Universidad Autónoma Metropolitana
Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Licenciatura en Ingeniería en Computación

Proyecto de Integración

Reporte Final

Prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla

Alumna:
Guerrero Angeles Claudia Ixchel
207205873

Profesora responsable:
Dra. Lizbeth Gallardo López

2014 Primavera

Fecha de entrega del reporte:
Agosto 29, 2014

Declaratorias:

Yo, Dra. Lizbeth Gallardo López, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Firma Asesor

Yo, Guerrero Angeles Claudia Ixchel, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Firma Alumno

Resumen

Para los niños con problemas del habla, comunicarse con los demás representa un reto. Estos niños necesitan terapia física, terapia psicológica y ejercicios de repetición de palabras para mejorar su comunicación. Se distinguen dos tipos de trastornos del habla: 1) Dislalia que es la dificultad en la producción de los sonidos requeridos para hablar y 2) Tartamudeo y falta de fluidez, los cuales son problemas a nivel de la calidad de la voz. En este documento se describe una herramienta de apoyo a niños de entre 5 y 7 años que presentan trastornos del habla. La herramienta es un juego computacional que hace uso de imágenes y audio que conducen al niño a la repetición de palabras. Esta herramienta podrá emplearse fuera del espacio terapéutico para aprovechar sus momentos de ocio. Consideramos que este juego puede ayudar en la práctica de la pronunciación de algunos fonemas que se conoce son un reto para estos niños.

Contenido

Índice de figuras

Índice de tablas

Índice de diagramas

1. Introducción

Para los niños con problemas del habla, comunicarse con los demás representa un reto. Estos niños necesitan terapia física, terapia psicológica y ejercicios de repetición para mejorar su comunicación. Se distinguen dos tipos de trastornos del habla: 1) Dislalia que es la dificultad en la producción de los sonidos requeridos para hablar. Las personas con dislalia pueden decir una palabra por otra o tener dificultad al pronunciar la “l” o la “r”; 2) Tartamudeo y falta de fluidez que son problemas a nivel de la calidad de la voz. Las personas con tartamudeo sufren de una interrupción en el flujo o ritmo del habla; con frecuencia se desconoce la causa [1] [2].

El proyecto "Prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla" tuvo como propósito construir una herramienta de apoyo a niños de entre 5 y 7 años que presentan trastornos del habla. La herramienta es un juego computacional que hace uso de imágenes y audio que conducen al niño a la repetición de palabras, esto fuera del espacio terapéutico, y aprovechando sus tiempos de ocio. El juego computacional consta de dos módulos principales: el módulo destinado al juego, el módulo entrenador del mismo juego y un módulo secundario que permite consultar el reporte de aciertos del niño. Para construir el juego, se adaptó un software de reconocimiento/análisis de voz, el cual permite obtener la pronunciación del niño para luego compararla con una lista de posibles pronunciaciones previamente establecidas.

Es importante mencionar que este juego no sustituye las terapias y ejercicios que el niño esté realizando en alguna clínica especializada; sin embargo, consideramos que puede servir de apoyo para el niño, su tutor y su instructor.

El documento consta de las secciones siguientes: en la sección 2 se describen los antecedentes sobre proyectos semejantes a este proyecto; en la sección 3 se presenta la justificación del proyecto para su aplicación como herramienta de apoyo a las terapias del niño; en la sección 4 se mencionan los objetivos general y específicos del proyecto; en la sección 5 se describe el marco teórico que respalda el desarrollo del proyecto; en la sección 6 se explica el desarrollo del proyecto, iniciando con la metodología empleada, el diseño del sistema, el uso del sistema y el terminando con el hardware y software necesario; en la sección 7 se presenta los resultados obtenidos; en la sección 8 se describe el análisis de los resultados; en la sección 9 se mencionan las conclusiones obtenidas; y finalmente, en la sección 7 se explica las conclusiones y el trabajo fututo del proyecto.

2. Antecedentes

En los campos de la psicología, pedagogía, educación y pediatría, actualmente se trabaja bajo programas terapéuticos que ayudan al niño a desarrollar la capacidad de producir sonidos; o bien, detener el deterioro en la calidad de la voz; estos programas terapéuticos, generalmente se realizan persona-persona, donde a través de ejemplos y repeticiones se le enseña al niño a pronunciar correctamente las palabras. Sin embargo, también se han realizado esfuerzos en ámbitos como biomédica y computación, para proponer dispositivos electrónicos y software que apoyen a las terapias presenciales; a continuación se mencionan algunos proyectos de este tipo.

2.1. Referencias externas

Ingenieros en Biomédica de la Universidad Iberoamericana, han desarrollado un dispositivo que ayuda a personas con problemas del habla en su comunicación con otras personas; incluso en los casos donde la persona solo puede mover una parte del cuerpo; por ejemplo la mano, el pie, la cabeza, el tronco o un párpado. Actualmente este dispositivo se encuentra en etapa de pruebas [3]

IBM ha desarrollado un programa para PC llamado *Speechviewer III;* utiliza retroalimentación visual y auditiva para analizar y mejorar las habilidades del habla en personas con trastornos del habla, del lenguaje o de la audición. *Speechviewer III* ofrece las siguientes ventajas: 1) permite visualizar los parámetros del sonido mostrando imágenes de como se articula la boca (posición de labios y lengua para producir sonidos); y 2) proporciona una retroalimentación de la producción del habla; además, 3) presenta información sobre el rendimiento y progreso del paciente [4] [5].

Di es un software que se distribuye gratuitamente por el Ministerio de Educación (MEC) del Gobierno de España a través del Centro Nacional de Información y Comunicación Educativa. Está orientado a niños con problemas de audición para que aprendan la lectura de los labios. Aunque también sirve para ayudar a aquellas personas que tienen problemas de articulación de las palabras. El programa consta de imágenes que muestran la articulación de los fonemas en una palabra o de los fonemas aislados [6].

2.2. Referencias internas

Actualmente, en la Universidad Autónoma Metropolitana Azcapotzalco, encontramos un reporte de proyecto computacional orientado a personas discapacitadas titulado “Sistema de reconocimiento del alfabeto dactilológico utilizando procesamiento de imágenes”, el objetivo del proyecto fue desarrollar una herramienta para el reconocimiento de las letras, con excepción de la “j” y la “z”, expresadas con las manos. Este proyecto fue realizado por la alumna Marín Díaz Lidia [7].

En cuanto a proyectos terminales concluidos de software de juegos se encontraron tres en la página electrónica de la División de Ciencias Básicas e Ingeniería en su apartado de Proyectos Terminales concluidos de Ingeniería en Computación: “Control software de ayuda para personas con limitaciones en sus extremidades inferiores” por la alumna Diana Romero Hernández. Cabe mencionar que los dispositivos electrónicos, que realizan las acciones, ya existían como resultado de otro proyecto terminal [8]; “Máquina de aprendizaje en un juego de ajedrez”, por el alumno Sánchez Sánchez Guillermo Augusto [9]; y “Plataforma de juego programable para Quoridor”, por los alumnos Hernández Hernández Fabrizio Alonso y Hernández Piña Hugo César [10]. Sin embargo, no pudimos acceder ni al reporte final, ni a la propuesta de estos proyectos.

Existe un proyecto terminal, concluido titulado “Prototipo para un sistema de apoyo a la comunicación de niños con problemas del habla” por parte de la alumna Erica Solano Duarte [11], el cual tiene un propósito similar al nuestro. Solano Duarte, propone una aplicación destinada a comunicar a un niño, con problemas del habla, con su tutor (cuando ambos están suficientemente lejos para emplear el lenguaje de señas). Cuando el niño tiene una necesidad del tipo: “quiero ir al baño”, “tengo sed”, etc., manda un mensaje a su tutor a través de un dispositivo móvil (PDA's). El niño tendrá una interfaz gráfica que revele las necesidades y el tutor recibirá un mensaje tipo SMS. En tanto que nuestra propuesta, está encaminada a proporcionar una herramienta de auto-aprendizaje para superar algunas deficiencias en la pronunciación de algunas palabras, mediante su repetición. Esta herramienta tiene forma de un juego computacional.

3. Justificación

El proyecto "Prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla" tuvo como propósito construir una herramienta de apoyo a niños de entre 5 y 7 años que presentan trastornos del habla. La herramienta es un juego computacional que busca favorecer el ejercicio cotidiano del niño, a partir de la pronunciación de palabras, cuya dificultad es reconocida por la literatura especializada. El juego emplea una serie de palabras asociadas cada una de ellas a una imagen, el niño interactúa con estas imágenes para escuchar la pronunciación correcta, seguido de un intento, por su parte, de repetir la misma palabra.

En ocasiones, el niño no puede asistir a las terapias que se tienen planeadas, o el tiempo para la siguiente cita es muy prolongado, y por lo tanto dejan de realizar los ejercicios. En otras ocasiones, los padres del niño no pueden realizar con él los ejercicios de repetición. En este sentido, el juego computacional que proponemos permite: i) Que el niño no interrumpa los ejercicios de pronunciación, ii) Que el niño de manera autónoma practique frente a la computadora, en cualquier momento, iii) Que el tutor y el terapeuta revisen los aciertos y los errores que el juego registra mientras el niño está realizando los ejercicios, iv) Que el tutor o el terapeuta amplíen la base de palabras e imágenes del juego.

4. Objetivos

4.1. Objetivo general

Desarrollar un juego por computadora (PC) que ayude a los niños de entre 5 y 7 años de edad con trastornos del habla a mejorar la pronunciación de algunas palabras.

4.2. Objetivos particulares

- Elección de herramienta de reconocimiento de voz.
- Diseñar y modelar el juego y su interfaz.
- Implementar un prototipo para el juego.
- Realizar pruebas de usabilidad con al menos un usuario especialista en el tema.
- Elaboración de entregables.

5. Marco teórico

5.1. Juegos por Computadora

Un juego se define como una actividad destinada a la diversión y esparcimiento de los involucrados, gracias a este tipo de actividades hemos aprendido a relacionarnos con nuestro entorno. Entre la gran variedad de juegos existentes, se encuentran aquellos cuyo objetivo es servir como una herramienta educativa para la comprensión de algunos temas. Con el desarrollo de la modernidad los juegos han ido evolucionando en la manera como se

llevan a cabo y el tipo de interacción que tienen los participantes de estos, con las nuevas tecnologías se ha incrementado el interés de los usuarios por el uso de la computadora por lo que los juegos han tenido que adaptarse a este medio, siendo lo que ahora conocemos como juegos por computadora [12].

Los juegos por computadora son populares en personas de todas las edades, principalmente porque son atractivos visual y auditivamente; existen varios tipos, siendo los de acción y simulación los más populares. Un juego por computadora se maneja a través de diversos dispositivos de entrada, por ejemplo: una palanca y un botón, un control con varios botones, el ratón y teclado de la computadora. Los juegos por computadora permiten desde un punto de vista educativo [13]:

- Adquirir nuevos conocimientos.
- Poner en práctica conocimientos conceptuales, procedimentales y actitudinales.
- Desarrollar habilidades psicomotrices, así como la coordinación mano-vista.
- Desarrollar habilidades de pensamiento crítico, estrategia y toma de decisiones.
- Adquirir habilidades relacionadas con el mundo digital.
- Desarrollar actitudes de superación y autoestima.
- Aprender a compartir y colaborar con el otro.
- Potenciar la fantasía, la imaginación y la creatividad.

5.2. Autoaprendizaje

El término autoaprendizaje se puede definir como la manera de aprender por uno mismo. Consiste en aprender mediante la búsqueda individual de información y la realización, también individual, de prácticas o experimentos [14]. El autoaprendizaje se puede realizar desde temprana edad sin que uno note que lo realiza, esto es a través de juegos y actividades recreativas. Jugar, aunque a veces no se tiene presente, tiene la función principal de aprender nuevas habilidades o mejorar las que ya se poseen.

En el Aprendizaje Auto-dirigido la persona aprende de forma autónoma a través de la lectura, el análisis, la reflexión, la realización de tareas, la búsqueda de información y otras actividades que le permitan desarrollar habilidades, actitudes y valores para desarrollarse en la sociedad [15].

Entre las ventajas del autoaprendizaje se encuentran [15]:

- Fomenta la curiosidad, la investigación y sobre todo la autodisciplina.
- Se aprende a resolver problemas por uno mismo.
- Como al principio puede comenzar como un juego, esto puede ser muy divertido, lo cual hace casi transparente el aprendizaje.
- El hecho de no tener que seguir el ritmo de un grupo de personas determinado, da la libertad de dedicar más tiempo a lo que realmente nos gusta o a aquello que nos es difícil aprender.
- Es constructivo, esto quiere decir que trae consecuencias positivas y ayuda al crecimiento tanto intelectual como emocional del individuo.
- No solo permite adquirir o desarrollar una habilidad en concreto sino formar nuestra

personalidad de forma positiva y dinámica.

5.3. Trastornos del habla

Un “trastorno del habla o lenguaje” se refiere a los problemas de la comunicación ó a problemas en las funciones motoras orales. Estos trastornos varían desde simples substituciones de sonido hasta la inhabilidad de comprender o utilizar el lenguaje. Algunas causas de los impedimentos del habla o lenguaje incluyen la pérdida auditiva, trastornos neurológicos, lesión cerebral, discapacidad intelectual, abuso de drogas, impedimentos tales como labio leporino, y abuso o mal uso vocal. Sin embargo, con mucha frecuencia se desconoce la causa [16].

Se distinguen dos tipos de trastornos del habla: 1) Dislalia que es la dificultad en la producción de los sonidos requeridos para hablar. 2) Tartamudeo y falta de fluidez que son problemas a nivel de la calidad de la voz.

El niño con dislalia, omite (sopa=.opa) o sustituye el sonido por otro (sopa=topa). Con frecuencia sabe pronunciar las sílabas separadas, pero expresa incorrectamente la unión de fonemas; por ejemplo, /b/ (be) la expresa bien, pero al pronunciar la palabra "bola" probablemente dirá "pola" [17].

La tartamudez es un trastorno caracterizado por repeticiones y prolongaciones de las unidades del habla: sonidos y sílabas, lo cual ocurre frecuentemente y es difícil de controlar. Este trastorno se acompaña de manifestaciones motoras del aparato fonoarticulatorio o de cualquier otra parte del cuerpo, dando la apariencia de un esfuerzo relacionado con el acto de hablar. Con frecuencia también se acompaña de un estado emocional de magnitud variada: una tensión, un estado de ansiedad, vergüenza y miedo [17].

La falta de fluidez (farfulleo) es un trastorno caracterizado por un ritmo rápido del habla con interrupciones en la fluidez, pero sin repeticiones o indecisiones. El habla es errática y sin ritmo, con súbitos brotes de excitación que generalmente implican formas incorrectas de la construcción de las frases; por ejemplo, alternancia de pausas y explosiones del habla dando lugar a una oración con grupos de palabras sin una correcta estructura gramatical [18].

5.4.Sistemas de reconocimiento de voz

Este tipo de sistemas permiten que las computadoras "escuchen" la voz de una persona y reconozcan lo que dijo. Para lograr una precisión en el reconocimiento de voz y un tiempo de respuesta razonables, los reconocedores de voz actuales acotan lo que escuchan a través del uso de gramáticas; por lo tanto, los reconocedores de voz actuales aún no tienen la capacidad de escuchar cualquier discurso, con un contexto determinado y transcribirlo con precisión [24].

El JSpeech Grammar Format (JSGF) define una plataforma que describe un tipo de gramática, con un conjunto de reglas conocido como gramática regular. Una gramática regular utiliza una representación textual que es legible y editable tanto para los desarrolladores como para las computadoras. Esta representación textual puede incluirse en el código fuente de alguna aplicación para el reconocimiento de voz. Una regla gramatical especifica (mediante una frase escrita) los tipos de expresiones que una persona podría decir de manera oral. Por ejemplo, una gramática simple permite distinguir entre dos

comandos sencillos para una computadora, tales como: "cerrar el archivo" y "cerrar la ventana" [24].

Algunos ejemplos de sistemas de reconocimiento de voz son:

- **FreeTTS:** Sintetizador de voz de código abierto escrita completamente en el lenguaje de programación Java [25].
- **"Speechfor Java" de IBM**: Aplicación basada en el producto ViaVoice de IBM, que apoya en el dictado, mando y control, y síntesis de voz. Es compatible con todas las versiones de idiomas europeos reconocidos por ViaVoice, a saber: inglés de EE.UU. y Reino Unido, francés, alemán, italiano, español y japonés [26]. Para su uso se debe pagar una licencia.
- **CMU Sphinx**: Sistema de reconocimiento de voz desarrollado en la Universidad de Carnegie Mellon. Incluye una serie de programas para el reconocimiento de voz y un entrenador acústico [27]. Es de código abierto.
- *TalkingJava SDK* es *framework* que implementa a *Java Speech API (JSAPI)* [23] para el reconocimiento de voz*,* y es propiedad de Cloud Garden. Con esta framework es posible realizar un dictado simple, el framework reconoce las palabras expresadas de manera oral. El framework estuvo disponible en [28] en el año de 2010 pero actualmente no está disponible.

6. Desarrollo del proyecto

6.1. Metodología empleada para el desarrollo del proyecto

La metodología de desarrollo que se empleó es el *Rational Proceso Unificado (RUP),* principalmente por establecer un proceso de desarrollo iterativo e incremental, lo cual permite una mejora continua del sistema.

El RUP es un modelo de software, ver Figura 1, que permite el desarrollo a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad, como el estándar ISO 9000-3 y el modelo CMMI para el Desarrollo, por mencionar unos ejemplos [19].

El RUP emplea el *Unified Modeling Language* (UML) en la preparación de todos los artefactos del sistema, por ejemplo: casos de uso de texto, diagrama de clases, diagrama de secuencia, entre otros [20].

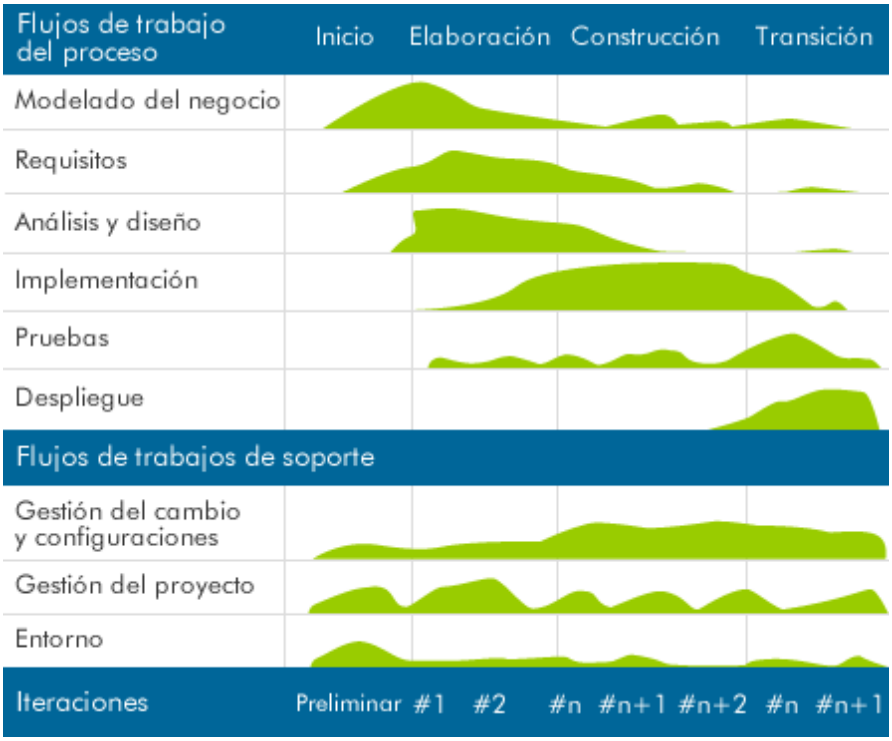

Figura 1. Ciclo de vida iterativo e incremental del RUP

(Fuente: http://www.orangeti.com/)

El RUP o se divide en cuatro fases y en nueve disciplinas. Cada fase se subdivide en iteraciones. En cada iteración se desarrolla en secuencia un conjunto de disciplinas o flujos de trabajos Las fases son: Inicio, Elaboración, Construcción y Transición. Las disciplinas del proceso son: Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas, Despliegue, y las disciplinas de soporte son: Gestión del cambio y configuraciones, Gestión del proyecto, Entorno. A continuación explicamos cada fase y las disciplinas realizadas para el proyecto "Prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla**".**

Fase de inicio. En esta fase se definió el alcance del proyecto, a partir de los requisitos del sistema. Los principales módulos del sistema son tres: i) Un módulo destinado al juego, ii) Un módulo entrenador del mismo juego y iii) Un módulo que permite generar un reporte de aciertos del niño. Se estableció la planificación del proyecto: número de iteraciones para cada una de las disciplinas (*workflows*) que marca la metodología.

Fase de elaboración. En esta fase se realizó el análisis y el diseño de la aplicación, a través varias iteraciones sobre los casos de uso de texto: jugar palabras, entrenar palabras, generar reporte, y consultar reporte; también se planteó la arquitectura general del sistema; además, se lingüística inicio a la implementación de módulo destinado al juego.

Fase de construcción. En esta fase se implementaron los módulos: Entrenar palabras y Generar reporte, iterando varias veces a partir de los resultados que revelaron las pruebas de unidad de cada uno de los módulos.

Fase de transición. En esta fase se realizó la integración de los módulos y se realizaron pruebas de sistema, corrigiendo los errores y defectos encontrados en estas pruebas. El sistema no ha sido puesto en marcha, para ello se requiere una fase de pruebas de

usabilidad con los terapeutas y personal especializado.

6.2. Diseño del sistema

6.2.1. Diagrama de casos de uso

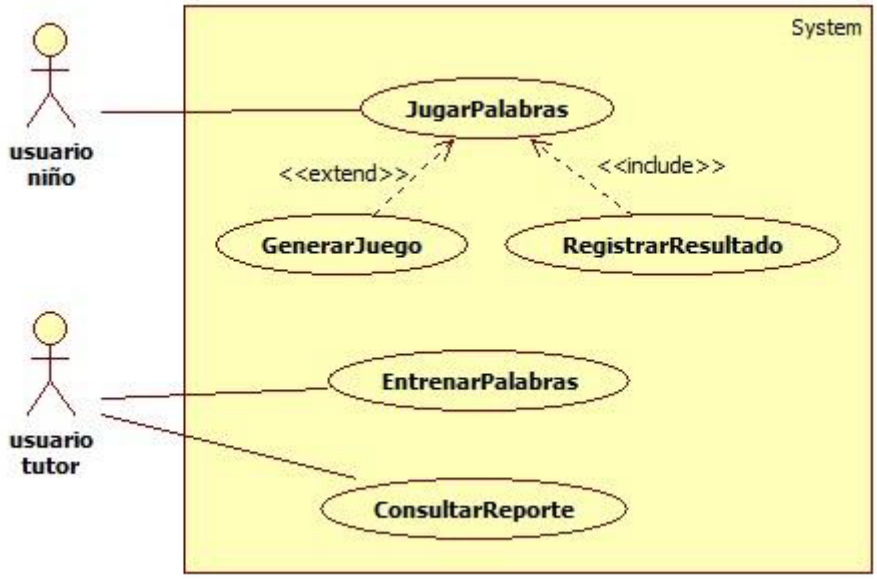

Diagrama 1. Diagrama de Casos de Uso General

El diagrama de Casos de Uso General (ver Diagrama 1), está conformado por cinco casos de uso: `JugarPalabras`, `GenerarJuego`, `RegistrarResultado`, `EntrenarPalabras` y `ConsultarReporte`. Los actores del sistema son: niño y tutor (o instructor).

- **`Niño`:** Este actor está relacionado con el caso de uso `JugarPalabras`, es quien elige entre las imágenes y repite la palabra asociada a esta imagen. El caso de uso `JugarPalabras` extiende hacia el caso de uso `GenerarJuego` e incluye el caso de uso `RegistrarResultado` de las pronunciaciones realizadas por el niño.
- **`Tutor`:** Este está relacionado con los casos de uso: `EntrenarPalabras` y `ConsultarReporte`, es quien de acuerdo al progreso del niño, gestiona el sistema añadiendo, modificando, eliminando y consultando las palabras, sus imágenes y audio asociados. Además puede consultar el reporte sobre los aciertos y errores del niño registrados durante el juego.

6.2.2. Especificación funcional de los módulos del sistema.

El sistema está conformado por dos módulos: `JugarPalabras` y `EntrenarPalabras`.

JugarPalabras recibe las entradas del niño a través de las imágenes proporcionadas, y éste debe contar con las siguientes operaciones: Generar el juego y registrar el resultado del juego.

- Generar el juego implica mostrar una serie de imágenes seleccionadas al azar de entre las disponibles, además de cargar el audio asociado a cada una de las imágenes.
- Registrar el resultado del juego implica hacer persistente la pronunciación del niño, y esta debe compararse con las pronunciaciones que se tienen registradas para la palabra en cuestión.

EntrenarPalabras, incluye las operaciones: dar de alta, modificar, consultar y borrar. Este módulo interactúa con el tutor, y permite entrenar el juego con nuevas palabras e imágenes asociadas.

- Alta. El tutor agrega una nueva palabra, la imagen y el audio asociados; además proporciona las posibles pronunciaciones de dicha palabra.
- Modificar. El tutor actualiza la imagen y audio de una palabra del acervo.
- Consultar. El tutor consulta la palabra, su imagen y audio asociados, que ya se encuentra en el acervo. Esto permite al tutor notar posibles errores, tales como: una palabra repetida, una imagen o audio que no correspondan.
- Borrar. El tutor elimina una palabra, su imagen y audio asociados cuando no ya no se requiera o cuando encuentre algún error.

6.2.3. Casos de uso de texto.

A continuación se muestran dos casos de uso de texto: JugarPalabras y RegistrarResultado.

- **JugarPalabras**

Caso de Uso	JugarPalabras
Actores	niño
Tipo	Básico
Resumen	El niño elegirá una de las imágenes mostradas y repetirá la palabra indicada por el sistema.
Pre-condiciones	Se obtienen de manera aleatoria 6 imágenes del directorio imagen y se presentan en la pantalla.
Post-condiciones	Se reproducirá un sonido dependiendo de la imagen seleccionada y el niño la pronuncia después.
Flujo principal	1. El sistema carga 6 palabras con sus respectivas imágenes y su audio (pronunciación correcta y las pronunciaciones posibles), y presenta las

	imágenes en la pantalla principal. 2. El niño elige una de las imágenes. 3. El sistema emite la pronunciación correcta de la palabra correspondiente. 4. El niño pronuncia la misma palabra. 5. El sistema registra la voz del niño. 6. El sistema reconoce la voz del niño y busca en el archivo diccionario.txt la pronunciación más cercana y la almacena. 7. El sistema registra el resultado de la pronunciación correcta y la pronunciación del niño: RegistrarResultado. 8. El caso de uso se continúa en el paso 2.
Subflujo	a) El niño en todo momento puede seleccionar salir, en ese caso el sistema termina su ejecución. b) El niño en todo momento puede seleccionar ver las instrucciones del juego, en ese caso el sistema la muestra en una ventana.

- **RegistrarResultado**

Caso de Uso	RegistrarResultado
Tipo	Inclusivo
Resumen	Se registrará en un archivo de texto plano la palabra correcta y la pronunciación del niño
Pre-condiciones	Se está realizando el caso de uso JugarPalabras.
Post-condiciones	Se termina el caso de uso en espera de un nuevo evento
Flujo principal	1. El sistema crea el archivo que nombrará con la fecha actual al momento de iniciar el juego. 2. El sistema abre el archivo recién creado para registrar la palabra correcta y la palabra dicha por el niño. 3. El sistema regresa al caso de uso JugarPalabras para continuar en el paso 2.
Subflujo	

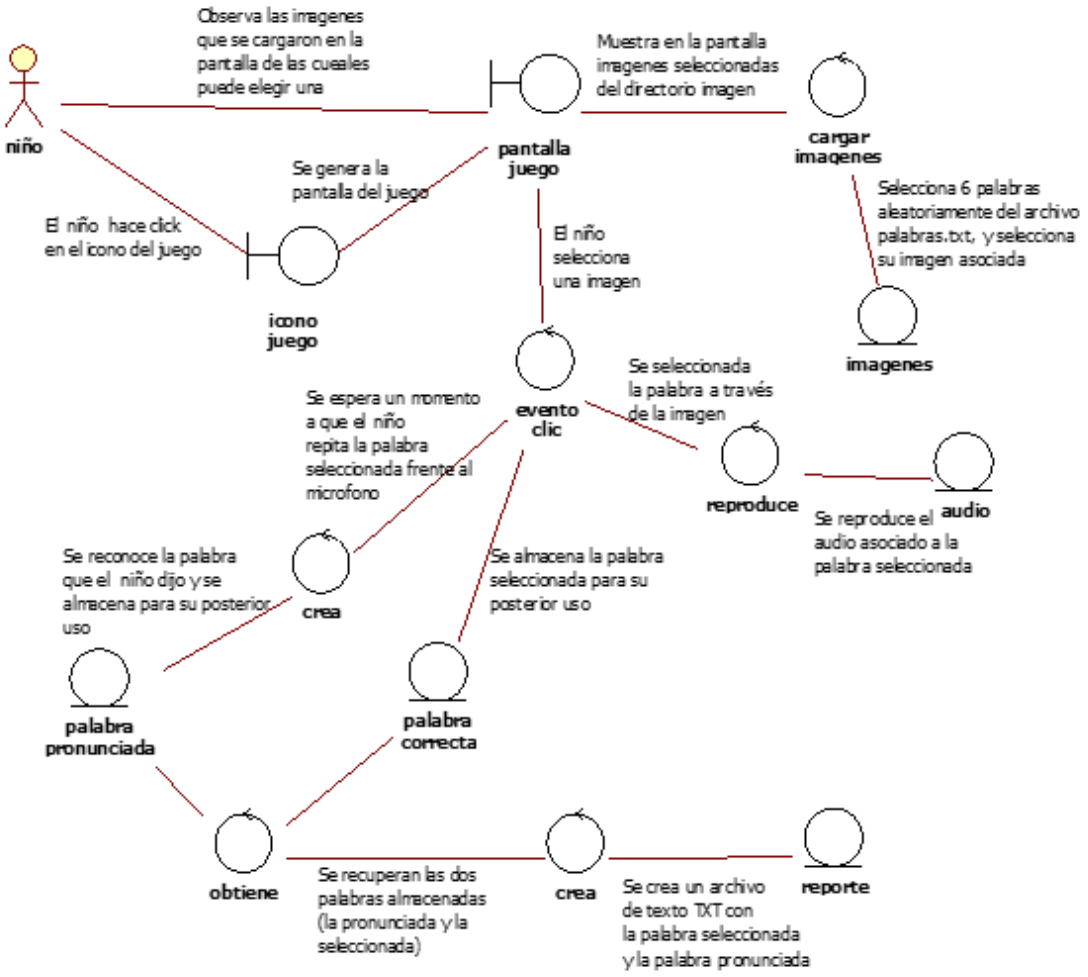

Diagrama 2. Diagrama de Robustez – Casos de uso `JugarPalabras` *y* `RegistrarResultado`

6.2.4. Diagrama de clases

- Para el proyecto se plantearon dos diagramas de clases para cada uno de los módulos: `JugarPalabras` y `EntrenarPalabras`.

- **`JugarPalabras`.** El Diagrama 3 muestra tres clases principales: `Juego`, `Escucha`, `Escribe`.

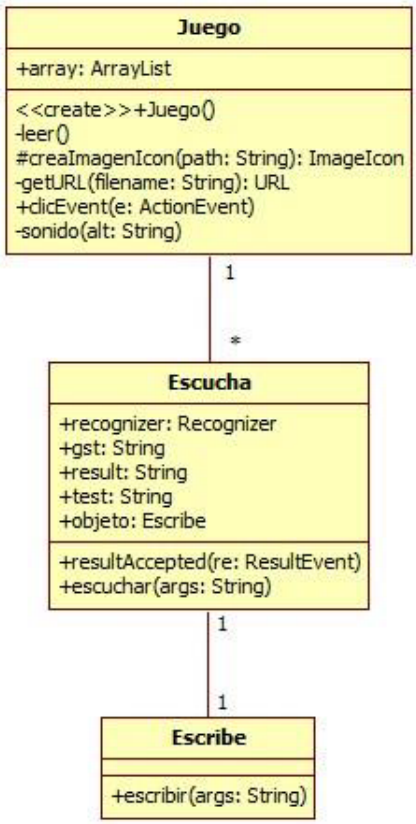

Diagrama 3. Diagrama de Clases – `Juego`

La **clase `Juego`** carga la interfaz que interactúa con el niño. Además, los métodos `leer()` y `creaImagenIcon()`, `getURL()`, `clicEvent()` y `sonido()` están encargados de generar el juego. La descripción de los elementos con los que cuenta esta clase se muestra en la Tabla 1.

Nombre	Observaciones
Atributos	
`array`	Tipo: `ArrayList`.
Constructor	
`Juego()`	Inicializa los componentes *JavaSwing*, y muestra las imágenes aleatoriamente en base a los elementos almacenados en el atributo `array`.
Métodos	
`leer()`	Lee el archivo `palabras.txt` y almacena su contenido en el atributo `array`.
`creaImagenIcon()`	Recibe como parámetro, una cadena que será la ruta de acceso a la imagen, el método lo recibe, y si la URL es correcta,

	regresará `ImagenIcon` con la imagen proporcionada para mostrarse en los botones inicializados por el constructor `Juego()`.
`getURL()`	Recibe como parámetro una cadena que será la ruta de acceso al audio, de acuerdo a la imagen seleccionada con un clic, de ser correcta, regresará la URL del audio al método sonido
`clicEvent()`	Cada que el niño da clic en alguna de las imágenes en la interfaz, se creará un objeto de tipo `Escucha()`, y se llamará al método `sonido()`.
`Sonido()`	Recibe como parámetro una cadena que será la ruta de acceso al audio, esta se valida a través del método `getURL()`, y de recibir una URL valida, se reproducirá el audio de acuerdo al botón seleccionado.

Tabla 1. Elementos de la Clase Juego

La **clase `Escucha`** fue adoptada de *Cmop* [21], clase empleada en el *framework TalkingJava SDK*. La **clase `Escucha`** recibe la voz del niño, cuando se le pide repetir una palabra. Los elementos con los que cuenta esta clase se muestran en la Tabla 2.

Nombre	**Observaciones**
Atributos	
`recognizer`	Tipo: `Recognizer`
`gst, result, test`	Tipo: `String`
Objeto	Tipo: `Escribe`
Métodos	
`resultAccepted()`	Recibe como parámetro la voz del niño, la cual es almacenada en memoria. El resultado es pasado como parámetro al objeto de tipo `Escribe` a través de su método `escribir()`.
`Escuchar()`	Recibe como parámetro la palabra correcta, por parte de la clase `Juego` en su método `clicEvent()` y espera a que se pronuncie algo a través del micrófono.

Tabla 2. Elementos de la Clase Escucha

La **clase `Escribir`** se encarga de generar el reporte que podrá consultar el tutor en otro

momento para evaluar el avance del niño. Los elementos que forman esta clase son:

Nombre	**Observaciones**
Método	
`escribir()`	Recibe como parámetro la comparación de la palabra pronunciada y la palabra correcta, por parte de la clase `Escucha` en su método `resultAccepted()`

Tabla 3. Elementos de la Clase `Escribir`

Un objeto creado a partir de la clase `Juego`, puede crear varios ejemplares de la clase `Escucha`, y por cada objeto de la clase `Escucha`, se crea un ejemplar de la clase `Escribir`.

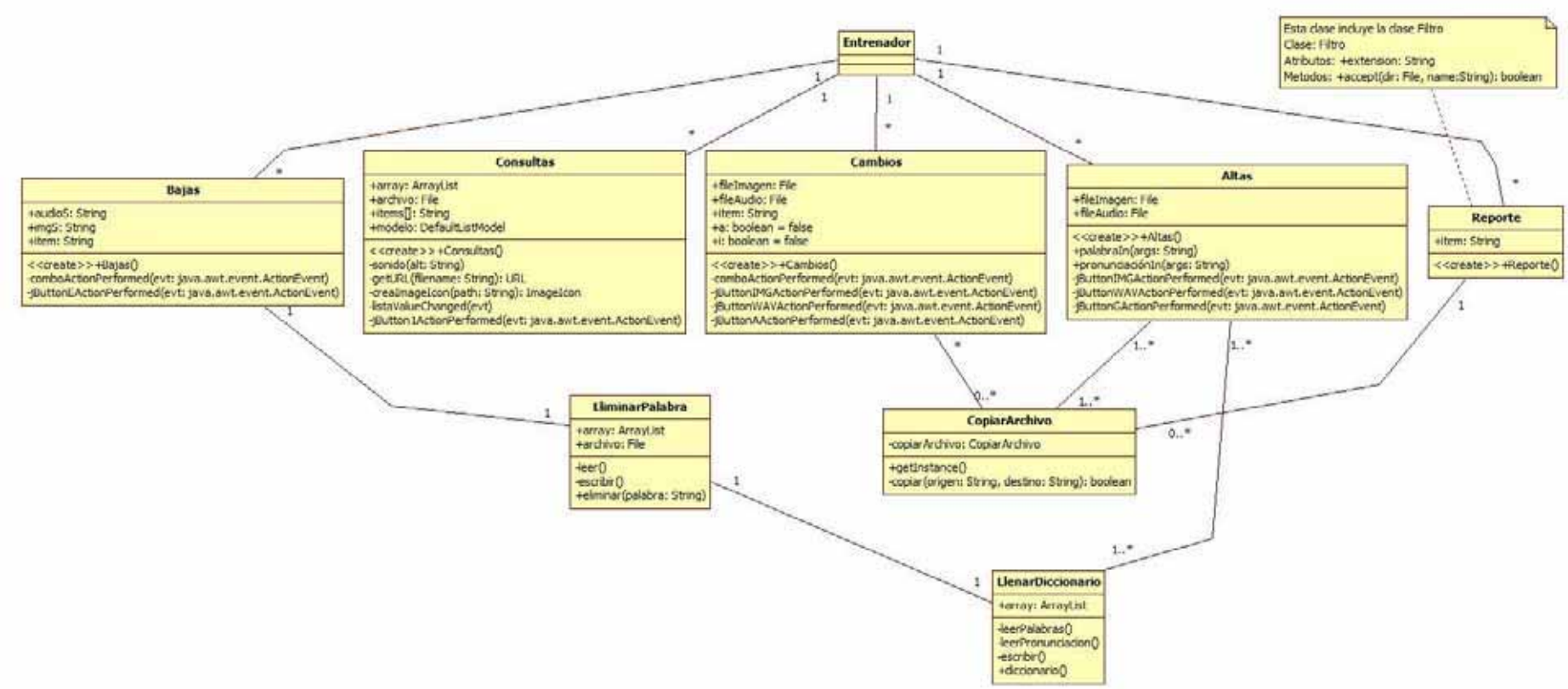

Diagrama 4. Diagrama de Clases - `Entrenador`

- **`EntrenarPalabras`**. El Diagrama 4**¡Error! No se encuentra el origen de la referencia.**, muestra las nueve clases que lo conforman, a saber: `Entrenador`, `Altas`, `Bajas`, `Cambios`, `Consultas`, `Reporte`, `EliminarPalabras`, `CopiarArchivo`, `LlenarDiccionario`.

La **clase `Entrenador`**, solo muestra la interfaz inicial, que consta de un menú de botones para que el tutor pueda seleccionar entre las acciones siguientes: Dar de alta una palabra, Eliminar, Consultar, Modificar y Consultar. Los elementos de la clase `Entrenador` son:

Nombre	**Observaciones**
Atributos	
`bAlta, bBaja, bCambios, bConsulta, bReportes`	Tipo: `Jbutton`
`jLabel2`	Tipo: `Jlabel`
Constructor	
`Entrenador()`	Inicializa la interfaz gráfica
Métodos	
`initComponents()`	Inicializa la interfaz
`bCambiosActionPerformed()`	Crea un objeto de la clase `Cambios`, crea un `Jframe` el cual contendrá el nuevo objeto de la clase `Cambios`, y lo despliega
`bAltaActionPerformed()`	Crea un objeto de la clase `Altas`, crea un `Jframe` el cual contendrá el nuevo objeto de la clase `Altas`, y lo despliega
`bBajaActionPerformed()`	Crea un objeto de la clase `Bajas`, crea un `Jframe` el cual contendrá el nuevo objeto de la clase `Bajas`, y lo despliega
`bConsultaActionPerformed()`	Crea un objeto de la clase `Consultas`, crea un `Jframe` el cual contendrá el nuevo objeto de la clase `Consultas`, y lo despliega
`bReportesActionPerformed()`	Crea un objeto de la clase `Reporte`, crea un `Jframe` el cual contendrá el nuevo objeto de la clase `Reporte`, y lo despliega

Tabla 4. Elementos de la Clase `Entrenador`

La **clase Altas** permite, a través de una interfaz gráfica, añadir una nueva palabra por escrito con su imagen y su audio; este último incluye las posibles pronunciaciones. Los elementos de esta clase son:

Nombre	Observaciones
Atributos	
`fileImagen, fileAudio`	Tipo: `File`
Constructor	
`Altas()`	Inicializa la interfaz gráfica
Métodos	
`palabraIn()`	Recibe como parámetro una palabra por escrito, y la añade al archivo `palabras.txt`
`pronunciacionIn()`	Recibe como parámetro la lista de posibles pronunciaciones, y las añade al archivo `pronunciacion.txt`
`jButtonIMGActionPerformed()`	Permite buscar la imagen que será asociada a la nueva palabra, y se almacena en el atributo `fileAudio`.
`jButtonWAVActionPerformed()`	Permite buscar el audio que será asociado a la nueva palabra y se almacena en el atributo `fileImagen`
`jButtonGActionPerformed()`	Cuando el usuario de clic en guardar, se ejecutarán los métodos `palabraIn()` y `pronunciacionIn()`, y se hace un llamado a la clase `CopiarArchivo` pasando como parámetros los atributos `fileImagen` y `fileAudio`

Tabla 5. Elementos de la Clase `Altas`

La **clase `Bajas`** es la encargada de eliminar la imagen y el audio asociados a una palabra seleccionada por el `tutor`. Los elementos de esta clase son:

Nombre	Observaciones
Atributos	
`audioS, imgS, item`	Tipo: `String`
Constructor	
`Bajas()`	Inicializa la interfaz, y genera la lista de las palabras existentes mostrándola en un *combobox*
Métodos	
`comboActionPermormed()`	Cuando el tutor selecciona una palabra, la URL de la imagen y el audio asociados a esta palabra se muestran en pantalla y se

	almacenan en los atributos `audioS` e `imgS`, y la palabra seleccionada se almacena en `item`
`jButtonEActionPerformed()`	Cuando el tutor seleccione `Eliminar`, tanto el audio como la imagen almacenados en los atributos `audioS` e `imgS`, serán eliminadas de los directorios correspondientes del programa. La palabra almacenada en `item`, es pasada como parámetro al objeto de la clase `EliminarPalabras`...

Tabla 6. Elementos de la Clase `Bajas`

La **clase `Cambios`** permite al `tutor` modificar la imagen o el audio de una palabra seleccionada en la interfaz gráfica. Los elementos de esta clase son:

Nombre	**Objetivo**
Atributos	
`fileImagen, fileAudio`	Tipo: `File`
`item`	Tipo: `String`
`a, i`	Tipo: `boolean`
Constructor	
`Cambios()`	Inicializa la interfaz gráfica, y genera la lista de las palabras existentes mostrándola en un *combobox*
Métodos	
`comboActionPerformed()`	Cuando el tutor selecciona una palabra, la URL de la imagen y el audio asociados a esta palabra se muestran en pantalla
`jButtonIMGActionPerformed()`	Permite buscar la nueva imagen que será asociada a la palabra seleccionada. La URL se almacenará en el atributo `fileImagen`
`jButtonWAVActionPerformed()`	Permite buscar el audio que será asociado a la palabra seleccionada. La URL se almacenará en el atributo `fileAudio`
`jButtonAActionPerformed()`	Cuando el usuario de clic en el botón `Actualizar`, se revisarán los estados de las banderas `a` e `i`, si estas marcan *false*, es que no hubo cambios y no se realizará ninguna acción, pero si el estado de alguna

	de las dos es *true*, se llama a la clase `CopiarArchivo` y únicamente se pasarán los parámetros del audio ó de la imagen de acuerdo a la bandera.

Tabla 7. Elementos de la Clase `Cambios`

La **clase `Consultas`** permite revisar tanto el audio como la imagen asociados a una palabra. Los elementos de esta clase son:

Nombre	**Observaciones**
Atributos	
`Array`	Tipo: `ArrayList`
`archivo`	Tipo: `File`
`items[]`	Tipo: `String`
`modelo`	Tipo: `DefaultListModel`
Constructor	
`Consultas()`	Inicializa la interfaz, y genera la lista de palabras que se mostrarán tomándolas del archivo `palabras.txt`
Métodos	
`sonido()`	Recibe como parámetro la palabra seleccionada, esta se valida a través del método `getURL`, y de existir una URL válida para esa palabra, se reproducirá el audio
`getURL()`	Recibe como parámetro una cadena que corresponde a la ruta de acceso al audio, de ser correcta, regresará la URL del audio al método sonido.
`creaImagenIcon()`	Recibe como parámetro una cadena que corresponde a la ruta de acceso a la imagen, si la URL es correcta, regresará `ImagenIcon` con la imagen proporcionada.
`listValueChanged()`	De acuerdo a la palabra seleccionada, este método llamará al método `creaImagenIcon` de la misma clase `Consultas,` y mostrará la imagen contenida en el *label* correspondiente.

jButton1ActionPerformed()	Cuando el tutor de clic sobre el botón Reproducir, se llamará al método sonido el cual reproducirá el audio asociado a la palabra seleccionada.

Tabla 8. Elementos de la Clase Consultas

La **clase Reporte** muestra en la interfaz, el contenido del archivo de texto que seleccione el tutor, también puede recuperarlo en su equipo. Los elementos de esta clase son:

Nombre	**Observaciones**
Atributos	
item	Tipo: String
Constructor	
Reporte()	Generará una lista de todos los reportes existentes, y la mostrará en un *combobox*
Métodos	
comboActionPerformed()	Cuando el tutor seleccione alguna de las opciones mostradas, se desplegará el reporte sobre la caja de texto
jButtonGActionPerformed()	Si el tutor da clic en el botón Guardar, se llamará a la clase CopiarArchivo proporcionándole como parámetros: la ubicación que el usuario eligió para guardar el archivo, y la ruta almacenada en el atributo item
Clases incluidas	
Filtro()	Esta clase nos restringe el tipo de archivos que se mostrarán en el listado. Fue tomada de [22].

Tabla 9. Elementos de la Clase Reporte

La **clase EliminarPalabras** elimina, tanto de la lista palabras.txt como del diccionario.txt, la palabra elegida por el tutor en la clase Bajas. Los elementos de esta clase son:

Nombre	**Observaciones**
Atributos	
array	Tipo. ArrayList

`archivo`	Tipo: `File`
Métodos:	
`leer()`	Almacena las palabras del archivo `palabras.txt` en el atributo `array`
`escribir()`	Sobre escribe el archivo `palabras.txt`, con el contenido del atributo `array`
`eliminar()`	Recibe como parámetro la palabra a eliminar, llama al método `leer()`, elimina la palabra proporcionada del atributo `array` y llama al método `escribir()`. Hace una instancia de la clase `LlenarDiccionario` y la llama en su método `diccionario`

Tabla 10. Elementos de la Clase `EliminarPalabras`

La **clase `LlenarDiccionario`** toma las palabras y las posibles pronunciaciones, y crea un documento llamado `diccionario.txt`, el cual interactúa con la clase `Escucha` del módulo `Juego`. Los elementos de esta clase son:

Nombre	**Observaciones**
Atributos	
`array`	Tipo: `ArrayList`
Métodos	
`leerPalabras()`	Almacena la lista de palabras existentes en el atributo array, tomadas del archivo `palabras.txt`
`leerPronunciacion()`	Almacena la lista de pronunciaciones del archivo `pronunciacion.txt` y las almacena en el atributo `array`
`escribir()`	Sobre escribe el archivo `diccionario.txt`, con las características necesarias para emplear el API y las bibliotecas *JavaSpeech*, y los elementos almacenados en el atributo `array`
`diccionario()`	Llama a los métodos `leerPalabra()`, `leerPronunciacion()` y `escribir()`

Tabla 11. Elementos de la Clase `LlenarDiccionario`

La **clase `CopiarArchivo`** permite copiar un archivo a la ubicación deseada. Los parámetros que recibe son: la ruta del archivo origen y la ruta del archivo destino, hay que

hacer notar que también se le tiene que proporcionar el nombre y tipo del archivo. Esta clase se relaciona con las clases `Altas`, `Cambios` y `Reporte`.

6.2.5. Arquitectura del sistema

La arquitectura (ver Diagrama 5) está compuesta de los módulos: `JugarPalabras` y `EntrenarPalabras;` además de un conjunto de archivos de configuración para el manejo de palabras y un conjunto de archivos auxiliares. Como lo mencionamos antes, el sistema hace uso de la *framework TalkingJava SDK* para el reconocimiento de voz. A continuación describiremos cada uno de ellos.

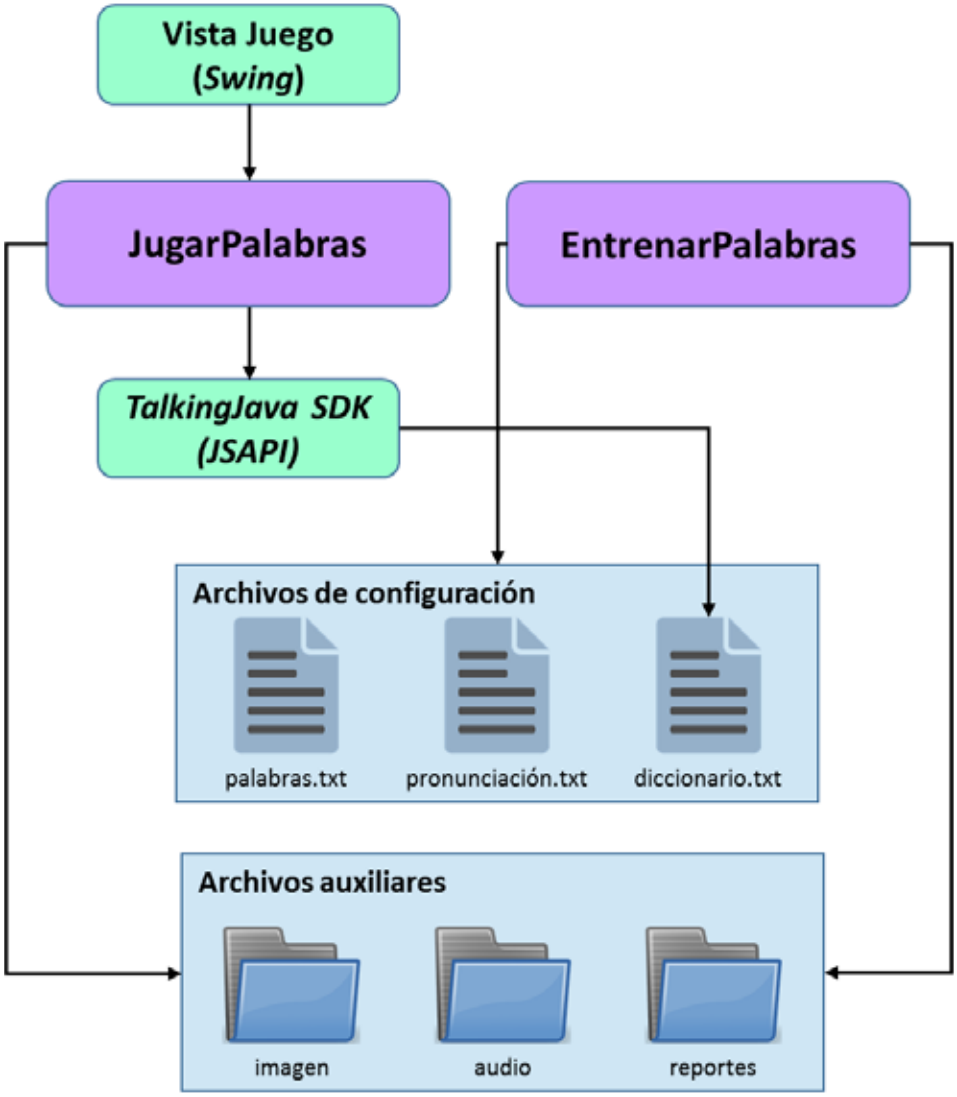

Diagrama 5. Diagrama de la Arquitectura del Sistema

La *Vista del juego* se enfoca principalmente a la interfaz que se presenta al niño, empleando los archivos auxiliares de imágenes y audio. Cuando el niño interactúa con la Vista, ésta se comunica con el JSAPI por medio de *TalkingJava SDK de Cloud Garden*.

Empleamos el *framework TalkingJava SDK* para el reconocimiento de voz. Es importante

mencionar que el uso de nuestra aplicación, en la versión actual, está condicionado a la compra de la licencia de este *framework*. Sin embargo, los módulos que se desarrollaron y que forman parte del "prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla" permiten adoptar (con las modificaciones pertinentes) otros *frameworks* para el reconocimiento de voz. Esto nos da la posibilidad de usar el prototipo para fines educativos y de investigación.

Los *Archivos auxiliares* corresponden a los archivos de imagen y los archivos de audio asociados a las palabras, y a los reportes generados.

Los *Archivos de configuración* almacenan tanto las palabras, y las posibles pronunciaciones, como el diccionario usado por JSAPI.

El *Entrenador* tiene la principal función de gestionar las acciones de: altas, bajas, modificación, y consultas de palabras, pronunciaciones, imágenes y audio del sistema. Por lo tanto, interactúa directamente con los archivos de configuración y los archivos auxiliares. El Entrenador puede agregar información sobre los archivos de configuración ya existentes o crear nuevos archivos auxiliares.

6.2.5.1. Archivos de configuración

`palabras.txt`

Es una lista de palabras separadas por salto de línea (ver Figura 2). El uso de este documento es sumamente importante para todo el sistema, de él dependen cuatro funcionalidades: i) la selección aleatoria de las imágenes para la Vista, ii) el registro de nuevas palabras, iii) la selección de una palabra, su imagen y su audio asociados con el propósito de modificar, eliminar o consultar esa palabra o alguno de sus elementos.

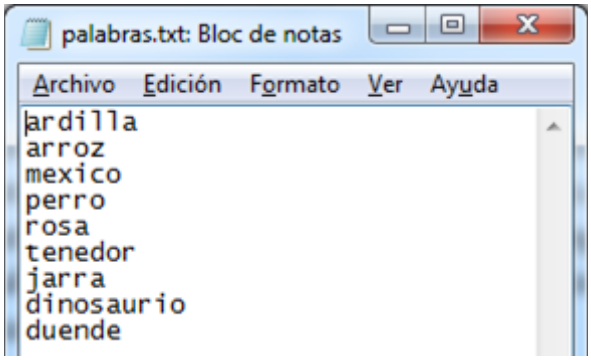

Figura 2. Ejemplo del archivo `palabras.txt`

`pronunciación.txt`

Al igual que el archivo `palabras.txt`, `pronuciacion.txt` contiene una lista de palabras separadas por un salto de línea (ver Figura 3), este documento es empleado para el llenado del `diccionario.txt` en el módulo `EntrenarPalabras`; esto se realiza en la clase `LlenarDiccionario`, la cual es auxiliar de las clases `Altas` y `Bajas`; cuando se crea o elimina una palabra, el diccionario debe ser actualizado, para reconocerla o dejar de reconocerla.

`pronuciacion.txt` almacena todas pronunciaciones (la correcta y las incorrectas) que

el niño pueda pronunciar; mientras este archivo contenga más pronunciaciones, mejor será el resultado en el reconocimiento de voz.

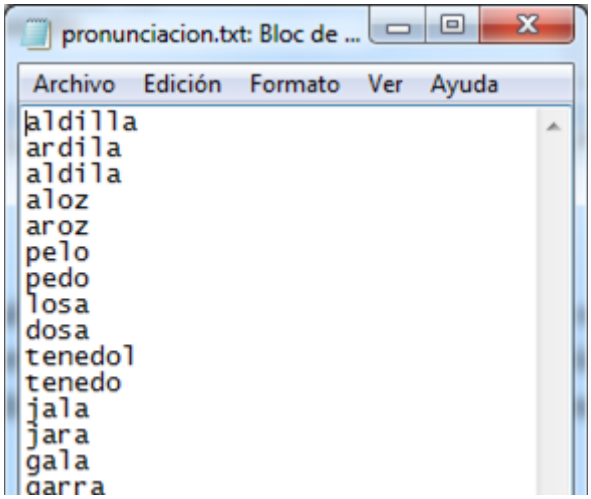

Figura 3. Ejemplo del archivo `pronunciación.txt`

`diccionario.txt`

Este archivo se precarga con un muestreo de palabras al momento de realizar la instalación de nuestra aplicación, y con cada alta o baja de alguna palabra, se actualiza. `diccionario.txt` contiene las reglas de gramática con la que trabajará *TalkingJava SDK*. Por ejemplo, el sistema espera que se diga la palabra "*jarra"* y el niño pronuncia *"arra",* y esta se encuentra como parte del `diccionario.txt`, entonces reconocerá eficazmente que el niño cometió un error; pero, si el niño dice *"parra"* y esta no se encuentra en el `diccionario.txt`, entonces el sistema tomará la pronunciación más cercana a la del niño que se encuentre en el `diccionario.txt`, con menoscabo en el tiempo empleado para realizar el reconocimiento de la voz. El contenido de la Figura 4 incluye un encabezado especificando que versión del *JSGF* es utilizada:

```
#JSGF V1.0;
```

A continuación se nombra a la gramática utilizada:

```
grammar palabras;
```

El cuerpo de la gramática está formado por las reglas que va a seguir el *TalkingJava SDK*. El orden de las reglas no es significativo para la gramática.

Esta regla se define publica ya que será la regla activa utilizada por *TalkingJava SDK* para determinar las palabras correctas y posibles pronunciaciones que pueden ser expresadas oralmente por el `niño`.

```
public <palabras> = <palabra>;
```

La siguiente regla es el conjunto de palabras, ordenadas entre paréntesis y separadas por el operador OR "|", indicando un conjunto de palabras excluyente; es decir, que el `niño` puede decir una de las palabras dentro de la agrupación y por lo tanto, el sistema escogerá aquella que corresponda con la pronunciación del `niño`.

```
<palabra>=(ardilla | arroz |...);
```

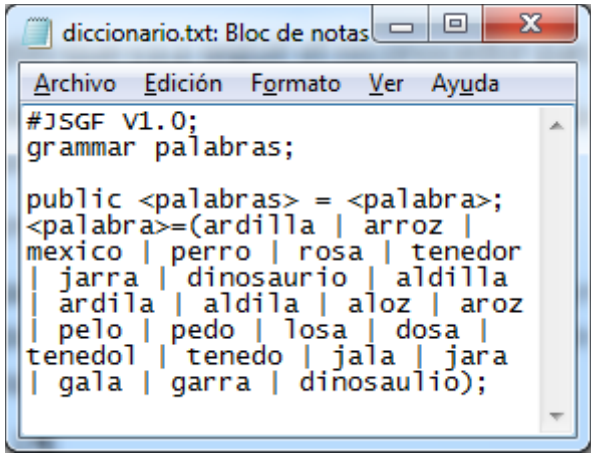

Figura 4. Ejemplo del archivo `diccionario.txt`

6.2.5.2. Archivos auxiliares

Los archivos auxiliares, son principalmente: i) las imágenes y los audios almacenados y reconocidos por el sistema, y ii) los reportes sobre el desarrollo del juego que genera el sistema. Estos archivos no modifican los valores o acciones ni de la Vista, ni *TalkingJava SDK*, ni del `Entrenador`; tampoco modifican los archivos de configuración.

La Figura 5 muestra el árbol de directorios del sistema.

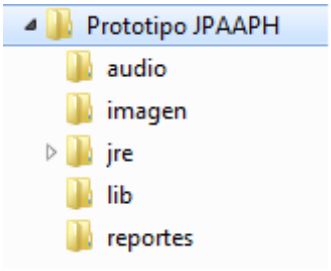

Figura 5. Árbol de directorios de la aplicación

El directorio imagen, almacena todas las imágenes asociadas a las palabras registradas en el sistema. El `Entrenador` es el encargado de dar de alta, modificar, eliminar o consultarlas de este directorio. Las imágenes deben tener el formato GIF y deben tener una dimensión de 200x229 pixeles (ver Figura 6).

Figura 6. Ejemplo de una imagen utilizada para el sistema

El directorio audio almacena las grabaciones de sonido asociadas a las palabras, estos archivos tienen que ser en formato WAV. El `Entrenador` da de alta, modifica, borra y consulta los archivos del directorio audio.

El directorio reportes almacena los reportes generados durante el juego. Los archivos son en texto plano en formato TXT. El `Entrenador` es el módulo que podrá consultarlos. La estructura de un archivo que contiene un reporte es la siguiente (ver Figura 7):

- El nombre del archivo se genera automáticamente y está formado por la fecha en que se generó el reporte, empezando por los últimos dígitos del año, luego los dos dígitos del mes y al final los dos dígitos del día, ejemplo: `140201.txt`.
- El archivo contiene como encabezado la hora en que se empezó el juego. A continuación se establecen dos títulos de columnas: Esperada y Pronunciada, refiriéndose a la palabra correctamente dicha y a la palabra que captó el *TalkingJava SDK*.
- Cada vez que el niño elige una imagen de la interfaz, se genera un renglón en el archivo, dividido en dos columnas, la primera con la palabra pronunciada de manera correcta y la segunda con la palabra proporcionada por el niño y captada por el *TalkingJava SDK*.

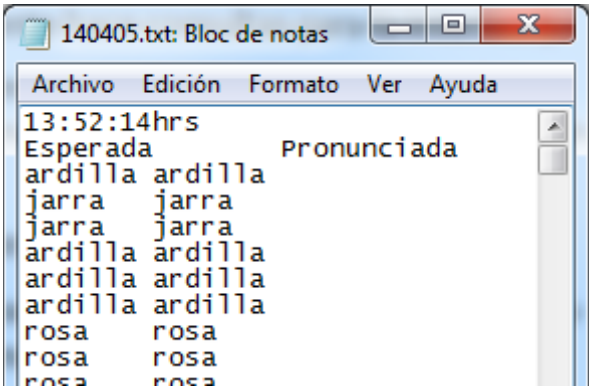

Figura 7. Ejemplo del contenido de un reporte

6.3. Uso del sistema

El sistema cuenta con dos funcionalidades principales: `JugarPalabras` y `EntrenarPalabras`.

6.3.1. JugarPalabras.

Muestra una serie de imágenes que el niño podrá seleccionar. El sistema pronuncia la palabra asociada a la imagen, y posteriormente se espera que el niño pronuncie la misma palabra. La pronunciación del niño pasa por un proceso donde se registra la palabra correcta y la palabra que el niño pronunció, tal y como la reconoció el *TalkingJava SDK*. El sistema repetirá estas acciones siempre que el niño seleccione una imagen (ver Figura 8).

Figura 8. Pantalla principal del prototipo

Las clases involucradas con la funcionalidad de esta pantalla son: `Juego`, `Escucha` y `Escribe`

La clase `Juego` (Figura 9) crea la interfaz que se muestra, lee en el archivo `palabras.txt`, de las palabras obtenidas, muestra solo 6 seleccionadas aleatoriamente, y espera a que el niño de clic en alguna para realizar el evento de acuerdo a la palabra (reproducir el audio de la palabra e instanciar a la clase `Escucha`).

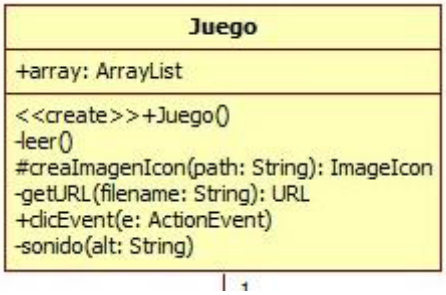

Figura 9. Clase `Juego`, atributos y métodos más importantes

La clase `Escucha` (Figura 10) es instanciada por la clase `Juego` en el momento que el niño cliquea alguna imagen, esta clase espera a que se hable a través del micrófono y capta lo dicho, lo compara con alguna palabra existente en el archivo `diccionario.txt` y regresa la palabra más parecida a lo que escuchó, tanto la palabra correcta (proporcionada por la clase `Juego`) tanto la palabra que escucho, son almacenadas y pasadas como parámetro a la instancia de la clase `Escribe`.

Figura 10. Clase `Escucha`, atributos y métodos más importantes

La clase `Escribe` (Figura 11) es instanciada por la clase `Escucha` después de captar lo que se dijo por el micrófono y después de buscarlo en el archivo `diccionario.txt`, esta recibe como parámetros en su método escribir, tanto la palabra propiamente dicha, como lo devuelto por el *TalkingJava SDK*. Esta clase es la encargada de generar el archivo reporte del día (si no existe el archivo, se crea; si existe el archivo se agrega el resultado).

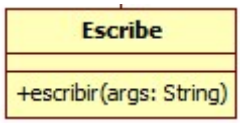

Figura 11. Clase `Escribe`

6.3.2. EntrenarPalabras

El sistema también cuenta con la funcionalidad de gestionar el juego, permitiendo al tutor las acciones de: dar de alta, modificar, eliminar y consultar las palabras que forman parte del juego. La Figura 12, muestra la ventana a través de la cual es posible dar de alta una nueva palabra, su imagen y audio asociados.

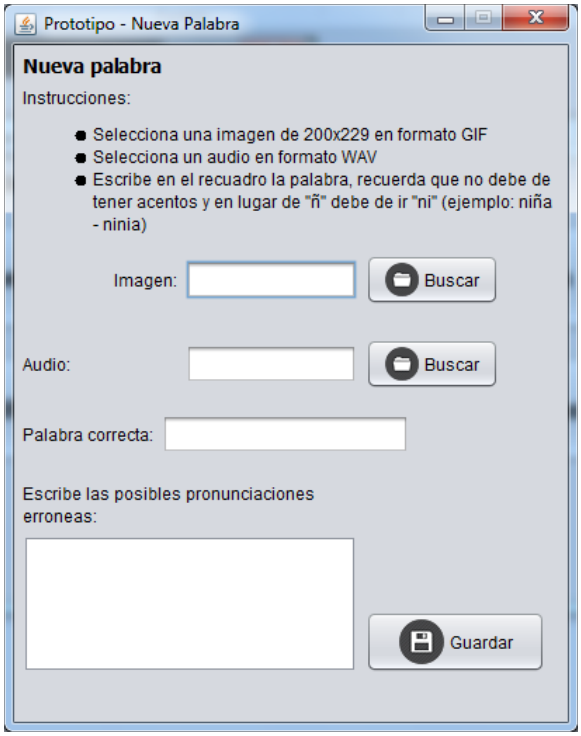

Figura 12. Pantalla para dar de alta una nueva palabra

Las clases involucradas en esta funcionalidad son: `Altas`, `CopiarArchivo`, `LlenarDiccionario`.

La clase `Altas` (Figura 13) permite que el tutor escriba la nueva palabra y sus posibles pronunciaciones, en los cuadros de texto respectivos. También le permite buscar la imagen y el audio para asociarlos a la palabra recién ingresada; esto se logra con el llamado de los métodos de la clase `JfileChooser`. Cuando el usuario seleccione Guardar, se crearán dos objetos de las clases `CopiarArchivo` y `LlenarDiccionario`.

Figura 13. Clase `Altas`, atributos y métodos más importantes

Cuando el usuario decide guardar una imagen, un audio y las palabras proporcionadas se crean objetos del tipo `CopiarArchivo` (Figura 14), tal objeto recibe los parámetros: ruta origen, donde se encuentra el archivo seleccionado, y la ruta destino, la cual será seleccionada por el tipo de archivo. Por ejemplo si se trata de la palabra perro, y se ha seleccionado su imagen el archivo se nombrará perro.gif y se guardará en el directorio `imagen`; si se ha seleccionado su audio, entonces se nombrará perro.wav y se guardará en el directorio `audio`.

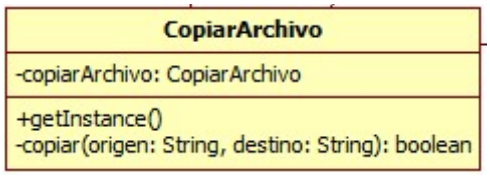

Figura 14. Clase `CopiarArchivo`

La clase `LlenarDiccionario` (Figura 15) es instanciada por la clase `Altas` cada vez que el usuario da clic sobre el botón Guardar. Esta clase lee los archivos de configuración `palabras.txt` y `pronunciacion.txt`, el contenido de ambos lo almacena en el archivo `diccionario.txt`

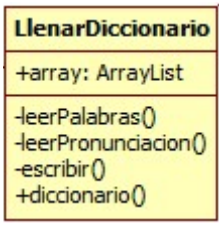

Figura 15. Clase `LlenarDiccionario`

6.4. Hardware y software necesario

6.4.1. Tecnología para el desarrollo de la aplicación

Para la realización de proyecto se emplearon los siguientes recursos:

Hardware

- Laptop Toshiba Satellite T215D
- Tarjeta de video ATI Mobility Radeon HD 4200, Internal DAC 400MHz y memoria de gráficos 1 GB.
- Procesador AMD Athlon II Neo K125 Processor 1.70GHz
- Memoria RAM 3.00 GB
- Micrófono y bocinas

Software

- Sistema Operativo Windows 7 Enterprice. Es importante señalar que la Java Speech API y la librería Cloud Garden TalkingJava SDK funciona únicamente bajo el Sistema Operativo Windows.
- NetBeans IDE 6.9.1para el desarrollo de la aplicación.
- Plataforma Java (JDK versión 7 update 67), principalmente por su portabilidad entre equipos y plataformas.
- *TalkingJava SDK de Cloud Garden* para el reconocimiento de voz, este *framework* es compatible con Java Speech API.

6.4.2. Tecnología para la instalación y puesta en marcha de la aplicación.

Los requerimientos mínimos para la instalación y buen funcionamiento de la aplicación son:

- Hardware:
    - Computadora personal Processor 1.70GHz o superior, memoria RAM 3.00 GB y tarjeta de video de 400MHz o superior y memoria de gráficos de 1 GB.
    - Micrófono, de preferencia no embebido en el equipo de cómputo para obtener mayor fidelidad en la voz del niño.
    - Bocinas, pueden ser bocinas integradas o audífonos externos.
- Software:
    - Sistema Operativo Windows 7 ó superior.
    - Plataforma Java JDK versión 7 ó superior.

Si el usuario cuenta con los siguientes recursos, se tendrá un mejor aprovechamiento de la aplicación:

- Software para grabar audio
- Software para editar imágenes

7. Resultados

Para alcanzar los objetivos propuestos al inicio de este proyecto, se realizaron una serie de artefactos que permiten observar el desarrollo de la aplicación.

- Documentos de requisitos funcionales y no funcionales.
- Documento de diseño, el cual incluye: Diagrama de casos de uso, diagrama de robustez, casos de uso de texto, diagrama de clases.
- Código fuente de la aplicación
- Casos de pruebas y corrección de errores
- Aplicación terminada y lista para usarse
- Manual de usuario
- Manual técnico

8. Análisis y discusión de resultados

Las pruebas que se han realizado para probar el sistema son básicamente de unidad y de sistema. Es decir, se realizaron pruebas sobre los módulos: `EntrenarPalabras` y `JugarPalabras`, y posteriormente, se han realizado pruebas sobre los módulos integrados.

Para el módulo `EntrenarPalabras` se realizaron un estimado de 10 pruebas para: dar de alta, baja, modificación y consulta de palabras. Actualmente el sistema cuenta con 20

palabras correctas precargadas, 20 archivos, cada uno contiene una imagen (de libre distribución), 20 archivos de audio, y cerca de 60 posibles pronunciaciones.

Para el módulo `JugarPalabras`, se realizaron aproximadamente 40 pruebas que implicaron iniciar el juego e interactuar con el aproximadamente de 5 a 15 minutos. Es importante mencionar que en las pruebas han intervenido aproximadamente 5 sujetos en edades de entre 25 a 35 años. No consideramos pertinente realizar pruebas con niños porque aún no contamos con la evaluación del sistema por parte de un terapeuta; sin embargo, se ha probado que si el sujeto pronuncia de manera incorrecta la palabra, el sistema registra la pronunciación de manera esperada. El sistema cuenta con un 70% de aciertos, consideramos que esto es debido a las condiciones con que se evaluó: ruido de ambiente, rapidez en la respuesta y un diccionario con pocas palabras; por ejemplo: la persona pronuncia la palabra *rosa* en dos ocasiones, en la primera ocasión el sistema la reconoce como *rosa* y en la segunda como *gala*.

9. Conclusiones

El proyecto "Prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla" tuvo como propósito construir una herramienta de apoyo a niños de entre 5 y 7 años que presentan trastornos del habla. Este juego computacional busca favorecer el ejercicio cotidiano del niño, a partir de la pronunciación de palabras, cuya dificultad es reconocida por la literatura especializada. El juego emplea una serie de palabras asociadas cada una de ellas a una imagen, el niño interactúa con estas imágenes para escuchar la pronunciación correcta, seguido de un intento, por su parte, de repetir la misma palabra.

En términos de los cinco objetivos planteados en este proyecto terminal, presentamos una tabla donde se describe el cumplimiento de cada uno de ellos.

Objetivos Específicos	**Alcance**
Elección de herramienta de reconocimiento de voz.	Se eligió el *TalkingJava SDK* por su fácil empleo; es decir, es sencillo de enriquecer la base de palabras para que realice el reconocimiento de voz. Sin embargo, este *framework* tiene una licencia para su uso en instituciones y su uso comercial.
Diseñar y modelar el juego y su interfaz.	Se realizó el análisis y diseño de los módulos principales del sistema, a saber: JugarPalabras y EntrenarPalabras. La interfaz gráfica se elaboró a partir de la *framework* Java Swing.
Implementar un prototipo para el juego.	El prototipo se logró y funciona a través de los módulos principales: JugarPalabras y EntrenarPalabras. Para el módulo de JugarPalabras se cuenta con una base de 20 palabras, sus imágenes y audios asociados. Es importante señalar que las imágenes son de libre distribución. Además, gracias

	al módulo de EntrenarPalabras, es posible enriquecerlo con nuevas palabras.
Realizar pruebas de usabilidad con al menos un usuario especialista en el tema.	Este objetivo se alcanzó parcialmente; en las pruebas que realizamos intervinieron 5 sujetos que no son especialista, pero que interactuaron con los módulos principales del sistema.
Elaboración de entregables.	Los entregables descritos en la propuesta del proyecto se encuentran finalizados.

Tabla 12. Alcance de los objetivos específicos

10. Perspectivas del proyecto

Como futuras mejoras para el "Prototipo para un juego por computadora que ayude en el auto-aprendizaje de niños con problemas del habla" podemos mencionar: i) adaptarlo para dispositivos móviles, actualmente se cuenta con los recursos necesarios para que nuestro sistema funcione; ii) el sistema puede servir para probar diferentes reconocedores de voz y medir la eficiencia de ellos en el contexto de auto-aprendizaje de niños con problemas del habla.

Referencias bibliográficas.

[1] Perelló, J. Trastornos del habla. 5 ª ed. Elsevier-MASSON 1995, pp. 1, 131-138, 285-287.

[2] "Trastornos del Habla y Lenguaje", Family NET Works (oct. 12) [en línea]. Disponible en: http://www.family-networks.org/espanol_speechimp.PDF

[3] Bonilla, A., "Crean software que facilita habla a personas con discapacidad", La Crónica en línea (oct. 12) [en línea]. Disponible en: http://www.cronica.com.mx/nota.php?id_nota=484536

[4] Belloch O. C., "SPEECHVIEWER III", Unidad de Tecnología Educativa. Universidad de Valencia (oct. 12) [en línea]. Disponible en: http://www.uv.es/bellochc/logopedia/speechviewer3.pdf

[5] SPEECHVIEWER III Visualizador Fonético (oct. 12) [en línea]. Disponible en: http://www.adaptat.com/productos/Comunicacion/speevi.htm

[6] "DI" (oct. 12) [en línea]. Disponible en: http://www.ite.educacion.es/w3/recursos/pntic98/fichas/di.htm

[7] Marín Díaz, L., "Sistema de reconocimiento del alfabeto dactilológico utilizando procesamiento de imágenes". Proyecto terminal, Universidad Autónoma Metropolitana Azcapotzalco, 2010.

[8] Medina González A., Hernández Silva D. R.,"Control BT / WI de servicio orientado a personas con limitantes en sus extremidades inferiores". Proyecto terminal, Universidad Autónoma Metropolitana Azcapotzalco, 2008.

[9] Sánchez Sánchez, G. A., “Máquina de aprendizaje en un juego de ajedrez”. Proyecto terminal, Universidad Autónoma Metropolitana Azcapotzalco, 2008

[10] Hernández Hernández, F. A., Hernández Piña, H. C., “Plataforma de juego programable para Quoridor”. Proyecto terminal, Universidad Autónoma Metropolitana Azcapotzalco, 2010.

[11] Solano Duarte, E., “Prototipo de una aplicación, para PDA, que facilite la comunicación de las necesidades de un niño con problemas del habla”. Proyecto terminal, Universidad Autónoma Metropolitana Azcapotzalco, 2011.

[12] Barberá, Ó. y San José, V., “Juegos de simulación por ordenador: un útil para la enseñanza a todos los niveles” Enseñanza de las ciencias: revista de investigación y experiencias didácticas (jul. 2014) [en línea] Disponible en: http://www.raco.cat/index.php/Ensenanza/article/view/51291/93037

[13] Rossaro, A. L., “Aprender jugando: Los videojuegos y su potencial educativo” [en línea] Disponible en: http://ineverycrea.net/comunidad/ineverycrea/recurso/Aprender-jugando-Los-videojuegos-y-su-potencial-e/ba4316fb-a533-4db4-9d91-92e0768dd9e4 [consultado 11/07/2014]

[14] Cervera, M. G., “Las tecnologías de la información y la comunicación como favorecedoras de los procesos de autoaprendizaje y de formación permanente” Universitat Rovira i Virgili. Departament de Pedagogía (jul. 2014) [en línea] Disponible en: http://www.quadernsdigitals.net/datos/hemeroteca/r_73/nr_780/a_10552/10552.pdf

[15] Autoaprendizaje (jun. 2014) [en línea] Disponible: http://autoaprendizaje7.wordpress.com

[16] “Trastornos del Habla y Lenguaje” Centro Nacional de Diseminación de Información para Niños con Discapacidades (jun. 2014) [en línea] Disponible en: http://nichcy.org/espanol/discapacidades/especificas/lenguaje

[17] Castañeda, P. F., “El lenguaje verbal del niño.” Universidad Nacional Mayor de San Marcos (jun. 2014) [en línea] Disponible en: http://sisbib.unmsm.edu.pe/bibvirtual/libros/linguistica/leng_ni%C3%B1o/ni%C3%B1o_tras_habla.htm

[18] Launay, C., Trastornos del lenguaje, la palabra y la voz en el niño. MASSON 1986, pp. 374-380

[19] Santiago Zaragoza, M., “Desarrollando aplicaciones informáticas con el Proceso de Desarrollo Unificado (RUP)”. Universidad Tecnológica del Valle del Mezquital (oct. 2013) [en línea]. Disponible en: http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm

[20] “El Proceso Unificado de Desarrollo de Software”, Universidad Autónoma de Baja California (Oct. 2013) [en línea]. Disponible en: http://yaqui.mxl.uabc.mx/~molguin/as/RUP.htm

[21] “Java, Reconocimiento y Sintetización de Voz con Cloud Garden TalkingJava SDK with Java Speech API implementation“, Cmop (ago. 2010) [en línea] Disponible en: http://cmop17.wordpress.com/2010/06/09/javareconocimiento-y-sintetizacion-de-voz-con-cloud-garden-talkingjava-sdk-with-java-speech-api-implementation

[22] Granco García, A. (feb. 2013) ”Programación en el lenguaje Java, Archivos y directorios” Universidad del País Vasco [en línea]. Disponible en: http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/archivos/file.htm

[23] “API Specifications” Oracle (ago. 2014) [en línea] Disponible en: http://www.oracle.com/technetwork/java/index-140170.html

[24] “Jspeech Grammar Format”, W3 (ago. 2010) [en línea] Disponible en: http://www.w3.org/TR/jsgf

[25] FreeTTS 1.2 (ago. 2014) [en línea] Disponible en: http://freetts.sourceforge.net/docs/index.php

[26] “developerWorks” IBM (ago. 2014) [en línea] Disponible en: https://www.ibm.com/developerworks/community/groups/service/html/communityview?communityUuid=18d10b14-e2c8-4780-bace-9af1fc463cc0

[27] “Sphinx 4” CMU Sphinx (ago. 2014) [en línea] Disponible en: http://cmusphinx.sourceforge.net/wiki/sphinx4:webhome

[28] Could Garden TalkingJava (oct. 2010) [en línea] Disponible en: http://www.cloudgarden.com/