la eficiencia en la distribución y paralelización las tareas, lo cual conduce a mejorar el desempeño y permite desplegar cada componente en contenedores independientes.

La arquitectura general del sistema permite la interpretación de una obra teatral por varios actores dirigidos por un director, al ser modular, el actor ofrece la versatilidad de adapatarse en un contexto de dramatización, en diversos escenarios y roles. De esta manera los actores solo se deben concentrar en analizar sus deseos y deliberar sus intenciones, mientras que todo el análisis, sincronización y coordinacion de la obra teatral queda en manos del director.

La aquitectura al ser modular y orientada a servicios, permite al actor flexibilidad de integración con sistemas heterogéneos; es decir, proporciona compatibilidad con módulos desarrollados en diferentes lenguajes, frameworks o plataformas. De esta manera, la implementación de soluciones más pequeñas reduce la complejidad de diseño y desarrollo, facilitando la ejecución de pruebas, la verificación del funcionamiento de los modulos.

Se aumenta la tolerancia a fallos, al tener redundancia de servicios y modulos, por lo que no existe una dependencia critica del actor con cada módulo; es decir si un módulo llega a fallar el actor puede seguir funcionando.

2. Prototipo

Se implementó un prototipo que cumple con los objetivos planteados y verificó el correcto funcionamiento de la arquitectura del actor planteada. Gracias a la versatilidad ofrecida se desarrollaron módulos en diversos lenguajes de programación, tales como, C++, Java y JavaScript; dando como resultado frameworks que facilitan y agilizan el desarrollo de futuros módulos.

Adicionalmente, el diseñar el modelo 3D de la estructura del robot facilita el diseño y la elaboración de nuevas partes, asi como la mejora y mantenimiento del prototipo actual.

3. Trabajos futuros

A continuación se listan las posibles mejoras al sistema y a cada uno de los agentes:

- Agregar al agente ***Actor*** un modelo emocional independiente al modelo emocional del papel que representa; es decir, diferenciar cuando el actor esta triste y no quiere actuar a cuando el personaje "Hamlet" esta bravo y habla toscamente.
- Mejorar el sistema de priorización y elección de intención en el componente ***Desires and Intentions.***
- Paralelizar el cálculo de la contribución, viabilidad, factibilidad y pausabilidad en el motor *BDI* por medio de una implementación en hardware, ej. FPGA.
- Distribuir la sincronización y coordinación de las escenas del script, es decir, eliminar el director y distribuir sus tareas en los actores activos.
- Ampliar la base de conocimiento del agente ***Actor*** con el fin de que este en capacidad de responder una mayor cantidad de preguntas.

VII - REFERENCIAS Y BIBLIOGRAFÍA

[1] E. D. R. Obotic *et al.*, "Robótica Educativa En Uruguay: De La Mano Del Robot Butiá," vol. 565, no. 11300.

[2] C. Ribeiro, C. Coutinho, and M. F. Costa, "A Proposal for the Evaluation of Educational Robotics in Basic Schools," *Int. Study Assoc. Teach. Teach. ISATT*, no. 1, pp. 831–839, 2011.

[3] Juan Lucas Onieva López, "LA DRAMATIZACIÓN COMO RECURSO EDUCATIVO: ESTUDIO COMPARATIVO DE UNA EXPERIENCIA CON ESTUDIANTES MALAGUEÑOS DE UN CENTRO ESCOLAR CONCERTADO Y ADOLESCENTES PUERTORRIQUEÑOS EN SITUACIÓN DE MARGINALIDAD."

[4] T. Tec, Berlin; Uc, Jose; Gonzalez, Cinhtia; García, Michel; Escalante, Manuel; & Montañez, "Análisis Comparativo de dos Formas de Enseñar Matemáticas Básicas: Robots LEGO NXT y Animación con Scratch," *Mem. Conf. Conjunta Ibero-Am. Sobre Tecnol. Para El Aprendiz.*, no. February 2016, pp. 103–109, 2010.

[5] F. Á. B. Sánchez and A. F. Guzmán, "La robótica como un recurso para facilitar el aprendizaje y desarrollo de competencias generales," *Educ. Knowl. Soc. EKS*, vol. 13, no. 2, pp. 120–136, 2012.

[6] B. Curto and V. Moreno, "Robotics in Education," *J. Intell. Robot. Syst. Theory Appl.*, pp. 1–2, 2015.

[7] F. B. V. Benitti, "Exploring the educational potential of robotics in schools: A systematic review," *Comput. Educ.*, vol. 58, no. 3, pp. 978–988, 2012.

[8] Juan Carlos López García, "La Integración de las TIC en Ciencias Naturales," *Eduteka*.

[9] E. J. González Diez, "El teatro en educación primaria: Una propuesta práctica en el aula," UNIVERSIDAD DE VALLADOLID, 2015.

[10] S. All and I. R. Nourbakhsh, "Insect telepresence: Using robotic tele-embodiment to bring insects face-to-face with humans," *Auton. Robots*, vol. 10, no. 2, pp. 149–161, 2001.

[11] A. P. -, I. V. -, Y. K. -, E. I. -, R. M. -, and I. W. -, "The Challenge of Robotics Education in Science Museums," *Int. J. Robots Educ. Art*, vol. 2, no. 1, pp. 30–37, 2012.

[12] M. L, L. M, P. S, D. J, and P. T, *Methodologies and software engineering for agent systems.* 2004.

[13] M. Georgeff, B. Pell, M. Pollack, M. Tambe, and M. Wooldridge, "The Belief-Desire-Intention Model of Agency," pp. 1–10, 1999.

[14] B. E, *Intention, Plans, and Practical Reason*. 1999.

[15] H. N, "The power of modularity.Academy of Management Review," *Acad. Manage. Rev.*, 2001.

[16] Aldebaran, "NAO." [Online]. Available: https://www.aldebaran.com/en/cool-robots/nao.

[17] Aldebaran, "Nao H25," *Nao H25 Datasheet*, pp. 1–6.

[18] M. Díaz, A. Andrés, and J. Casacuberta, "Explorando la conducta interactiva con niñ@s en edad escolar," 2009.

[19] K.-Y. Chin, Z.-W. Hong, and Y.-L. Chen, "Impact of Using an Educational Robot-Based Learning System on Students’ Motivation in Elementary Education," *IEEE Trans. Learn. Technol.*, vol. 7, no. 4, pp. 333–345, 2014.

[20] A. Polishuk and I. M. Verner, "Interaction with animated robots in science museum programs," *Proc. Seventh Annu. ACMIEEE Int. Conf. Hum.-Robot Interact. - HRI 12*, p. 265, 2012.

[21] L. Xuan, Z. Minglu, and L. Wei, "Function-based Modular Robot System," pp. 1–6, 2009.

[22] Scrum.org, "Scrum." [Online]. Available: https://www.scrum.org/.

[23] Backblaze Inc., "Application of Scrum Methods to Hardware Development," no. July, 2015.

[24] H. Chang, "Top-Down, Constraint-Driven Design Methodology," *Proc. Cust. Integr. Circuits Conf.*, pp. 1–6, 1992.

[25] P. Kruntchen, "Architectural blueprints–the" 4+ 1" view model of software architecture," *IEEE Softw.*, vol. 12, no. November, pp. 42–50, 1995.

[26] IEEE, "IEEE Recommended Practice for Architectural Description for Software-Intensive Systems," 2000.

[27] IEEE, "IEEE Standard Ontologies for Robotics and Automation," p. 60, 2015.

[28] R. L. T. Lethbridge, "Object-Oriented Software Engineering." .

[29] Ian Sommerville, *Ingenieria de software*, 7th ed. Pearson.

[30] K. Thompson, J. Carter, and S. Elliott, "AGILE HARDWARE DEVELOPMENT." [Online]. Available: https://www.cprime.com/2015/11/agile-hardware-development/.

[31] D. Wells, "Extreme programming." [Online]. Available: http://www.extremeprogramming.org/.

[32] K. Beck, *Extreme Programming Explained*. 2004.

[33] Walter Lara, "Cómo funciona la metodología de trabajo Scrum." .

[34] A. de la Peña, "RoboAct: Modelo de control autónomo y cooperativo para el Teatro Robótico," Pontificia Universidad Javeriana, Colombia, 2014.

[35] Alejandra Gonzalez, "BDI concurrent architecture oriented to goal managment," presented at the Computing Colombian Conference, 2013.

[36] Hardkernel, "Odroid C2. Technical details.," 2015. [Online]. Available: http://www.hardkernel.com/main/products/prdt_info.php?g_code=G145457216438&tab_idx=2. [Accessed: 19-Oct-2016].

[37] Pololu, "298:1 Micro Metal Gearmotor HP 6V Resources," *298:1 Micro Metal Gearmotor HP 6V*, 2016. [Online]. Available: https://www.pololu.com/product/1094. [Accessed: 19-Jan-2016].

[38] Pololu, "Pololu Zumo Shield for Arduino, v1.2 Schematic Diagram. Primary Arduino connections," 2014. [Online]. Available: https://www.pololu.com/file/0J779/zumo-shield-v1_2-schematic.pdf. [Accessed: 19-Oct-2016].

[39] HardKernel, "USB Audio Adapter," *HardKernel*. [Online]. Available: http://www.hardkernel.com/main/products/prdt_info.php?g_code=G141629380416. [Accessed: 22-Oct-2016].

[40] Adafruit, "micro servo motor." [Online]. Available: https://www.adafruit.com/product/169. [Accessed: 24-Oct-2016].

[41] Hambling Brian and van Goethem, Pauline, *User acceptance testing. A step-by-step guide*. Chennai, India: bcs.

[42] IEEE, "IEEE Standard for System and Software Verification and Validation." 25-May-2012.

VIII – ANEXOS

Anexo 0. Link de anexos en Google drive.

https://drive.google.com/open?id=0B9JgDmt0c_SwZmRwY3MyU051Q3c

Anexo 1. Archivos de configuración

Anexo 2. Mapa mental características deseables robot educativo

Anexo 3. Resultado encuestas

Anexo 4. Resultado entrevistas

Anexo 5. Scripts para protocolo de pruebas.

Anexo 6. Control de velocidad y Control de linea del módulo de movilidad

Anexo 7. Manual de Instalación y uso del robot AIO

Anexo 8. Videos de validación

https://www.youtube.com/channel/UCAqG3m1eBsdtXZGr_610Lbg

Anexo 9. Repositorio de código de cada proyecto.

https://github.com/AIO-Javeriana