UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA: INGENIERÍA DE SISTEMAS

Tesis previa a la obtención del título de: INGENIERO DE SISTEMAS

TEMA:
DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE UBICACIÓN Y ENRUTAMIENTO PARA PERSONAS CON DISCAPACIDAD VISUAL EN UN AMBIENTE CONTRALADO POR MEDIO DE ALGORITMOS INTELIGENTES Y MEDIOS ÓPTICOS.

AUTORES:
CRISTHIAN ALBERTO MIRANDA GANCHOZO
EDISON WLADIMIR ROMERO CHICAIZA
DANILO EFRAIN TACO VARGAS

DIRECTOR:
DANIEL GIOVANNY DÍAZ ORTIZ

Quito, octubre de 2013

DECLARATORIA DE RESPONSABILIDAD Y AUTORIZACIÓN DE USO DEL TRABAJO DE GRADO

Nosotros, Cristhian Alberto Miranda Ganchozo, Edison Wladimir Romero Chicaiza y Danilo Efrain Taco Vargas, autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de grado y su reproducción sin fines de lucro.

Además declaramos que los conceptos y análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

_____

Cristhian Alberto Miranda Ganchozo
CI: 1310479231

_____

Edison Wladimir Romero Chicaiza
CI: 1718901281

_____

Danilo Efrain Taco Vargas
CI: 1718342585

DEDICATORIA

El presente trabajo es un compendio de los esfuerzos, los sueños y anhelos de los seres a los cuales debo mi existencia, formación, mi manera de ver la vida y de afrontarla, mis padres y mis familiares.

Por tanto dedico el presente trabajo y todas sus repercusiones futuras a mi Padre fuente inagotable de inspiración, ejemplo de ser humano, profesional y caballero a carta cabal, quien inculco en mí desde los inicios de mi existencia su tenacidad, convicción y principios de bien.

Dedico además el presente a mi madre fuente de sabiduría, apoyo incondicional y pilar fundamental de mi vida quien con su infinito amor de madre supo guiar, comprender y formar al ser humano que soy hoy por hoy.

Dedico además el éxito que implica el presente trabajo a mis dos madres de corazón mi abuela madre quien ya no está entre nosotros y mi tía mamá Petita quienes con su amor y ejemplo inculcaron en mí, principios y guías de vida desde mis primeros pasos.

A mi familia en general y al Creador por estar junto a mi durante todo el transcurso de este viaje convolusionado que es la vida.

Cristhian Alberto Miranda Ganchozo

2013

DEDICATORIA

Dedico este trabajo a mis padres, César y Lourdes quienes año tras año sacrificaron mucho por mi bienestar.

A mi padre César de quien tengo el apoyo incondicional, quien me ayudó a conseguir mi sueño, el ser un profesional.

A mi madre querida Lourdes quien con su amor me supo guiar para tomar las decisiones correctas, quien me enseñó a levantarme después de una caída aunque esta sea dolorosa. Y quien me ayudó a cumplir este sueño.

También dedico este trabajo a mi hermana Adriana a quien desde pequeño vi como un gran ejemplo a seguir, por ser una persona fuerte y luchadora.

Dedico a mis hermanos queridos Marco y Lissette que siempre con sus bromas me impulsaban a seguir adelante.

Dedico a toda mi familia, puesto que sin su apoyo no hubiera redactado estas palabras.

Edison Wladimir Romero Chicaiza

2013

DEDICATORIA

Es mi más profundo deseo dedicar este trabajo

A mis padres

Quienes con su infinito amor, perseverancia, comprensión y bendiciones diarias me han inculcado los mejores valores qupe me guiaran a lo largo de la vida. A mi padre quien con su forma de ver el mundo me enseño que no importa los obstáculos que se presenten, sino cómo se desenvuelve uno ante ellos, siendo humilde y perseverante ante todo; he aprendido a ser más humano. A mi madre que con su amor, sabiduría y ternura supo cuando decir no y cuando decir si, discerniendo de esta manera el mejor camino a seguir y enseñándome a ver lo hermoso que puede ser el mundo si uno es el que toma el control.

Danilo Efrain Taco Vargas

2013

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El trabajo de investigación y desarrollo fue concebido con el objetivo de atacar la problemática que causa la inserción laboral de personas con capacidades diferentes en tareas cotidianas, debido que en la mayoría de los lugares son subutilizados en labores estáticas y repetitivas. Además de facilitar al empleador el cumplimiento de la ley orgánica de discapacidades en el artículo 47, a través de la fusión de tres tecnologías como son: Aforge (Object Tracking) es la encargada del seguimiento del individuo dentro del frame de video durante la ejecución del sistema, LSharp que funciona como motor para los algoritmos de búsqueda informada que calculan y retornan las ruta a seguir y por último Voice Recognition que permite la comunicación entre el individuo y la aplicación en ambas vías, a través de una plataforma común C# (perteneciente al paquete de Microsoft .Net), es el valor agregado que tiene el proyecto de tesis, puesto que permite que se localice, comunique y guie a una persona no vidente en un ambiente controlado, facilitando de esta manera su movimiento dentro de lugares cómo: bodegas, bibliotecas entre otros en los cuales pueden ser empleados en tareas de logística.

ABSTRACT

This thesis is a research and development job, which was conceived with the aim of attacking the problem caused by the labor insertion of people with disabilities in everyday tasks, because in most places and almost all the time people with disabilities are underutilized at work static and repetitive. In addition to facilitating the employer compliance with disability law in article 47, throws the fusion of three kinds of technologies described bellow: AFORGE (Object Tracking) responsible for tracking of the person inside of video frame in real time during the system execution, LSHARP work like a motor to execution of informed search algorithms in order to calculate and provide the route to follow by the user, and the last part Voice Recognition used in communication between user and software Lazar-I-Soft in both ways, through a common platform designed on C#, LSHARP and AFORGE. This is the additional value of this project, the perfect mix between those technologies because may guide, communicate and track the user inside of a real controlled environment in real time, in order to guide and start the move the user in places such as warehouse, libraries in which may be employed people with visual disabilities in logistical tasks

INTRODUCCIÓN

Ecuador en los últimos años se ha convertido en un referente en temas relacionados con la inclusión y mejoramiento de la calidad de vida de personas con capacidades diferentes. Dichas iniciativas datan desde el Gobierno del Dr. Gustavo Noboa Bejarano que mediante decreto de ley N° 2000-25 reforma la Ley de Discapacidades que fue publicada en el Registro Oficial N° 171 del 26 de septiembre del 2000; continuando con esta tan notable y admirable labor el Lic. Lenin Moreno Vicepresidente de la República del Ecuador, en el periodo 2007-2012 crea la Misión Solidaria Manuela Espejo e impulsa de manera eficiente la creación de proyectos que estén encaminados al mejoramiento de la calidad de vida de la población ecuatoriana.

Por tales méritos el Ecuador se ha convertido en el primer y más importante puntal en Sur América en temas de Geo posicionamiento de personas con discapacidad, legislación, protección de derechos y más.

Desde enero de 2010, rige la disposición que reza: "El Código del Trabajo establece, a partir del 2010, contratar al menos el 4% del total de los trabajadores de la empresa, siendo ese el porcentaje fijo que se aplicará en los años sucesivos (Art. 42 núm. 33 CT)", misma que habilita al Ministerio de Relaciones Laborales para la inspección y sanción en caso de incumpliendo.

Luego de iniciar la vigencia del artículo 42, las empresas públicas y privadas se ven en la obligación de cumplir con el mismo, y surgen inconvenientes relacionados a la ubicación de dicho personal.

Con estos antecedentes se tiene la iniciativa de crear un Sistema Inteligente que ayude a personas con discapacidades visuales a involucrarse en un ambiente laboral en donde puedan desempeñar diferentes actividades.

El sistema básicamente consta de tres módulos interconectados entre sí, siendo estos: enrutamiento, visión artificial y reconocimiento de voz. Cada modulo mencionado cumple funciones específicas las cuales permiten que una persona no vidente pueda efectuar una actividad en un ambiente controlado.

Fusionando software y hardware se construye un prototipo (maqueta) a escala que representa un ambiente real en donde se ejecutan las pruebas y las demostraciones de la funcionalidad del sistema inteligente.

CAPÍTULO 1
ASPECTOS GENERALES

1.1. Planteamiento del problema

Según la Organización Mundial de la Salud (OMS), "en el mundo existen aproximadamente 285 millones de personas con discapacidad visual, de las cuales 39 millones son ciegas y 246 millones presentan baja visión, del universo de estas personas el 90% del global de personal con discapacidad visual viven en países en vías de desarrollo" (Organización Mundial de la Salud (OMS). Estadísticas, 2013), como es el caso del Ecuador.

En el Ecuador existen alrededor de 331.106 personas con algún tipo de discapacidad, de las cuales 37.759 padecen discapacidad visual como lo muestran las estadísticas del Consejo Nacional de Discapacidades (CONADIS). La mayor concentración de este sector de la población se encuentra en las provincias de Manabí, Guayas y Pichincha y puesto que en estas últimas dos provincias se encuentra el asentamiento mayoritario del aparato productivo, nace la necesidad de crear un medio que permita la inclusión de estos ciudadanos en el mismo, desarrollando así un Sistema Inteligente con capacidades de guía, enrutamiento y gestión de personas con discapacidad visual, para así facilitar la inclusión de los mismos en el mundo laboral y el mejoramiento de su calidad de vida.

Mediante la implementación del sistema Lazar-I-Soft en operaciones de logística, se podrá contar con la intervención de estos ciudadanos en actividades destinadas para personas con plenas capacidades visuales, además de facilitar el cumplimiento de la ley para los empleadores ya que estos se encuentran obligados por el artículo 42 de la constitución vigente, a contratar del total de su planta, un 4% de personas con capacidades diferentes.

1.2. Objetivos

1.2.1. Objetivo general

Diseñar y Construir un sistema informático que sea capaz de ubicar, enrutar y guiar personas no videntes en un ambiente controlado, a través de la implementación de algoritmos de búsquedas informadas, reconocimiento de imágenes y reconcomiendo de voz.

1.2.2. Objetivos específicos

- Diseñar y construir los módulos primarios, que serán los encargados de los procesos vitales del sistema como son:
  - Enrutamiento.
  - Monitoreo y guía.
  - Comunicaciones.
- Crear e implementar el módulo de reconociendo de imágenes por gama de color (Object Tracking), mediante el cual se realizará tareas y procesos de monitoreo y ubicación del individuo.
- Crear e implementar el módulo de reconcomiendo de voz y comunicación que se encargará de mantener la interacción directa del individuo con el sistema.
- Crear e implementar los módulos de: enrutamiento, guía y búsqueda, que son los encargados de interconectar el módulo de reconocimiento de imagen con el módulo de comunicación para generar las rutas y comunicar las órdenes.
- Diseñar y construir los módulos suplementarios, mismo que serán los encargados de brindar información y soporte a los módulos primarios.
  - Módulo para la alimentación de datos correspondientes a los ítems.
  - Interface de administración del sistema Lazar-I-Soft.

1.3. Justificación del proyecto

El presente proyecto se ha desarrollado con el fin de demostrar los conocimientos adquiridos durante el transcurso de la carrera de Ingeniería de Sistemas, además de afianzar la formación humana que se recibió durante el mismo periodo.

Un dato adicional al proyecto es que fue desarrollo en versión beta para participar en el concurso internacional Imagen Cup que organiza la transnacional Microsoft Coorporation, mismo que cumple con lo inicialmente planteado, amalgamando de manera óptima los dos grandes puntales de la formación recibida que son la excelencia académica y la formación humana de calidad.

Por todo cuanto se describe, Lazar-I-Soft no es solo una pieza de software aislada de nuestra realidad puesto que es la muestra más clara de la vinculación con la colectividad y el trabajo, el proyecto es la respuesta a una realidad que nos impacta y a la vez desconcierta la cual es la subutilización de personas con capacidades diferentes (problemas visuales), mismas que se encuentran en muchos casos desplazadas de actividades que son reservadas para personas con capacidades visuales plenas.

Nuestros objetivos buscan generar un sistema informático que permita la movilidad de personas no videntes en un ambiente controlado, para de este modo poder atacar el problema desde su núcleo logrando que dichas personas puedan tener mayor comodidad al realizar tareas de personas videntes, incluyéndolas en el mundo laboral de una manera más cómoda y natural, además de que se amplía el universo de posibilidad para la ubicación de las mismas.

1.4. Alcance del proyecto

El proyecto será diseñado y construido para su funcionamiento sobre una maqueta a escala determinada de 1:17, debido a las dificultades para presentar en un espacio a escala 1:1.

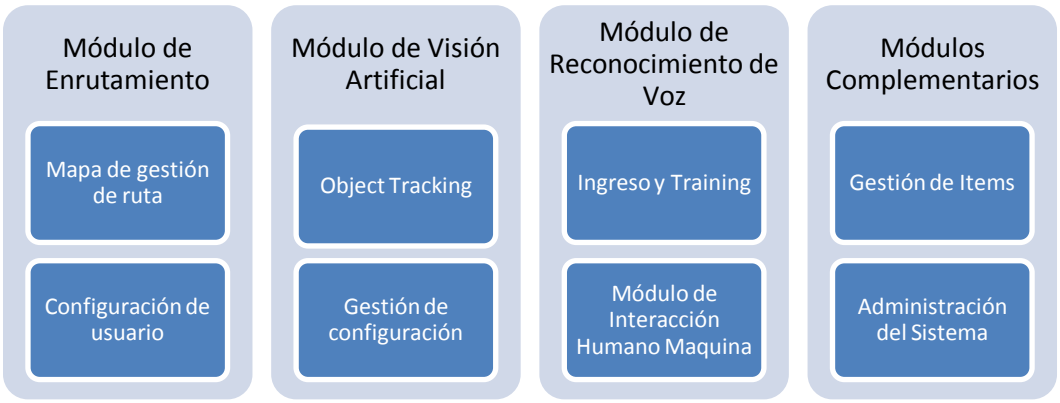
El sujeto de prueba será representado por medio de un muñeco, que será controlado por una persona, misma que interactuará directamente con el sistema y ejecutará las órdenes del mismo sobre el ambiente simulado, el sistema captará la posición exacta del individuo (muñeco) mediante el procesamiento de la imagen obtenida de la cámara web, que a su vez será procesada por el módulo de visión artificial.

La comunicación entre la máquina y la persona será procesada mediante el módulo de Reconocimiento de voz, mismo que tomará las órdenes del sistema y las comunicará al individuo y viceversa.

El procesamiento de las rutas o caminos que seguirá el individuo en el sistema prototipo, será el producto del módulo de algoritmo de enrutamiento, el cual recibe el punto a dónde quiere llegar el individuo.

El sistema respetará la estructura siguiente: Véase Figura 1 Product break down.

Figura 1 Product break down



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

1.5. Metodología

Etimológicamente hablando la palabra metodología proviene de tres vocablos griegos: *metà* (“más allá”), *odòs* (“camino”) y *logos* (“estudio”), por lo cual la Real Academia de la Lengua Española la define como: Un conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal. Razón por la cual todo trabajo de investigación o proyecto de cualquier área esté relacionada a él una metodología, específicamente el proyecto de tesis utiliza la metodología que se describe a continuación.

1.6. Metodología Extreme Programing “XP”

XP es una metodología ágil para el desarrollo de proyectos informáticos que está focalizada “en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software” (Beck, 1999), incentivando y promoviendo el trabajo en

equipo, prestando especial atención en el aprendizaje de los desarrolladores y brindando un ambiente de trabajo agradable en donde todos los miembros del equipo se sientan los más cómodos posible.

XP se basa principalmente en la realimentación de información continua con el cliente y el equipo de desarrollo, la comunicación correcta y fluida con todos los participantes, la mayor simplicidad posible en las soluciones implantadas y la determinación y decisión para afrontar los cambios. Razones por las cuales XP es la metodología por excelencia para proyectos que no poseen requisitos bien definidos o muy cambiantes, con pocos miembros en el equipo de desarrollo y en los cuales existe un alto grado de riesgo técnico.

En XP los principios y prácticas son propios del sentido común pero al borde del extremo, de ahí su nombre. A continuación se presenta las características básicas de la metodología en cuestión, agrupadas en tres ítems.

- ***Historias de usuarios***

  XP utiliza las historias de usuarios como técnicas de recolección de requisitos, sean estos funcionales o no funcionales, a través de tarjetas de papel en las cuales los clientes describen brevemente las características que el sistema deber tener. El tratamiento de estas tarjetas es muy dinámico y flexible, debido a que las historias de usuarios en cualquier momento pueden ser remplazadas por otras más detalladas o modificadas por información que otras historias poseen. Cada historia de usuario es lo más consistente, clara y delimitada posible, de tal manera que permite a los programadores realizar implementaciones en pocas semanas.

  En cuanto al contenido y formato de las historias de usuario, el universo de plantillas es muy amplio debido a que no existe un consenso en este tema. A continuación se describe la plantilla planteada por el mentor de la metodología Kent Beck la que contiene: “fecha, tipo de actividad (nueva, corrección, mejora), prueba funcional, número de historia, prioridad técnica y

del cliente, referencia a otra historia previa, riesgo, estimación técnica, descripción, notas y una lista de seguimiento con la fecha, y comentarios". (Letelier & Penadés, 2006)

- ***Roles***

  Los roles que se describen a continuación respetan la propuesta original de XP presentada por Kent Beck.

  - *Programador*

    Es el encargado de escribir el código y definir las pruebas unitarias. La comunicación entre este y los demás roles del equipo debe ser la mejor posible.

  - *Cliente*

    Sus responsabilidades son las de escribir las historias de usuarios y las pruebas funcionales. Además de ser el que asigna la prioridad a las historias y decide el orden de la implementación en las iteraciones focalizándose en aportar mayor valor al negocio.

  - *Encargado de las pruebas*

    Se encarga de colaborar con el cliente en la descripción de las pruebas funcionales, ejecutar las pruebas, difundir los resultados al equipo y es responsable de las herramientas de soporte de las pruebas.

  - *Encargado del seguimiento*

    Su función es la de realizar las labores de seguimiento que se describen a continuación:

    - Verificar el grado de acierto entre las estimaciones de tiempo realizadas y el tiempo real invertido.
    - Comunicar los resultados de las estimaciones para poder realizar ajustes en las mismas de ser el caso.
    - Evaluar en cada iteración si los objetivos son alcanzables con relación al tiempo y los recursos.
    - Decidir si se deben realizar cambios en las iteraciones para lograr alcanzar los objetivos planteados para las mismas.

- *Entrenador (Coach)*

  Es responsable de todo el proceso en general, debe conocer a fondo la metodología a fin de que la pueda hacer cumplir y guiar a los miembros del equipo.

- *Consultor*

  Es un ente externo al equipo que posee una expertise en un tema específico que es necesario en el proyecto, es el encargado de guiar al equipo para resolver un problema en particular.

- *Gran jefe*

  Es el enlace entre los clientes y programadores, su labor primordial es la coordinación. (Beck, 1999, págs. 46-53)

- ***Procesos***

  El ciclo de desarrollo con XP implica los siguientes pasos:

  a) El cliente define el valor de negocio a implementar.
  b) El programador estima el esfuerzo necesario para su implementación.
  c) El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
  d) El programador construye ese valor de negocio.
  e) Vuelve al paso a.

  - *Fase I Exploración*

    En la presente fase el cliente desarrolla a groso modo las historias de usuario que son de interés para la primera entrega del producto, a la vez que los demás miembros del equipo se familiarizan con las herramientas, tecnologías y demás temas del proyecto.

    En esta fase se esbozan y prueban las posibles arquitecturas del sistema y se construye un prototipo. El tiempo que toman estas tareas va desde pocas semanas a pocos meses dependiendo de la envergadura del proyecto.

  - *Fase II Planificación de Entrega*

    En la planificación de entrega los clientes establecen la prioridad de cada una de las historias de usuarios, para que paralelamente los

programadores realicen la estimación de tiempo y esfuerzo de cada una de ellas. Se toman acuerdos con respecto a la primera entrega y se realiza el cronograma conjuntamente con el cliente, cabe recalcar que una entrega no debe tomarse más de tres meses.

La unidad de medida que utilizan los programadores para las estimaciones de esfuerzo asociadas a las historias de usuarios es el punto, partiendo del hecho de que un punto equivale a una semana de programación idealizada, las historias de usuarios por lo general implican de uno a tres puntos. Por otro lado ya definida la unidad de medida el equipo de desarrollo debe llevar el registro de la velocidad del desarrollo que estará basado en puntos por iteración.

La planificación puede ser realizada tomando como referencia el tiempo o el alcance. Si se realiza la planificación con base en el tiempo "se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar" (Beck, 1999), si la planificación es con base en el alcance del sistema, "se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación" (Beck, 1999, pág. 53).

- *Fase III Iteraciones*

  En la presente fase se incluyen iteraciones sobre el sistema, previas a la entrega. El plan de entrega está formado por iteraciones menores a cuatro semanas. En la primera iteración se debe definir la arquitectura del sistema que deberá ser utilizada en el transcurso del proyecto, para lo cual se deben escoger las historias que fuercen la creación de la arquitectura, con las debidas restricciones que soliciten o exijan los clientes. Los elementos que se deben tomar en cuenta a la hora de crear el plan de iteraciones son:

  - Las historias de usuario no abordadas.
  - La velocidad del proyecto.

- Las pruebas de aceptación no superadas en la iteración anterior.
- Las tareas no terminadas en la iteración anterior.

- *Fase IV Producción*

  En la Fase en cuestión se requiere realizar pruebas y revisiones de rendimiento adicionales previas a la instalación del sistema en el entorno del cliente. Al mismo tiempo que se deben tomar decisiones sobre la inclusión de nuevas características en la versión actual o la reducción de las mismas por cambios ocurridos en esta fase.

- *Fase V Mantenimiento*

  Una vez el proyecto migrado a producción en su primera versión, se debe mantener el mismo en funcionamiento a la vez que se desarrollan nuevas iteraciones. Esta fase puede implicar una disminución en la velocidad de desarrollo y en algunos casos la introducción de más miembros en el equipo para compensar dicha disminución.

- *Fase VI Muerte del proyecto*

  La muerte del proyecto puede darse en tres escenarios: el primero es cuando el cliente no posee historias de usuario para ser incluidas al sistema, es decir el caso ideal, de darse de esta manera se genera la documentación final y ya no se realizan cambios en el sistema. La segunda de las causas por las cuales puede morir el proyecto es cuando el sistema no genera los beneficios esperados por el cliente. Y por último en caso del que el proyecto ya no sea sostenible

CAPÍTULO 2
VISIÓN ARTIFICIAL

2.1. Generalidades

David Marr define la visión cómo: "Vision is the process of discovering from images what is present in the world, and where it is" (Noë, 2002, pág. 229).

Siendo una parte importante de la rama de la Inteligencia Artificial, se puede definir a la Visión Artificial como una tecnología que permite analizar y procesar imágenes digitales capturadas mediante un ente tecnológico externo como una cámara web.

Uno de los objetivos principales de la visión artificial es simular la capacidad que tiene el ser humano de identificar imágenes, procesarlas y utilizarlas dependiendo de las necesidades.

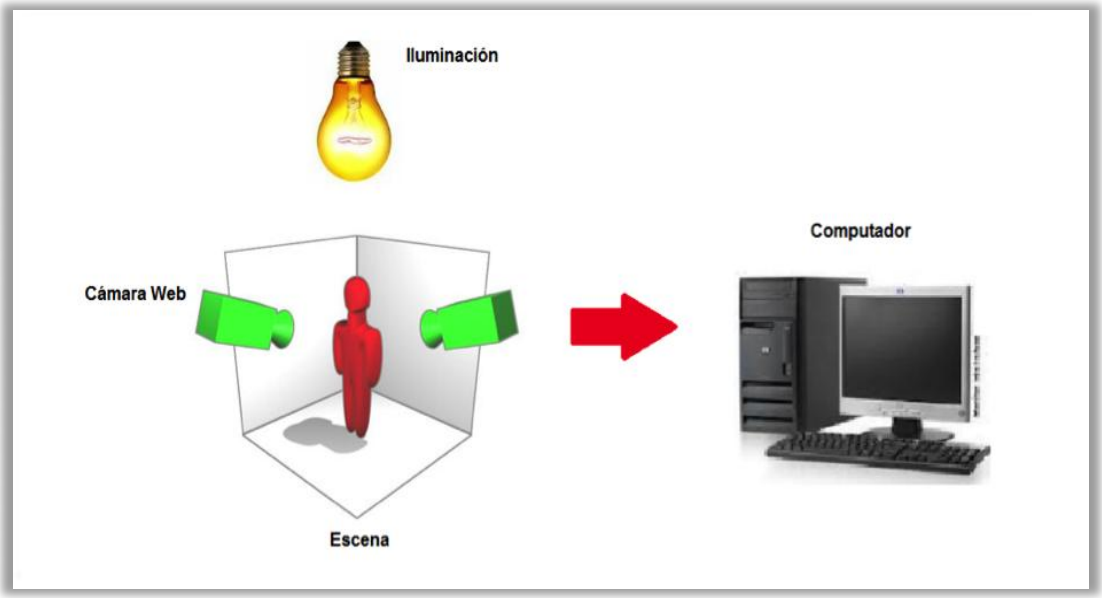
En la actualidad la rama de la Visión Artificial ha sido implementada en varias aplicaciones tanto de escritorio como web, obteniendo buenos resultados en los sistemas construidos. A continuación se mencionan algunos sistemas en los cuales es posible implementar esta tecnología:

- Sistemas de procesamiento digital de imágenes.
- Reconocimiento óptico de caracteres.
- Procesamiento automatizado de huellas dactilares.
- Navegación de robots en entornos controlados.
- Seguimiento de objetos, entre otros.

2.1.1. Componentes de un sistema de visión artificial

Un sistema básico de Visión Artificial está conformado por varios elementos que intervienen en el análisis y procesamiento de una escena específica, cada uno de estos complementándose entre sí. En la Figura 2 Componentes básicos de un sistema de visión artificial, se puede observar los componentes que intervienen en el sistema.

Figura 2 Componentes básicos de un sistema de visión artificial



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.1.2. Etapas de un sistema de visión artificial

Independiente del sistema que se esté desarrollando, es posible identificar las etapas (adquisición de imágenes, procesamiento**,** segmentación, representación-descripción y reconocimiento) de un sistema de visión artificial. Estas etapas se pueden agrupar en dos grandes pilares; el primer pilar "*Formación de las Imágenes*" conformado por la iluminación, captación de la imagen y la adquisición y el segundo pilar "*Procesamiento de las imágenes*" conformado por todos los algoritmos utilizados para el procesamiento.

A continuación se detalla brevemente cada una de estas etapas:

- ***Adquisición de imágenes***

  Es la captura de la imagen que va hacer procesada, esta captura de imagen depende mucho del tipo de iluminación, cámara, etc.

- ***Procesamiento***

  En la etapa de procesamiento se trata de mejorar la calidad de la imagen eliminando defectos que se hayan generado al momento de realizar la captura de la imagen, así mismo se resaltan las partes de esta imagen.

- ***Segmentación***

  En la segmentación se decide que secciones de la imagen van a ser analizadas y que secciones no van a ser analizadas.

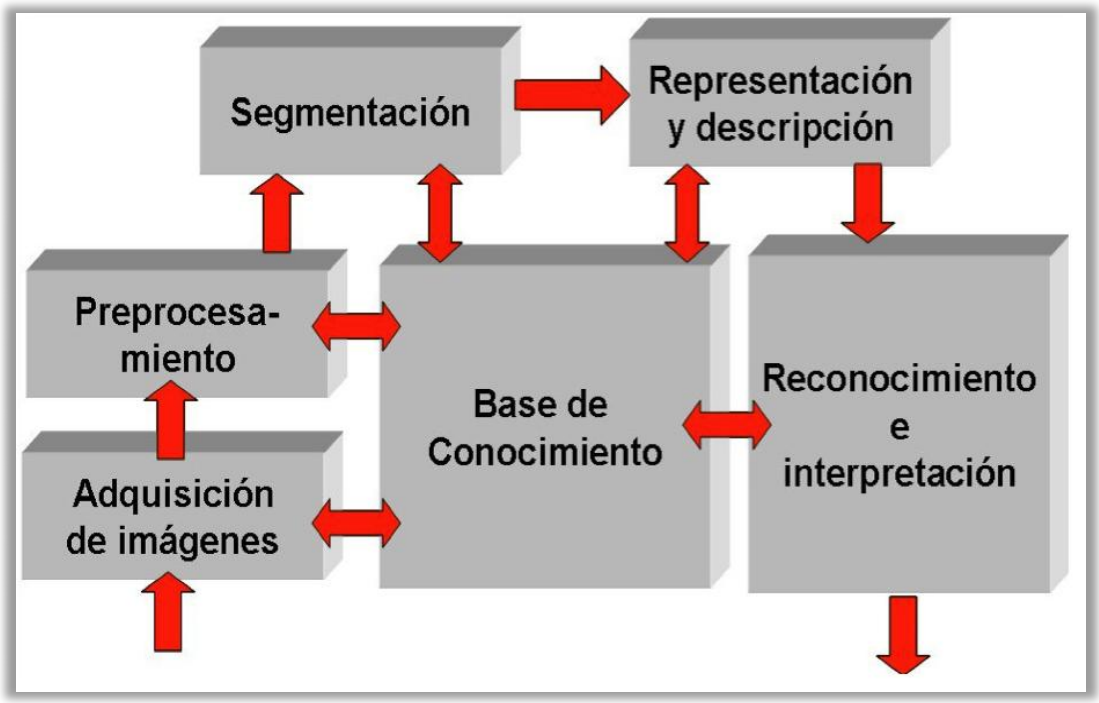
- ***Representación y descripción***

  Se definen las características que se toman en cuenta para realizar el análisis de las imágenes.

- ***Reconocimiento***

  Es una de las etapas más importantes ya que con una correcta segmentación y una correcta representación se puede realizar un buen reconocimiento de dicha imagen.

Figura 3 Etapas de un sistema de visión artificial



Fuente: (Visión artificial e interacción sin mandos, 2010)

2.2. Adquisición de imágenes

Un sistema de visión artificial, se realiza la adquisición de imágenes en un plano bidimensional 2D, para posteriormente procesarlas, con el fin de obtener la información requerida de dicha imagen.

Para que la adquisición de imagen sea un completo éxito depende de varios factores como: iluminación, dispositivo de captura de imagen, escenario.

2.2.1. Iluminación

Es uno de los elementos más importantes que intervienen en los sistemas de visión artificial, el sistema de iluminación debe ser elegido dependiendo del entorno en donde se está ejecutando la aplicación, ya que al tomar una mala decisión en la iluminación puede provocar graves problemas los cuales pueden reducir la calidad de la imagen. Los problemas más comunes que se presenta en la adquisición de la imagen son: sombras, niveles muy bajos de contrastes, distorsiones en la imagen, etc.

Así mismo, una buena selección de la iluminación logra que el escenario en donde se ejecuta la aplicación esté impecable para el procesamiento de las imágenes.

2.2.1.1. Fuentes de luz

- ***Incandescentes***

  "La luz incandescente se produce cuando los átomos se calientan y empiezan a liberar alguna de su vibración termal como radiación electromagnética" (Fuentes de Luz, 2009).

- ***Luminiscentes***

  A lo contrario de las fuentes de luz incandescentes, las fuentes luminiscentes se producen a bajas temperaturas. Cuando un electrón se encuentra a niveles de energía bajos libera una pequeña cantidad de energía, esta se convierte en un fotón o luz de colores.

2.2.1.2. Técnicas de iluminación

- ***Iluminación posterior***

  Permitiendo visualizar la silueta del objeto, la iluminación posterior es una técnica la cual se la utiliza para obtener un mayor contraste de la escena, perfiles bien detallados y sobre todo debido a que es una técnica muy fácil de implementar.

- ***Iluminación frontal***

  Si el objetivo es obtener una imagen clara, bien detallada y sobre todo evitar sombras de la escena es recomendable utilizar la técnica de iluminación frontal la cual es esencial para sistemas de visión artificial.

- ***Iluminación lateral***

  Si se requiere dos efectos de iluminación en la misma escena, es recomendable utilizar la técnica lateral la cual permite visualizar dos costados, el un costado iluminado y claro, y por el otro costado oscuro con sombra.

- ***Iluminación de día nublado***

  Esta técnica es muy utilizada en superficies muy irregulares en las cuales el objetivo puede confundirse con la superficie.

- ***Iluminación en campo oscuro***

  La iluminación en campo oscuro es utilizada para dar mayor visibilidad al objeto debido a que la fuente de luz se encuentra muy cerca de la superficie en la que encuentra el objeto.

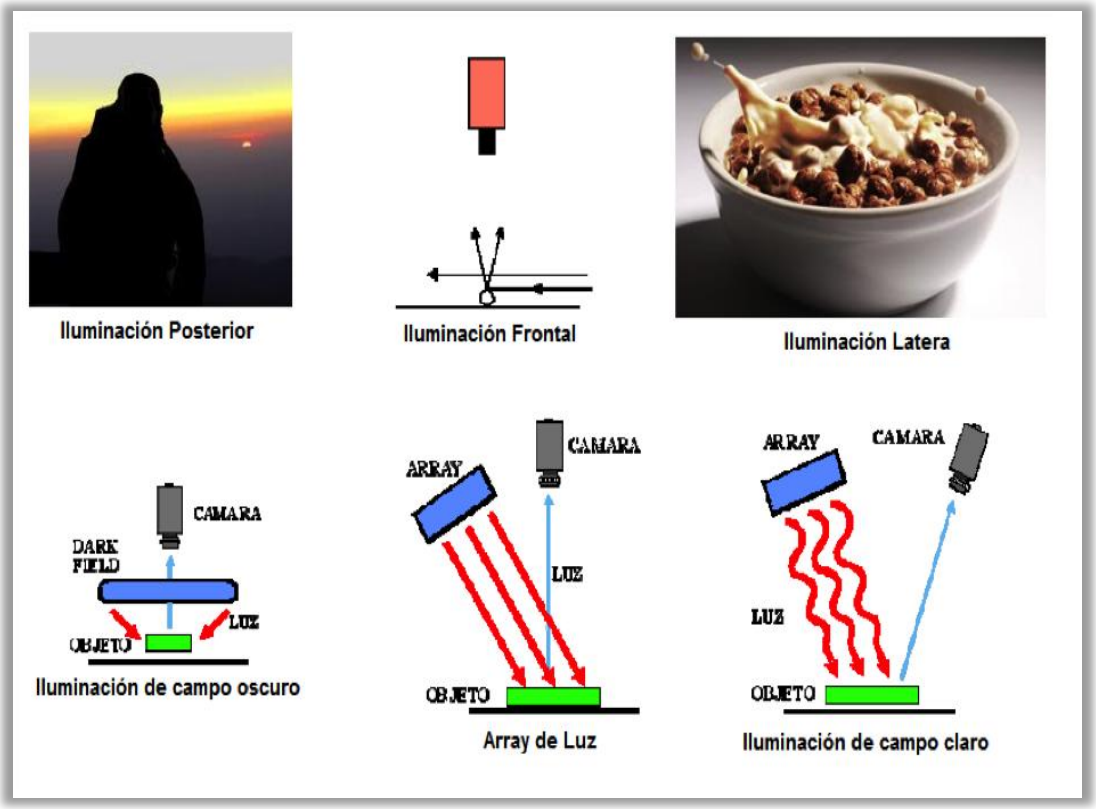
- ***Array de luces***

  Para resaltar la imagen y ganar un alto contraste de la misma, se debe utilizar la técnica de iluminación array de luces, debido a que esta permite a la luz llegar directamente al objeto obteniendo el resultado mencionado.

- ***Iluminación en campos claros***

  Al utilizar esta técnica se debe tomar en cuenta el ángulo en que se encuentra la iluminación, debido a que puede provocar sombras no esperadas en la imagen procesada.

Figura 4 Técnicas de iluminación



Fuente: (Iluminación para las aplicaciones de Visión Artificial, 2005)

2.2.2. Cámara

La función principal de las cámaras en un sistema de visión artificial es capturar las imágenes y enviarlas al sistema de procesamiento.

2.2.2.1. Estructura

La estructura básica de una cámara digital ha ser utilizada en un sistema de visión artificial consta de los siguientes elementos:

- ***Objetivo fotográfico***

  "Formado por una combinación de lentes convergentes y divergentes, recubiertos con una capa antirreflejos azulada" (García, 2009).

  La función de este es localizar los objetivos a largas o cortas distancias utilizando el zoom óptico.

- ***Sensor fotográfico CCD***

  Consiste en capturar los fotones de las imágenes convirtiéndolos en impulsos eléctricos de corriente alterna.

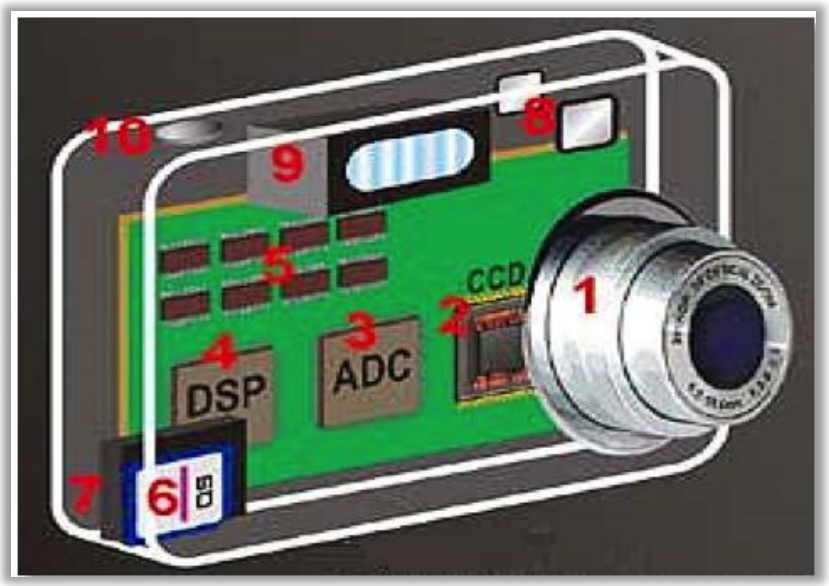
- ***Dispositivo ADC (Convertidor Analógico-Digital)***

  Toma los impulsos electrónicos y los convierte en código numérico binario.

- ***Dispositivo DSP (Procesador de Señal Digital)***

  Procesa las imágenes para enviarlas al almacenamiento de memoria.

Figura 5 Estructura de una cámara digital



Fuente: José Antonio García.

2.2.2.2. Características

Una cámara que va ser utilizada en un sistema de visión artificial debe poseer las siguientes características:

- ***Sensibilidad***

  Sensibilidad al espectro electromagnético para que genere una señal eléctrica proporcional a la recibida.

- ***Digitalizador***

  Permite la conversión de señal eléctrica a señal digital.

- ***Velocidad de obturación***

  Velocidad al captar las imágenes de una escena en movimiento, mientras más rápida sea la velocidad de obturación se obtendrá mejores capturas de movimiento y aumentará la nitidez de la imagen, pero así mismo mientras mayor sea la velocidad de obturación se deberá aumenta la iluminación.

2.2.2.3. Tipo de cámaras utilizadas en sistemas de visión artificial

- ***Cámaras lineales***

  "Construyen la imagen línea a línea realizando un barrido del objeto junto con un desplazamiento longitudinal del mismo" (Infaimon, 2011).

  Requiere de una alta precisión y mayor sincronización para obtener una imagen de alta calidad.

- ***Cámaras matriciales***

  Las cámaras matriciales cubren el área con una matriz de pixeles.

2.2.3. Aforge.net framework

Es un framework opensource diseñado para desarrolladores e investigadores en los campos de la visión por computadora, la inteligencia artificial, el procesamiento de imágenes, redes neuronales, algoritmos genéticos, robótica y máquinas de aprendizaje.

Aforge es un compendio de librerías y aplicaciones de ejemplo que permiten al investigador obtener una vista global de las características que posee el framework.
A continuación un listado de las principales librerías que posee Aforge.net FrameWork.

- ***AForge.Imaging***

  Es una librería que contiene rutinas de procesamiento de imágenes y filtros.

- ***AForge.Vision***

  La librería en cuestión contiene temas relacionados con visión computarizada o visión artificial.

- ***AForge.Neuro***

  La librería es utilizada para la construcción y estudio de redes neuronales.

- ***AForge.Genetic***

  Es una librería enfocada en "evolution programming" y programación genética.

- ***AForge.Fuzzy***

  Es una librería especializada en lógica difusa y computación fuzzy.

- ***AForge.MachineLearning***

  Librería para máquinas de aprendizaje.

- ***Aforge.Robotics***

  Librería que provee soporte para un conjunto de kits de robótica.
- ***AForge.Video***

  Grupo de librerías de procesamiento de videos.

2.2.4. Object Traking. (Seguimiento de objetos)

Es una parte muy importante dentro del campo de la visión computarizada, en los últimos años el desarrollo acelerado de computadores de alto rendimiento, la disponibilidad de alta calidad a bajo costo en cámaras de video de última generación y la creciente necesidad de análisis de video acelerado, han generado un gran interés en el estudio y desarrollo de algoritmos de seguimiento de objetos.

En este proceso existen tres pasos clave a la hora de analizar el video:

- Detección de objetos en movimiento.
- Seguimiento de los objetos detectados en el frame de video.
- Análisis de los objetos seguidos para reconocer su comportamiento.

Por lo expuesto el "Object Traking", es una herramienta útil para monitorear tareas como:

- Movimiento basado en reconocimiento de imagen, es decir, es útil para identificar tareas humanas basándose en el modo de caminar "patrón de movimiento".
- Vigilancia automatizada, se refiere al seguimiento de una escena para detectar actividades poco probables o sospechosas.
- Indexación de video, la anotación para la recuperación automática de video.
- El control de tráfico, a través de recopilación en tiempo real de estadísticas de tráfico desde el flujo del mismo.
- Navegación, se refiere a la planificación de rutas basada en video y evasión de obstáculos en tiempo real.

El seguimiento o rastreo se puede definir como la estimación de la trayectoria de un objeto en una imagen plana cuando se mueve alrededor de una escena, es decir que

asigna etiquetas de seguimiento de conformidad con los objetos localizados en un frame de video.

Para simplificar el estudio del seguimiento de objetos, se lo puede hacer mediante la imposición de restricciones sobre el movimiento y/o apariencia de los objetos. Por ejemplo casi en su mayoría los algoritmos de rastreo parten de la premisa de que el movimiento de los objetos es suave y sin cambios abruptos y de que se puede limitar el movimiento de los objetos a ser de velocidad o aceleración constante, en función de la importancia de la información, previo al conocimiento del número y tamaño de los objetos y la apariencia o forma de los mismos.

2.2.5. Representación de objetos

En un ambiente controlado de seguimiento, un objeto se define como un ente que es de interés para el análisis y seguimiento, por ejemplo en el mar los barcos, autos en la carretera, personas en un almacén, maletas en la central de buses, son un conjunto de objetos que pueden ser de interés para el seguimiento en un ambiente o dominio específico. Los objetos pueden ser representados por:

- ***Puntos***

  El objeto puede ser representado por un punto como lo muestra la Figura 6 Representación de objetos (a), que sería el centro o por un grupo de puntos (b), esta representación es la adecuada para el seguimiento de objetos que ocupen pequeñas regiones del frame de video.

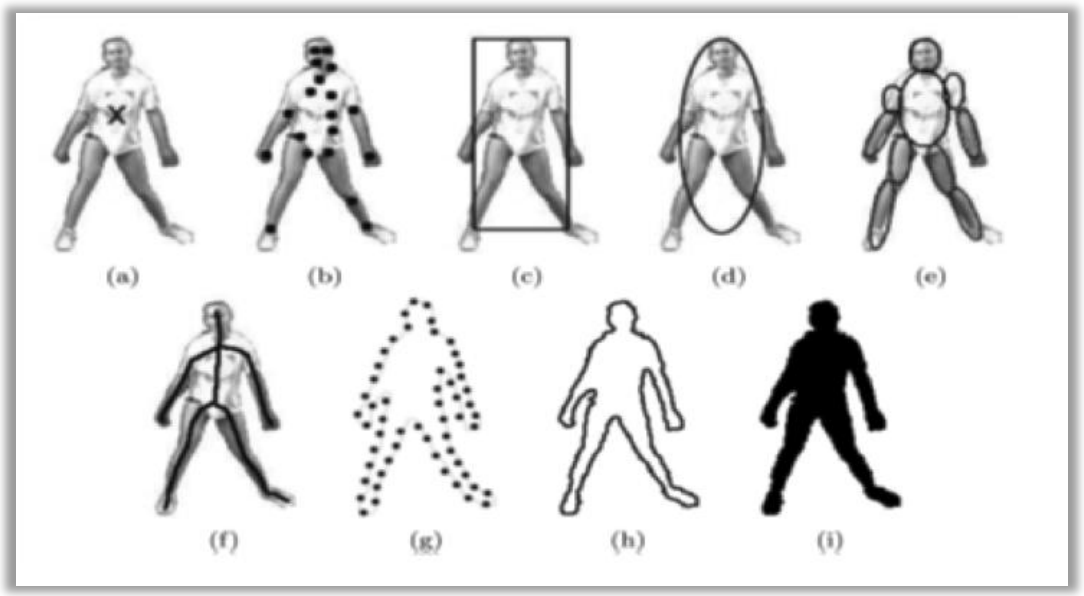
- ***Primitivas formas geométricas***

  La forma total del objeto es representado por un rectángulo como lo muestra la Figura 6 Representación de objetos (c), elipse literal (d), cuadrado o cualquier forma geométrica base. El movimiento del objeto para tal representación es usualmente modelado por este tipo de conversión, puesto que a través de figuras geométricas simples es mucho más sencilla la ubicación y representación de objetos rígidos, aunque también se utiliza para el seguimiento de objetos no rígidos.

- ***Silueta y contorno***

  La representación del contorno define el límite de un objeto como se aprecia en la Figura 6 Representación de objetos (g,h) y la región que se encuentra del contorno como la silueta del objeto literal (i). Estas representaciones son adecuadas para el seguimiento de complejas formas no rígidas.

Figura 6 Representación de objetos



Fuente: Aforge.com

2.2.6. Motion detection

AForge.net provee un conjunto de clases que implementan la detección de movimiento (Motion Detection) y algoritmos de detección de movimiento. Los algoritmos de detección de movimiento están dirigidos solo a la detección de movimiento en fotogramas de video continuo que proporciona la cantidad de movimiento detectado y el marco de movimiento, lo cual muestra todas las regiones del movimiento detectado. Los algoritmos de procesamiento del movimiento tienen como objetivo llevar a cabo el post procesamiento del movimiento detectado en las regiones del realce del movimiento, la contabilidad de los objetos en movimiento y el seguimiento del mismo.

Las diferentes clases [1] que se encargan de la detección del movimiento pueden usar distintos algoritmos de detección de movimiento, pero todas ellas son similares en cómo obtienen las imágenes para el análisis y como estos reportan los niveles de movimiento detectado. Todas estas clases proporcionan la propiedad de nivel de movimiento, la cual muestra el nivel de moviente en rango de [0,1]. Por ejemplo si la propiedad indica 0.05, entonces esto implica que dicha clase detectó un 5% de movimiento, analizando los valores de esta propiedad y comparándolo con el umbral predefinido permite lanzar alarmas cuando se detecta una variación en el movimiento.

El fragmento de código que se muestra en la Figura 7, representa un ejemplo sencillo que demuestra la idea principal de trabajar con detección de movimiento y algoritmos de procesamiento de movimiento.

Figura 7 Seudo algoritmo de ilustración de motion detection

```
// create motion detector
MotionDetector detector = new MotionDetector(
    new SimpleBackgroundModelingDetector( ),
    new MotionAreaHighlighting( ) );

// continuously feed video frames to motion detector
while ( ... )
{
    // process new video frame and check motion level
    if ( detector.ProcessFrame( videoFrame ) > 0.02 )
    {
        // ring alarm or do somethng else
    }
}
```

Fuente: (AForge.NET, 2.2.4)

2.2.7. Algoritmos de detección de movimiento

Aforge.Vision provee un conjunto de algoritmos de detección de movimiento los cuales se describen a continuación:

[1] Clases en programación es la definición de propiedades y comportamiento de un tipo de objeto concreto.

- ***Two frames difference motion detector. (Detección de movimiento por diferencias en entre 2 frames.***

  Este tipo de detector de movimiento es el más simple y rápido de todos, la idea de este detector se basa en encontrar diferencias entre dos frames consiguientes del flujo de video, es decir, mientras mayor sea la diferencia entre los frames mayor es el nivel de movimiento, este tipo de algoritmo se recomienda utilizar en tareas en las cuales sólo se requiere detectar movimiento.

Figura 8 Two frames differnce motion detector

Fuente: (AForge.NET, 2.2.4)

- ***Simple background modeling detector***

  En contraste con el algoritmo anteriormente mencionado, el presente algoritmo está basado en encontrar la diferencia entre un frame de video concurrente y un frame que representa el fondo. Este detector de movimiento trata de utilizar un técnica simple de modelado de escenas de fondo y actualizado a través del tiempo para tener en cuenta los cambios de escena.

  La característica de modelado de fondo de este detector de movimiento permite que la capacidad de resaltado sea mucho más precisa en las regiones de movimiento.

Figura 9 Simple background modeling motion detector

Fuente: (AForge.NET, 2.2.4)

2.3. Procesamiento básico de imágenes digitales

Para realizar un procesamiento de imágenes es necesario relacionar varias áreas entre sí, algunas de esas áreas son las matemáticas, computación y ciertos conocimientos de los diferentes órganos del cuerpo humano como los ojos que intervienen en la percepción de las imágenes (Escalante, 2006).

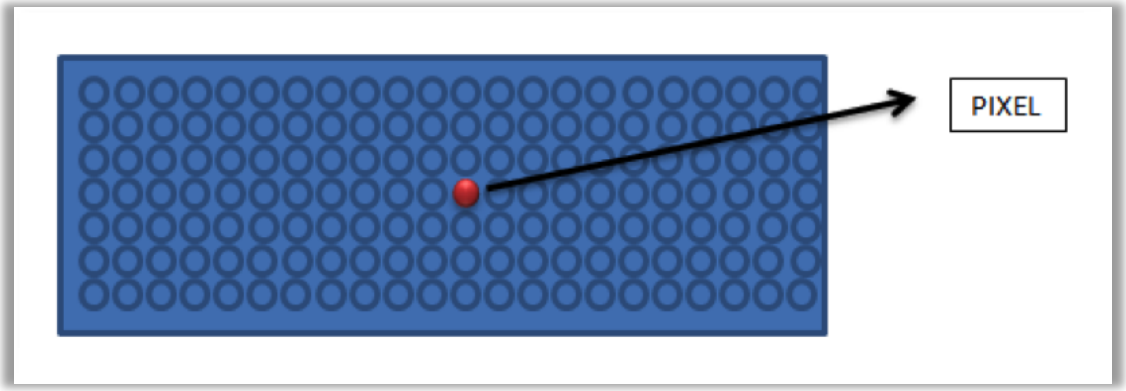
Para mejorar el procesamiento de las imágenes el hombre ha buscado imitar ciertas características del cuerpo humano como un apoyo para la solución de problemas. Algunos de estos avances se ven reflejados en la medicina, la astronomía, geología y microscopía. Un ejemplo más claro de estos avances es la ecografía que se realiza a mujeres embarazadas, el cual ha cambiado notablemente mostrando una mejor definición en sus resultados.

La detección de colores es un factor importante para definir la ubicación de la persona no vidente dentro del ambiente controlado, este factor permite al sistema Lazar-I-Soft detectar, analizar y calcular la ubicación exacta del pixel mientras se traslada de un nodo a otro.

Con la ayuda del framework AFORCE el sistema Lazar-I-Soft, puede agilizar este procesamiento de imágenes sin que este cálculo o detección de colores consuma demasiados recursos de la máquina permitiendo un análisis más rápido y en tiempo real.

Para utilizar el framework es necesario tener algunos conocimientos sobre el procesamiento de imágenes, para poder aplicarlo de mejor manera en el sistema.

Figura 10 Representación de un pixel



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.3.1. Relaciones entre píxeles

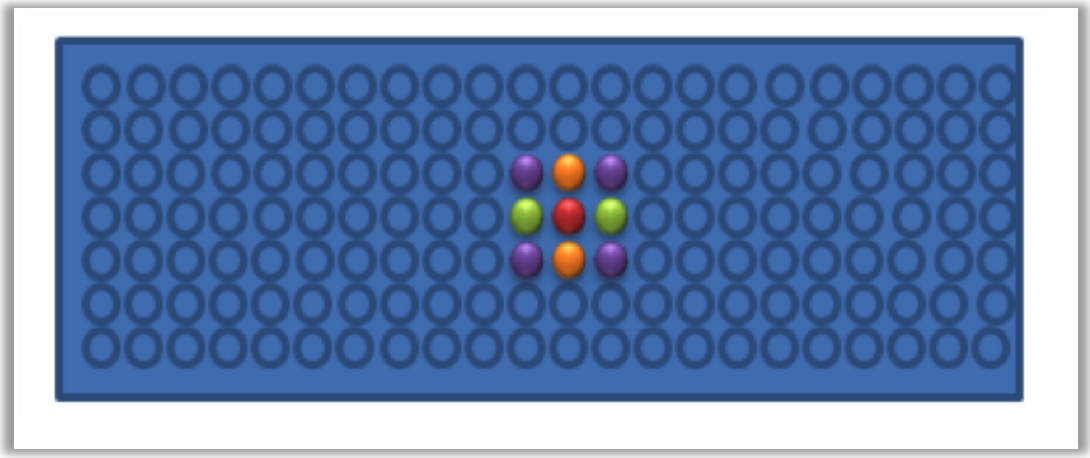
Los cinco tipos de relaciones que existen entre los pixeles son: "vecindad, distancia, contornos, conectividad y adyacencia" (Albizuri, 2010), pero estos tipos de relaciones tienen básicamente las mismas características.

2.3.1.1. Vecindad

La vecindad se basa en tres categorías en función de los pixeles que están a su alrededor.

- Horizontal: Pixeles de la izquierda y derecha. (Color verde)
- Vertical: Pixeles de arriba y abajo. (Color naranja)
- Diagonal: Pixeles faltantes para completar el ruedo de un pixel seleccionado. (Color morado).

Figura 11 Relaciones entre pixeles

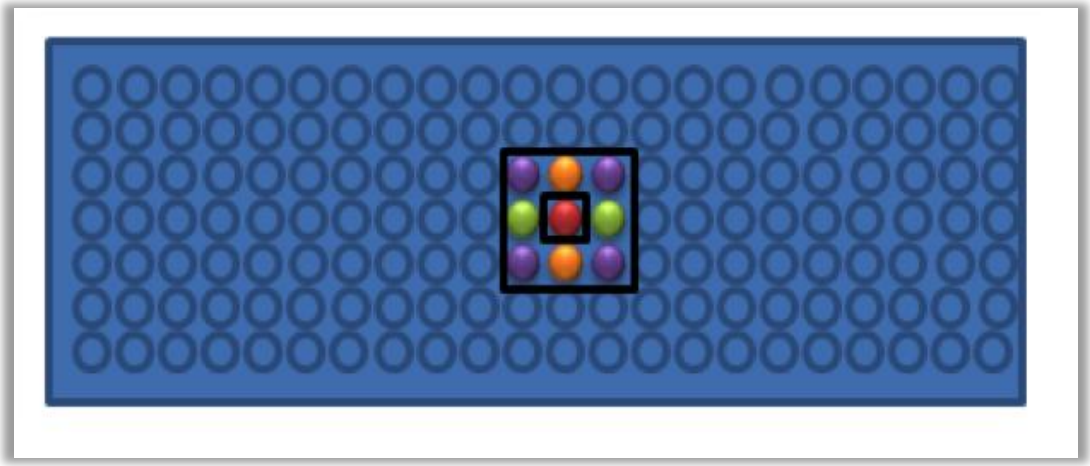


Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.3.1.2. Conectividad

Determina las relaciones entre los pixeles de forma que se pueda establecer límites o áreas de colores a las regiones. Normalmente se usa para determinar los límites de una imagen y poder encontrar los componentes de las áreas.

Figura 12 Conectividad entre pixeles



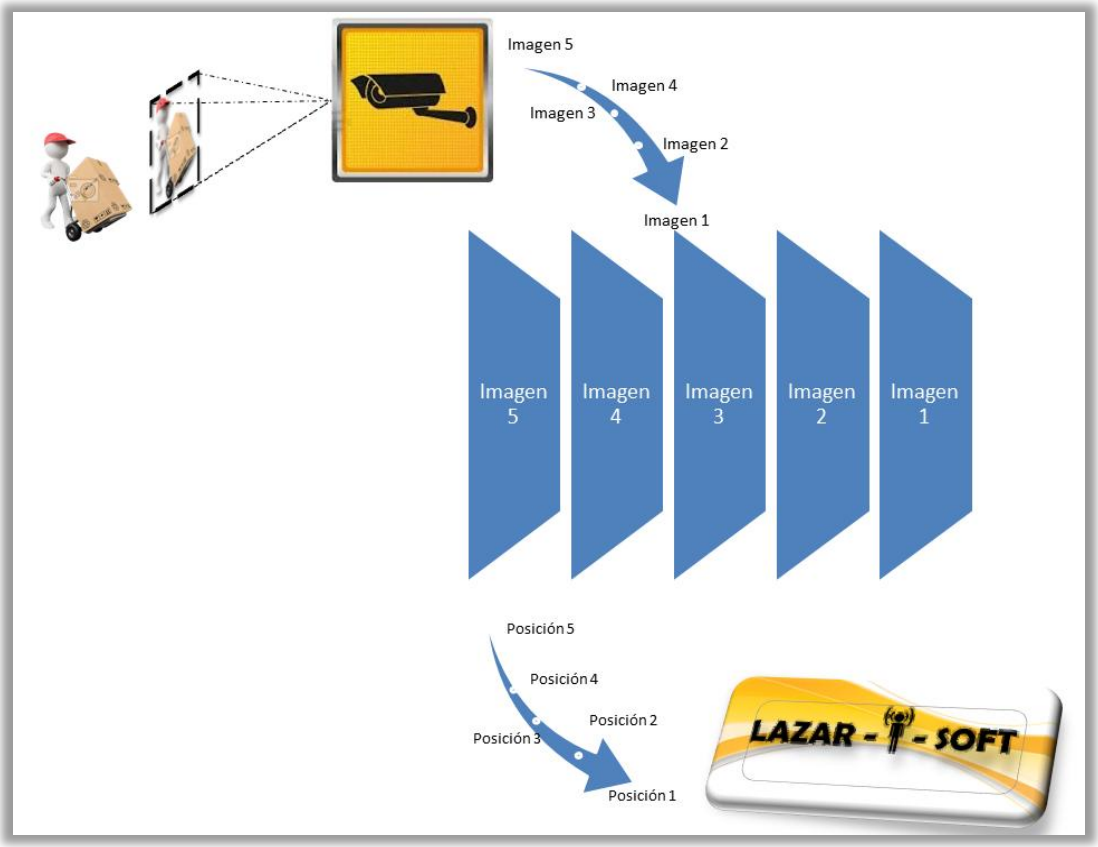
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.3.2. Detección de figuras geométricas

Existen operaciones geométricas que permiten identificar una figura dentro de una imagen digital, estas operaciones analizan un conjunto de valores almacenados en vectores que representan geométricamente una imagen. El sistema Lazar-I-Soft

utiliza el framework AFORCE para realizar estas operaciones geométricas sobre las imágenes capturadas por la cámara de video y de esta manera identificar la ubicación exacta de la persona no vidente dentro del ambiente controlado.

Figura 13 Procesamiento de imágenes



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.3.3. Histograma de una imagen

Es el escaneo de los valores de una imagen almacenados en vectores para obtener los datos de cada uno de sus píxeles representados entre 0 y 255, estos valores indican la densidad de probabilidad en escala de grises, tratando de que sean lo más llanos y separados posibles para que los píxeles se distribuyan ampliamente a lo largo de la imagen, logrando de esta manera resaltar los detalles de la imagen (Atienza Vanacloig, 2010).

Este factor es muy importante para que el sistema Lazar-I-Soft detecte el color rojo configurado dentro del sistema, si este color es alterado se distorsionan los valores

afectando el desempeño del sistema al momento de detectar la ubicación de una persona no vidente dentro del ambiente controlado.

2.3.3.1. Brillo y contraste

Es el aumento o disminución de los valores de los píxeles, dentro del vector que almacenan la representación geométrica de la imagen con el fin de resaltar ciertos colores antes de que sean analizados por el sistema Lazar-I-Soft. Estos valores varían de 0 a 255 donde 0 es igual al color negro y 255 igual al color Blanco.

Figura 14 Brillo y contraste

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.3.3.2. Reconocimiento de objetos

Es el proceso que identifica un objeto dentro de una escena determinada.

Para realizar el reconocimiento automático de un objeto es necesario identificar las características discriminantes o rasgos únicos del mismo, estos rasgos permiten localizar un objeto dentro del universo desconocido o imagen, actualmente existen varios algoritmos que permiten realizar estos cálculos utilizando:

- Detección de formas
- Análisis geométricos
- Comparación de patrones
- Mediciones
- Detección de objetos difusos, etc.

La intensidad y ubicación de la fuente de luz también son factores que ayudan a discriminar ciertos valores dentro del ambiente controlado, ya que el color emitido por la fuente de luz no puede estar en dos lugares al mismo tiempo y tampoco puede

alejarse de forma esporádica hacia otro punto; esta intensidad y luminosidad ayudan a detectar de mejor manera la ubicación de la persona no vidente mientras se traslada de un nodo a otro.

Figura 15 Iluminación del ambiente

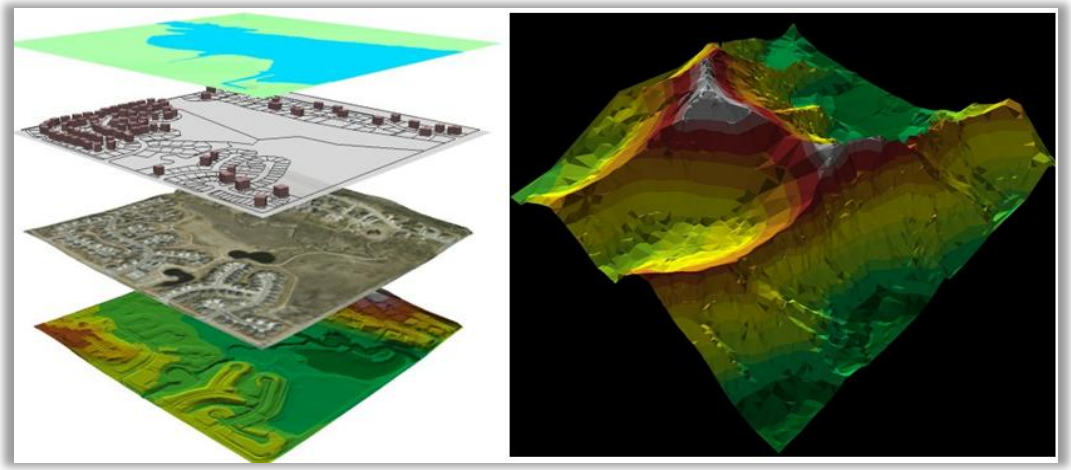
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

2.3.3.3. Reconocimiento de profundidad

Una imagen digital almacena mucha información del ambiente capturado, esta información puede ser interpretada o analizada de diferentes maneras para encontrar la solución a un problema.

Normalmente este tipo de análisis es utilizado para generar mapas topográficos, como el sistema Quantum GIS. (GIS)

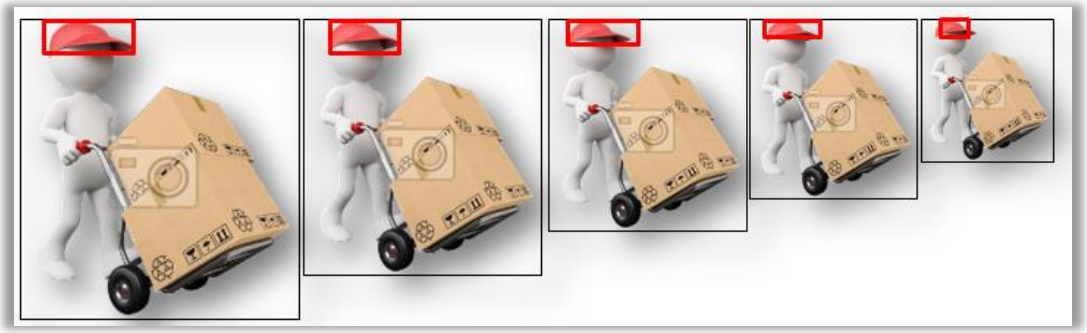
Figura 16 Reconocimiento de profundidad



Fuente: Gestor de Archivos Geo referenciados

Los colores y la intensidad de los mismos ayudan a definir una distancia entre el objeto y la fuente de luz, esta representación de colores es receptada por el sensor y emitida al sistema Lazar-I-Soft quien analiza estos datos para definir el movimiento de la persona no vidente dentro del ambiente controlado.

Figura 17 Distancia del objeto



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

CAPÍTULO 3
ALGORITMOS DE INTELIGENCIA ARTIFICIAL

3.1. Introducción

A través de los tiempos la evolución constante que ha tenido el hombre desde la aparición de los primeros homínidos hasta el advenimiento del homo sapiens, ha planteado una interrogante muy compleja y atractiva a la vez, el entender: ¿Cómo se piensa?, es decir, el vislumbrar cómo un cúmulo de materia encefálica puede comprender, percibir, vaticinar y manipular el mundo de su alrededor que indudablemente la supera, así misma en complejidad y dimensión. Es por ello que los esfuerzos del ser humano van más allá, tratando de no solo descifrar, sino también emular la misma y así poder crear entidades inteligentes.

La mayor parte de autores de libros de inteligencia artificial como: "Inteligencia Artificial, un enfoque moderno" de Stuart J. Russell y Peter Norvig, están de acuerdo que esta ciencia relativamente joven tiene sus orígenes en la Segunda Guerra Mundial y que fue recopilada y difundida a mediados de la década de los cincuenta del siglo pasado. La inteligencia artificial junto con otras ciencias como la biología molecular, pertenecen a un grupo selecto dentro de la rama de la investigación; muy apetecido por científicos de todas las disciplinas, esto se debe a que otras ciencias como la física y matemática poseen la mayoría de sus leyes ya establecidas, la inteligencia artificial ofrece una infinidad de huecos sin cerrar en los que se podría investigar.

Hoy por hoy la inteligencia artificial, se ha convertido en un gran contenedor de sub campos, que van desde áreas muy generales, como el aprendizaje y la percepción hasta otras muy específicas como la demostración de teoremas matemáticos, diagnósticos, análisis entre otros. La inteligencia artificial esquematiza y automatiza actividades intelectuales siendo potencialmente relevante para cualquier rama intelectual del ser humano, resumida en pocas palabras es un campo versátil difícilmente de emular.

3.2. Inteligencia artificial IA

3.2.1. Definición

La introducción a la inteligencia artificial hace gala de la majestuosidad de este campo de la investigación humana, pero aún no se ha revelado una de las interrogantes más importantes de este campo: ¿Qué es? Por lo tanto en la Figura 18 Matriz de definiciones de IA, se hace referencia a las definiciones recopiladas de ocho distintos libros de textos que mantiene concordancia con este tema de estudio.

Las primeras que se encuentran en la parte superior de la Figura 18 hacen referencia a los procesos mentales y razonamientos del ser humano, mientras que los que se encuentran en la parte inferior aluden a la conducta del mismo, a su vez en la parte izquierda se mide el éxito en función de la fidelidad tomando en cuenta la forma de actuar de los seres humanos, mientras que en la derecha tienen como estandarte un concepto ideal de inteligencia, denominado racionalidad o la acción de realizar lo correcto, en función del conocimiento que se posee.

Estos cuatro enfoques planteados en la Figura 18 Matriz de definiciones de IA, muestran un enfrentamiento entre los enfoques centralizados del ser humano y los que giran en torno a su racionalidad, puesto que la perspectiva centralizada del comportamiento humano es más empírica que racional. Este comportamiento presenta hipótesis y confirmación por la vía de la experimentación mientras que la racional implica modelos matemáticos e ingeniería.

Figura 18 Matriz de definiciones de IA

Sistemas que piensan como humanos	Sistemas que piensan racionalmente
«El nuevo y excitante esfuerzo de hacer que los computadores piensen… máquinas con mentes, en el más amplio sentido literal». (Haugeland, 1985) «[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje…» (Bellman, 1978)	«El estudio de las facultades mentales mediante el uso de modelos computacionales». (Charniak y McDermott, 1985) «El estudio de los cálculos que hacen posible percibir, razonar y actuar». (Winston, 1992)
Sistemas que actúan como humanos	**Sistemas que actúan racionalmente**
«El arte de desarrollar máquinas con capacidad para realizar funciones que cuando son realizadas por personas requieren de inteligencia». (Kurzweil, 1990) «El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor». (Rich y Knight, 1991)	«La Inteligencia Computacional es el estudio del diseño de agentes inteligentes». (Poole *et al.*, 1998) «IA… está relacionada con conductas inteligentes en artefactos». (Nilsson, 1998)

Fuente: Inteligencia Artificial, un enfoque moderno.

3.2.2. Aprendizaje

El aprendizaje puede verse desde diferentes puntos, pero para los fines investigativos de este documento, se toma el aprendizaje convencional y el aprendizaje de una máquina, por los siguientes motivos:

- ***Aprendizaje convencional***

  "Cuando los organismos se ajustan o adaptan al conjunto de estímulos que provienen del entorno; reciben la información y la almacena con el fin de reutilizarla en situaciones o patrones de estímulos semejantes." (Pesquera, 2010)

- ***Máquina que aprende***

  "Sistema organizado que transforma un mensaje de entrada en una salida, de acuerdo con un principio de transformación. Si tal principio está sujeto a cierto criterio de validez y el método de transformación se ajusta a fin de que tienda a mejorar el funcionamiento; se dice que el sistema aprende." (Pesquera, 2010)

El aprendizaje en la inteligencia artificial puede representarse como un proceso que recibe un parámetro X como mensaje de entrada y lo transforma en un Y como parámetro de salida, por medio de una función de transformación que de estar sujeta algún criterio de validación será alterada para obtener un mejor resultado, emulando el aprendizaje convencional en cuanto a la adaptabilidad.

La mejor manera de comprender esto es con una simple analogía: Cuando una persona intenta cruzar la calle ejecuta una serie de pasos consecutivos de manera intuitiva; primero recopila la información al mirar hacia ambos lados introduciendo de esta manera las percepciones a su "programa agente", mismo que realiza una elección racional gracias a la "función del agente" que serían las experiencias a la fecha, para poder tomar la decisión, lo que implica que no solo el agente recopila información, si no que deba aprender lo máximo que pueda; de las percepciones que recibe del entorno.

En algunos casos muy particulares el agente cuenta con el conocimiento necesario del entorno a priori, provocando que el mismo no necesite percibir ni aprender alguna información adicional del entorno, ya que simplemente actúa de manera correcta. Este tipo especial de agentes son muy frágiles puesto que funcionan en ambientes idealizados, por lo tanto un agente de éxito debe ser aquel que esté atento al cambio, lo que se plantea para tal fin, es la división del cálculo de las tareas del agente en tres espacios:

- Cuando se está diseñando el agente.
- Cuando se está pensando en la siguiente operación.
- Cuando se está aprendiendo de la experiencia y se lleva a cabo más cálculos para modificar su comportamiento.

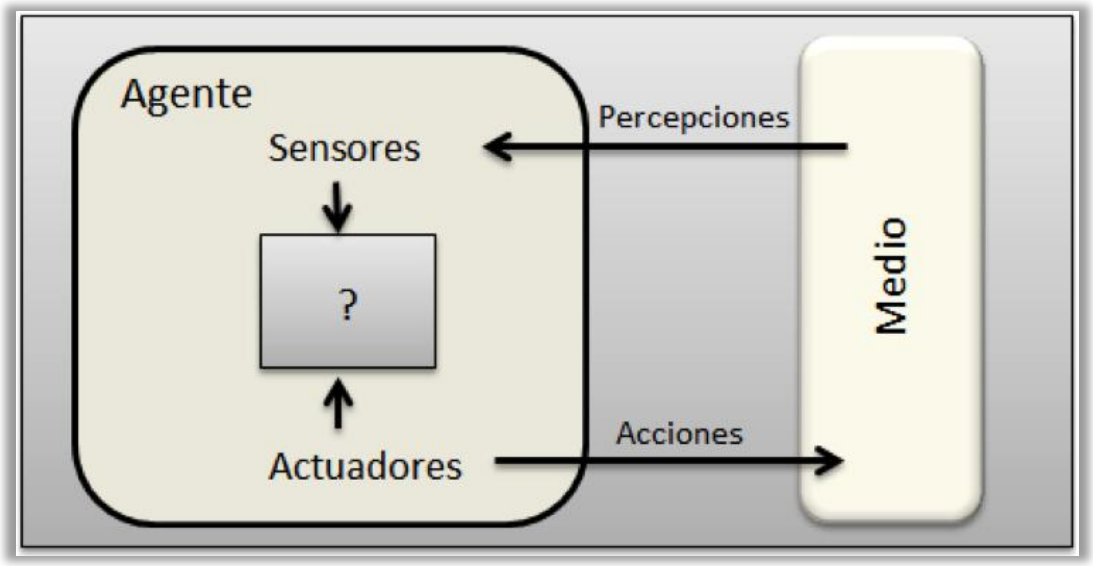
3.2.3. Agentes inteligentes

Para comenzar a hablar sobre los agentes inteligentes es necesario aclarar ciertos temas sobre los agentes racionales.

Los agentes son entes de razonamiento que simplemente ejecutan tareas, pero los agentes informáticos o agentes racionales diseñados con la finalidad de alcanzar el mejor resultado o el mejor esperado dentro de un ambiente de incertidumbre, normalmente esperan tener un conjunto de atributos que los diferencie de los programas convencionales, tales como: controles autónomos de persistencia que se ejecuten en un lapso de tiempo prolongado, controles de percepción del ambiente que alcancen objetivos diferentes y por último adaptación a cambios.

Desde el enfoque de la inteligencia artificial al tratarse de agentes racionales todo el énfasis está dirigido a realizar las correctas inferencias, puesto que una manera racional de actuar o llegar a la conclusión lógica de una determinada acción es alcanzar el objetivo deseado. Pero para realizar dicha acción, un agente debe ser capaz de percibir el entorno en el que se encuentra con ayuda de sensores para interactuar con los actuadores[2] de un medio predeterminado.

Figura 19 Interacción simple de agente con el medio ambiente



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

La Figura 19 Interacción simple de agente con el medio ambiente, muestra de manera gráfica cómo interactúan un agente y el entorno o medio que lo rodea; para lo cual se puede realizar la siguiente analogía: Un agente humano posee órganos sensoriales

[2]Actuadores: Se usa este término para indicar el elemento que reacciona a un estímulo realizando o una acción.

como ojos, manos, oídos, piernas y boca, que le permiten interactuar con el medio ambiente, visto de esta manera un agente inteligente puede recibir pulsaciones desde un teclado, archivo de datos, paquete de red, datos provenientes de una cámara de video, etc. a manera de entradas sensoriales y al igual que un agente humano, el agente inteligente puede actuar por medio de ficheros enviando paquetes de datos, salidas visuales, salidas sonoras, pulsaciones, etc.

Matemáticamente un agente inteligente se lo puede definir de la siguiente manera: "Se permite decir que el comportamiento del agente viene dado por la función del agente que proyecta una percepción dada en una acción determinada". (Russell & Norvig, 1996)

3.2.4. Estructura de un agente

La estructura básica de un agente está dada por dos componentes base, estos componentes son el programa del agente más la arquitectura del mismo; es decir, que el programa del agente que implementa la función tendrá algún tipo de computador base con sensores físicos y actuadores.

La arquitectura debe ser apropiada para el programa agente y el mismo apropiado para la arquitectura de tal manera que si el agente recomienda acciones como instrucciones para caminar, la arquitectura debe tener bocinas para comunicar esta acción. Y si la arquitectura plantea la recolección de datos del entorno por medio de una cámara, el programa debe tener un módulo de procesamiento de estos datos. En otras palabras, todo debe funcionar como engranes que embonan unas con otras para poder ejecutar la función del agente.

Los programas agentes se definen como aquellos entes que reciben o perciben datos actuales desde los sensores y comunican una determinada acción a los actuadores, marcando la diferencia con la función agente al solo recibir los datos actuales, más no el histórico de estos datos.

Entre los principales programas para agentes se encuentran:

- Agentes reactivos simples.
- Agentes reactivos basados en modelos.
- Agentes reactivos basados en objetos.
- Agentes basados en la utilidad.

3.3. Algoritmos de inteligencia artificial

3.3.1. Definición

Un algoritmo es un conjunto de instrucciones, comandos o parámetros, claramente especificados que deben seguirse para resolver un problema específico.

- Un algoritmo debe ser secuencial y preciso que indique el orden de realización paso a paso.
- Un algoritmo debe estar definido. Ya que el algoritmo ejecutado N número de veces, debe dar el mismo resultado.
- Un algoritmo debe ser finito o en otras palabras debe tener un principio y un fin con un número finito de pasos.

La definición de un algoritmo se puede describir en tres partes: entrada, proceso y salida, siendo el caso del sistema Lazar-I-Soft que tiene como objetivo el de enrutar a la persona no vidente a su objetivo, la entrada seria el posicionamiento inicial, el proceso los cálculos y pasos ejecutados por el sistema de enrutamiento y posicionamiento y la salida las instrucciones emitidas por el sistema hacia el usuario.

3.3.2. Primero el mejor

Es una estrategia de control exhaustiva en amplitud y profundidad, su principal diferencia con algoritmos de búsqueda informada como A*, es que el primer nodo que encuentre es el mejor, para obtener los nodos de una ruta. En este caso se elige el primer nodo que al parecer es el mejor sin tener en cuenta su posición en el árbol es decir, se expande el nodo más prometedor para llegar a la solución.

Los tipos de nodo que utiliza el algoritmo primero el mejor se describen a continuación:

- ***Nodos abiertos***

  Contiene los nodos que han sido generados a través de la función heurística que genera una COLA DE PRIORIDADES, en la que los elementos con mayor prioridad son los que tienen los valores más prometedores, (el mejor camino o el nodo más cercano).

- ***Nodos cerrados***

  Contiene los nodos que han sido examinados, para que la búsqueda sea en forma de grafo y no en forma de árbol, costo, distancia o un valor que represente prioridad al problema, este valor ayuda a buscar primero por los senderos que son o parecen más prometedores para llegar al punto final.

3.3.3. Búsqueda de grafos

Es un conjunto de nodos unidos en una red o los nodos de un camino que existen dentro de un ambiente controlado como intersecciones, curvas, lugares estratégicos o puntos, en los cuales el sistema necesite realizar una parada antes de tomar una decisión.

Al trasladarse dentro de este ambiente controlado existe una definición que se repite constantemente mientras el no vidente se traslada de un nodo a otro; siempre al nodo anterior se le define como predecesor y al nodo destino como sucesor. Adicional a esto existe un coste vinculado al desplazamiento entre nodos, esto ayuda a la evaluación antes del desplazamiento.

Un algoritmo de búsqueda siempre trata de encontrar el camino más prometedor u óptimo entre los nodos. Una de las principales diferencias entre los algoritmos de búsqueda es la información que guardan acerca del ambiente o grafo, normalmente esta información es el coste de viajar desde el origen hasta el nodo destino, incluso una estimación de lo prometedor que es un nodo para conducir al objetivo. Algunos algoritmos no guardan información sobre sus ambientes simplemente expanden su búsqueda desde el nodo inicial, hasta el nodo final.

La expansión de la búsqueda se realiza en forma de árbol. Partiendo del nodo inicial a los nodos vecinos y trasladándose entre cada uno de ellos hasta que uno de los nodos sea el nodo objetivo.

Esta definición ayuda a elegir una ruta lógica entre los nodos del ambiente controlado para llegar al nodo objetivo o nodo final, esta ruta guía a la persona no vidente dentro del ambiente para que se traslade de un nodo a otro utilizando la información precargada de la ruta. Es imposible que una persona no vidente se salte un punto o cruce una pared para llegar a su objetivo.

3.3.4. A asterisco

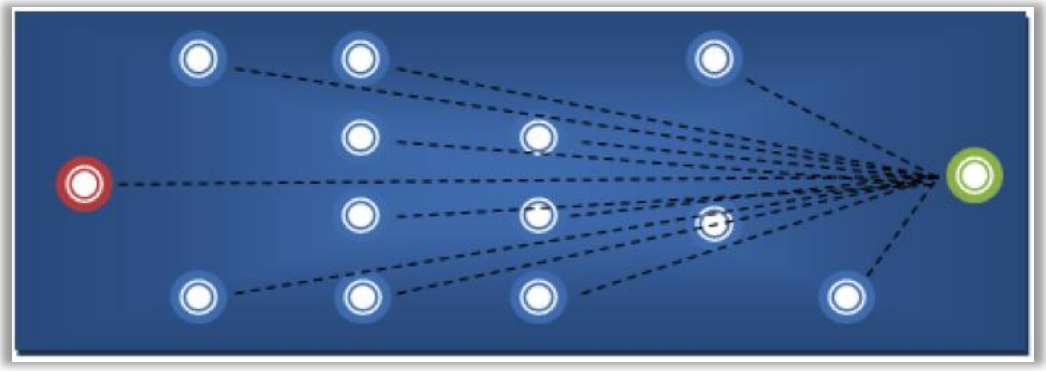
Está clasificado dentro de los algoritmos de búsqueda en grafos. Presentado en 1968 por Peter E. Hart, Nils J. Nilsson y Bertram Raphael, este algoritmo encuentra su respuesta, siempre y cuando se cumplan unas determinadas condiciones o funciones establecidas; esta evaluación se basa en la matriz de información que contiene la ubicación de cada uno de los nodos y el coste que poseen entre cada uno de ellos, así como también la relación que existe entre los mismos.

Para muchas aplicaciones, es conveniente definir la función f, como la suma de dos funciones (g y h).

$$f = g + h$$

La función g es una medida del costo de llegar desde el nodo inicial al nodo actual.

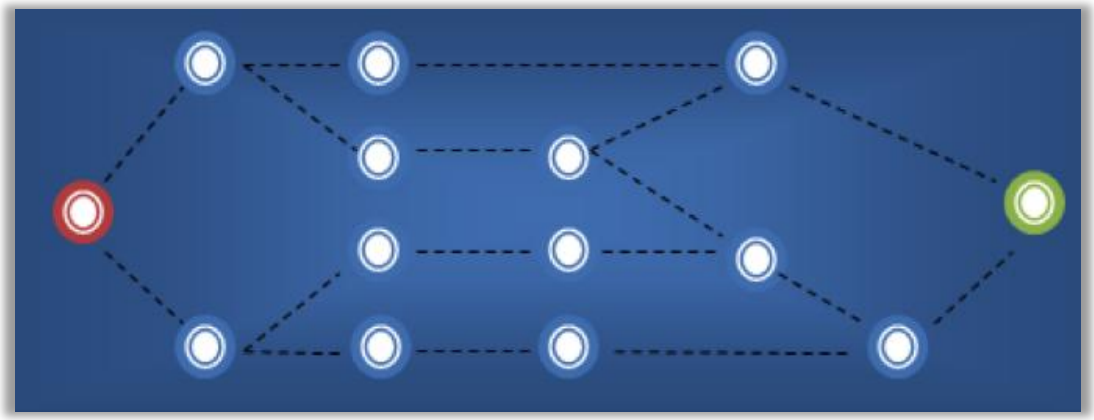
Figura 20 Definición de G' (Costo por distancia en línea recta)



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

La función h es una estimación del costo adicional para llegar desde el nodo actual al estado objetivo.

Figura 21 Definición de H' (Costo por distancia)



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

La definición de estos literales ayudan a la representación lógica del ambiente controlado. Para que el sistema Lazar-I-Soft defina cuál es la ruta más prometedora a seguir el no vidente dentro del ambiente controlado.

Como resultado el sistema Lazar-I-Soft siempre despliega el camino con menor coste entre el nodo origen y el nodo objetivo. Este algoritmo es el seleccionado para el objetivo de esta tesis, ya que contempla en conjunto los literales 3.3.2 y 3.3.3.

3.4. Lisp

3.4.1. Definición

Lisp es uno de los primeros lenguajes de programación de computadora de tipo multiparadigma con una larga historia y una sintaxis completamente entre paréntesis. Su creador John McCarthy lo presentó por primera vez en 1958.

Lisp es el segundo más viejo lenguaje de programación de alto nivel y de extenso uso hoy en día; solamente Fortran es más viejo que lisp.

Lisp ha cambiado en muchos dialectos desde sus comienzos, los más ampliamente conocidos son: CommonLisp y Scheme.

Originalmente lisp fue creado como una notación matemática práctica para los programas de computadora, basada en el cálculo lambda [3], en la actualidad es utilizada como lenguaje de programación por excelencia para la investigación de la inteligencia artificial. Además de ser la precursora de varios paradigmas de la computación como las estructuras de datos en árbol, el manejo de almacenamiento automático, tipos dinámicos y el compilador auto contenido.

3.4.2. Recursividad

Un algoritmo recursivo consta de una parte recursiva y una condicional o iterativa.
La parte recursiva siempre necesita de un condicional para su finalización caso contrario el programa jamás parará y se generará un ciclo infinito.
En cambio una condicional no necesariamente existe dentro de un algoritmo recursivo.

Existen dos tipos de recursividades:

- Recursividad directa: cuando un subprograma se llama a sí mismo una o más veces directamente.
- Recursividad indirecta: cuando se definen una serie de subprogramas llamándose unos a otros.

3.4.3. Csharp

Csharp (C#), es un lenguaje de programación orientado a objetos desarrollado por Microsoft, está implementado en la plataforma. NET, es un lenguaje de programación diseñado para la infraestructura de lenguaje común. Su sintaxis básica deriva de C/C++ y es similar a Java, aunque incluye mejoras derivadas de otros lenguajes.

[3] El cálculo lambda es un sistema formal diseñado para investigar la definición de función.

3.4.4. RDNZL y L-Sharp

RDNZL es una capa o etiqueta de common lisp, que permite a las aplicaciones de Lisp interactuar con bibliotecas .NET, siendo una interface de funciones externas para. NET como C # construida sobre de la interfaz C externa.

RDNZL viene con una licencia tipo BSD por lo que básicamente se puede hacer con ella varios aplicativos. (BSD es compatible con GPL; pero no es considerada libre por la Free Software Fundation).

Lamentablemente no existen manuales para la utilización o programación de los diferentes comandos que vienen pre cargados en el RDNZL; para lo cual se recomienda tener un conocimiento básico de los comandos lisp, ya que estos ayudarán a la interpretación de los mismos. La programación que se realiza en el RDNZL es de tipo L-csharp y se recomienda analizar el código antes de utilizar este lenguaje de programación, Además no consta con funciones complejas o de simplificación de algún tipo, no discrimina mayúsculas de minúsculas y la programación se realiza en consola (console aplication) de Csharp.

Nota: Para poder aplicar el RDNZL en el aplicativo Lazar-I-Soft fue necesario cambiar la compilación del RDNZL original cambiándola de Consola a Windows Forms, este cambio implicó la adaptación de librerías y funciones del RDNZL. Una vez terminado este cambio las funciones del L-csharp se mantienen en memoria hasta la finalización del sistema Lazar-I-Soft.

CAPÍTULO 4
RECONOCIMIENTO DE VOZ

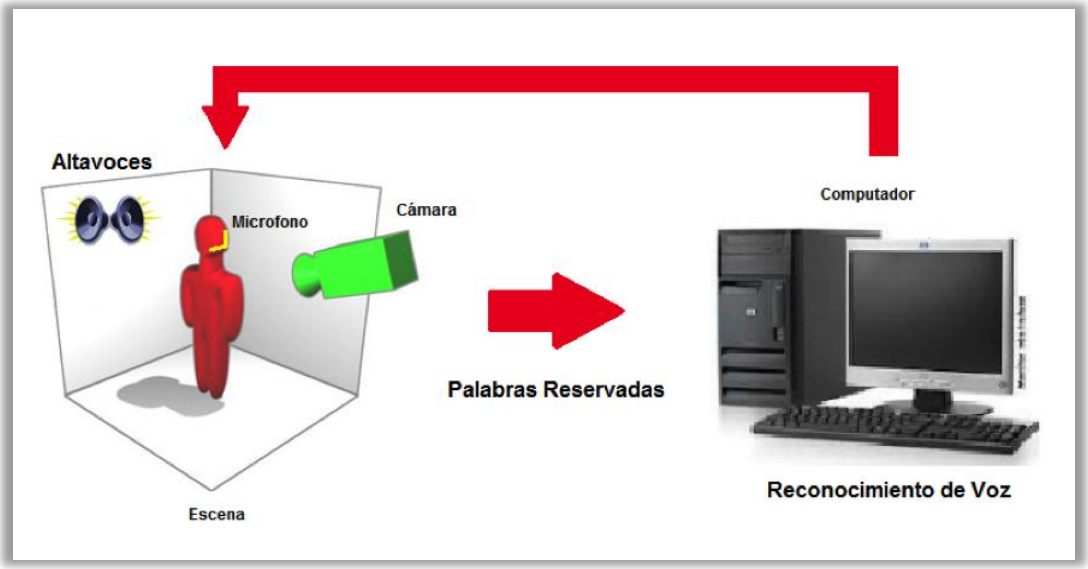
4.1. Introducción

En los últimos años se ha visto la necesidad de crear software que facilite la utilización de las máquinas a personas con discapacidades, una de ellas llamada Reconocimiento de Voz, esta herramienta permite ejecutar comandos para realizar una orden específica en el computador logrando así la interacción entre la máquina y el usuario sin la necesidad de utilizar algún tipo de dispositivo externo como un mouse o un teclado.

Microsoft ha incorporado en sus sistemas operativos una herramienta Reconocimiento de Voz, la cual después de ejecutar un entrenamiento al sistema, es capaz de reconocer comandos vocales generados por una persona, a través de un simple micrófono, una persona puede indicar comandos para manipular aplicaciones, logrando así la navegación por todo el sistema operativo.

El sistema Lazat-I-Soft está conformado por varios módulos, uno de los cuales es el módulo de Reconocimiento de Voz, éste permite a la persona no vidente comunicarse con el computador a través palabras reservadas, mediante un procedimiento de identificación de palabras, el sistema genera instrucciones las cuales indican a la persona no vidente la ruta por la cual debe seguir, logrando una comunicación entre la máquina y el usuario.

Figura 22 Sistema de reconocimiento de voz



Elaborado por: Cristhian Miranda, Edison Romero, Danilo Taco,

4.2. Reconocimiento de voz

Denominado en Ingles Speech o Voice Recognition, es la capacidad de recibir información a través de un micrófono para luego procesarla (generalmente convertir la voz de una persona en binarios utilizando algoritmos conocidos) y transformarlas en comandos reconocibles por un computador para realizar una acción específica.

Existen varias técnicas que se pueden emplear para el reconocimiento de voz, las más conocidas y utilizadas son: Comparación de Patrones, Modelos Ocultos de Markov, Redes Neurales o Neuronales.

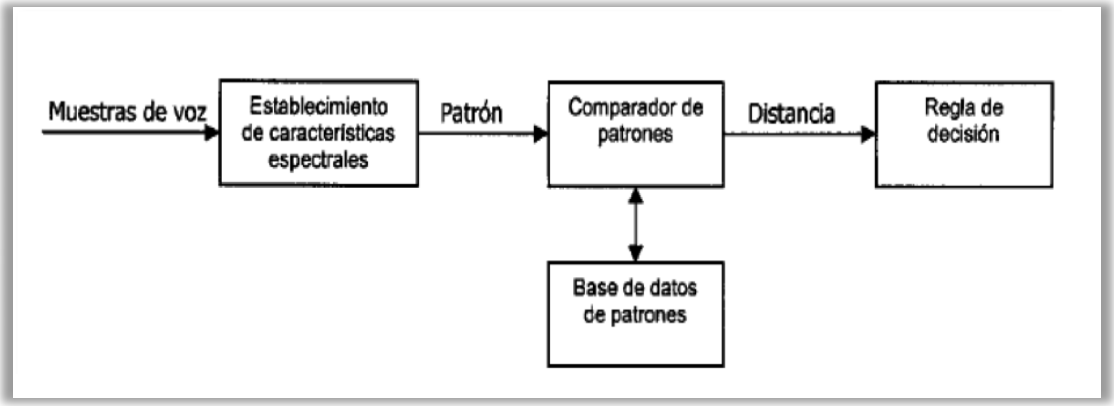
4.2.1. Comparación de patrones

La comparación de patrones consiste en realizar una relación y comparación con un conjunto de plantillas[4], las cuales contienen las unidades a reconocer. Esta técnica utiliza el algoritmo Dynamic Time Warp (DTW) el cual es capaz de comparar similitudes entre dos secuencias de audio. Debido a que utiliza plantillas para la comparación de los patrones, esta técnica puede generar problemas por escasas palabras en la base de datos de patrones.

[4] La plantilla no es más que un conjunto de características acústicas ordenadas en el tiempo (secuencia de vectores de parámetros o índices de una librería de centroides o codebook).

Otra desventaja de esta técnica es que las palabras que van a ser comparadas nunca se pronuncian a la misma velocidad lo que puede provocar distorsiones en la comparación.

Figura 23 Comparación de patrones



Fuente: Comparación de Patrones

4.2.2. Modelos ocultos de Markov (HMM)

Esta es una de las técnicas más utilizadas en los últimos tiempos, debido a que realiza la comparación de los patrones con una gran cantidad de datos, utilizado para el reconocimiento de lenguaje natural y continuo.

Siendo un modelo estadístico la principal función es encontrar parámetros ocultos de una cadena, una vez que han sido escogidos los parámetros estos servirán para realizar el siguiente análisis, así secuencialmente hasta generar la posible salida.

4.2.3. Redes neuronales

Es una de las técnicas más utilizadas actualmente, debido a que genera buenos resultados al momento de realizar el reconocimiento de palabras. Las redes neuronales han sido utilizadas para aplicaciones de reconocimiento de voz tomando patrones de comparación; las características de un fonema.

A igual que todas las redes neuronales requieren de entrenamiento, debido a su gran robustez puede procesar el reconocimiento de una forma mucho más rápida que las anteriores técnicas.

4.3. Lenguaje

El lenguaje es una de las partes más importantes de un sistema de reconocimiento de voz, ya que gracias a este se logra realizar el correcto reconocimiento.

La pronunciación de las palabras es una sección fundamental en los sistemas que utilizan reconocimiento de voz, se debe pronunciar correctamente las palabras y sílabas que intervengan en la aplicación, ya que los algoritmos son muy sensibles en el reconocimiento, estos podrán confundir fácilmente la pronunciación como por ejemplo la pronunciación de un SI con la pronunciación de un SIP.

Dependiendo del sistema que se esté empleando, el vocabulario afecta en este; por ejemplo para un sistema como Lazar-I-Soft se requiere poco vocabulario y palabras exactas, pero para un sistema como un diccionario se requiere de un vocabulario extenso ya que al no poseerlo puede generar incoherencias en el reconocimiento.

4.4. Base de datos

Lazar-I-Soft utiliza una base de datos en la cual se almacena un banco de palabras que permite el correcto funcionamiento del sistema. Las palabras que se almacenan en la base de datos, son comandos que permiten identificar acciones las cuales debe realizar la persona.

4.4.1. Palabras reservadas

Estas palabras permiten la comunicación con la persona no vidente indicando la ruta por la cual debe seguir hasta llegar a un punto específico.

Mediante las herramientas Microsoft Agent y Microsoft Speech API se generan los sonidos que representan cada una de las palabras las cuales son: Adelante, Girar a la izquierda, Girar a la derecha, Detenerse, entre otras.

En los siguientes apartados se detalla a mayor profundidad el procedimiento que se utiliza para pronunciar las palabras utilizando las herramientas mencionadas.

4.5. Ruido

Desde el enfoque del Reconocimiento de Voz el ruido "es una señal acústica o eléctrica que contamina la señal de voz que se requiere reconocer". (El reconocedor de habla natural)

4.5.1. Definición

El ruido se define como "todo sonido no deseado que interfiere en la comunicación entre las personas o en sus actividades". (Academic, 2012)

Como se indicó en el apartado anterior, el punto más débil de un sistema de reconocimiento de voz es el ruido que puede provocar cualquier tipo de ente externo a la escena en la cual se encuentra el sistema.

Para que un sistema de reconocimiento de voz sea procesado correctamente se debe tomar en cuenta el entorno en el cual se está ejecutando éste, debido a que cualquier sonido involuntario puede provocar confusiones en el reconocimiento.

4.6. Microsoft Agent

Desarrollada por Microsoft, es una tecnología que incorpora distintos personajes animados, módulos de transformación de texto a voz, y reconocimiento de voz, con el fin de realizar una interacción entre el usuario y la máquina.

Microsoft Agent tiene la capacidad de generar una salida de voz, todo gracias al motor de voz Microsoft Speech API (SAPI), la cual es distinta en cada uno de sus personajes siendo estos: Peedy, Merlín, Genie, y Robby gratuitos de Microsoft.

Microsoft Agent alcanzó la popularidad al ser introducido en programas como Microsoft Office, en el cual ofrece tipos de ayuda, búsquedas instantáneas, etc.

Figura 24 Personaje de la herramienta Microsoft Agent

Fuente: (Microsoft, 2009)

4.6.1. Microsoft Speech API

Es una herramienta que permite el desarrollo de aplicaciones relacionadas con el habla como reconocimiento de voz. Esta herramienta consta de varios API's los cuales se detallan a continuación:

- ***Voice Command API Referencia***

  Permite la ejecución de comandos en aplicaciones Windows.
- ***Direct SpeechRecognition API Referencia***

  API que permite el reconocimiento de habla en aplicaciones Windows.
- ***Voice Text API Referencia***

  Esta herramienta permite incorporar el texto-hablado en aplicaciones Windows.
- ***Voice Dictation API Referencia***

  Genera dictado por voz en diferentes aplicaciones.

4.6.2. Funcionalidades

Utilizado principalmente en aplicaciones de Microsoft Office (hasta versiones de Windows XP), la herramienta Microsoft Agent es utilizada para generar ayudas y búsquedas personalizadas.

Lazar-I-Soft utiliza Microsoft Agent para generar la comunicación que existe entre la máquina y usuario, indicando la guía por la cual la persona no vidente debe seguir para llegar a un punto fijo. Mediante un procedimiento de programación el personaje

seleccionado (Peddy) indica palabras reservadas como: SIGA, ADELANTE, GIRE A LA DERECHA, GIRE A LA IZQUIERDA, DETENGASE, entre otras.

4.6.3. Características

Tecnología utilizada en aplicaciones desarrolladas en plataformas de Visual Basic, Visual Studio para escritorio, y para aplicaciones Web desarrollas en plataformas VBScript.

Provee distintos personajes, cada uno de ellos con diferentes características, ofreciendo así diferentes animaciones y diferentes interacciones.

"El motor de voz en sí mismo es impulsado por Speech API Microsoft (SAPI), en la versión 4 o superior. Microsoft SAPI ofrece un panel de control fácil de instalar y cambiar." (Microsoft, 2009)

4.6.4. Características técnicas

Para la utilización de la herramienta Microsoft Agent y Speech API Microsoft es recomendable las siguientes características técnicas.

Tabla 1 Características de la herramienta Microsoft Agent

Componente	Descripción
Sistema Operativo	Microsoft Windows 95, 98, Me, NT 4.0, 2000, XP, Windows 7 (modo compatible)
Navegadores	Internet Explorer 3.0 o versiones anteriores
Procesador	Pentium I en adelante con 100 Mhz
Almacenamiento	1 MB de espacio libre en disco
Memoria RAM	16 MB
Almacenamiento para Microsoft Speech Recognition Engine	22 MB
Dispositivos	Micrófono , Parlantes
Tarjetas de Sonido	Cualquier tarjeta de sonido compatible con Windows

Fuente: (Microsoft, 2009)

Microsoft Agent debe ser ejecutado en modo compatibilidad en sistemas operativos actuales como es Windows 7, debido a las características técnicas que tienen estos sistemas.

CAPÍTULO 5
DISEÑO DEL SISTEMA

5.1. Definición de requisitos

Como se ha mencionado anteriormente Lazar-I-Soft consta de tres módulos principales los cuales son: módulo de visión artificial, módulo de enrutamiento y módulo de reconocimiento de voz. A continuación se detallan los requisitos funcionales y los requisitos no funcionales de cada uno de ellos.

5.1.1. Requisitos funcionales

- ***Módulo de visión artificial***

  EL módulo debe identificar el color establecido por el administrador de la aplicación. El sistema debe poseer una interfaz para administrar el tipo de color que se utiliza en la aplicación.

- ***Módulo de enrutamiento***

  El módulo de enrutamiento es el encargado de generar la mejor ruta de un punto A hacia un punto B para la búsqueda de un item.

  El algoritmo A*, utilizado en el sistema debe ser capaz de brindar información constante al módulo de reconocimiento de voz para que este se lo indique al usuario.

- ***Módulo de reconociendo de voz***

  El módulo debe soportar la autenticación a través de la voz de una persona para iniciar la aplicación.

  El módulo debe ser capaz de reproducir palabras y frases totalmente claras, para el buen entendimiento de la persona.

  La guía de voz de la aplicación debe ser seguida y con palabras puntuales para evitar retrasos y confusiones.

- ***Interfaces de administración***

  El sistema debe contar con interfaces para la administración de la aplicación, además de entornos gráficos para el ingreso de empleados, ingresos de nuevos pedidos al sistema, ingresos de coordenadas del mapa, entre otros.

5.1.2. Requisitos no funcionales

Debido a la complejidad para presentar el sistema Lazar-I-Soft en un ambiente real (biblioteca o bodega de tamaño natural), se ha definido una maqueta la cual representa el ambiente de una biblioteca real, para ello se ha considerado una escala de 1:17, donde un metro es el equivalente a 17 metros.

La maqueta diseñada con material de cartón, representa el escenario de una biblioteca, el mismo que cuenta con divisiones que indican las secciones que tiene una biblioteca real.

El sujeto de prueba es representado por un modelo (muñeco), el cual es guiado dependido de las instrucciones que indique el módulo de enrutamiento y el de reconocimiento de voz.

El módulo de visión artificial tiene como objetivo principal el identificar las posiciones en las que se encuentra el individuo (muñeco) en sus diferentes estados, para lo cual se utiliza una cámara web de alta definición debido a que esta puede capturar imágenes claras y así se evita inconvenientes al momento de realizar la detección del movimiento.

El reconocimiento de movimiento se lo realiza identificando un color determinado, es recomendable que el color escogido sea un color intenso siendo estos los posibles (rojo, azul, amarillo, verde, violeta).

Cabe mencionar que para tener éxito en el reconocimiento, se debe evitar el movimiento de entes externos a los requeridos para el funcionamiento del sistema tales como: personas extras o no programadas, colores utilizados en paredes, pisos, anaqueles similares al utilizado para la identificación.

La orden para ejecutar el inicio de la aplicación, es indicada por la persona que guía el muñeco, esta orden se la menciona por medio de un micrófono, el cual debe estar aislado del ruido externo para evitar confusiones en el reconocimiento de voz.

Una vez tomadas estas órdenes en el módulo de reconocimiento de voz, conjuntamente el módulo de visión artificial y el módulo de enrutamiento generan las instrucciones de seguimiento o guía, las cuales son emitidas a través de parlantes logrando que se escuche independientemente de donde se encuentre ubicada la persona (muñeco).

La iluminación es una parte fundamental en el escenario debido a que gracias a ella se pueden obtener mejores resultados en el seguimiento de objetos a través de video, no se debe tener una iluminación demasiada clara y cerca de la maqueta esto puede provocar cambios de contraste en los colores y por ende afecta al seguimiento de objetos, por lo cual se recomienda una iluminación natural, la que no provoque sombras, contraste altos ni bajos.

Los Sistemas Operativos soportados por la aplicación Lazar-I-Soft son: Windows XP y Windows 7. En lo correspondiente a características de hardware es recomendable utilizar una máquina con un procesador superior a Dual Core con memoria RAM superior a 2 GB para así cubrir con todas las funcionalidades del sistema.

El sistema Lazar-I-Soft debe cumplir con una disponibilidad del 99% en los módulos de visión artificial, enrutamiento y reconocimiento de voz.

Para almacenar la data del negocio se utiliza Mysql debido a su gran capacidad de almacenamiento de datos.

Para el reconocimiento de voz se debe utilizar un micrófono de gran capacidad para receptar los sonidos de la manera más clara posible.

5.2. Definición de parámetros para el control del ambiente

El posicionamiento o asignación de los puntos en la ruta controlada dentro del ambiente seleccionado se realiza utilizando parámetros (interconexiones, posiciones en eje X y Y) y una secuencia alfabética establecida para que Lazar-I-Soft pueda trabajar y representar lógicamente el entorno de una biblioteca.

Como el ambiente (entorno de una biblioteca) puede variar a lo largo, ancho, en su forma o en la distribución de los objetos que puede poseer en su interior, Lazar-I-Soft necesita definir los siguientes puntos:

- La puerta de entrada del ambiente siempre es el punto de inicio o punto de partida.
- El punto de inicio es representado por la letra A con la posición (0,0) dentro del ambiente controlado, como si se tratara de un punto dentro del plano cartesiano.

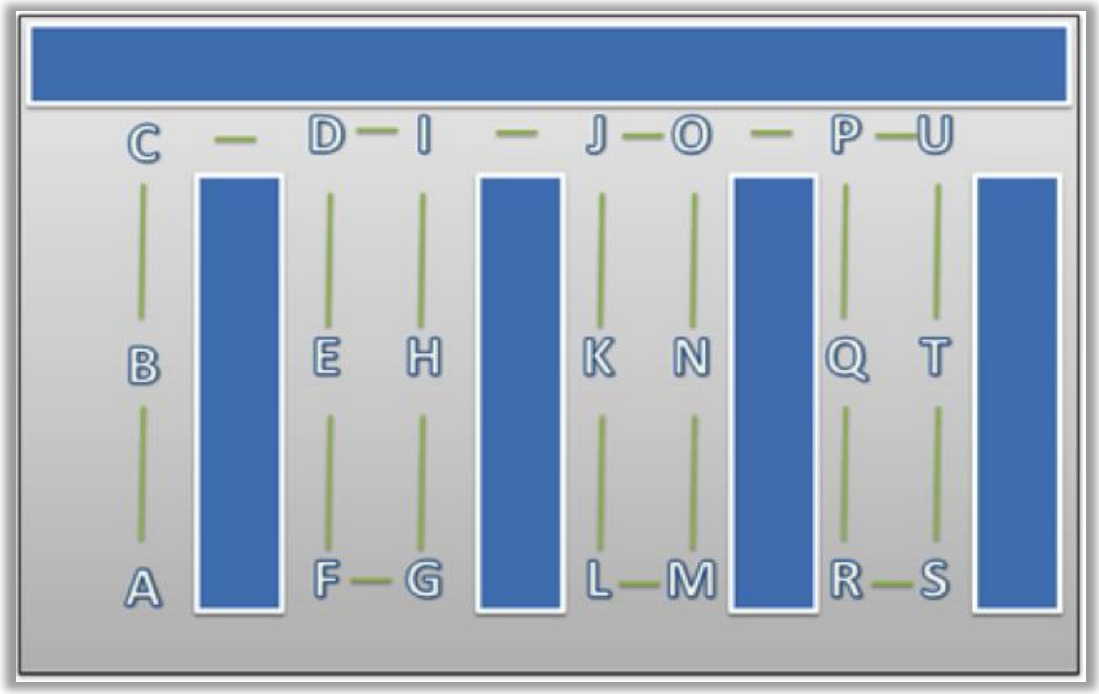
Para la definición de los puntos que le siguen al punto A en el ambiente, se requiere tener listos ciertos prerrequisitos:

- Poseer la estructura del mapa de caminos a seguir.
- Tener la cámara ubicada en la posición definitiva dentro del ambiente controlado, ya que la misma ayuda a detectar el posicionamiento del usuario no vidente y la ubicación de los puntos o camino dentro del mismo.
- La cámara debe capturar todo el espacio posible o en su defecto la gran mayoría del ambiente a monitorear.

Además, se debe tomar en cuenta que mientras más puntos se encuentren colocados dentro del ambiente, Lazar-I-Soft es más preciso para el enrutamiento del usuario, pero a su vez se vuelve más lento en los cálculos a realizar, aunque estos tiempos son casi imperceptibles, no hay que descartarlos del análisis. Es preferible asignar puntos en lugares estratégicos como las intersecciones, esquinas, finalización de un callejón o pasadizo y lugares donde se desee realizar un descanso, aunque este no se lo realice siempre.

La asignación de un identificador (una letra del alfabeto) a cada punto deseado se lo realiza de abajo hacia arriba y de derecha a izquierda, es decir que una vez ubicados en el punto A, el siguiente punto sobre A es el punto B y así sucesivamente, como se lo observa en la siguiente figura.

Figura 25 Representación de puntos en el ambiente



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Como se observa en la Figura 25, el punto A tiene una sola concatenación con el punto B, esto quiere decir que el usuario no vidente para trasladarse del punto A hacia cualquier otro punto, siempre debe pasar por el punto B; en cambio el punto M tiene concatenación con los puntos L y N esto quiere decir que tiene dos caminos que llevan a M; por lo tanto el camino puede variar dependiendo del punto destino.

Una vez ubicados e identificados alfabéticamente los puntos, estos deben ser ingresados en la base de datos, tanto el identificador como también su ubicación (x, y) y la lista de los puntos a concatenar.

La representación del usuario (muñeco) dentro del ambiente tiene igual importancia que la ubicación de los puntos en el mismo, por lo tanto; para que el usuario no pase desapercibido en el análisis de las imágenes para detectar su ubicación; es necesario que el usuario no vidente se destaque de todo lo que está a su alrededor, por tal motivo se le asigna una color específico para que lleve un distintivo consigo y pueda ser identificado en el video como lo muestra la figura.

Figura 26 Identificación de un objeto

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

El Rojo es un color intenso que no se pierde con facilidad dentro de los colores que pueden existir a su alrededor, además es uno de los colores principales del RGB (Red, Green, Blue), modelo de color mediante el cual un tono se obtiene a partir de la suma de distintas cantidades de los tres colores aditivos primarios.

Esto no quiere decir que dicho color no pueda existir en el ambiente o que la captura de las imágenes realizadas por la cámara de video varíe debido a la intensidad de la luz que posea el ambiente y provoque distorsión en el análisis de la información.

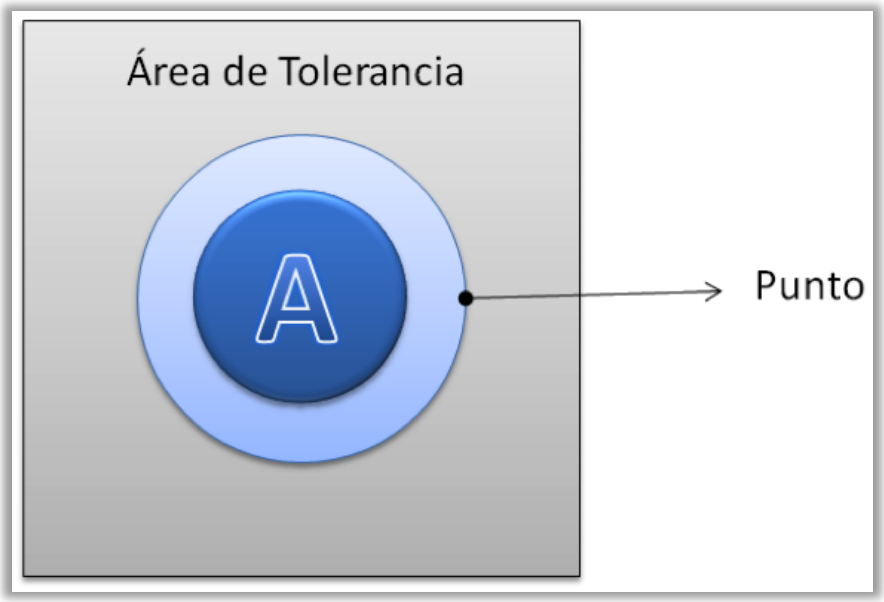
Para evitar estos inconvenientes se le provee al usuario no vidente con un LED "(Light-Emitting Diode: Diodo Emisor de Luz), que es un dispositivo semiconductor que emite luz incoherente de espectro reducido cuando se polariza de forma directa la unión PN en la cual circula por él una corriente eléctrica" (Diodo LED). Que emita el color con mayor intensidad al que está a su alrededor y con esto evitar la distorsión de colores que se produce al tener diferentes intensidades de luz en el ambiente.

Figura 27 Asignación de color al usuario no vidente

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

La cámara tiene una ubicación fija en el ambiente considerando que siempre hay puntos ciegos y no se puede esperar que el usuario pase exactamente por el punto esperado, se establece un radio de tolerancia a cada punto como se muestra en la Figura 28; esto quiere decir que al punto a ubicar al usuario no vidente, se le resta o suma un valor de tolerancia que ayuda a detectarlo de mejor manera.

Figura 28 Área de tolerancia



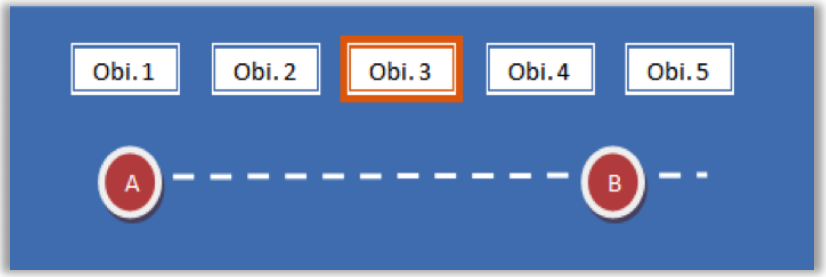
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

De la cantidad de puntos depende de la precisión o la efectividad deseada, aunque sin dejar de lado las observaciones mencionadas anteriormente, se debe también especificar los puntos en el lugar donde se encuentran los objetos a buscar, ya que si están demasiado separados el sistema no ubica al no vidente en el lugar específico si no, en un aproximado.

Ejemplo:

- Se solicita al sistema que busque un objeto 3 que se encuentra entre A y B, Lazar-I-Soft ubica al no vidente en el punto más cercano, en este caso el punto B, como lo muestra en la siguiente Figura.

Figura 29 Búsqueda de objetos entre puntos (1)



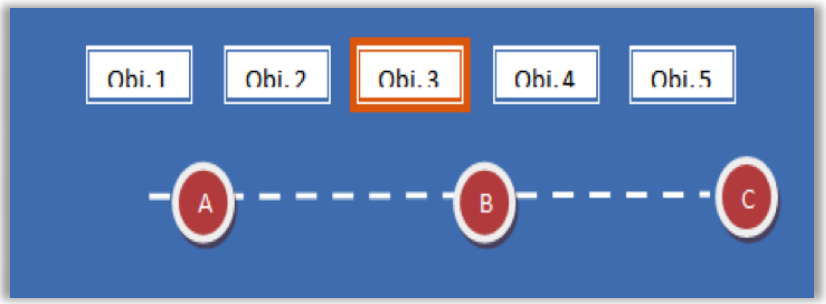
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Se generan puntos de forma compartida haciendo que un punto sirva para poder retirar dos objetos distintos a la vez, esta es la mejor opción al momento de definir puntos para no crear demasiados puntos innecesarios y mantener la efectividad del sistema.

Ejemplo:

- Como lo muestra la figura siguiente el caso del punto B, se optimiza la ubicación de un punto, ubicándolo justo en el medio de dos objetos, permitiendo al sistema ubicar al usuario en un punto determinado y dar la orden respectiva para recoger el objeto seleccionado, evitando la creación innecesaria de puntos en la ruta.

Figura 30 Búsqueda de objetos entre puntos (2)



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

5.3. Diseño de software

5.3.1. Casos de uso

Según Benet Campderrich, en su libro "Ingeniería del Software" publicado en 2003 un caso de uso se define como el instrumento que "documenta una interacción entre

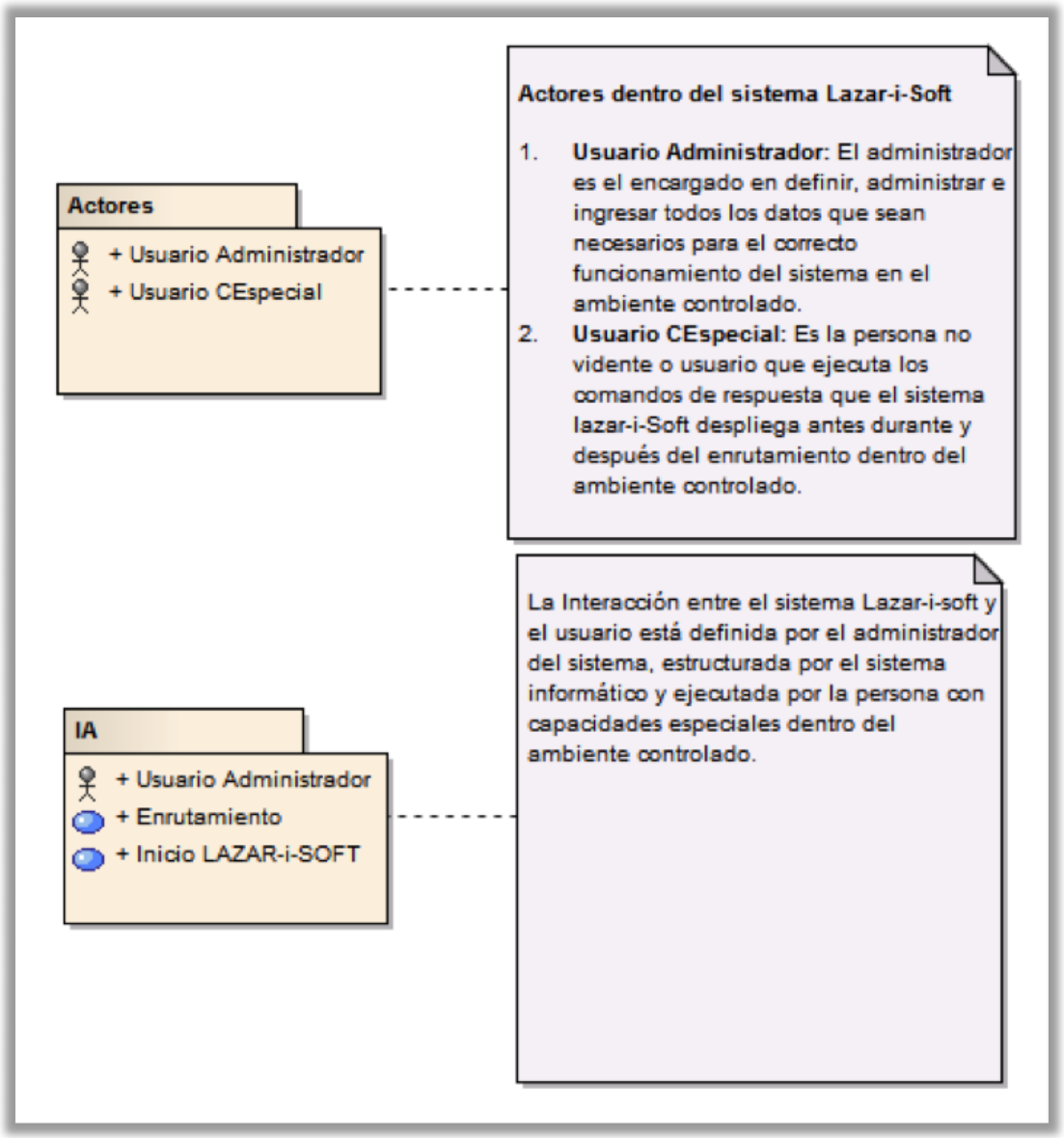
el *software* y un actor o más. Dicha interacción tiene que ser, en principio, una función autónoma dentro del *software*" (Campderrich, 2003, pág. 85), a continuación se describe dichas interacciones.

- ***Actores***

  En el sistema Lazar-I-Soft existen dos actores principales que interactúan y administran el sistema, estos actores se clasifican en dos: Usuario Administrador y Usuario "CEspecial" o Usuario con Capacidades Especiales.

  Los actores que intervienen en el sistema Lazar-I-Soft se describen en la figura siguiente.

Figura 31 Actores del sistema Lazar-I-Sof

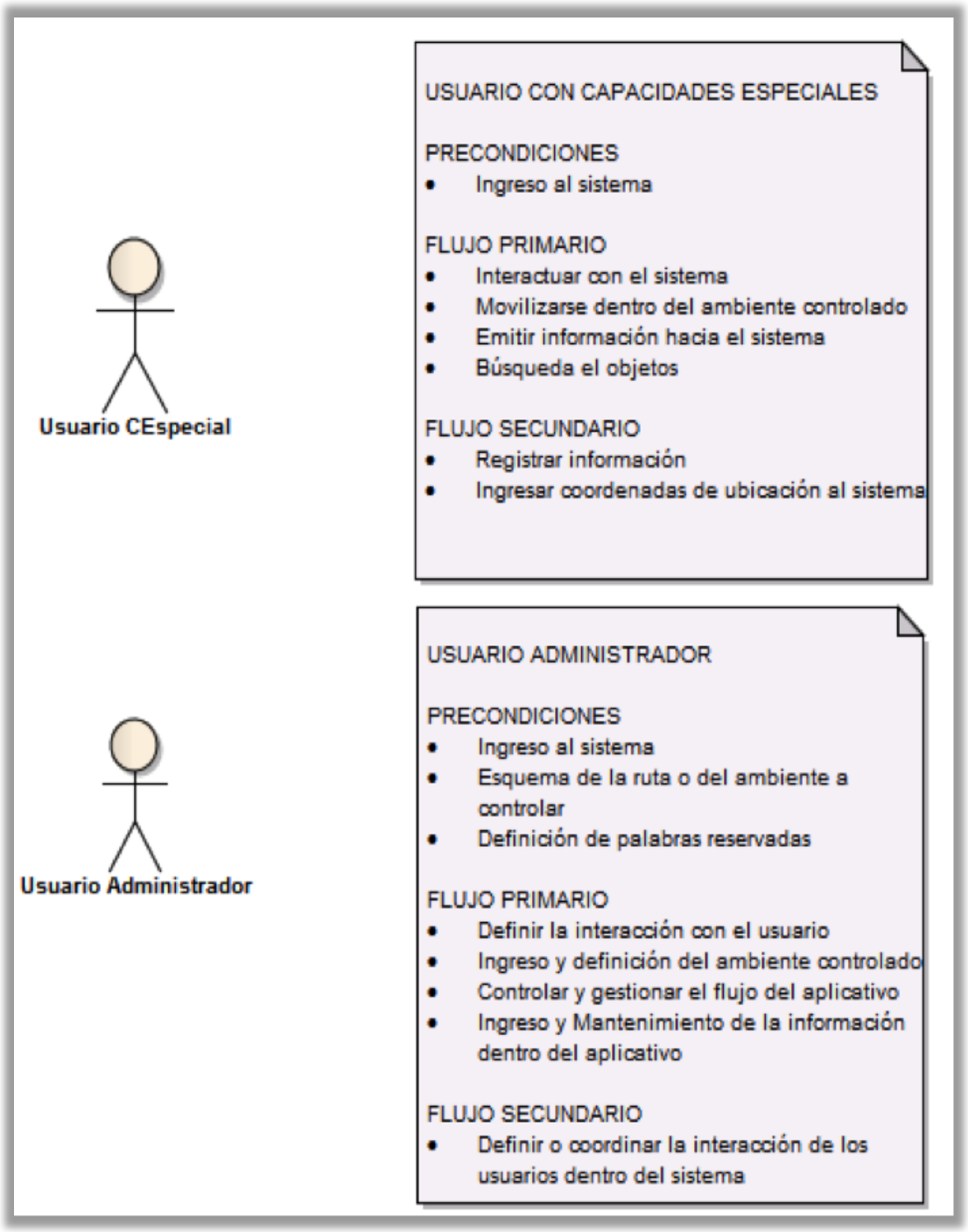


Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- ***Flujos***

  Para que el sistema Lazar-I-Soft funcione adecuadamente cada uno de sus actores posee ciertas precondiciones y flujos establecidos para interactuar con el sistema, mismos que son descritos en la siguiente figura.

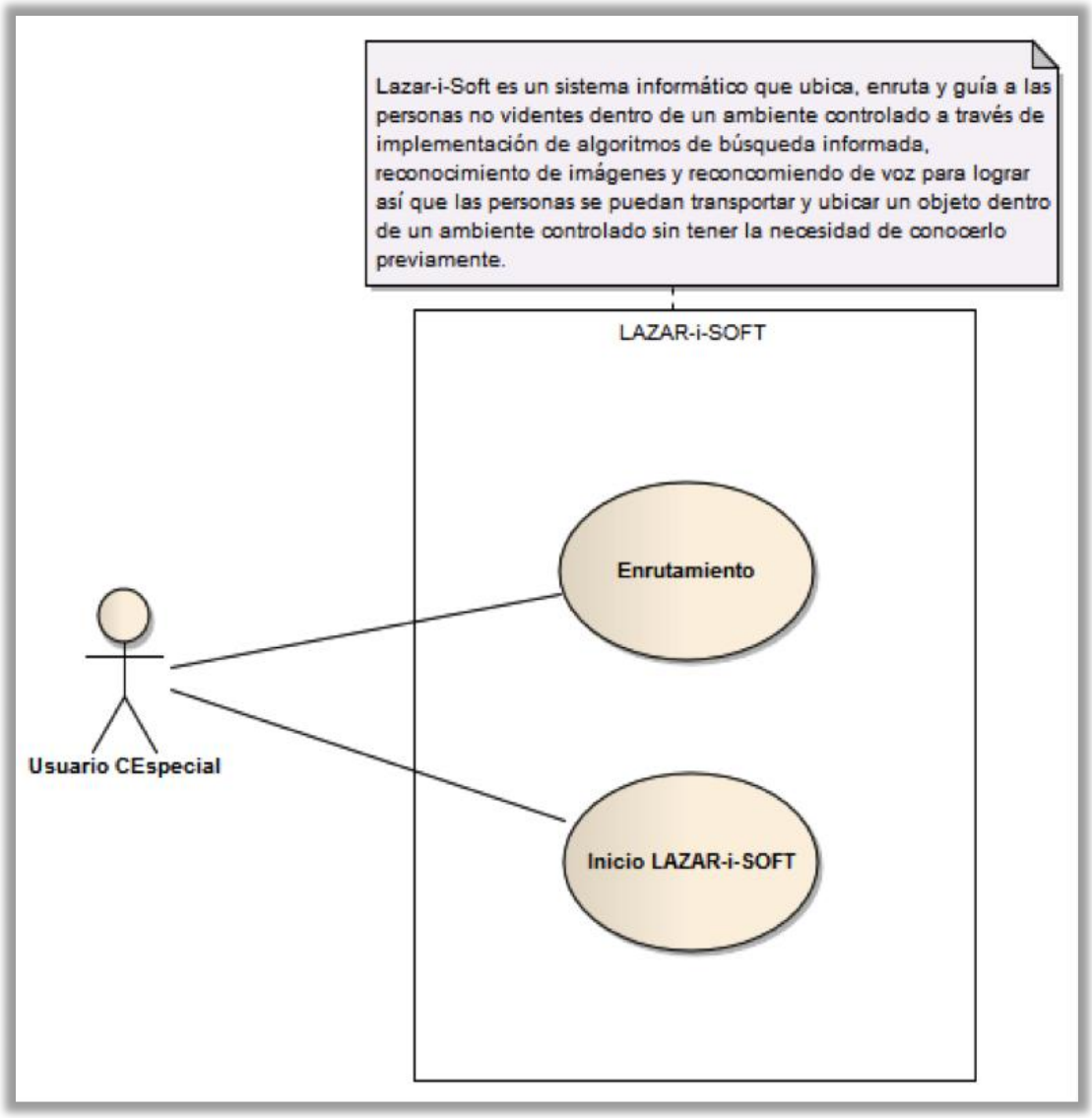
Figura 32 Flujos del sistema



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

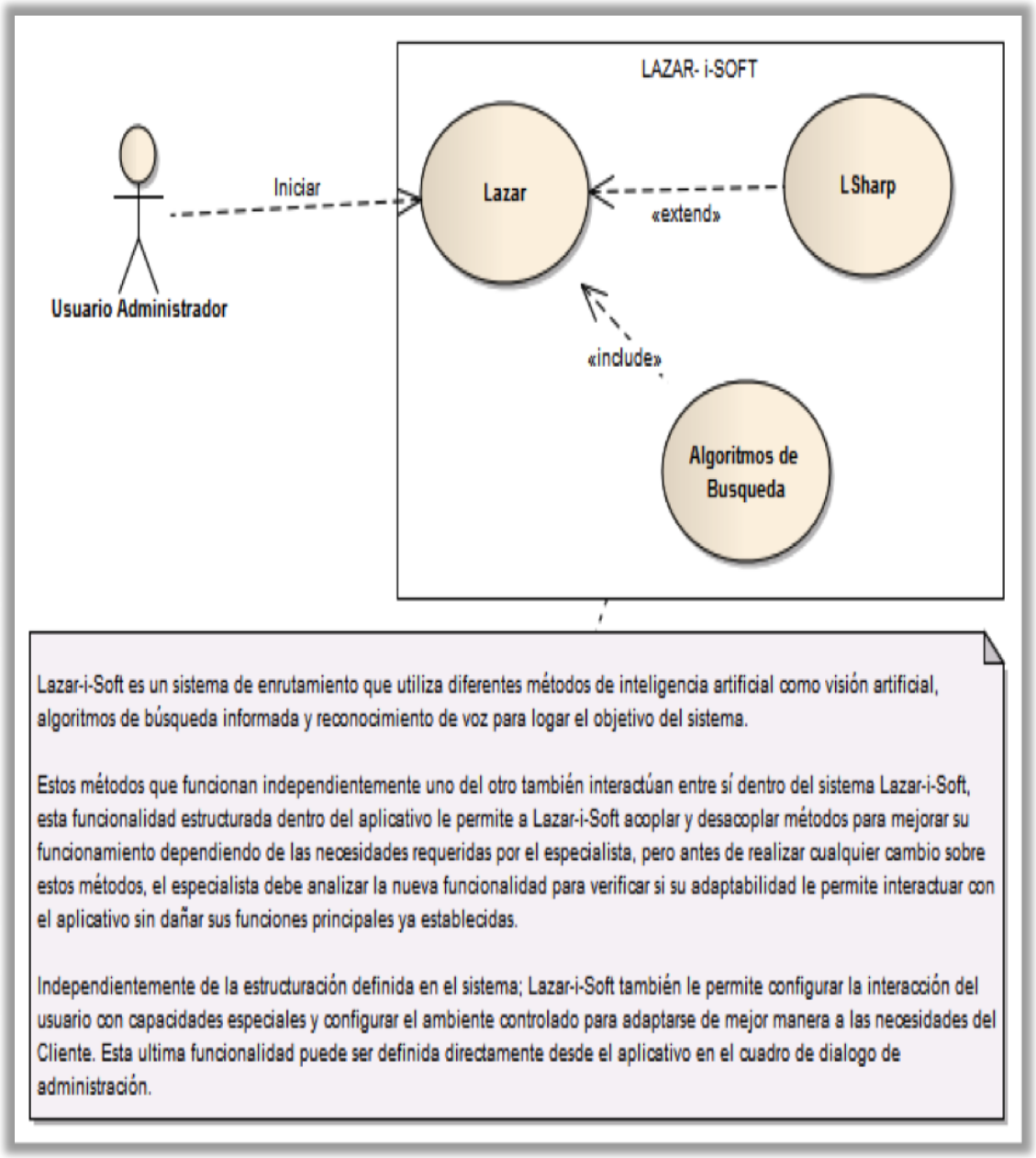
- ***Esquema de interacción entre los actores y el sistema Lazar-I-Soft.***

Figura 33 Esquema de interacción (1)



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

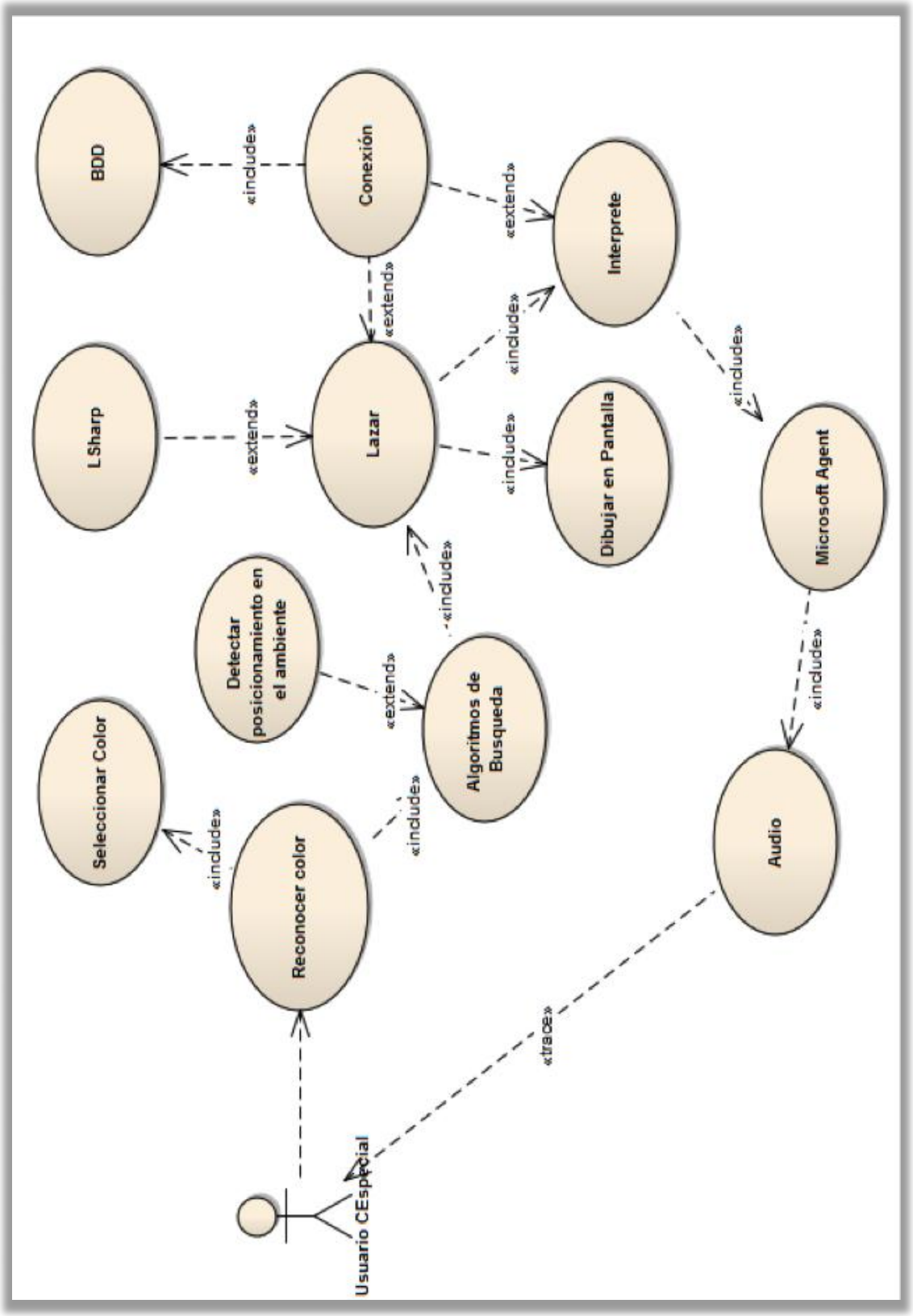
Figura 34 Esquema de interacción (2)



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

5.3.2. Diagrama de caso de uso

Figura 35 Diagrama de caso de uso



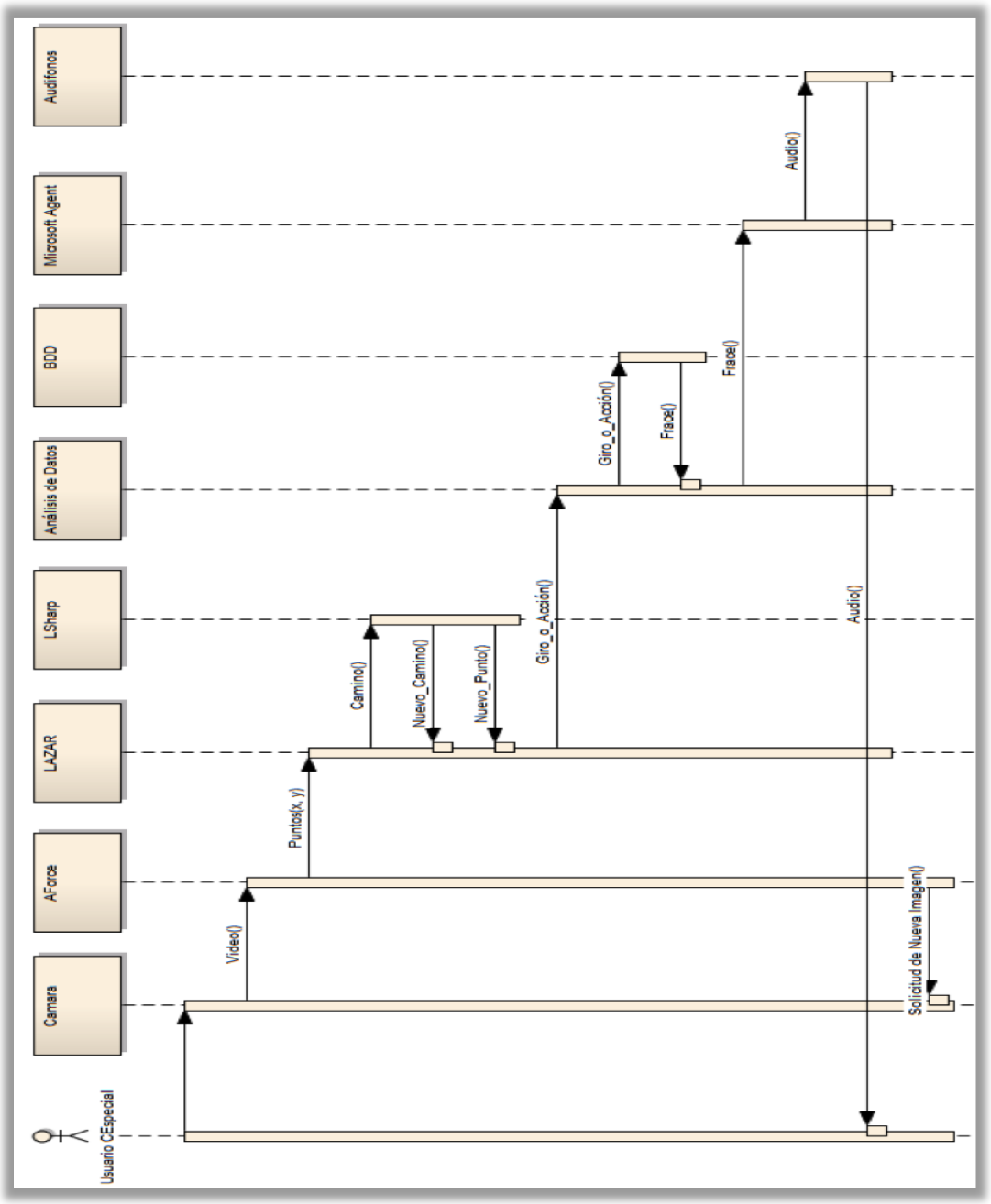
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

5.3.3. Diagrama de secuencias

Benet Campderrich en su libro "Ingeniería del Software" publicado en 2003 afirma sobre los diagramas de secuencia lo siguiente:

- En el diagrama de secuencia no se representan de forma explícita los papeles de asociaciones (quedan implícitos en los mensajes) y se representa explícitamente el orden en el tiempo, e incluso la duración, de los mensajes y de las operaciones que ponen en marcha (2003, pág. 95).

Figura 36 Diagrama de secuencias



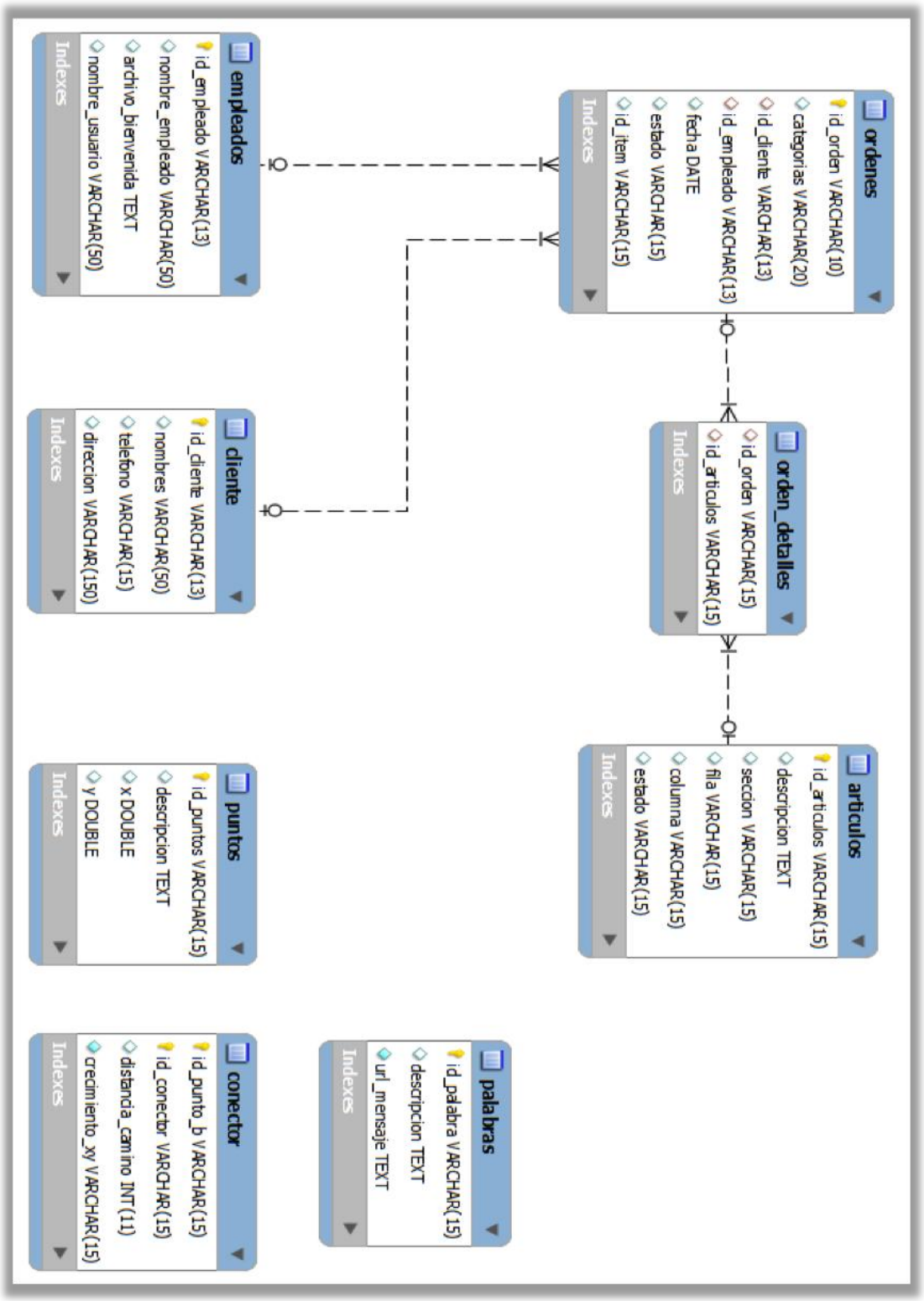
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

5.3.4. Diagramas de clases

Ver diagramas de clases en formato A3

5.3.5. Diagrama de base de datos

Figura 37 Diagrama de base de datos Lazar-I-Soft



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

5.3.5.1. Diccionario de datos

- articulos (Tabla que almacena los artículos de la librería)

Tabla 2 Artículos

Columnname	DataType	PK	NN	UQ	Default	Comment
id_articulos	VARCHAR(15)	√	√			Identificador del artículo
descripcion	TEXT				NULL	Nombre que se le asigna al artículo
seccion	VARCHAR(15)				NULL	Ubicación, sección en donde se encuentra alojado el artículo
fila	VARCHAR(15)				NULL	Fila de la sección en donde se encuentra el artículo
columna	VARCHAR(15)				NULL	Columna de la sección en donde se encuentra el artículo
estado	VARCHAR(15)				NULL	Estado del articulo

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- cliente (Datos informativos de las personas que utilizan el sistema)

Tabla 3 Cliente

Columnname	DataType	PK	NN	UQ	Default	Comment
id_cliente	VARCHAR(13)	√	√			Identificador de Cliente - cédula de identidad
nombres	VARCHAR(50)				NULL	Nombres del cliente
telefono	VARCHAR(15)				NULL	Teléfono de contacto del cliente
direccion	VARCHAR(150)				NULL	Dirección de domicilio del cliente

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- conector (Conectores)

Tabla 4 Conectores

Columnname	DataType	PK	NN	UQ	Default	Comment
id_punto_b	VARCHAR(15)	√	√			Punto de origen
id_conector	VARCHAR(15)	√	√			Punto de destino
distancia_camino	INT(11)				NULL	Distancia de recorrido entre puntos
crecimiento_xy	VARCHAR(15)		√			Dirección hacia dónde va el usuario X,Y

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- empleados (Datos informativos de los empleados)

Tabla 5 Empleados

Columnname	DataType	PK	NN	UQ	Default	Comment
id_empleado	VARCHAR(13)	√	√			Identificador del empleado - cédula de identidad
nombre_empleado	VARCHAR(50)				NULL	Nombre del empleado
archivo_bienvenida	TEXT				NULL	Ubicación del archivo de audio de bienvenida
nombre_usuario	VARCHAR(50)				NULL	Nombre identificador del usuario

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- orden_detalles

Tabla 6 Órdenes detalles

Columnname	DataType	PK	NN	UQ	Default	Comment
id_orden	VARCHAR(15)				NULL	Identificador de la orden
id_articulos	VARCHAR(15)				NULL	Identificador del articulo

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- ordenes (Órdenes que se genera del cliente al empleado)

Tabla 7 Órdenes

Columnname	DataType	PK	NN	UQ	Default	Comment
id_orden	VARCHAR(10)	√	√			Identificador de la orden
categorias	VARCHAR(20)				NULL	Categoría de la orden
id_cliente	VARCHAR(13)				NULL	Identificador del cliente que pertenece la orden
id_empleado	VARCHAR(13)				NULL	Identificador del empleado que pertenece la orden
fecha	DATE				NULL	Fecha en la que se genera la orden
estado	VARCHAR(15)				NULL	Estado de la orden

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- palabras (Utilizada para la traducción del Sistema)

Tabla 8 Palabras

Columnname	DataType	PK	NN	UQ	Default	Comment
id_palabra	VARCHAR(15)	√	√			Identificador de la palabra
descripcion	TEXT		√			Palabra para la traducción
url_mensaje	TEXT				NULL	Destino de la ubicación del mensaje en audio

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- puntos (Puntos de ubicación en el mapa Lazar-I-Soft)

Tabla 9 Puntos

Columnname	DataType	PK	NN	UQ	Default	Comment
id_puntos	VARCHAR(15)	√	√			Identificador del punto
descripcion	TEXT				NULL	Descripción de identificación del punto
x	DOUBLE				NULL	Coordenadas en X del punto
y	DOUBLE				NULL	Coordenadas en Y del punto

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

5.3.6. Definición de arquitectura de software

La arquitectura de software se define como el arte de construir estructuras de sistemas basadas en cánones definidos, que incluye sus componentes de software y la relación que hay entre ellos. De tal manera que se convierte en el marco de referencia para guiar la construcción del sistema en cuestión.

Por lo expuesto anteriormente cuando se construyen sistemas complejos con la interconexión de muchos componentes, y la organización de estos componentes se transforma en un gran problema de diseño, este es resuelto por medio de la generación de diagramas de componentes, diagramas de secuencia y demás componentes de la arquitectura de un sistema, permitiendo obtener la vista en "nivel de diseño" que es el análogo de los planos de edificio.

A nivel de arquitectura de software el sistema es diseñado basado en el modelo de n capas (n-Layer) que contempla la organización lógica del código, teniendo como principal objetivo la división modular de las lógicas, es decir, a cada módulo se le designa una misión específica, lo que permite un diseño arquitectónico escalable y menor impacto al momento de realizar cambios.

La principal ventaja de este tipo de arquitectura multinivel o de n capas, es que al momento del desarrollo cada grupo o equipo está completamente abstraído en su módulo determinado sin tener mayor competencia en los módulos que desarrollen los demás miembros en paralelo, además de que al momento de realizar cambios en un lugar determinado solo se deberá modificar ese componente aislado, sin necesidad de generar un cuello de botella en el desarrollo.

El ejemplo más clásico de esto es el de cambiar la lógica de conexión a una base de datos si esta capa se encuentra abstraída en un módulo determinado no tendrá mayor inconveniente si lo hace a través de conexión directa o a través de Apis de persistencia de base de datos, puesto que para las capas que se conecte con la capa de acceso de datos este tema será transparente.

El mayor inconveniente con este tipo de diseños arquitectónicos se presenta al momento de la integración puesto que demanda de un esfuerzo mayor.

El modelo arquitectónico por capas presenta dos estilos de diseño que son: el diseño en capas Estricto y el diseño de capas Laxo.

- ***Diseño en capas estricto***

  Limita la comunicación de los componentes de una capa de tal manera que solo se pueden comunicar entre los miembros de la misma capa o con los componentes de la capa inmediatamente inferior. Es decir que los componentes de la capa N de la figura solo podrán comunicarse con los componentes de la capa N-1, y los componentes de la capa N-1 con los componentes de la capa N-2 y así sucesivamente.

Figura 38 Modelo N-Capas

Fuente: Guía de arquitectura n-capas orientada al dominio con .net 4.0

- ***Diseño en capas laxo***

  Permite que los componentes de una capa determinada interactúen con todos los componentes de las capas inferiores a ella. De tal manera que los componentes de la capa N de la figura anterior podrán interactuar con todos los componentes de las demás capas, pero los de la capa 2 solo con los de la capa 1.

  Este tipo de aproximación de diseño conlleva consigo ventajas de rendimiento puesto que optimiza las llamadas a las capas ya que el sistema no debe realizar llamadas redundantes desde una capa a otra.

  El diseño de capas Laxo no proporciona el mismo nivel de aislamiento entre capas que proporciona el diseño de "capas estricto" entre las diferentes capas, haciendo más difícil la sustitución de un capa al poseer un alto grado de cohesión entre capas.

  Luego de analizar los paradigmas de diseño de capas, el que más se ajusta al proyecto Lazar-I-Sotf por las condiciones en las que se ha desarrollado, es diseño de capas estricto, puesto es la que mejor se ajusta a las condiciones de desarrollo del proyecto.

De tal manera que la capa de presentación tendrá acceso a las capas inferiores y se agilita el desarrollo independiente de módulos contenidos en cada una de las capas inferiores, puesto que la capa de presentación solo llamará los componentes de las capas de Negocio, Routing y Acceso a los datos, El esquema de las capas lo muestra la figura siguiente.

Figura 39 Modelo de N-Capas Lazar-I-Soft

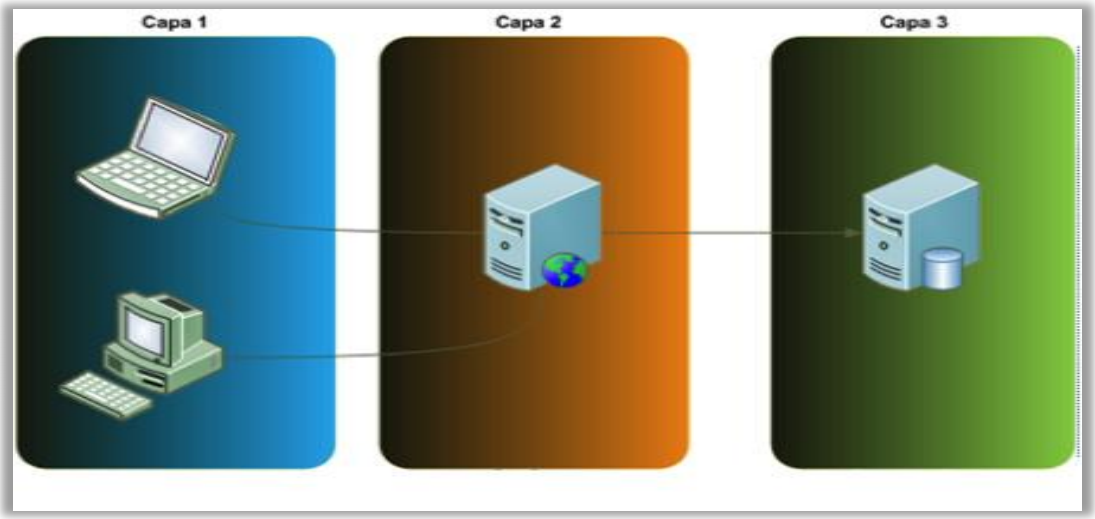
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

- Módulo capa presentación
  - Interface administración.
  - Módulo funcional.
- Módulos capa negocio
  - Definición de puntos.
  - Configuraciones.
- Módulos capa routing
  - Enrutamiento IA.
  - Ubicación (ObjectTraking).
  - Comunicación (VoiceRecognition).
- Módulos capa acceso a datos
  - Conexión a base de datos

5.3.7. Definición de arquitectura de hardware

La arquitectura hardware plantea la distribución física de las capas de la arquitectura de software por lo cual la arquitectura que se utiliza en el proyecto Lazar-I-Soft, es la arquitectura de N-Tier como lo ilustra la figura siguiente.

Figura 40 Modelo N-Tier



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Al poseer una arquitectura dividida en N-Tier el sistema de software se vuelve más robusto puesto que los componentes no se encuentran centralizados, y de darse el caso de fallar alguno se cambiaría solo el equipo físico de ese Tier y el resto de componentes seguirían funcionando sin mayores contratiempos.

Debido a que el proyecto se ejecuta en un ambiente simulado el número de tier será de 1 ya que todos los componentes de la arquitectura de n-capas se ejecutan dentro de un mismo equipo.

CAPÍTULO 6
CONSTRUCCIÓN DEL SISTEMA

6.1. Entorno de trabajo

Para el correcto funcionamiento del sistema Lazar-I-Soft, se debe tomar en cuenta los diferentes factores que intervienen en el entorno de trabajo debido a que existen varios factores que alteran la información antes de que esta llegue a su destino o al sistema Lazar-I-Soft, estos factores definen el comportamiento del sistema y su buen funcionamiento.

- ***Factor de iluminación***

  Dentro del ambiente controlado el factor de iluminación define los colores y la tonalidad de los mismos, estos datos son recopilados por la cámara de video y enviados hacia el sistema Lazar-I-Soft, por lo que es indispensable tener una adecuada iluminación dentro del ambiente controlado.

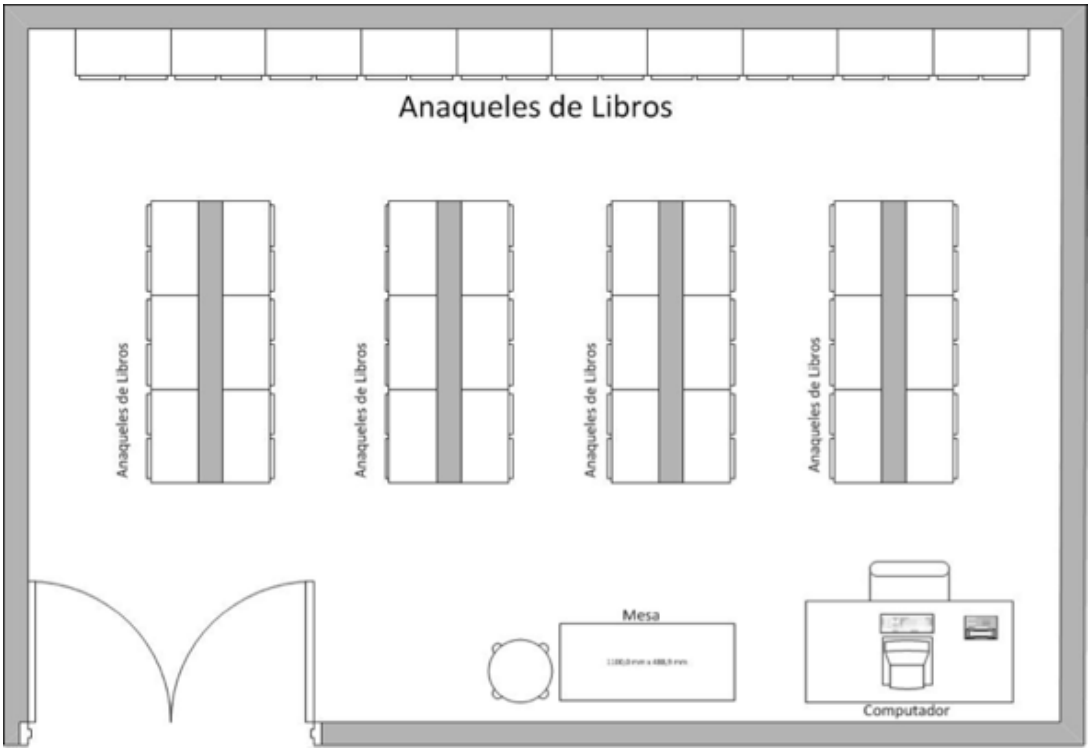
- ***Factor de ruido***

  Lazar-I-Soft requiere que la comunicación existente entre la aplicación y la persona no vidente que sea lo más nítida posible, para que los comandos solicitados por el aplicativo y emitidos por la persona no vidente se ejecuten correctamente.

- ***Factor de distancia***

  La ubicación de la cámara de video dentro del ambiente controlado define, parametriza y monitorea el ambiente en tiempo real, por lo que es indispensable ubicar la cámara a una distancia adecuada, donde el lente de la cámara abarque toda el área de monitoreo y mitigue los puntos ciegos del ambiente controlado, mejorando así el desempeño del sistema.

Figura 41 Entorno de trabajo



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

6.2. Implementación y adaptación de módulos

El sistema Lazar-I-Soft se encuentra estructurado por módulos independientes que interactúan entre sí mediante parámetros de entrada y códigos de salida, estos módulos le permiten al sistema Lazar-I-Soft adaptarse de mejor manera a cualquier cambio o mejora del sistema; siempre y cuando se cumpla con los códigos de salida.

Lazar-I-Soft utiliza la visión artificial para reconocer la ubicación del no vidente mediante algoritmos que identifican un color específico dentro del ambiente controlado, estos colores son analizados por el sistema para encontrar la ubicación exacta de un color específico dentro del ambiente controlado en tiempo real; estos puntos son interpretados por el sistema como ubicaciones exactas del no vidente dentro del ambiente controlado como puntos o coordenadas de un plano cartesiano; estos puntos son analizados por los algoritmos de inteligencia artificial y procesados mediante listas para encontrar el resultado final o ruta, esta respuesta es emitida al reconocimiento de voz quien es el encargado de analizar, buscar y transformar los

códigos en comandos de voz, para que estos sean ejecutados por la persona no vidente, mientras se traslada de un punto a otro.

6.3. Programación del software

6.3.1. Definición del algoritmo de giros

Este algoritmo permite al sistema Lazar-I-Soft analizar la ubicación de los nodos dentro del ambiente controlado y los movimientos que realiza la persona no vidente, para definir los giros que se deben realizar al trasladarse de un nodo a otro.

```
(DEF GIROS (AT AC SG AVNZ_AT_AC AVNZ_AC_SG BND_AVNZ)
        (IF (IS AT SG)
                8
                (IF (IS AT AC)
                        11
                        (IF (IS AVNZ_AT_AC AVNZ_AC_SG)
                                11
                                (IF (IS AVNZ_AT_AC 'Y)
                                        (IF (> (CAAR (RL1 L1 AC))
                                                        (CAAR (RL1 L1 SG)))
                                                (IF (> (CADR(CAR (RL1 L1 AT)))
                                                                (CADR(CAR (RL1 L1 SG))))
                                                        4
                                                        5
                                                )
                                                (IF (> (CADR(CAR (RL1 L1 AT)))
                                                                (CADR(CAR (RL1 L1 SG))))
                                                        5
                                                        4
                                                ))
                                        (IF (> (CADR(CAR (RL1 L1 AC)))
                                                  (CADR(CAR (RL1 L1 SG)))
                                                )
                                                (IF (< (CAAR (RL1 L1 AT))
                                                                (CAAR (RL1 L1 SG)))
                                                        4
                                                        5
                                                )
                                                (IF (< (CAAR (RL1 L1 AT))
                                                                (CAAR (RL1 L1 SG))
                                                        )
                                                        5
                                                        4
                                                )
                                        )
                                )
                        )
                )
        )
)
```

6.3.2. Definición de la lista de puntos

La siguiente lista L1 es la estructura que utiliza el sistema Lazar-I-Soft para analizar la ubicación de cada nodo dentro del ambiente controlado, esta estructura contiene la ubicación en X y Y que son utilizados por el sistema para calcular las distancias existentes entre los nodos

```
(SET L1 '(
(100 20 A)
(100 285 B)
(255 285 C)
(255 50 D)
(265 50 E)
(265 285 F)
(375 285 G)
(375 50 H)
(385 50 I)
(385 285 J)
(435 285 K)
(450 20 L)
)
```

6.3.3. Definición de la lista de enlaces

La siguiente lista L2 define los enlaces o los posibles caminos que existen dentro del ambiente controlado que utiliza el sistema Lazar-I-Soft para buscar el camino con menor coste desde un nodo a otro.

```
(SET L2 '(
( A B )
( B A C )
( C B F D )
( D C E )
( E D F )
( F C E G )
( G F J H )
( H G I )
( I H  J )
( J G K I )
( K J  L)
( L K )
 )
)
```

6.3.4. Definición del algoritmo de enrutamiento

Este algoritmo realiza todos los cálculos necesarios dentro del sistema Lazar-I-Soft para analizar, comparar y seleccionar la ruta con menor coste desde un nodo origen a un nodo destino.

```
(DEF BUSCA_MENOR_LC (ACUM PI PF LISTA P_ANT)
        (SET PUNTOMENORLISTA (EL_MENOR_HP_MAS_GP LISTA (CAR
LISTA)))
        (SET PUNTOACTUAL (CAR PUNTOMENORLISTA))
        (SET VALORPUNTOACTUAL (CADR PUNTOMENORLISTA) )
        (SET LISTASIGUIENTE (SUBLIS G_P PUNTOACTUAL))
        (IF (IS PI PF)
                NULL
                (IF (IS PF P_ANT)
                        NULL
                        (IF (IS PF PUNTOACTUAL)
                                (LIST PUNTOACTUAL)
                                (IF (CDR LISTA)
                                                (CONS
                                                        PUNTOACTUAL
(BUSCA_MENOR_LC (+ ACUM VALORPUNTOACTUAL) PI PF
LISTASIGUIENTE PUNTOACTUAL)
                                                )
                                                PUNTOACTUAL
                                )
                        )
                )
        )
)
```

6.4. Pruebas y ajustes

6.4.1. Pruebas de caja negra (Funcionales)

Las pruebas funcionales se enfocan en los resultados finales emitidos por el sistema, al igual que los servicios y sus funcionalidades, estas pruebas analizan el desenvolvimiento del sistema vs el entorno. Apuntando siempre a la comparación del resultado esperado vs el resultado obtenido.

A continuación se describe de manera detalla y explicita los resultados obtenidos de las pruebas realizadas al sistema Lazar-I-Soft en los siguientes módulos:

- Acceso.
- Reconocimiento usuario.
- Enrutamiento.
- Reconocimiento de comandos vocales.
- Reconocimiento-seguimiento de objeto (object tracking).

Tabla 10 Pruebas de acceso Lazar-I-Soft

Caso de prueba acceso Lazar-I-Soft						
Id_Caso de Prueba	**Descripción del Caso de Prueba**	**Pre Requisito**	**Resultado Esperado**	**Resultado Obtenido**	**Estado**	**Observación**
LZ-I-S_01_AC	Acceso al Sistema	Palabras reservadas creadas en la Base de Datos	El sistema solicita el Nombre de Usuario	El sistema no responde	Sin respuesta	Se debe revisar las palabras reservadas y la conexión al módulo
LZ-I-S_02_AC	Acceso al Sistema	Palabras reservadas creadas en la Base de Datos	El sistema solicita el nombre de usuario	El sistema solicita el nombre de usuario	Iniciado	N/A
LZ-I-S_03_AC	Acceso al Sistema	Usuario creado en la Base de Datos	El sistema permite el acceso y da la bienvenida	El sistema permite el acceso y da la bienvenida	Autenticado	N/A
LZ-I-S_04_AC	Acceso al Sistema	Palabras reservadas creadas en la Base de Datos	El sistema solicita el nombre de usuario	El sistema solicita el nombre de usuario	Iniciado	N/A
LZ-I-S_04_AC	Acceso al Sistema (Configurar)	Usuario creado en la Base de Datos con privilegios	El sistema permite el acceso y redirige a la Administración	El sistema permite el acceso y redirige a la Administración	Autenticado	N/A

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Tabla 11 Pruebas de configuración Lazar-I-Soft

Caso de prueba configuración Lazar-I-Soft						
Id_Caso de Prueba	**Descripción del Caso de Prueba**	**Pre Requisito**	**Resultado Esperado**	**Resultado Obtenido**	**Estado**	**Observación**
LZ-I-S_01_CS	Configuración del Sistema	Usuario dentro del sistema, base de datos creada con tablas de configuración	Registro satisfactorio de punto en el Sistema	Registro satisfactorio de punto en el Sistema	Puntos Ingresados	N/A
LZ-I-S_02_CS	Configuración del Sistema(obtención dinámica de puntos)	Que el sistema posea acceso a una cámara web	Se coloca el objeto a rastrear en el video y se obtienen los puntos en XY. Para que los valores de los campos de texto cambien dependiendo de la ubicación del objeto	Se coloca el objeto a rastrear en el video y se obtienen los puntos en XY. Para que los valores de los campos de texto cambien dependiendo de la ubicación del objeto	Completado	N/A
LZ-I-S_03_CS	Configuración del Sistema (interconexión de puntos)	Puntos registrados en el Sistema	Se interconectan los puntos y el sistema muestra el grafico de interconexión	Se registra la interconexión de los puntos en a nivel lógico pero el grafico no muestra las conexiones	Iniciado	Se debe refrescar la interface para que se muestren los puntos interconectados. Se recomienda efectuar un repaint de la imagen para visualizar el cambio
LZ-I-S_04_CS	Configuración del Sistema	Puntos registrados en el Sistema	Se interconectan los puntos y el sistema muestra el grafico de interconexión	Se interconectan los puntos y el sistema muestra el grafico de interconexión	Completado	Se realiza en el código el cambio y se puede concluir con la prueba
LZ-I-S_05_CS	Configuración del Sistema	Usuario Creado en la Base de Datos con privilegios	El Sistema permite el acceso y redirige a la interface de Administración	El Sistema permite el acceso y redirige a la interface de Administración	Autenticado	N/A

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Tabla 12 Pruebas de enrutamiento Lazar-I-Soft

Caso de prueba módulo enrutamiento						
Id_Caso de Prueba	**Descripción del Caso de Prueba**	**Pre Requisito**	**Resultado Esperado**	**Resultado Obtenido**	**Estado**	**Observación**
LZ-I-S_01_ME	Acceso al módulo principal de enrutamiento	Que el Usuario haya pasado satisfactoriamente el menú vocal de bienvenida.	Inicio del módulo de enrutamiento principal	Inicio del módulo de enrutamiento principal	Completado	N/A
LZ-I-S_02_ME	Se usa un test por defecto que recorre una ruta en L con giro.	Que se encuentre el usuario en módulo de enrutamiento	El sistema genera las órdenes avanza, gira a la derecha eizquierda, detente	El sistema genera las órdenes avanza, gira a la derecha eizquierda, detente	Completado	N/A
LZ-I-S_03_ME	Se acopla el acceso a los puntos que obtiene desde el frame de video con el gráfico que se muestra en la interface	Que se encuentre en la base de datos ingresados los puntos del ambiente controlado	Se pintan los puntos del mini plano que se muestra en la interface conforme se avanza en la maqueta	Se pintan los puntos del mini plano que se muestra en la interface conforme se avanza en la maqueta	Iniciado	Se debe poseer la intensidad del LED de identificación bien definida.
LZ-I-S_04_ME	Obtención de artículos	Que se encuentre configurado en la base de datos la ubicación del ítem a buscar	El sistema obtiene la orden de búsqueda y envía al módulo de enrutamiento el origen y destino	El sistema obtiene la orden de búsqueda y envía al módulo de enrutamiento el origen y destino.	Completado	N/A
LZ-I-S_05_ME	Se completa el circuito	Que complete las tareas de guía y ruteo	Va del punto alfa a omega recupera y retorna al inicio	Va del punto alfa a omega recupera y retorna al inicio	Completado	N/A

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

6.4.2. Pruebas de caja gris (Desarrollo)

Las pruebas de caja gris consisten en analizar las respuestas emitidas por los métodos o funciones durante el proceso de construcción de la solución, estas respuestas son registradas y clasificadas en una tabla dependiendo de las características de los métodos que las generan. Además estas pruebas ayudan a encontrar una mejor solución a los problemas que se presentan en cada uno de los métodos, comparando los resultados obtenidos vs los resultados esperados. A estas pruebas también las denomina pruebas de escritorio.

Durante la etapa de implementación y adaptación de módulos del sistema Lazar-I-Soft se realizó y registró las siguientes pruebas a los métodos: de visión, device, algoritmos de búsqueda informada y reconocimiento de comandos vocales.

Tabla 13 Pruebas de visión – device

Caso de prueba visión-device						
Id_Caso del Prueba	**Descripción del Caso de Prueba**	**Pre Requisito**	**Resultado Esperado**	**Resultado Obtenido**	**Estado**	**Observación**
LZ-I-S_01_VI	Obtención en línea temporal de coordenadas XY	Poseer una cámara en el sistema	Durante la ejecución de la prueba se debe monitorear el movimiento en X y Y de las coordenadas	El sistema no responde	Sin respuesta	Se modifica la configuración del Timer y la llamada al objectTraking
LZ-I-S_02_VI	Obtención en línea temporal de coordenadas XY	Poseer una cámara en el sistema	Durante la ejecución de la prueba se debe monitorear el movimiento en X y Y de las coordenadas	Durante la ejecución de la prueba se debe monitorear el movimiento en X y Y de las coordenadas	Completada	Se aprecia comolos valores cambian en los campos de textomostrando las coordenadas XY en tiempo real.
LZ-I-S_03_VI	Reconocimiento y Objecttraking		El Sistema permite el acceso y da la bienvenida	El Sistema permite el acceso y da la bienvenida	Autenticado	N/A
LZ-I-S_04_VI	Reconociendo de Dispositivo	Poseer una cámara en el sistema	Obtener el listado de todos los dispositivos de video que se encuentran conectados al ordenador	Obtener el listado de todos los dispositivos de video que se encuentran conectados al ordenador	Completado	N/A

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Tabla 14 Pruebas de reconocimiento de voz

Caso de prueba reconocimiento de voz						
Id_Caso del Prueba	**Descripción del Caso de Prueba**	**Pre Requisito**	**Resultado Esperado**	**Resultado Obtenido**	**Estado**	**Observación**
LZ-I-S_01_RV	Reconociendo de Frase	Poseer una línea de entrada de audio, lugar libre de exceso de ruido ambiental	Cambiar de pantalla dependiendo de la palabra reconocida	El sistema se satura de ruido ambiental, por lo cual no muestra lo que debe mostrar en el caso de reconocimiento exitoso	Sin respuesta	N/A
LZ-I-S_02_AC	Reconocimiento de palabras	Poseer una línea de entrada de audio , lugar libre de exceso de ruido ambiental	Mostrar un msg si la palabra pronunciada es la esperada	Mostrar un msg si la palabra pronunciada es la esperada	Completado	N/A
LZ-I-S_03_RV	Reconociendo la Frase	Poseer una línea de entrada de audio , lugar libre de exceso de ruido ambiental	Cambiar de pantalla dependiendo de la palabra reconocida	Cambiar de pantalla dependiendo de la palabra reconocida	Completado	Se disminuye el ruido del ambiente usando un protector sobre la línea de entrada

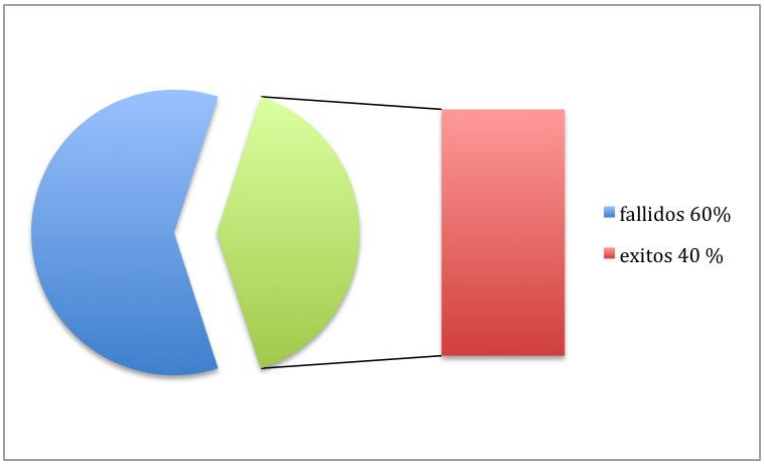
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

CONCLUSIONES

- Luego de la construcción e implementación del módulo de reconocimiento de voz, específicamente durante el periodo de pruebas de caja gris, se obtuvieron los siguientes resultados.

Número total de pruebas en reconocimiento de voz por "Frase": 40

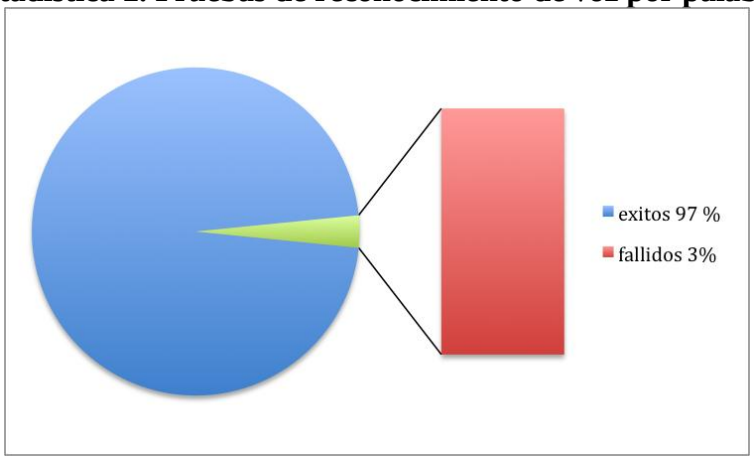
Figura 42 Estadística 1. Pruebas de reconocimiento de voz por frases



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Número total de pruebas en reconocimiento de voz por "Palabras": 40

Figura 43 Estadística 2. Pruebas de reconocimiento de voz por palabras



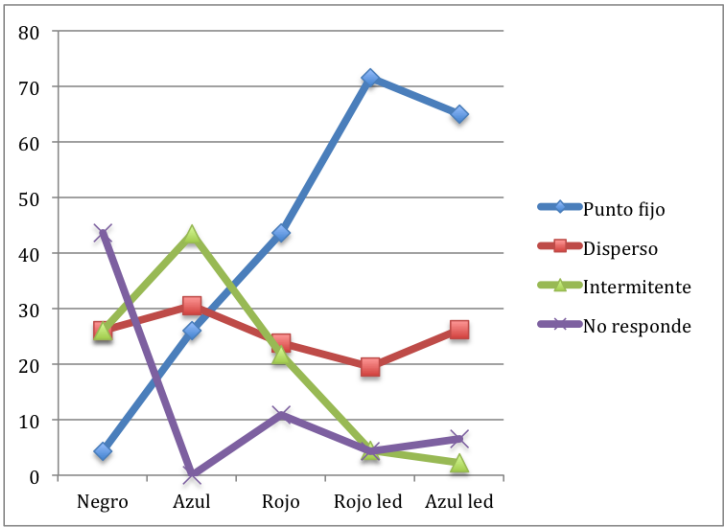
Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Por lo cual se concluye que al momento de utilizar métodos de reconcomiendo de voz en un ambiente con ruido, es preferible utilizar el método del reconocimiento de voz por "Palabra", puesto que es mucho más

eficiente que el método por “Frase”. Como se demuestra en la Figura 44, el método por “Frase” presenta una eficiencia menor al 40%, mientras que el método por “Palabra” es mucho más efectivo, brindando respuestas exitosas en el 97% del tiempo.

- Durante la ejecución de las pruebas de caja gris sobre el módulo de ObjectTraking, se ejecutaron las mismas pruebas con varios grupos de gamas de colores, para el análisis de factibilidad de uso de una determinada gama de color en RGB dentro del videoframe, de las cuales se obtuvieron los siguientes resultados.

Figura 44 Estadística pruebas gamas de colores



Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

Tabla 15 Relación entre colores

	Negro	Azul	Rojo	Rojo Led	Azul Led
Punto fijo	4,34	26,08	43,56	71,61	65
Disperso	26,04	30,48	23,87	19,53	26,2
Intermitente	26,04	43,4	21,68	4,43	2,24
No responde	43,56	0	10,85	4,39	6,52

Elaborado por: Cristhian Miranda, Edison Romero y Danilo Taco

De los colores analizados se determinó que los colores con mayor cantidad de brillo permiten que la aplicación ubique un punto determinado de una manera más estable y constante, puesto que sus gamas de colores son muy diferentes

a las de los objetos que se encuentran en un ambiente cotidiano de una bodega o biblioteca.

De los colores que se analizaron, los que poseen un brillo muy pobre son factibles de ubicar pero al momento de entrar en movimiento generan mayor cantidad de pérdidas en la señal que identifica al individuo en el frame de video.

Por lo cual se concluye que las gamas de colores brillantes y en especial el color rojo intenso que proporcionan los diodos led, es el más factible a la hora de implementar la aplicación de seguimientos de puntos sobre video en tiempo real.

- En cuanto a lo correspondiente al tema de enrutamiento el sistema está diseñado de manera que utiliza un conjunto de puntos interconectados entre sí formando un árbol de nodos conectados unos con otro, de tal manera que simulan una red por la cual transita la persona no vidente.

  Dicho principio simplifica la abstracción del ambiente a un esquema menos análogo y mucho más digitalizado facilitando el procesamiento del mismo, mejorando su precisión con cada aumento de un nodo o punto. El incremento del universo de nodos dentro del algoritmo de enrutamiento genera una carga de procesamiento en memoria directamente proporcional al número de puntos aumentados, es decir, que si incrementa el número de nodos incrementa el consumo de memoria disminuyendo en un grado muy ínfimo pero no depreciable en cuanto al performance del mismo.

  Por lo cual se concluye que para mejorar el rendimiento del sistema en cuanto a precisión se deben colocar los puntos de los nodos en lugares estratégicos y apuntando a la optimización en su número total.

- De la metodología de desarrollo y de gestión de proyecto podemos rescatar los siguientes puntos:

  - La utilización de un modelo de desarrollo multicapa orientado a abstraer la solución en módulos removibles e independientes es la mejor opción para un proyecto como este, puesto que su desarrollo fue impulsado para ser entregado en un concurso dentro de un cronograma muy apretado y con respuestas puntales, además de que por el tamaño del equipo humano que lo desarrolló y el corto tiempo para su estabilización y puesta en marcha, la filosofía de engranes que propone nuestra arquitectura es muy flexible y permite que todos los módulos se construyan en stand alone, sin importar que uno u otro esté como entrada del siguiente, lo importante es concretar las entradas y salidas de comunicación tal cual se lo hace cuando se interopera en sistemas SOA.

  - El utilizar una metodología de desarrollo ágil como lo es Extream Programing facilitó el desarrollo del proyecto de software puesto que al tener que solucionar temas tan apegados a la experiencia que tiene el cliente (persona no vidente), como son poder interactuar con él, y guiarlo desde un aplicativo autónomo, lo primordial es que la persona esté completamente cómoda y conforme con el resultado de cada una de las iteraciones con el sistema. Por lo cual se puede concluir que este tipo de metodología es la más apropiada al momento de querer captar el mayor porcentaje de aceptación por parte de cliente, además que es perfecta para grupos pequeños de desarrolladores que se comunican todo el tiempo.

RECOMENDACIONES

Luego del presente proceso de construcción y puesta en marcha del prototipo Lazar-I-Soft, el análisis de sus funcionalidades y las pruebas realizas, se espera que existan mejoras futuras que puedan obtener mejores resultados con tecnologías de mayor potencia que aún no están a nuestro alcance, por lo cual a continuación se describen las siguientes recomendaciones.

- En cuanto al algoritmo de enrutamiento A* que se utiliza en el presente proyecto, se recomienda analizar la posibilidad de mejorar el modelo matemático o implementar una ampliación en cuanto a los parámetros que utiliza la función heurística de búsqueda, o abrazar la posibilidad de utilizar otro tipo de estructura como las que se utilizan en redes, tales como "Algoritmos de enrutamiento por vector de distancia" que proporcionan mayor cantidad de parámetros para discernir cual es la ruta más óptima.

- Como se especifica en el documento, el proyecto Lazar-I-Soft, es un prototipo concebido bajo la premisa de que sus partes funcionalmente motoras como lo son: reconocimiento, visión y enrutamiento, sean removibles y que permita el acople. De tal manera se recomienda que en el caso de emprender un desarrollo sobre la base actual, primero se esté consiente en alto porcentaje de todo el aparataje que proporciona el proyecto actual, para poder empezar a cambiar uno a uno los componentes deseados.

- Por ser una solución diseñada para personas no videntes, el core de la misma debe ser configurada siendo muy prolijos a la hora de colocar los puntos en la Base de Datos, además de que la configuración debe ser realizada con un test en paralelo de lo que se está ingresando a la base de datos para así reducir el factor humano que podría colocar incoherencias en la data, desencadenando un efecto en domino de falla del sistema. Por lo que se recomienda que la persona asignada a realizar las configuraciones del sistema, se encuentre apoyado mínimo de una persona especialista, que le ayude a verificar y testear el software.

LISTA DE REFERENCIAS

Libros

- Beck, K. (1999). *Extreme Programming Explained. Embrace Change.* Pearson Education.
- Campderrich, B. (2003). Ingeniería del Software. Barcelona: UOC.
- Miranda, C., Romero, E., & Taco, D. (2012-2013). *Autores de la Tesis.* Quito.
- Noë, A. (2002). Vision and Mind. En *Selected Readings in the Philosophy of Perception* (pág. 229).
- Russell, S. J., & Norvig, P. (1996). *Inteligencia Artificial un enfoque Moderno.*

Netgrafía

- Academic. (2012). *Definición de Ruido.* Recuperado el 2 de 11 de 2012, de http://enciclopedia_universal.esacademic.com/63888/Ruido_%28sonido%29
- AForge.NET. (2.2.4). *Motion Detection.* Recuperado el 17 de junio de 2012, de http://www.aforgenet.com/framework/features/motion_detection_2.0.html
- Albizuri, X. (2010). *Procesamiento de imágen digital.* Recuperado el 12 de agosto de 2013, de http://www.sc.ehu.es/ccwalirx/gwdip3/imageprocessing.pdf
- Anónimo. (2010). *RDNZL - A.Net layer for Common Lisp*. Recuperado el 18 de junio de 2012, de http://weitz.de/rdnzl/
- Anónimo. (2001). *Técnicas más utilizadas aplicadas al Reconocimiento de Habla*. Recuperado el 15 de abril de 2012, de http://elies.rediris.es/elies12/cap2414.htm
- Anónimo. (2012). *Visión Artifical*. Recuperado el 16 de junio de 2012, de http://www.etitudela.com/celula/downloads/visionartificial.pdf
- Atienza Vanacloig, V. (2010). *El histograma de una imagen digital.* Recuperado el 14 de diciembre de 2013, de

http://riunet.upv.es/bitstream/handle/10251/12711/El%20histograma%20una%20imagen%20digital.pdf

- Cobarrubias Garcia, M. (2011). *Mapas Cartografía Digital*. Recuperado el 19 de diciembre de 2012, de http://sigmapascartografiadigital.blogspot.com/2011/09/cartografia-y-tipos-de-mapas.html
- *Diccionario de la Lengua Española.* (2013). Recuperado el 2013, de http://lema.rae.es/drae/
- *Diodo LED.* (s.f.). Recuperado el 13 de agosto de 2013, de http://www.monografias.com/trabajos60/diodo-led/diodo-led.shtml
- *El reconocedor de habla natural.* (s.f.). Recuperado el 11 de mayo de 2013, de http://www.lpi.tel.uva.es/~nacho/docencia/ing_ond_1/trabajos_02_03/Recon_voz/reconocedor.html
- Escalante, B. (2006). *Procesamiento Digital de Imágenes.* Recuperado el 12 de agosto de 2013, de verona.fi-p.unam.mx/boris/teachingnotes/Introduccion.pdf
- *Fuentes de Luz.* (2009). Recuperado el 12 de abril de 2012, de http://www.electricalfacts.com/Neca/Science_sp/light/sources_sp.shtml
- García, J. A. (2009). *Cámara Digital*. Recuperado el 19 de agosto de 2012, de http://www.asifunciona.com/practico/pr_camara_digital/camdig_4.htm
- GIS, Q. (s.f.). *Open Source Geographic Information System (GIS).* Recuperado el 21 de agosto de 2013, de http://www.qgis.org/
- Gómez, J. C. (2004). *Reconocimiento automático de Voz basado en técnicas de Comparación de Patrones*. Recuperado el 10 de noviembre de 2012, de http://www.fceia.unr.edu.ar/prodivoz/RAV_Comparacion_Patrones_bw.pdf
- *Iluminación para las aplicaciones de Visión Artificial.* (2005). Recuperado el 13 de abril de 2012, de Tipos de Iluminación 1-13: http://iaci.unq.edu.ar/materias/vision/archivos/apuntes/Tipos%20de%20Iluminaci%C3%B3n.pdf
- Infaimon. (2011). *Análisis de imágen y visión por ordenador.* Recuperado el 29 de mayo de 2012, de http://pserv.udg.edu/Portal/Uploads/4103862/CAMARAS_Infaimon.pdf

- Letelier, P., & Penadés, C. (2006). *Metodologías ágiles para el desarrollo de software*. Recuperado el 24 de julio de 2012, de http://www.cyta.com.ar/ta0502/b_v5n2a1.htm
- Microsoft. (2009). *Microsoft Agent in the MSDN Library*. Recuperado el 14 de noviembre de 2012, de http://www.microsoft.com/products/msagent/main.aspx
- *Organización Mundial de la Salud (OMS). Estadísticas*. (2013). Recuperado el 2 de agosto de 2013, de http://www.who.int/mediacentre/factsheets/fs282/es/
- Pesquera, A. (2010). *Aprendizaje en Inteligencia Artificial.* Recuperado el 17 de agosto de 2013, de http://www.sindominio.net/~apm/articulos/IAIC/aprendizaje/aprendizaje.pdf
- Wikipedia. (2009). *Algoritmo de búsqueda A**. Recuperado el 17 de agosto de 2013, de www.itescam.edu.mx/principal/sylabus/fpdb/recursos/r66692.DOCX
- Wikipedia. (2010). *Algoritmos de búsqueda en grafos*. Recuperado el 17 de agosto de 2013, de http://es.wikipedia.org/wiki/Algoritmos_de_b%C3%BAsqueda_en_grafos

ANEXOS

Manual de Instalación

Instalación y Configuración del Sistema Lazar-I-Soft

SISTEMA DE UBICACIÓN Y ENRUTAMIENTO PARA PERSONAS CON DISCAPACIDAD VISUAL EN UN AMBIENTE CONTROLADO POR MEDIO DE ALGORITMOS INTELIGENTES Y MEDIOS ÓPTICOS.

Quito, agosto 2013

Documento: Manual de Instalación – Instalación y Configuración del Sistema Lazar-I-Soft
Versión 1.0
Fecha de emisión 02/08/2013

1. INTRODUCCIÓN

En este manual se detalla el proceso de instalación de cada uno de los componentes que conforman el sistema Lazar-I-Soft en un entorno Windows, siendo estos: Base de Datos Mysql (motor de base de datos en donde se almacena las palabras reservadas), Microsoft Agent 2.0 (modo compatibilidad con Windows 7 para voz del sistema), Aplicación Lazar-I-Soft.

2. ENTORNO

Este manual de instalación fue escrito utilizando el siguiente entorno:

- Hardware
  - Mac Os X v. 10.7.5.
  - Procesador 2.9 GHz Intel Core i7.
  - Memoria RAM 8GB.
- Software
  - Sistema Operativo: Windows 7 Home Basic (Virtual).
  - 4GB de memoria RAM asignada.

3. INSTALACIONES

3.1. Instalación Mysql 5.1.68-Community

Ejecutar la aplicación descargada desde la página oficial de Mysql. Si se requiere realizar la descarga de este aplicativo, se la puede realizar desde el siguiente link: http://dev.mysql.com/downloads/mysql/

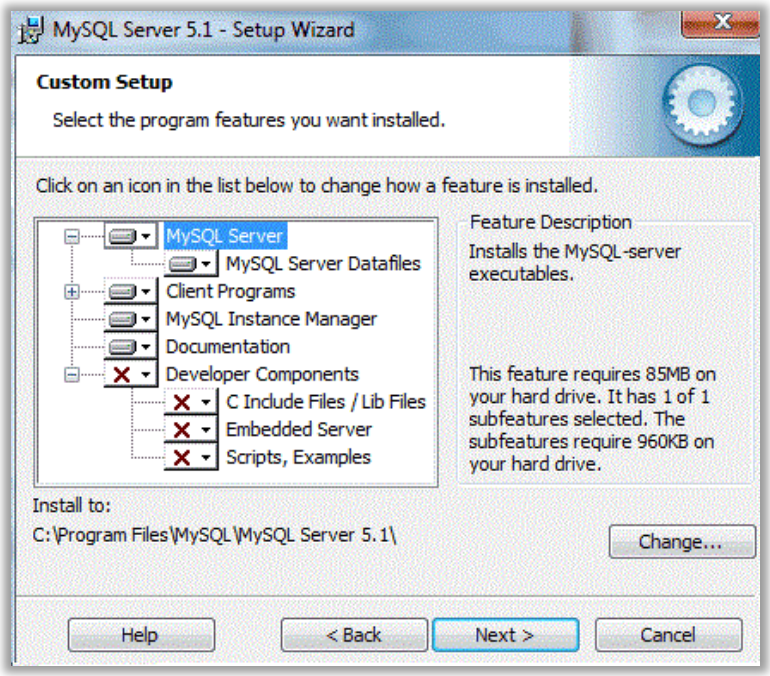
Figura 45 Bienvenida a la instalación de Mysql Server

El asistente de instalación de Mysql provee de tres tipos de instalación (Típica, Completa y Personalizada), para este manual se utiliza la instalación Personalizada.

Figura 46 Instalación personalizada

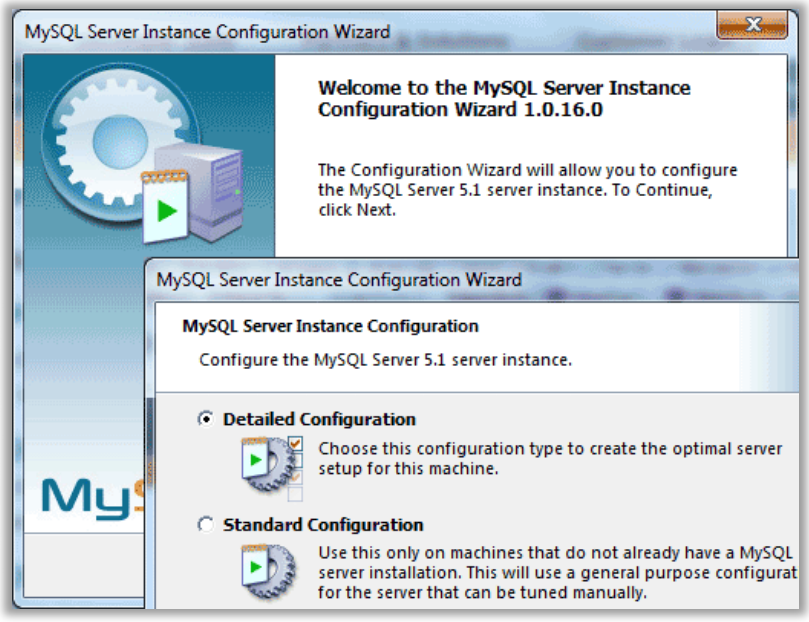
Seguidamente indica que complementos se pueden instalar en el sistema, se recomienda dejar todos los complementos para MysqlServer, Clientes y Documentación.

Figura 47 Complementos de instalación



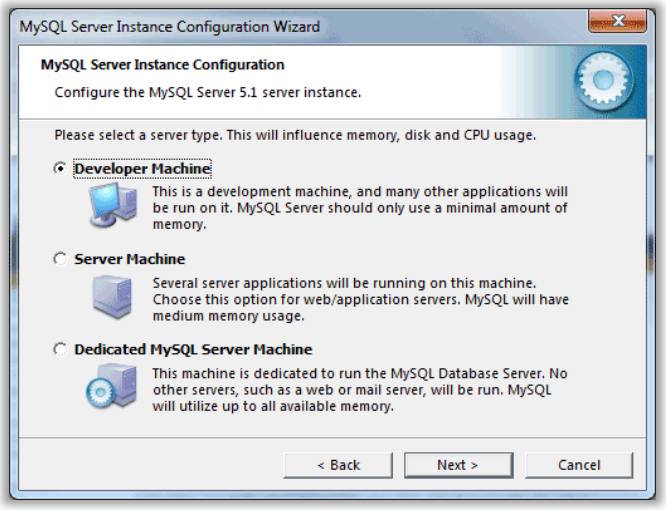
Al terminar la instalación se realiza la configuración del servidor, se debe realizar una configuración detallada, como se ve en la siguiente imagen.

Figura 48 Configuración detallada de instalación



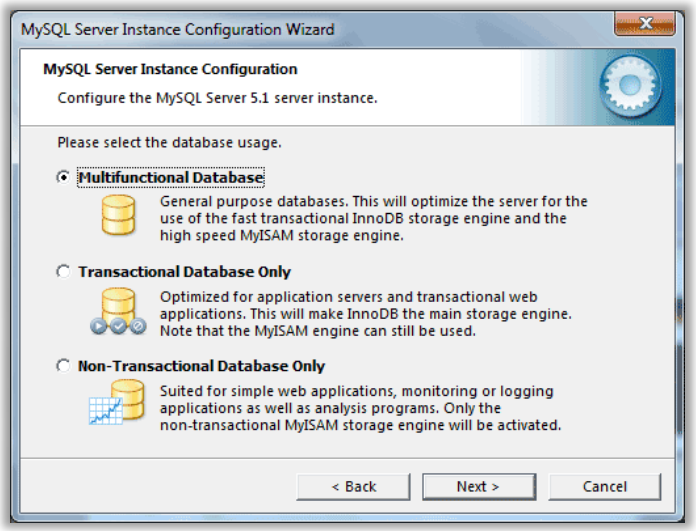
Al servidor de Mysql se recomienda configurar como máquina de desarrollo debido a que únicamente se la utiliza para guardar la información generada por el sistema Lazar-I-Soft.

Figura 49 Mysql para desarrollo



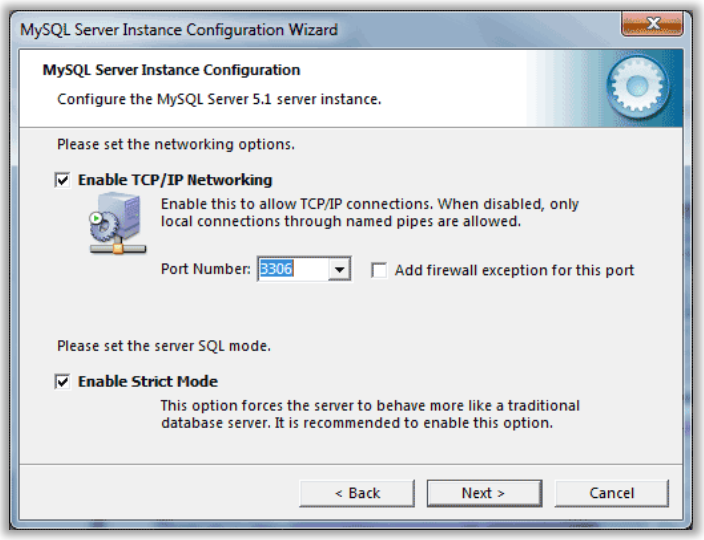
Se debe indicar que tipo de base de datos se va a utilizar en el servidor, es recomendable escoger base de datos multifuncional, para crear n esquemas de base de datos en caso de ser necesario.

Figura 50 Tipo de base de datos



Posteriormente se selecciona el tamaño de espacio en disco, número de conexiones concurrentes al servidor, y puerto en el cual se va a ejecutar, que por default y estándar lo se debe utilizar en el puerto 3306.

Figura 51 Puerto de ejecución servicio Mysql

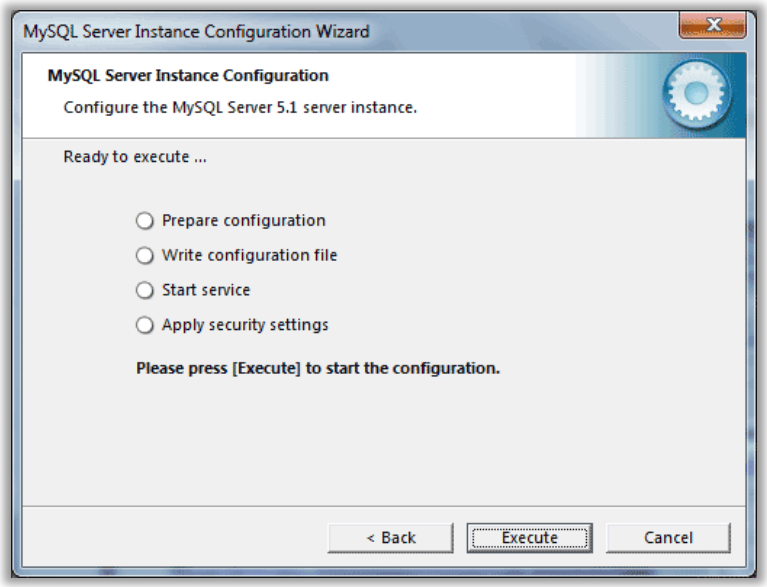


Se debe realizar una configuración de servicio de Windows para que se ejecute cada vez que inicie el Sistema Operativo, y se recomienda asignar una contraseña para el usuario root de Mysql.

Figura 52 Contraseña usuario root

Una vez terminada la configuración personalizada, el asistente de instalación, comienza a realizar sus tareas y configuraciones, finalmente termina la instalación de Mysql.

Figura 53 Finalización de instalación Mysql



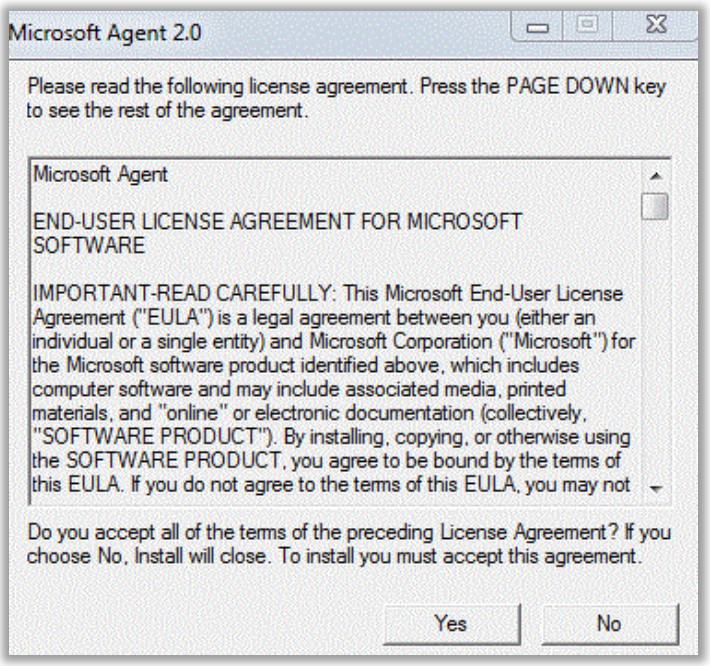
3.2. Instalación Microsoft Agent

Para la comunicación entre el usuario y la máquina se utiliza los complementos de Microsft Agent, los paquetes requeridos son los siguientes:

- MSagent.exe, utilizado en .Net.
- Peedy.exe para la animación de la voz.
- spchapi.exe para la pronunciación de la voz de la animación.

Cada uno de estos componentes los puede descargar de la siguiente url: http://microsoft-agent-spanish-pack.softonic.com/

Figura 54 Microsoft Agent



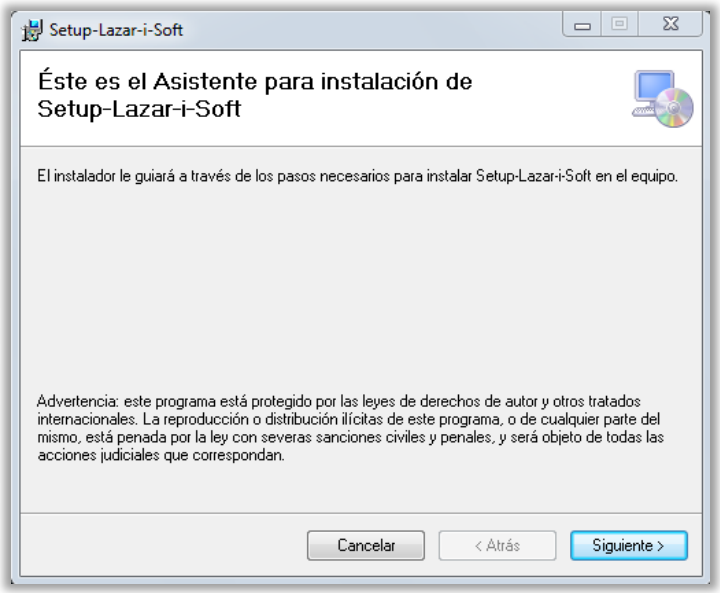
Dado a que estas aplicaciones no tienen pasos extensos en su instalación, se ha obviado pantallas de instalación.

3.3. Instalación Lazar-I-Soft

A continuación se detalla paso a paso la instalación de la aplicación Lazar-I-Soft en el sistema.

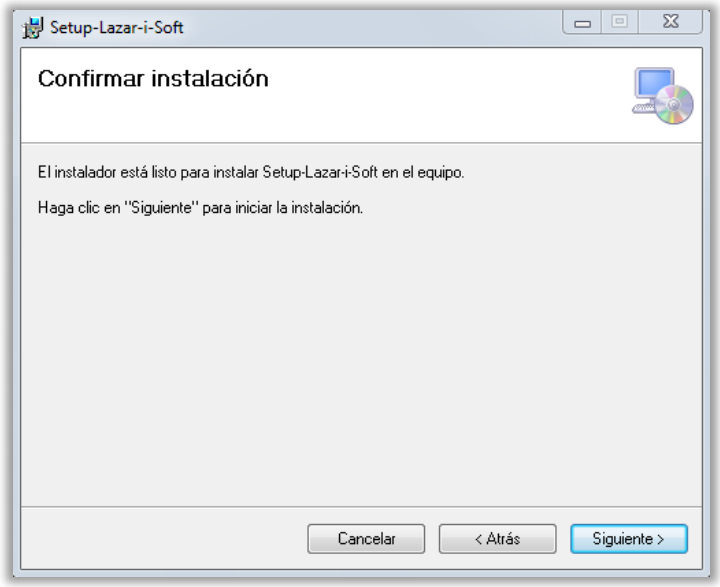
Ejecutar la aplicación (Setup) que será proporcionada por los estudiantes tesistas, seguidamente muestra el asistente de instalación indicando los derechos de la aplicación.

Figura 55 Bienvenida Asistente de instalación Lazar-I-Soft



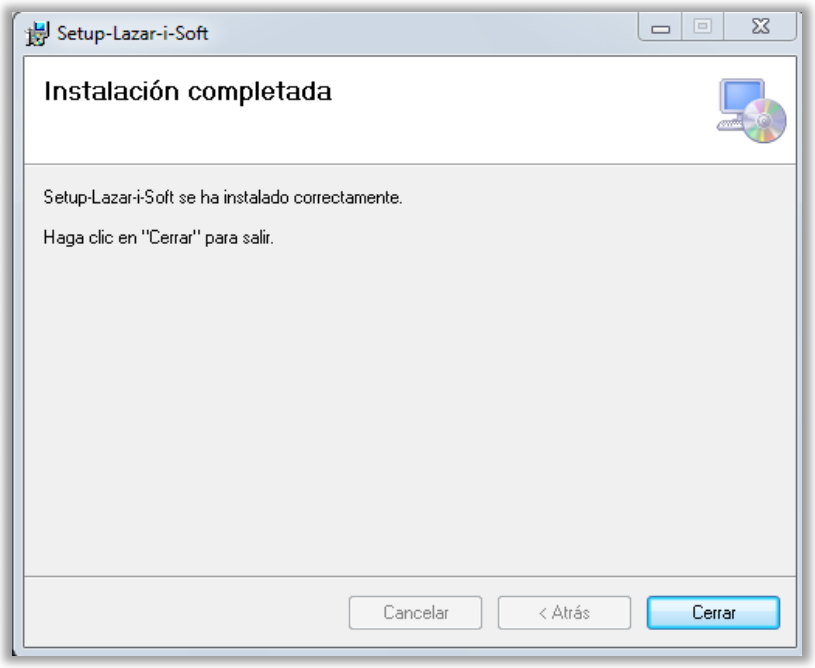
Una vez leído el mensaje sobre las leyes de la aplicación, se debe seleccionar la carpeta en donde la aplicación se va a instalar, y así mismo se indica para qué usuarios se va a instalar, lo recomendable es instalar únicamente para el usuario actual.

Figura 56 Selección de instalación



Indicando las configuraciones anteriores se procede a iniciar la instalación, dependiendo de las características de la máquina en donde se encuentra realizando dicha operación, demorará en terminar la instalación.

Figura 57 Finalización de instalación



Una vez terminada la instalación, puede utilizar la aplicación. Se espera que sea de su agrado y sea de mucha utilidad para la sociedad.

Manual de Usuario

Administración y Uso del Sistema Lazar-I-Soft

SISTEMA DE UBICACIÓN Y ENRUTAMIENTO PARA PERSONAS CON DISCAPACIDAD VISUAL EN UN AMBIENTE CONTROLADO POR MEDIO DE ALGORITMOS INTELIGENTES Y MEDIOS ÓPTICOS

Quito, agosto de 2013

Documento: Manual de Usuario – Administración y uso del sistema Lazar-I-Soft.
Versión: 1.0
Fecha de emisión: 02/08/2013

1. INTRODUCCIÓN

El presente documento tiene como objetivo describir de manera clara y concisa la forma de utilizar el sistema Lazar-I-Soft.

El sistema Lazar-I-Soft está desarrollado para enrutar a personas no videntes dentro de un ambiente desconocido para ellos, por lo que se recomienda consultar este manual antes de utilizar y configurar los diferentes componentes que posee el sistema. Estos componentes tienen características, funcionalidades, interfaces y métodos que le permitirán interactuar de mejor manera con el ambiente controlado.

2. GUÍA DE USO

Inicie Lazar-I-Soft. Haga clic en Inicio > Lazar-I-Soft > Lazar o si tiene el acceso directo en el escritorio, haga doble-clic sobre él.

Figura 58 Acceso directo al sistema Lazar-I-Soft

Figura 59 Interfaz de bienvenida del sistema Lazar-I-Soft

Al iniciar Lazar-I-Soft elija la opción que desea ejecutar. Pronuncie la palabra reservada ‘Iniciar’ para ejecutar inmediatamente la interfaz de enrutamiento o haga clic sobre los botones ‘Administración’, ‘Lazar’ para ejecutar la opción deseada.

Figura 60 Botones de acceso a interfaces de Lazar-I-Soft

3. INTERFACES

Tener siempre presente que el sistema Lazar-I-Soft es un sistema direccionado a mitigar las capacidades faltantes de un usuario en específico (Persona no vidente), por lo que el sistema carece de una interfaz gráfica para este tipo de usuarios, pero utiliza otros dispositivos para comunicarse con el mismo. Estos dispositivos son los auriculares y el micrófono, que receptan y emiten respectivamente los comandos parametrizados en el sistema.

3.1. Interfaz de Enrutamiento

La interfaz de enrutamiento del sistema Lazar-I-Soft es la encargada de enlazar las tres características principales del aplicativo y enrutar a la persona no vidente dentro del ambiente controlado, mientras el mismo se traslada de un nodo a otro, estas características son: visión artificial, inteligencia artificial y reconocimiento de voz.

Figura 61 Interfaz de enrutamiento

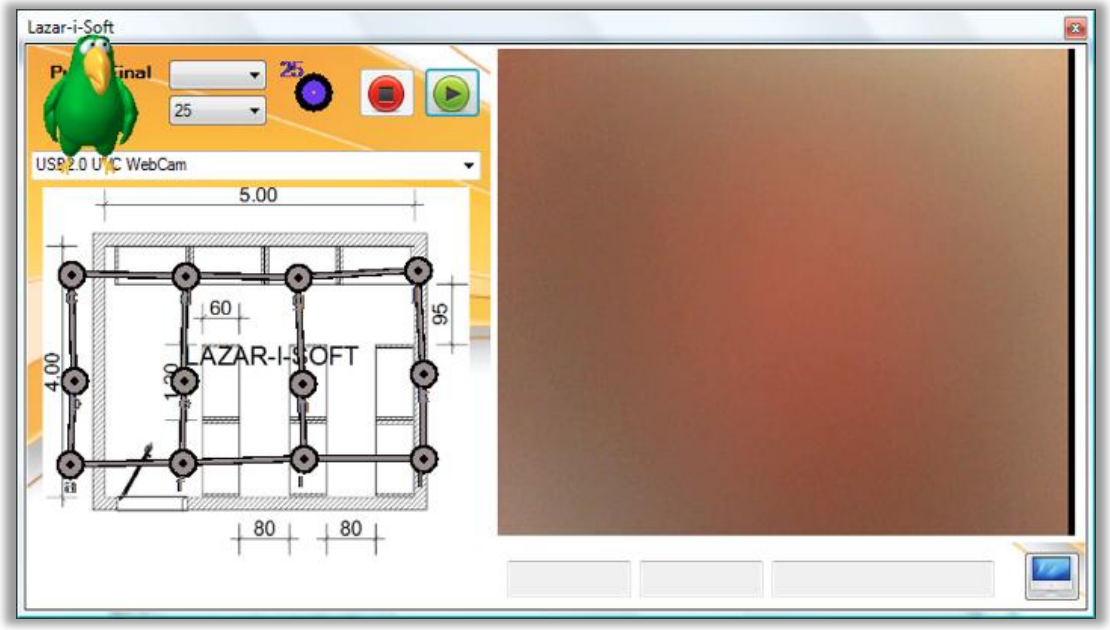
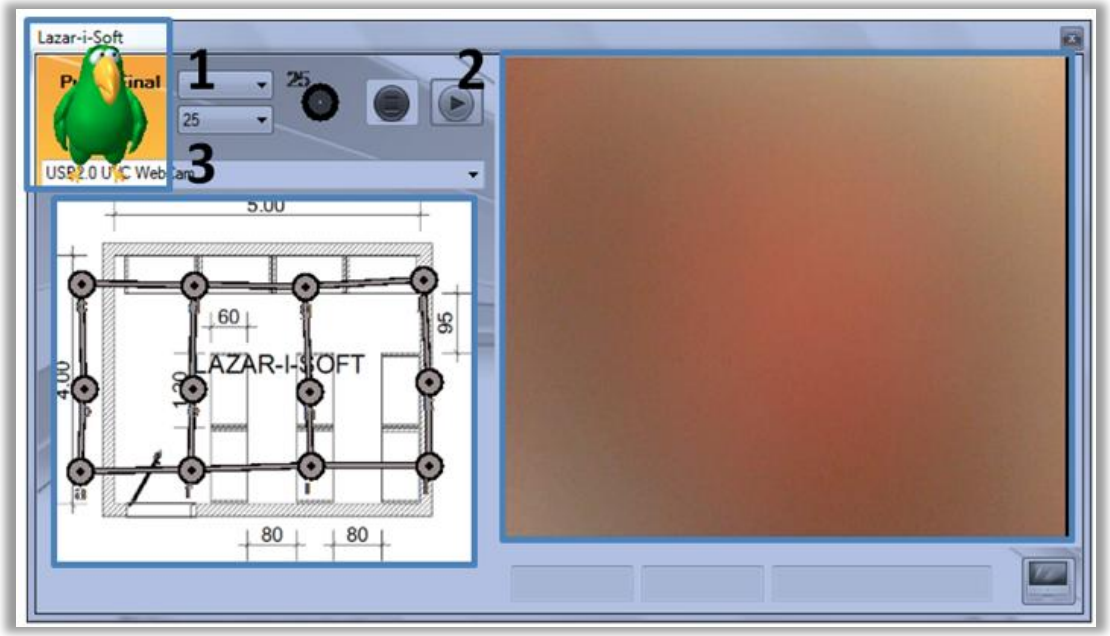


Figura 62 Interfaz de enrutamiento clasificado por componentes



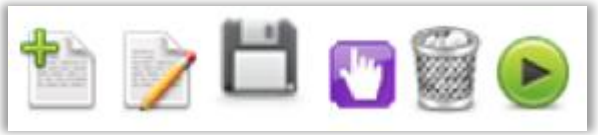
3.2. Interfaz de Administrador

Lazar-i-Soft posee una interfaz de administración la cual permite definir, configurar y editar los comandos de interacción con el usuario. Para acceder a estas opciones es necesario hacer clic en el botón de Administración, definido con el icono de la Figura 63.

Figura 63 Icono de administración del sistema Lazar-I-Soft

El botón de Administración despliega una interfaz que permite cambiar las diferentes opciones del sistema. Para esto se puede utilizar los diferentes botones de la Figura 7 que permiten ejecutar opciones especificas dependiendo de sus necesidades; entre estos botones están las opciones de Nuevo, Editar, Guardar, Seleccionar, Eliminar, Ejecutar, etc.

Figura 64 Botones del sistema Lazar-I-Soft



3.3.1. Administración de Comunicación

La interfaz de comunicación permite definir las palabras o comandos que utiliza el sistema para interactuar con el usuario final, entre estos comandos existen mensajes de advertencia, bienvenida, sonidos, instrucciones y palabras reservadas que son ejecutadas o emitidas por la persona no vidente dentro del ambiente controlado.

Figura 65 Interfaz de administración de comunicación

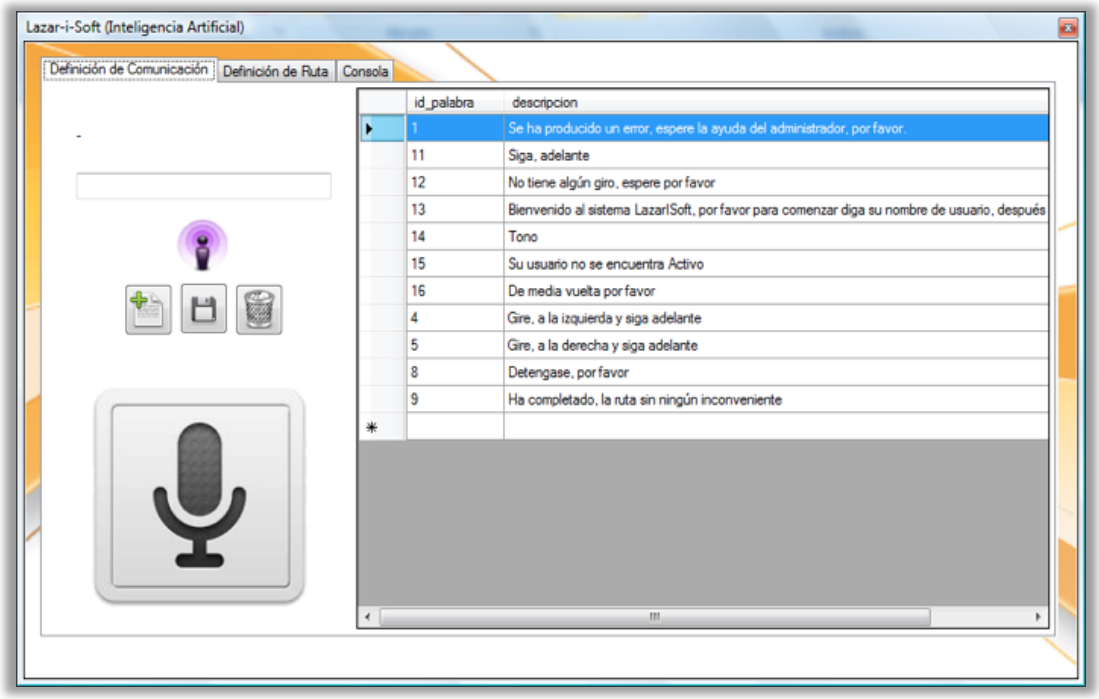
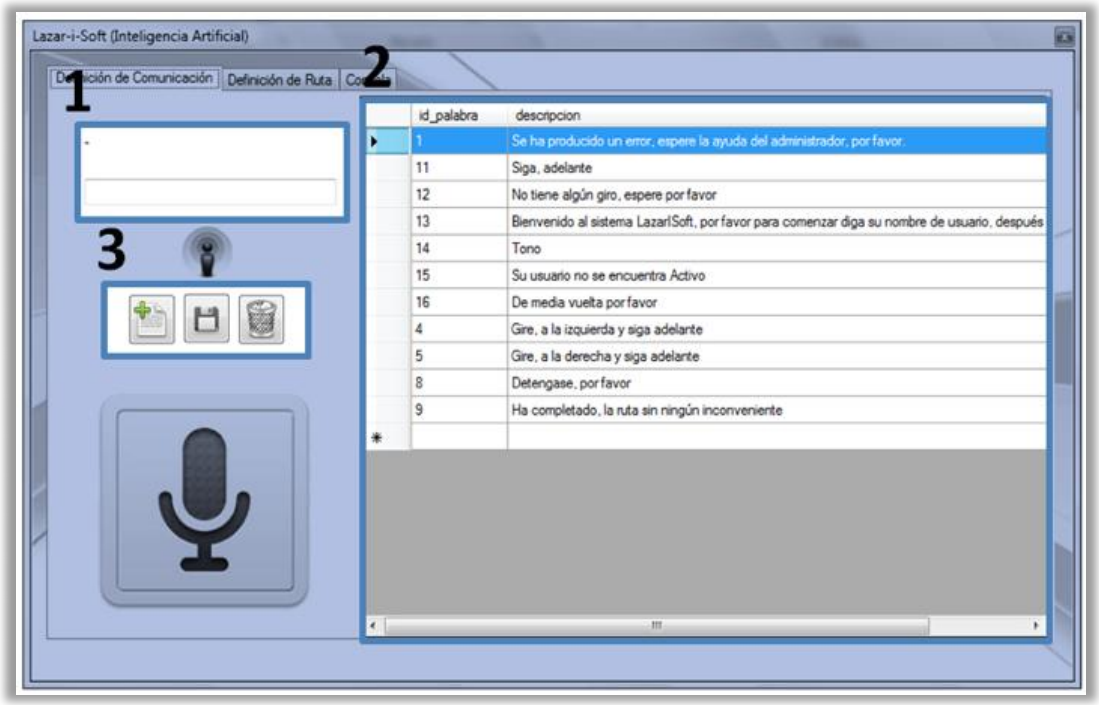


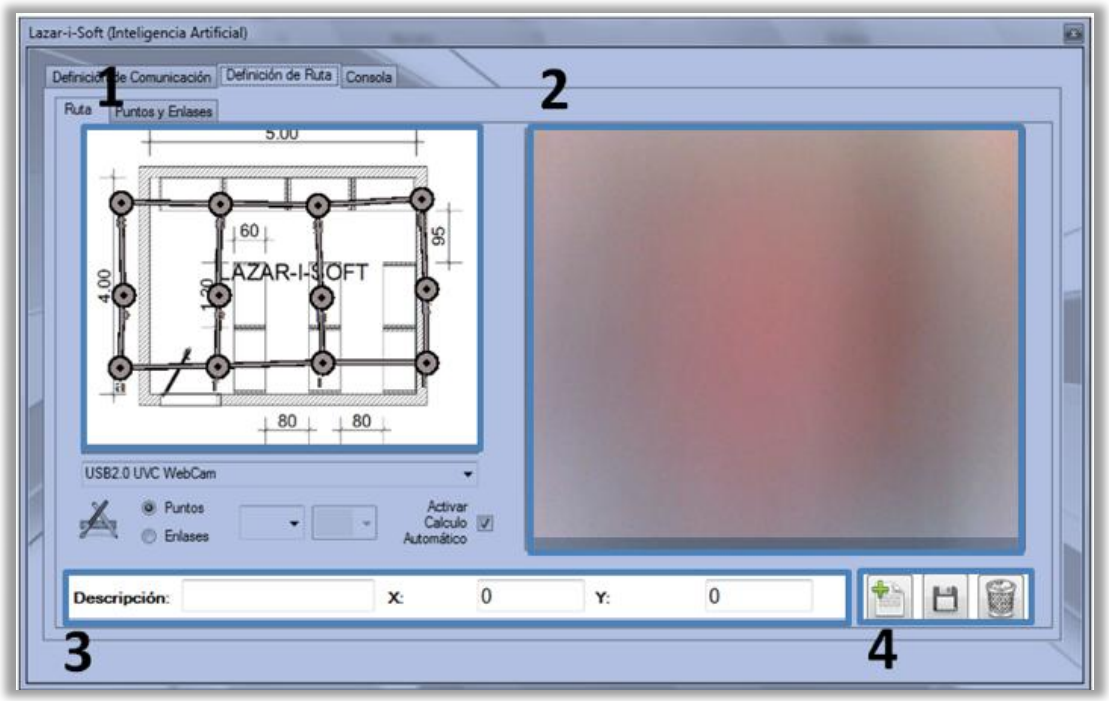
Figura 66 Interfaz de administración de comunicación clasificado



3.3.2. Administración de Enrutamiento

Esta interfaz permite reestructurar y definir el camino a seguir por la persona no vidente dentro del ambiente controlado, esta interfaz no define automáticamente la ruta si no que interactúa con la visión artificial para que sea más fácil la definición de la misma dentro del aplicativo.

Figura 67 Interfaz de administración de Rutas

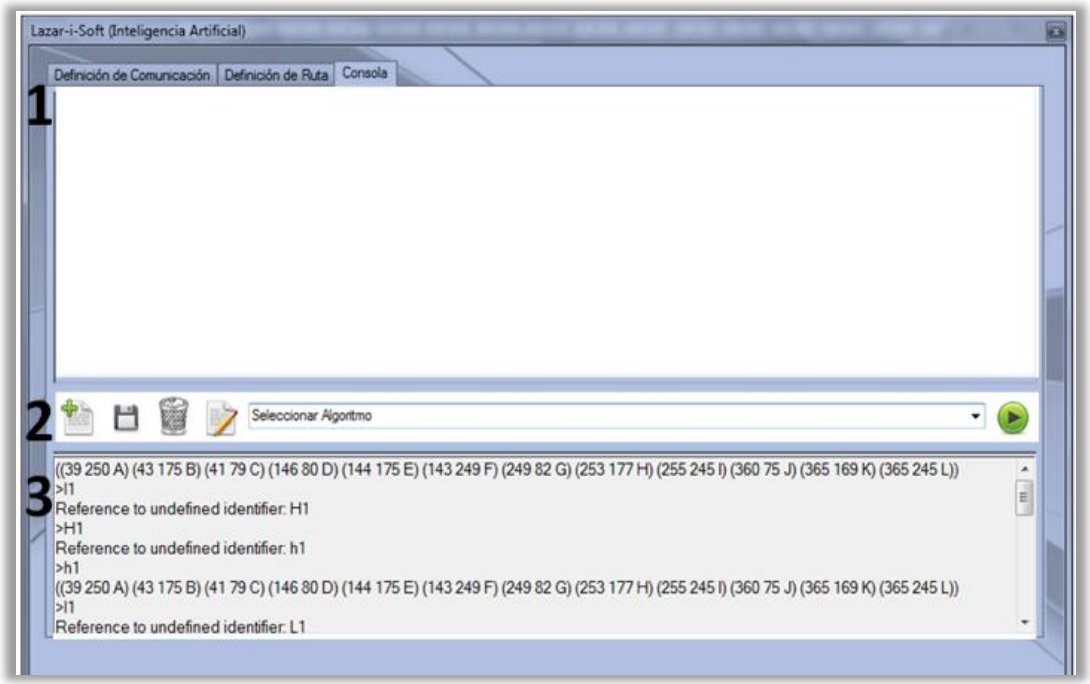


1	• Representación gráfica del ambiente controlado.
2	• Vision artificial.
3	• Area de edicion de Nodos.
4	• Botones de administración de Rutas

3.3.3. Administración de algoritmos de búsqueda

Permite administrar todos los algoritmos de búsqueda informada precargados en el sistema. Esta interfaz también permite monitorear los resultados desplegados por los algoritmos antes, durante y después de la generación de la ruta.

Figura 68 Interfaz de administración de algoritmos de búsqueda



1	•Cuadro de edición de Algoritmos de Búsqueda.
2	•Botones de administración de Rutas
3	•Cuadro de respuestas de algoritmos de busqueda.