BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

ENCRIPTACIÓN DE UN TEXTO EN ESPAÑOL UTILIZANDO LA FORMA SILÁBICA

PRESENTA:

CECILIA ESPINOZA GUEVARA

TESIS SOMETIDA COMO REQUISITO
PARA OBTENER EL GRADO DE:

LICENCIADA EN CIENCIAS DE LA COMPUTACIÓN

DIRIGIDA POR:

DRA. BÁRBARA SÁNCHEZ RINZA

PUEBLA, PUE., AGOSTO 2009

ÍNDICE

CAPÍTULO 3

ANÁLISIS

3.1 Descripción general del proyecto.

3.2 Metodología.

3.3 Estrategias de solución.

3.4 Diagrama de contexto.

3.5 Acciones y restricciones del programa.

3.6 Diagramas de casos de uso.

- 3.6.1 Caso de uso: Menú Principal.
- 3.6.2 Caso de uso: Abrir Archivo .txt.
- 3.6.3 Caso de uso: Encriptación.
- 3.6.4 Caso de uso: Desencriptación.
- 3.6.5 Caso de uso: Guardar en Archivo .txt.

3.7 Diagramas de secuencia.

- 3.7.1 Diagrama de secuencia: Menú Principal.
- 3.7.2 Diagrama de secuencia: Abrir Archivo .txt.
- 3.7.3 Diagrama de secuencia: Encriptación.
- 3.7.4 Diagrama de secuencia: Desencriptación.
- 3.7.5 Diagrama de secuencia: Guardar en Archivo .txt.

CAPÍTULO 4

DISEÑO

4.1 Diagrama de Arquitectura.

4.2 Diseño de la Interfaz.

4.3 Estructura de Datos.

- 4.3.1 Identificación de clases.

PRÓLOGO

Hoy en día la seguridad de la información ha dejado de ser un tema *secreto* y a pasado a ser un tema de gran importancia para la mayoría de personas que diariamente utilizan una computadora. La necesidad de proteger sus archivos; fotos, textos, programas, videos, canciones, grabaciones inclusive nuestras propias contraseñas, es inevitable.

Uno de los modos más básicos y simples que se han utilizado siempre, es almacenar esta información privilegiada en dispositivos de almacenamiento, como pueden ser disquetes, CD o más actualmente DVD. Hoy en día, este método es insuficiente y por ello los métodos de seguridad más populares utilizan la encriptación, el cual es el proceso de codificar la información de tal manera, que solo la persona (u ordenador) con una clave determinada, puede decodificarla y hacer uso de dicha información.

Este trabajo de tesis ofrece una herramienta para Encriptar un texto en español utilizando la forma silábica, por lo que la tesis está estructurada de la siguiente manera:

Capítulo uno: se plantea el problema y su justificación, los objetivos generales así como los específicos, y los antecedentes de la criptografía, y por último los alcances y limitaciones del programa.

Capítulo dos: contiene una breve introducción a cerca de las sílabas y su clasificación, se plantean las 10 reglas ortográficas para la segmentación de las palabras en sílabas, analizándolas previamente.

Capítulo tres: pertenece al análisis del proyecto, en este capítulo se describe el proyecto de manera general siguiendo una metodología y proponiendo una estrategia de solución. Este capítulo incluye también los casos de uso y los diagramas de secuencia del programa, esto, para saber la manera en que trabaja el programa.

Capítulo cuatro: El diseño. En este capítulo se diseña la interfaz, se escribirán los algoritmos y se explicarán las funciones sobresalientes del programa.

Capítulo cinco: una vez que ya se ha diseñado el programa, es necesario implementarlo por lo que en este capítulo se implementará el programa y posteriormente se realizaran pruebas para depurar algún error.

En las ***Conclusiones*** se exponen los objetivos logrados y las espectativas, en el ***Apéndice I,*** están un compendio de todas las sílabas del idioma español.

Finalmente en la ***Bibliografía*** se presentan las referencias bibliográficas y de documentos virtuales (Internet, y enciclopedias en CD–ROM).

CAPÍTULO 1

INTRODUCCIÓN

CAPÍTULO 1

INTRODUCCIÓN

1.1 Planteamiento del problema.

Dada la importancia que la encriptación tiene, ya que es una forma de seguridad para cualquier manejo de la información, se desarrollará una aplicación que consiste en encriptar silábicamente dicha información; por medio de un algoritmo que separará las palabras en sílabas; las separará tomando en cuenta su tamaño y ciertas reglas ortográficas.

Ya que las sílabas estén correctamente separadas, se empleará otro algoritmo; la función de este algoritmo es básicamente encriptar, es decir; codificar la información para que sea indescifrable a simple vista, de manera que una letra *"A"* pueda equivaler a *"5x5mBwE" o bien a "xQE9fq"*, el trabajo del algoritmo es precisamente determinar como será transformada la información de su estado original a otro que sea muy difícil de descifrar, ya que existen datos que no pueden estar expuestos a muchos usuarios, lo cual plantea la necesidad de que los datos estén protegidos durante su transmisión y almacenamiento.

Es importante tomar en cuenta que a cada elemento del dominio sólo le corresponde uno y solo un elemento del codominio, esto implica que la función debe ser biyectiva para no crear conflictos a la hora de desencriptar el texto, utilizando un amplio alfabeto por la gran cantidad de silabas, dicho alfabeto es el de las letras del abecedario, números y símbolos para poder realizar la encriptación.

1.2 Objetivo general.

El objetivo general se centra en el desarrollo de una aplicación para la encriptación de un texto en español utilizando la forma silábica.

1.3 Objetivos específicos.

Los objetivos específicos se describen a continuación:

- Diseñar e implementar un programa para asegurar y mantener la confidencialidad de los datos.
- Al implementar el programa se tomarán en cuenta los diferentes algoritmos de encriptación que existen, así como también las reglas para la segmentación de las sílabas del idioma español.
- Garantizar el secreto en la comunicación entre dos entidades (personas, organizaciones, etc.).
- Asegurar que la información que se envía es auténtica en un doble sentido: que el remitente sea realmente quien dice ser y que el contenido del mensaje enviado, habitualmente denominado criptograma, no haya sido modificado en su tránsito.

1.4 Justificación.

El proyecto cuenta con una pequeña aportación a la tecnología, aunque se han desarrollado diversos trabajos de encriptación alrededor del mundo, este proyecto puede ser utilizado por una microempresa o simplemente por cualquier persona que quiera mantener segura su información.

El siguiente trabajo se enfoca para ser de gran ayuda a quien lo utilice, que funcione eficazmente y rápidamente, pero sobretodo que al utilizarlo se adquiera la seguridad completa para enviar o mantener la información segura.

Al implementar este programa se debe de tomar en cuenta todas y cada una de las reglas fundamentales para determinar la separación de las sílabas de cada palabra.

1.5 Antecedentes del proyecto.

1.5.1 Historia de la criptografía.

La historia de la criptografía es larga y abunda en anécdotas. Ya las primeras civilizaciones desarrollaron técnicas para enviar mensajes durante las campañas militares de

forma que si el mensajero era interceptado la información que portaba no corriera el peligro de caer en manos del enemigo [1].

El origen de la criptografía data del año 2000 AC., con los egipcios y sus jeroglíficos. Los jeroglíficos estaban compuestos de pictogramas complejos, donde sólo el significado completo podría ser interpretado por algunos. Posiblemente, el primer criptosistema que se conoce fuera documentado por el historiador griego Polibio: un sistema de sustitución basado en la posición de las letras en una tabla.

También los romanos utilizaron sistemas de sustitución, siendo el método actualmente conocido como César, porque supuestamente Julio César(100 AC. a 44 AC.), lo utilizó en sus campañas, uno de los más conocidos en la literatura (según algunos autores, en realidad Julio César no utilizaba este sistema de sustitución, pero la atribución tiene tanto arraigo que el nombre de éste método de sustitución ha quedado para los anales de la historia). Otro de los métodos criptográficos utilizados por los griegos fue la escitala espartana, un método de trasposición basado en un cilindro que servía como clave en el que se enrollaba el mensaje para poder cifrar y descifrar [1].

No sólo los romanos, sino los árabes y los vikingos hicieron uso de sistemas de cifrado.

Gabriel de Lavinde hizo de la criptografía una ciencia más formal cuando publicó su primer manual sobre Criptología en 1379, Samuel Morse el Código Morse, desarrollado en 1832, aunque no es propiamente un código como los otros, es una forma de cifrar las letras del alfabeto dentro de sonidos largos y cortos [1].

Otro de los criptógrafos más importantes del siglo XVI fue el francés Blaise de Vigenere que escribió un importante tratado sobre "la escritura secreta" y que diseñó una cifra que ha llegado a nuestros días asociada a su nombre. A Selenus se le debe la obra criptográfica "Cryptomenytices et Cryptographiae" (Lüneburg, 1624). Durante los siglos XVII, XVIII y XIX, el interés de los monarcas por la criptografía fue notable. Las huestes de Felipe II utilizaron durante mucho tiempo una cifra con un alfabeto de más de 500 símbolos que los matemáticos del rey consideraban inexpugnable. Cuando el matemático francés François Viète consiguió criptoanalizar aquel sistema para el rey de Francia, a la sazón Enrique IV, el conocimiento mostrado por el rey francés impulsó una queja de la corte española ante del papa Pío V acusando a Enrique IV de utilizar magia negra para vencer a sus ejércitos. Por su parte, la reina María Estuardo, reina de los

Escoceses, fue ejecutada por su prima Isabel I de Inglaterra al descubrirse un complot de aquella tras un criptoanálisis exitoso por parte de los matemáticos de Isabel [1].

1.5.2 Criptografía medieval.

Fue probablemente el análisis textual del Corán, de motivación religiosa, lo que llevó a la invención de la técnica del análisis de frecuencias para romper los cifrados por sustitución monoalfabéticos, en algún momento alrededor del año 1000. Fue el avance criptoanalítico más importante hasta la Segunda Guerra Mundial. Esencialmente, todos los cifrados quedaron vulnerables a esta técnica criptoanalítica hasta la invención del cifrado polialfabético por Leon Battista Alberti (1465) inventó un nuevo sistema de sustitución polialfabética que supuso un gran avance de la época y muchos lo siguieron siendo desde entonces [1].

La criptografía se hizo todavía más importante (secretamente) como consecuencia de la competición política y la revolución religiosa. Por ejemplo, en Europa, durante el Renacimiento, ciudadanos de varios estados italianos, incluidos los Estados Pontificios y la Iglesia Católica, fueron responsables de una rápida proliferación de técnicas criptoanalíticas, de las cuales muy pocas reflejaban un entendimiento (o siquiera el conocimiento) del avance de Alberti. Los «cifrados avanzados», incluso después de Alberti, no eran tan avanzados como afirmaban sus inventores/desarrolladores/usuarios (y probablemente ellos mismos creían); puede que este sobre optimismo sea algo inherente a la criptografía, ya que entonces y hoy en día es fundamentalmente difícil saber realmente cómo de vulnerable es un sistema. En ausencia del conocimiento, son comunes las conjeturas y esperanzas, como es de esperar.

La criptografía, el criptoanálisis y la traición cometida por agentes y mensajeros en la conspiración de Babington, durante el reinado de la reina Isabel I de Inglaterra, provocaron la ejecución de María, reina de los escoceses. Un mensaje cifrado de la época de el hombre de la máscara de hierro (descifrado poco antes del año 1900 por Étienne Bazeries) ha arrojado algo de luz (no definitiva, lamentablemente) sobre la identidad real de ese prisionero legendario y desafortunado. La criptografía y su mala utilización estuvieron implicadas en la conspiración que condujo a la ejecución de Mata Hari y en la confabulación que provocó la ridícula condena y encarcelamiento de Dreyfus, ambos hechos acaecidos a principios del siglo XX. Afortunadamente, los criptógrafos también

jugaron su papel para exponer las maquinaciones que provocaron los problemas de Dreyfus; Mata Hari, en cambio, fue fusilada.

Fuera del Medio Oriente y Europa, la criptografía permaneció comparativamente subdesarrollada. En Japón no se utilizó la criptografía hasta 1510, y las técnicas avanzadas no se conocieron hasta la apertura del país hacia occidente en los años 1860 [1].

1.5.3 Criptografía desde 1800 hasta la Segunda Guerra Mundial.

Aunque la criptografía tiene una historia larga y compleja, hasta el siglo XIX no desarrolló nada más que soluciones ad hoc para el cifrado y el criptoanálisis (la ciencia que busca debilidades en los criptosistemas). Ejemplos de lo último son el trabajo de Charles Babbage, en la época de la Guerra de Crimea, sobre el criptoanálisis matemático de los cifrados polialfabéticos, redescubierto y publicado algo después por el prusiano Fiedrich Kasiski. En esa época, el conocimiento de la criptografía consistía normalmente en reglas generales averiguadas con dificultad; véase, por ejemplo, los escritos de Auguste Kerckhoffs sobre criptografía a finales del siglo XIX. Edgar Allan Poe desarrolló métodos sistemáticos para resolver cifrados en los años 1840. Concretamente, colocó un anuncio de sus capacidades en el periódico de Filadelfia Alexander's Weekly (Express) Messenger, invitando al envío de cifrados, que él procedía a resolver. Su éxito creó excitación entre el público durante unos meses. Más tarde escribió un ensayo sobre los métodos criptográficos que resultaron útiles para descifrar los códigos alemanes empleados durante la Primera Guerra Mundial [1].

Proliferaron métodos matemáticos en la época justo anterior a la Segunda Guerra Mundial (principalmente con la aplicación, por parte de William F. Friedman, de las técnicas estadísticas al desarrollo del criptoanálisis y del cifrado, y la rotura inicial de Marian Rejewski de la versión del Ejército Alemán del sistema Enigma). Tanto la criptografía como el criptoanálisis se han hecho mucho más matemáticas desde la Segunda Guerra Mundial. Aun así, ha hecho falta la popularización de los ordenadores y de Internet como medio de comunicación para llevar la criptografía efectiva al uso común por alguien que no sea un gobierno nacional u organizaciones de tamaño similar [1].

1.5.5 Criptografía de la Segunda Guerra Mundial.

En la Segunda Guerra Mundial, las máquinas de cifrado mecánicas y electromecánicas se utilizaban extensamente, aunque —allá donde estas máquinas eran poco prácticas— los sistemas manuales continuaron en uso. Se hicieron grandes avances en la rotura de cifrados, todos en secreto. La información acerca de esta época ha empezado a desclasificarse al llegar a su fin el periodo de secreto británico de 50 años, al abrirse lentamente los archivos estadounidenses y al irse publicando diversas memorias y artículos.

Los alemanes hicieron gran uso de diversas variantes de una máquina de rotores electromecánica llamada Enigma. El matemático Marian Rejewski, de la Oficina de Cifrado polaca, reconstruyó en diciembre de 1932 la máquina Enigma del ejército alemán, utilizando la matemática y la limitada documentación proporcionada por el capitán Gustave Bertrand, de la inteligencia militar francesa. Este fue el mayor avance del criptoanálisis en más de mil años. Rejewsky y sus colegas de la Oficina de Cifrado, Jerzy Różycki y Henryk Zygalski, continuaron desentrañando la Enigma y siguiendo el ritmo de la evolución de los componentes de la máquina y los procedimientos de cifrado. Al irse deteriorando los recursos financieros de Polonia por los cambios introducidos por los alemanes, y al irse avecinando la guerra, la Oficina de Cifrado, bajo órdenes del estado mayor polaco, presentaron a representantes de la inteligencia francesa y británica los secretos del descifrado de la máquina Enigma, el 25 de julio de 1939, en Varsovia.

Poco después de que estallará la Segunda Guerra Mundial el 1 de septiembre de 1939, el personal clave de la Oficina de Cifrado fue evacuado hacia el sureste; el 17 de septiembre, tras la entrada de la Unión Soviética en el este de Polonia, cruzaron Rumanía. Desde allí alcanzaron París, en Francia; en la estación de inteligencia polaco-francesa PC Bruno, cerca de París, continuaron rompiendo la Enigma, colaborando con los criptólogos británicos de Bletchley Park, que se habían puesto al día con el tema. Con el tiempo, los criptólogos británicos —en los que se incluían lumbreras como Gordon Welchman y Alan Turing, el fundador conceptual de la computación moderna— hicieron progresar sustancialmente la escala y tecnología del descifrado Enigma.

El 19 de abril de 1945 se ordenó a los oficiales superiores británicos que nunca debían revelar que se había roto el código de la máquina Enigma alemana, porque esto le dar-

ía la oportunidad al enemigo de decir que "no fueron vencidos justa y satisfactoriamente" [1].

Los criptógrafos de la Armada estadounidense (en cooperación con criptógrafos británicos y holandeses a partir de 1940) rompieron varios sistemas criptográficos de la Armada japonesa. La rotura de uno de ellos, el JN-25, condujo a la célebre victoria estadounidense de la Batalla de Midway. Un grupo del ejército estadounidense, el SIS, consiguió romper el sistema criptográfico diplomático japonés de alta seguridad (una máquina electromecánica llamada Púrpura por los estadounidenses) antes incluso del comienzo de la Segunda Guerra Mundial. Los estadounidenses llamaron a la inteligencia derivada del criptoanálisis, quizás en especial la derivada de la máquina Púrpura, como «Magic» (Magia). Finalmente los británicos se decidieron por «Ultra» para la inteligencia derivada del criptoanálisis, en especial la derivada del tráfico de mensajes cifrados con las diversas Enigmas. Un término británico anterior fue para Ultra fue «Boniface».

Los militares alemanes también desarrollaron varios intentos de implementar mecánicamente la libreta de un solo uso. Bletchley Park los llamó cifrados Fish, y Max Newman y sus colegas diseñaron e implementaron el primer ordenador electrónico digital programable del mundo, Colossus, para ayudarles con el criptoanálisis. La Oficina de Asuntos Exteriores alemana empezó a usar la libreta de un solo uso en 1919; parte de este tráfico fue leído en la Segunda Guerra Mundial como resultado de la recuperación de material importante en Sudamérica que fue desechado con poco cuidado por un mensajero alemán.

La Oficina de Asuntos Exteriores japonesa utilizó un sistema eléctrico lógico basado en uniselectores (llamado Púrpura por EEUU), y también utilizó varias máquinas similares para los agregados de algunas embajadas japonesas. Una de estas recibió el nombre de «Máquina M» por EEUU, y otra fue apodada «Red». Todas fueron rotas en mayor o menor grado por los aliados.

Las máquinas de cifrado aliadas utilizadas en la Segunda Guerra Mundial incluían la Typex británica y la SIGABA estadounidense; ambos eran diseños de rotores electromecánicos similares en espíritu a la Enigma, aunque con mejoras importantes. No se tiene constancia de que ninguna de ellas fuera rota durante la guerra. Los polacos utilizaron la máquina Lacida, pero se demostró que era poco segura y se canceló su uso.

Las tropas de campo utilizaron las familias M-209 y M-94. Los agentes SOE utilizaron inicialmente «cifrados de poema» (las claves eran poemas memorizados), pero más avanzada la guerra empezaron a utilzar libretas de un solo uso [1].

1.5.6 Criptografía moderna.

La era de la criptografía moderna comienza realmente con Claude Shannon, que podría decirse que es el padre de la criptografía matemática. En 1949 publicó el artículo Communication Theory of Secrecy Systems en la Bell System Technical Journal, y poco después el libro Mathematical Theory of Communication, con Warren Weaver. Estos trabajos, junto con los otros que publicó sobre la teoría de la información y la comunicación, establecieron una sólida base teórica para la criptografía y el criptoanálisis. Y, a la vez, la criptografía desapareció de la escena para quedarse dentro de las organizaciones gubernamentales secretas como la NSA. Muy pocos trabajos se hicieron públicos hasta mediados de los 70, cuando todo cambió, en estos años se vivieron dos importantes avances públicos. El primero fue la publicación del borrador del Data Encryption Standard en el Registro Federal estadounidense el 17 de marzo de 1975. La propuesta fue enviada por IBM, por invitación de la Oficina Nacional de Estándares (ahora NIST), en un esfuerzo por desarrollar sistemas de comunicación electrónica segura para las empresas como los bancos y otras organizaciones financieras grandes. Tras «asesoramiento» y ciertas modificaciones por parte de la NSA, fue adoptado y publicado como un Federal Information Processing Standard en 1977 (actualmente el FIPS 46-3). El DES fue el primer cifrado accesible públicamente que fue «bendecido» por una agencia nacional como la NSA. La publicación de sus especificaciones por la NBS estimuló una explosión del interés público y académico por la criptografía [1].

DES fue suplantado oficialmente por el Advanced Encryption Standard (AES) en 2001, cuando el NIST anunció el FIPS 197. Tras una competición abierta, el NIST seleccionó el Rijndael, enviado por dos criptógrafos belgas, para convertirse en el AES. El DES, y otras variantes más seguras (como el Triple DES; ver FIPS 46-3), todavía se utilizan hoy en día, y se han incorporado en muchos estándares nacionales y de organizaciones. Sin embargo, se ha demostrado que el tamaño de su clave, 56 bits, es insuficiente ante ataques de fuerza bruta (un ataque así, llevado a cabo por el grupo pro libertades civiles digitales Electronic Frontier Foundation en 1997, tuvo éxito en 56 horas —la his-

toria se cuenta en Cracking DES, publicado por O'Reilly Associates). Como resultado, hoy en día el uso sin más del cifrado DES es sin duda inseguro para los nuevos diseños de criptosistemas, y los mensajes protegidos por viejos criptosistemas que utilizan el DES, y de hecho todos los mensajes enviados desde 1976 que usan el DES, también están en riesgo. A pesar de su calidad inherente, el tamaño de la clave DES (56 bits) fue considerado por algunos como demasiado pequeño incluso en 1976; quizás la voz más sonora fue la de Whitfield Diffie. Había sospechas de que las organizaciones gubernamentales tenían suficiente potencia de cálculo para romper los mensajes DES; ahora es evidente que otros han logrado esa capacidad.

El segundo desarrollo, en 1976, fue quizás más importante todavía, ya que cambió de manera fundamental la forma en la que los criptosistemas pueden funcionar. Fue la publicación del artículo New Directions in Cryptography, de Whitfield Diffie y Martin Hellman. Introdujo un método radicalmente nuevo para distribuir las claves criptográficas, dando gran un paso adelante para resolver uno de los problemas fundamentales de la criptografía, la distribución de claves, y ha terminado llamándose intercambio de claves Diffie-Hellman. El artículo también estimuló el desarrollo público casi inmediato de un nuevo tipo de algoritmo de cifrado, los algoritmos de cifrado asimétrico.

Antes de eso, todos los algoritmos de cifrado útiles eran algoritmos de cifrado simétrico, en los que tanto el remitente como el destinatario utilizan la misma clave criptográfica, que ambos deben mantener en secreto. Todas las máquinas electromecánicas utilizadas en la Segunda Guerra Mundial eran de esta clase lógica, al igual que los cifrados César y Atbash y en esencia todos los cifrados y sistemas de códigos de la historia. La «clave» de un código es, por supuesto, el libro de códigos, que debe asimismo distribuirse y mantenerse en secreto.

En estos sistemas era necesario que la partes que se iban a comunicar intercambiaran las claves de alguna forma segura antes del uso del sistema (el término que se solía utilizar era «mediante un canal seguro»), como un mensajero de confianza con un maletín esposado a su muñeca, o un contacto cara a cara, o una paloma mensajera fiel. Este requisito nunca es trivial y se hace inmantenible rápidamente al crecer el número de participantes, o cuando no hay canales seguros disponibles para el intercambio de claves, o cuando las claves cambian con frecuencia (una práctica criptográfica sensata). En particular, si se pretende que los mensajes sean seguros frente a otros usuarios,

hace falta una clave distinta para cada par de usuarios. Un sistema de este tipo se conoce como criptosistema de clave secreta o de clave simétrica. El intercambio de claves D-H (y las posteriores mejoras y variantes) hizo que el manejo de estos sistemas fuera mucho más sencillo y seguro que nunca.

En contraste, el cifrado de clave asimétrica utiliza un par de claves relacionadas matemáticamente, en el que una de ellas descifra el cifrado que se realiza con la otra. Algunos (pero no todos) de estos algoritmos poseen la propiedad adicional de que una de las claves del par no se puede deducir de la otra por ningún método conocido que no sea el ensayo y error. Con un algoritmo de este tipo, cada usuario sólo necesita un par de claves. Designando una de las claves del par como privada (siempre secreta) y la otra como pública (a menudo visible), no se necesita ningún canal seguro para el intercambio de claves. Mientras la clave privada permanezca en secreto, la clave pública puede ser conocida públicamente durante mucho tiempo sin comprometer la seguridad, haciendo que sea seguro reutilizar el mismo par de claves de forma indefinida.

La efectividad de los algoritmos asimétricos depende de una clase de problemas matemáticos conocidos como funciones de un solo sentido, que requieren relativamente poca potencia de cálculo para ejecutarse, pero muchísima potencia para calcular la inversa. Un ejemplo clásico de función de un sentido es la multiplicación de números primos grandes. Es bastante rápido multiplicar dos primos grandes, pero muy difícil factorizar el producto de dos primos grandes. Debido a las propiedades matemáticas de las funciones de un sentido, la mayor parte de las claves posibles tienen poca calidad para su uso criptográfico; solo una pequeña parte de las claves posibles de una cierta longitud son candidatas ideales, y por tanto los algoritmos asimétricos requieren claves muy largas para alcanzar el mismo nivel de seguridad proporcionado por las claves simétricas, relativamente más cortas. Las exigencias de generar el par de claves y realizar el cifrado/descifrado hacen que los algoritmos asimétricos sean costosos computacionalmente. Como, a menudo, los algoritmos simétricos pueden usar como clave cualquier serie seudoaleatoria de bits, se puede generar rápidamente una clave de sesión desechable para uso a corto plazo. Por consiguiente, es una práctica común utilizar una clave asimétrica larga para intercambiar una clave simétrica desechable mucho más corta (pero igual de fuerte). El algoritmo asimétrico, más lento, envía de forma segura una

clave simétrica de sesión, y entonces el algoritmo simétrico, más rápido, toma el control para el resto del mensaje.

La criptografía de clave asimétrica, el intercambio de claves Diffie-Hellman y los famosos algoritmos de clave pública/clave privada (es decir, lo que se suele llamara algoritmo RSA), parecen haber sido desarrollados de manera independiente en una agencia de inteligencia británica antes del anuncio público de Diffie y Hellman en el 76. El GCHQ ha publicado documentos que afirman que ellos habían desarrollado la criptografía de clave pública antes de la publicación del artículo de Diffie y Hellman. Varios artículos clasificados fueron escritos en el GCHQ durante los años 60 y 70, que finalmente llevaron a unos sistemas esencialmente idénticos al cifrado RSA y al intercambio de claves Diffie-Hellman en 1973 y 1974. Algunos de ellos se acaban de publicar, y los inventores (James H. Ellis, Clifford Cocks y Malcolm Williamson han hecho público parte de su trabajo [1].

El hombre ha tenido como una de sus prioridades en el proceso de comunicación, la confidencialidad de sus mensajes de tal forma que solamente puedan ser interpretados correctamente por el emisor de los mismos y por el receptor al que van dirigidos.

Para esto se han implementado diferentes sistemas de protección, la Criptología ha demostrado con el tiempo ser una de las mejores técnicas para resolver esta cuestión, y actualmente es una de las técnicas más usadas para la protección de datos durante su transmisión y almacenamiento.

La "Criptografía" viene de "Kryptos" secreto o escondido y "Graphos" escritura, es decir, es la ciencia de la escritura secreta.

La Criptografía es una rama de las Matemáticas, que se complementa con el Criptoanálisis, que es la técnica de descifrar textos cifrados sin tener autorización para ello, es decir se realiza una Criptografía inversa. Ambas técnicas forman la Criptología [1].

La Criptografía es la parte de la Criptología que estudia como cifrar efectivamente los mensajes; es la técnica de transformar un mensaje inteligible, denominado texto claro, en otro que sólo puedan entender las personas autorizadas a ello, denominado criptograma o texto cifrado.

La base de la criptografía suele ser la aplicación de problemas matemáticos de difícil solución a aplicaciones específicas, denominándose criptosistema o sistema de cifrado a los fundamentos y procedimientos de operación involucrados en dicha aplicación.

Existen muchas opciones para hacer un sistema seguro mediante algoritmos de encriptación, ya que utilizada correctamente, la criptografía proporciona ciertos rasgos de seguridad estándar como son [2] [3]:

- *Autenticidad:* consiste en la seguridad de que las personas que intervienen en el proceso de comunicación son las que dicen ser; es decir, asegura que la gente con las que Usted trata no sea impostora.

  El método más usado para probar la autenticidad es la firma digital, basada en la criptografía, empleando llaves o claves. Las llaves son una secuencia bastante larga de caracteres y números, generadas por un procedimiento matemático. Su utilización no es, ni más ni menos, que un proceso por el cual los comunicantes poseen dos llaves: una privada, que mantienen en su poder y una pública, que está a disposición de los posibles intercomunicadores. La criptografía permite autentificar a la persona con quien se está realizando la comunicación.

- *Confidencialidad:* seguridad de que los datos que contiene el documento pertenecen ocultos a los ojos de terceras personas durante su transferencia.

  La confidencialidad se asegura mediante la encriptación del texto o archivo, por cualquiera de los métodos existentes, de tal forma que nadie conozca las claves con las que se ha enviado, así como también la información que contenga dicho archivo.

- *Integridad:* consiste en la seguridad de que los datos del documento no sufra ninguna modificación a lo largo de la transferencia (regularmente por medio de internet), es sí, es la no alteración del mensaje desde su salida hasta su llegada.

  La comprobación de la integridad se suele realizar mediante firmas electrónicas, generalmente basadas en funciones hash. La autenticidad es condición suficiente para la Integridad, razón por la cual se afirma que si un documento es auténtico es integro, pero no al revés.

- *No repudio:* una vez enviado el documento, su emisor puede negar haber sido el autor de dicho envío. El no repudio es la condición de imposibilidad de negación del envío de un mensaje, además de que es suficiente para la autenticidad por lo que si un documento es no repudiable es auténtico.

Cualquier sistema de transferencia segura basado en criptografía, debe abarcar estos cuatro aspectos, ya que los sistemas de clave simétrica ofrecen confidencialidad, pero ninguno de los otros factores, mientras que los de clave asimétrica ofrecen autenticidad, integridad, confidencialidad en el envío y no repudio si van asociados con una firma digital.

1.6 Alcances y limitaciones.

Los alcances de este programa de encriptación se describen a continuación:

- Es una herramienta tecnológica que sirve como apoyo para cualquier persona que quiera mantener segura su información
- Con esta herramienta al usuario se le facilitará asegurar su información.
- Es una herramienta basada en los algoritmos de encriptación silábicos que existen.
- Es una herramienta que puede abrir un archivo .txt con el texto claro o el usuario puede escribir el texto directamente en ella.
- Realiza el proceso de desencriptación solo con el texto encriptado, esto se realiza con la finalidad de comprobar que el proceso de encriptación fue realizado correctamente.
- Puede guardar la información encriptada en un archivo .txt para ser utilizada después para lo que se requiera.

Las limitaciones de este programa de encriptación se describen a continuación:

- No es una herramienta que trabaja de manera distribuida o de comunicación cliente/servidor.
- Si el texto encriptado fue el primero que se introdujo al programa no se puede realizar la desencriptación.

CAPÍTULO 2

SEGMENTACIÓN DE LAS PALABRAS EN SILABAS

CAPÍTULO 2

SEGMENTACIÓN DE LAS PALABRAS EN SÍLABAS

2.1 Introducción.

La división silábica tiene especial importancia en el escrito ya que en español sí se permite dividir las sílabas de una palabra cuando ésta no cabe en el renglón en uso. Aunque, a veces, los límites silábicos pueden ser difusos y existen tendencias a provocar diptongos o hiatos según la velocidad con la que se hable. En algunos idiomas el sistema de escritura es silábico, es decir, un carácter representa una sílaba.

¿Qué es la silaba?.- Es el conjunto de letras que se pronuncian juntas en una sola emisión de voz [4].

La división silábica de una palabra se suele anotar con guiones (-) y dependiendo del número de sílabas una palabra puede ser [4]:

- *Monosílaba.-* una sola sílaba; en español no se acentúa salvo que haya dos palabras iguales para diferenciarlas (tilde diacrítica) o en casos como ciertos exclamativos e interrogativos. Ejemplos: sol, sí, más, dos, sed...etc.
- *Bisílaba.-* dos sílabas. Ejemplos: calor, mano, árbol...etc.
- *Trisílaba.-* tres sílabas. Ejemplos: repetir, tímbrico, recoger…etc.
- *Polisílaba.-* más de tres sílabas. Ejemplos: azulado, diccionario, policlínica...etc.

En el español existen 30 letras, las cuales están clasificadas de acuerdo a su pronunciación en dos grupos: vocales y consonantes [5]. El grupo de las vocales está formado por seis letras, desde el punto de vista fonético, su pronunciación no dificulta la salida

del aire. La boca actúa como una caja de resonancia abierta en menor o mayor grado y de acuerdo a esto, las vocales se clasifican en abiertas, semiabiertas o cerradas (*Figura 2.1.1*).

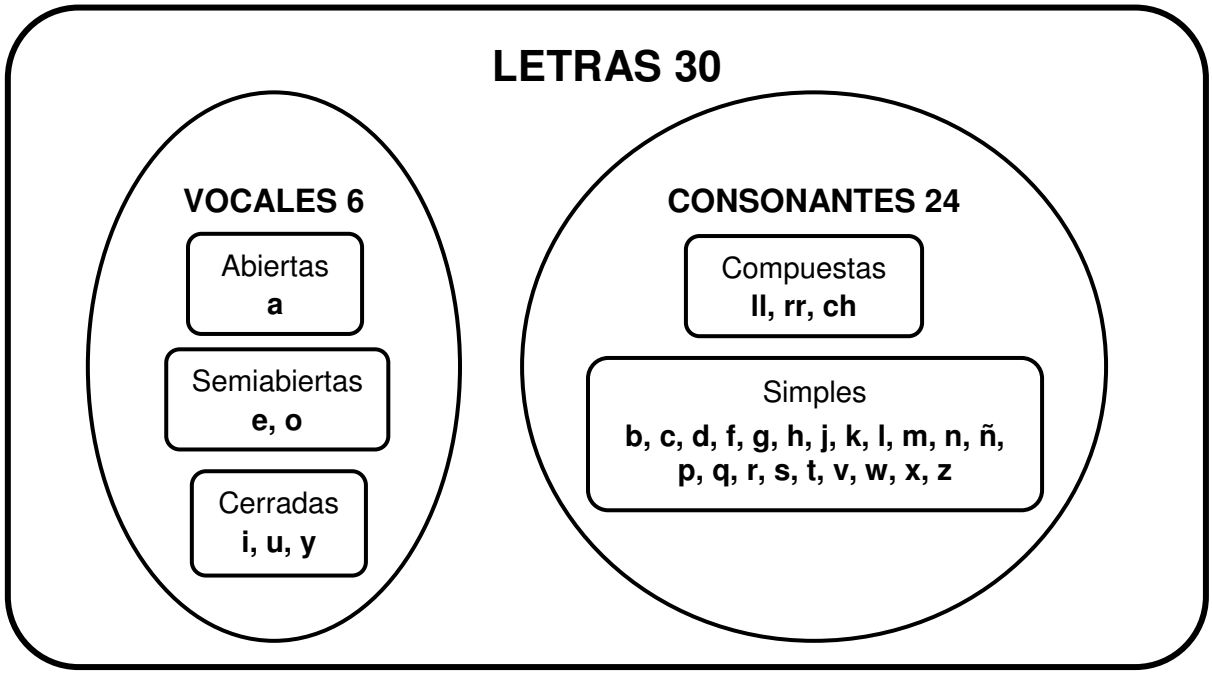


Figura 2.1.1 Las letras y su clasificación

2.2 Reglas del idioma para la formación de Sílabas.

Para expresar las reglas con claridad se utiliza la notación mostrada en la tabla 2.1.

Símbolo	Descripción
¤	Fin de palabra o carácter diferente a vocal o consonante (‘,’, ‘-’, ‘1’, etc.)
[CV]	Las siguientes dos letras son CV
[CVV]	Las siguientes dos letras pertenecen al grupo de consonantes inseparables
V*	Vocal acentuada
|	Utilizada para establecer posibilidades alternativas
Li	I-ésima letra de la sílaba o palabra
P	Palabra que se va a dividir en sílabas

Tabla 2.2.1 Notación utilizada

En el idioma Castellano existen 3108 sílabas incluyendo sílabas con acento y sin acento [6], se declaran en el apéndice I únicamente las palabras sin acento, ya que el sistema sólo encriptará palabras sin acento, además de once reglas fundamentales [2,3], las cuales determinan la separación de las sílabas de cada palabra. Estas reglas son listadas a continuación, mostrando enseguida de la regla ejemplos de la misma, así como excepciones.

- ***Regla 1.-*** En las sílabas, por lo menos, siempre tiene que haber una vocal. Sin vocal no hay sílaba.
- ***Regla 2.-*** Cada elemento del grupo de consonantes inseparables, mostrado en la *Figura 2.1.2*, no puede ser separado al dividir una palabra en sílabas.

br, bl, cr, cl, dr,
fr, fl, gr, gl, kr, ll,
pr, pl, tr, rr, ch

Figura 2.1.2 Consonantes inseparables

- ***Regla 3.-*** Cuando una consonante se encuentra entre dos vocales, se une a la segunda vocal, como se muestra en el ejemplo de la *Tabla 2.2.*2.

Palabra	**Sílabas**
u n e	u + ne

Tabla 2.2.2 Consonantes entre dos vocales

- ***Regla 4.-*** Cuando hay dos consonantes entre dos vocales, cada vocal se une a una consonante, como se muestra en el ejemplo de la *Tabla 2.2.3*.
  Regla de excepción.- Esto no ocurre en el grupo de consonantes inseparables (*Regla 2*).

	Palabra	**Sílabas**
Ejemplo	c o m p o n e r	com + po + ner
Ejemplo de excepción	a p r e n d e r	a + pren + der

Tabla 2.2.3 Dos consonantes entre dos vocales

- ***Regla 5.-*** Si son tres las consonantes colocadas entre dos vocales, las dos primeras consonantes se asociarán con la primera vocal y la tercera consonante con la segunda vocal, como se muestra en el ejemplo de la *Tabla 2.2.4*.

  Regla de excepción.- Esta regla no se cumple cuando la segunda y tercera consonante forma parte del grupo de consonantes inseparables.

	Palabra	**Sílabas**
Ejemplo	t r a n s p o r t e	trans + por + te
Ejemplo de excepción	c u m p l e	cum + ple

Tabla 2.2.4 Tres consonantes entre dos vocales

- ***Regla 6.-*** La *Tabla 2.2.5* nos muestra el caso cuando las palabras que contienen una h precedida o seguida de otra consonante, se dividen separando ambas letras.

Palabra	**Sílabas**
a n h e l o	an + he + lo

Tabla 2.2.5 Cuando existe h en una palabra

- ***Regla 7.-*** El diptongo es la unión inseparable de dos vocales. Se pueden presentar tres tipos de diptongos posibles:
  - (Una vocal abierta + una vocal cerrada)
  - (Una vocal cerrada + una vocal abierta)
  - (Una vocal cerrada + una vocal cerrada)

  Son diptongos sólo las siguientes parejas de vocales: ai, au, ei, eu, io, ou, ia, ua, ie, ue, oi, uo, ui, iu, ay, ey, oy, como se muestra en el ejemplo de la *Tabla 2.2.6*.

Palabra	**Sílabas**
j a u l a	jau + la

Tabla 2.2.6 Ejemplo de diptongo

La unión de dos vocales abiertas o semiabiertas no forman diptongo, es decir, deben separarse en la segmentación silábica. Pueden quedar solas o unidas a una consonante, como se muestra en el ejemplo de la *Tabla 2.2.7*.

Palabra	Sílabas
a e r e o	a + e + re + o

Tabla 2.2.7 Ejemplo que no forma diptongo

- ***Regla 8.-*** La h entre dos vocales, no destruye un diptongo, como se observa en el ejemplo de la *Tabla 2.2.8*.

Palabra	Sílabas
a h u y e n t a r	ahu + yen + tar

Tabla 2.2.8 La h entre dos vocales

- ***Regla 9.-*** La acentuación sobre la vocal cerrada de un diptongo provoca su destrucción, como se muestra en el ejemplo de la *Tabla 2.2.9*.

Palabra	Sílabas
M a r í a	Ma + rí + a

Tabla 2.2.9 Destrucción del diptongo

- ***Regla 10.-*** La unión de tres vocales forma un triptongo. La única disposición posible para la formación de triptongos es la que indica el esquema:

  Vocal cerrada + (vocal abierta | vocal semiabierta) + vocal cerrada

  Sólo las siguientes combinaciones de vocales, forman un triptongo: iai, iei, uai, uei, uau, iau, uay, uey.

2.3 Estructura de las sílabas y su clasificación.

De acuerdo a las reglas mencionadas anteriormente se deduce una estructura de las sílabas que son las siguientes:

a) Vocal (1 ó 2 vocales) (V).
b) Vocal (1 ó 2 vocales) + consonante (1 consonantes) (VC).

c) Consonante (1 ó 2 consonantes) + Vocal (1, 2 ó 3 vocales) (CV).
d) Consonante (1 ó 2 consonantes) + Vocal (1, 2 ó 3 vocales) + Consonante (1 ó 2 consonantes) (CVC).

Otra clasificación de las sílabas es por el número de letras que la componen, se muestran a continuación:

a) *Monolítera*.- formada por una sola letra. Ejemplos: a, e, i.
b) *Bilítera*.- formada por dos letras. Ejemplos: de, la, en.
c) *Trílitera*.- formada pro tres letras. Ejemplos: por, del sin.
d) *Cuadrilítera*.- formada por cuatro letras. Ejemplos: tres, tras.
e) *Pentalítera*.- formada por cinco letras. Ejemplo: trans.

Como se observa no existen sílabas con más de cinco letras.

2.4 Agrupación de las Sílabas.

De acuerdo a las reglas mencionadas anteriormente, se concluye que todas las posibles combinaciones de sílabas en el idioma español, caen dentro de los siguientes casos:

- **Caso 1. Inicio de sílaba: Vocal.**
  En la *Tabla 2.4.1* se muestran las posibles combinaciones de este caso, un ejemplo de cada combinación y el número de regla aplicada para la separación en sílabas de esa palabra.

Combinación	Ejemplo	Número de regla aplicada
V	A	1
VC	Ár + bol	4
VV	Au + tomóvil	7
VVC	Aun + que	7, 4

Tabla 2.4.1 Caso 1: Inicio de sílaba: Vocal

- **Caso 2.- Inicio de sílaba: Consonante + Vocal**

  De igual forma que en el caso anterior, la *Tabla 2.4.2* describe este caso con sus respectivos ejemplos.

Combinación	Ejemplo	Número de regla aplicada
C	Y	1
CV	La	1
CVC	Las	1
CVCC	Cons + tante	5
CVV	Jau + la	7
CVVC	Cuan + do	4
CVVV	Cuau + tla	10
CVVVC	Cuahu + temoc	10, 6

Tabla 2.4.2 Caso 2: Inicio de sílaba: Consonante + Vocal

- **Caso 3.- Inicio de sílaba: Consonante + Consonante.**

  En la *Tabla 2.4.3*, mostrada a continuación se describen las posibles combinaciones de sílabas para este caso.

Combinación	Ejemplo	Número de regla aplicada
CCV	Tra + po	2
CCVC	Tras + to	2, 4
CCVCC	Trans + porte	2, 5
CCVV	Trau + ma	2, 7
CCVVC	Claus + trofobia	2, 7, 5

Tabla 2.4.3 Caso 3: Inicio de sílaba: Consonante + Consonante

CAPÍTULO 3

ANÁLISIS

CAPÍTULO 3

ANÁLISIS

3.1 Descripción general del proyecto.

El sistema a implementar es una aplicación que encripta un texto en español utilizando la forma silábica. Para implementarlo, es necesario contar con todas las sílabas del idioma Español, para esto se necesita la separación silábica de cada palabra, tomando en cuenta las reglas ortográficas para separarlas, lo cual es importante ya que a partir de esto y de las letras con las que cuente cada sílaba se le asignará un valor, el cual corresponderá a una sílaba encriptada.

Una vez que se cuente con cada sílaba se convertirá el texto claro (el cual puede ser leído por cualquier usuario) a un texto cifrado (o transformación criptográfica y este ya no puede ser leído tan fácilmente).

Java es el lenguaje de programación en el que se implementará el programa para realizar la encriptación de un texto en español utilizando la forma silábica, el cuál es uno de los lenguajes adecuados para la realización de este tipo de proyectos por su versatilidad, eficiencia, portabilidad y la seguridad que aporta.

Este programa está enfocado a aquellas personas que manejan información importante la cual no puede ser modificada en su almacenamiento ni interceptada durante su transmisión por personas ajenas a ella.

El programa a implementar funcionará de la siguiente manera: cuando el usuario haya ingresado al programa podrá escoger entre dos opciones:

- **Abrir un archivo .txt que contenga texto claro**
- **Escribir el texto claro en un cuadro de texto**

Después de esto el usuario podrá encriptar los datos y enseguida aparecerán en otro cuadro de texto y por último el usuario podrá escoger entre:

- **Desencriptar**
- **Guardar en archivo**

La opción de ***Desencriptar***, como su nombre lo indica desencriptará el texto previamente encriptado; la opción ***Guardar en archivo***, creará un archivo .txt con el texto encriptado y únicamente funcionará después de realizar la encriptación y antes de desencriptar el texto.

3.2 Metodología.

La metodología que se utilizará en este proyecto de tesis es el *modelo de vida clásico* o *de cascada*. Fue ingeniado por Winston Royce, pero propuesto por Boehm en 1976. Este modelo se caracteriza por visualizar el proceso de desarrollo como una fábrica de producción en cadena, en la que cada fase empieza donde termino la anterior tras un proceso de prueba y validación [7].

El *modelo de vida clásico* fue desarrollado entre 1960 – 1980, sugiere un enfoque sistemático o más bien secuencial del desarrollo de software que comienza en un nivel de sistemas y progresa con el análisis, diseño, codificación, pruebas y mantenimiento. Este modelo es el paradigma de desarrollo de software más antiguo que existe, y el más usado a pesar de sus errores con los que se enfrentó.

El *modelo de vida clásico* es el enfoque metodológico que ordena rigurosamente las etapas del ciclo de vida del software, de tal forma que el inicio de cada etapa debe esperar la finalización de la inmediatamente anterior, desde la concepción de una idea hasta la entrega y el retiro del sistema [7].

Observemos la *Figura 3.2.1* que nos muestra el *ciclo de vida clásico* para entender mejor como funciona.

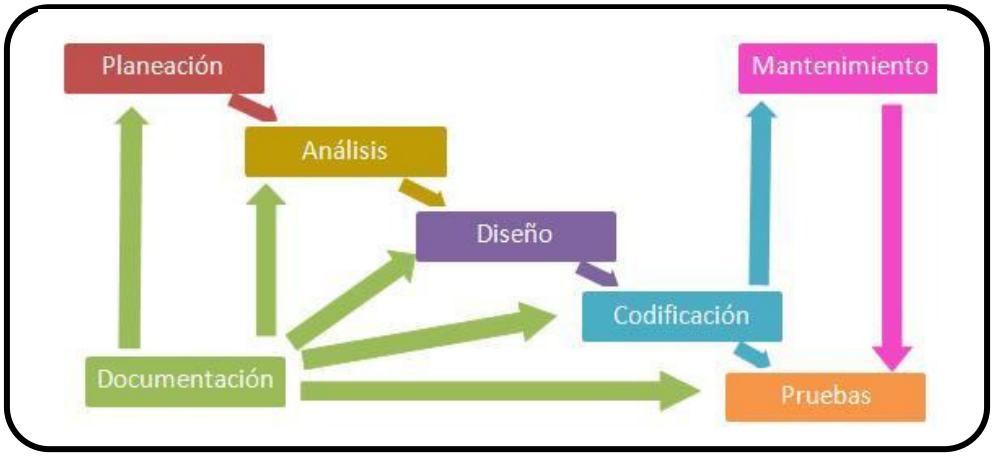


Figura 3.2.1 Ciclo de vida clásico

La descripción de las fases del ciclo de vida clásico es:

- ***Planeación.-*** Se define el problema a resolver, es importante señalar que en esta etapa se deben consensuar todo lo que se requiere del sistema y será a aquello lo que seguirá en las siguientes etapas, no pudiéndose requerir nuevos resultados a mitad del proceso de elaboración del software.
- ***Análisis.-*** Se trata de disponer de información detallada del ambiente de información y de función de software.
- ***Diseño.-*** Contiene la descripción de la estructura relacional global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera de que se combinan unas con otras. También se diseña el programa por medio de los algoritmos necesarios para el cumplimiento de lo requerimientos del usuario, así como también las características de la interfaz, todo esto para saber que herramientas usar en la etapa de codificación.
- ***Codificación.-*** El diseño se traduce a un lenguaje ejecutable, es decir, es la fase de la programación o implementación propiamente dicha. Aquí se implementara el código fuente, haciendo uso de prototipos. Así como pruebas y ensayos para corregir errores.
- ***Pruebas.-*** Se descubren errores lógicos en la implementación. Los elementos ya corregidos, se ensamblan para componer el sistema totalmente, ya que se comprueba que funciona correctamente, puede ser utilizado nuevamente.
- ***Mantenimiento.-*** Se asocia a la corrección de errores, evolución del entorno y a modificaciones

3.3 Estrategias de solución.

La estrategia de solución para la implementación del Sistema que encripta un texto en español utilizando la forma silábica es con el lenguaje de programación *Java*.

La elección de implementar el sistema con Java es por su versatilidad y eficiencia, la portabilidad de su plataforma y la seguridad que aporta, ya que la han convertido en la tecnología ideal para su aplicación en cualquier sistema. Java está diseñado para que un programa escrito en este lenguaje sea ejecutado independientemente de la plataforma (hardware, software y sistema operativo) en la que se esté actuando, además de

que existen diferentes libros, documentos, ejemplos, para su instalación y configuración, lo que ayuda a conocer más el lenguaje.

Un ejemplo claro es que se usa para portátiles a centros de datos, de consolas de juegos a súper equipos científicos, de teléfonos móviles a Internet, Java está en todas partes.

Con todo esto Java facilitará la implementación del Sistema, principalmente para proteger la información entre un usuario transmisor y un usuario receptor, para que dicha información no se pierda, así como también para proteger la información que se encuentra almacenada en una computadora para no exponerla a usuarios ajenos.

3.4 Diagrama de contexto.

La aplicación que encriptará un texto en español utilizando la forma silábica, tendrá dos funciones básicas: *Encriptación* y *Desencriptación.*

Para la *encriptación* el programa utilizará el texto claro contenido en un archivo .txt o el texto claro escrito por el usuario, ésta será la entrada al programa y la salida será el texto cifrado o encriptado. En la *Figura 3.4.1* se muestra el diagrama de contexto para la *encriptación*.

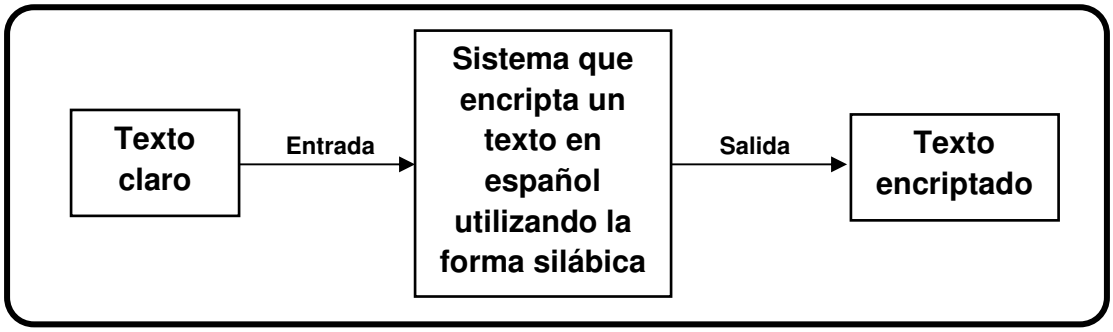


Figura 3.4.1 Diagrama de contexto para la encriptación

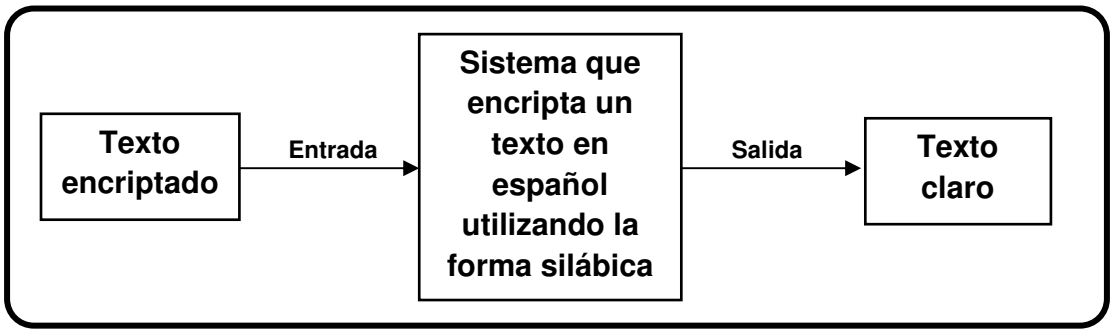


Figura 3.4.2 Diagrama de Contexto para la Desencriptación

Para la *desencriptación* el programa necesitará como entrada el texto cifrado o encriptado previamente y la salida será el texto desencriptado o texto claro. En la *Figura 3.4.2* se muestra el diagrama de contexto para la *desencriptación.*

3.5 Acciones y restricciones del programa.

La *Tabla 3.5.1* muestra las acciones y restricciones del programa:

Acción	Restricción
Acentos	Una de las más importantes restricciones, es que el programa no acepta las palabras acentuadas.
Ejecutar el programa	Es necesario ejecutar el programa correctamente para obtener los resultados deseados.
Abrir archivo	Si se quiere encriptar información contenida en un archivo .txt, se tiene que abrir el archivo que contiene el texto claro, ya que de lo contrario, no se podrán obtener los resultados esperados.
Encripta	Si el texto claro se obtuvo de un archivo .txt o el usuario lo introdujo en un campo de texto el sistema encripta esa información.
Desencriptar	Para desencriptar los datos en necesario encriptarlos previamente, sino se realiza así el sistema provocara un error.
Guardar Archivo	La información encriptada es la única que se puede guardar en un archivo y sólo se puede realizar después de la encriptación y antes de la desencriptación.

Tabla 3.5.1 Acciones y restricciones del programa

3.6 Diagramas de casos de uso.

A continuación se mostrarán los diagramas de casos de uso utilizados para la realización de esta aplicación.

3.6.1 Caso de uso: Menú Principal.

El siguiente caso de uso (*Figura 3.6.1*) muestra el comportamiento del Menú principal, en el cual el usuario escoge entre abrir un archivo .txt que contenga el texto claro o es-

cribir el texto claro en un campo de texto, enseguida puede encriptar el texto y por último puede escoger entre desencriptar el texto o guardar el texto encriptado.

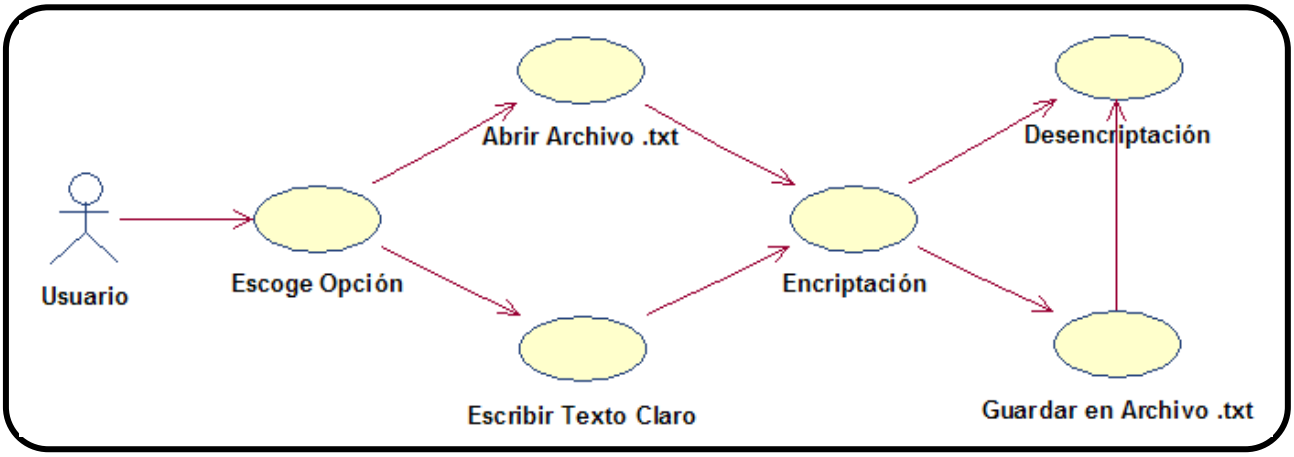


Figura 3.6.1 Caso de uso: Menú Principal

Secuencia normal de ejecución:

1. El usuario elige una de las dos opciones primordiales para realizar la encriptación: abrir archivo .txt o escribir texto claro.
2. Después el usuario escoge la encriptación del texto claro.
3. El usuario puede escoger entre la desencriptación o guardar en archivo .txt.

Excepciones:

1. Si el usuario escoge guardar en archivo .txt después de realizar la encriptación, entonces podrá hacer la desencriptación del texto claro.

3.6.2 Caso de uso: Abrir Archivo .txt.

El siguiente caso de uso (Figura 3.6.2) muestra el comportamiento de *Abrir Archivo .txt,* en la cual el usuario abre el archivo .txt que contiene la información en texto claro, enseguida el sistema carga el texto claro y por último es mostrado en pantalla.

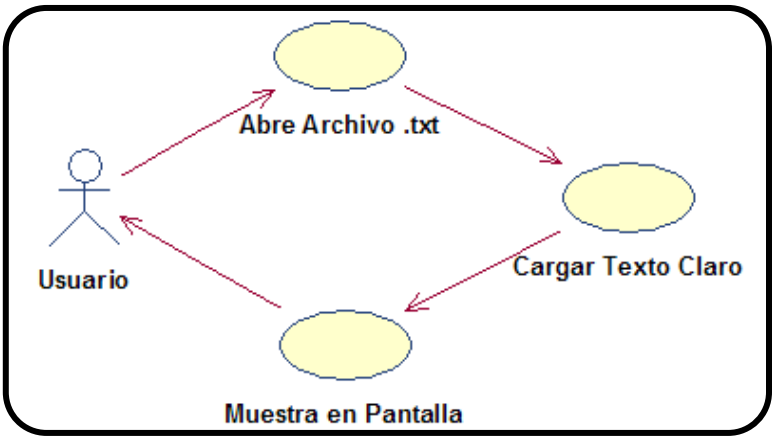


Figura 3.6.2 Caso de uso: Abrir Archivo .txt

Secuencia normal de ejecución:

1. El usuario abre un archivo .txt previamente guardado, éste debe de contener el texto claro para que sea encriptado.
2. El sistema carga el texto claro del archivo .txt.
3. Se muestra el texto claro en pantalla en el campo de texto y listo para ser encriptado.

Excepciones:

1. En caso de haber un error al cargar el archivo .txt el sistema mostrara un mensaje y no podrá realizarse el proceso de encriptación.

3.6.3 Caso de uso: Encriptación.

El siguiente caso de uso (*Figura 3.6.3*) muestra el comportamiento del proceso *Encriptación.*

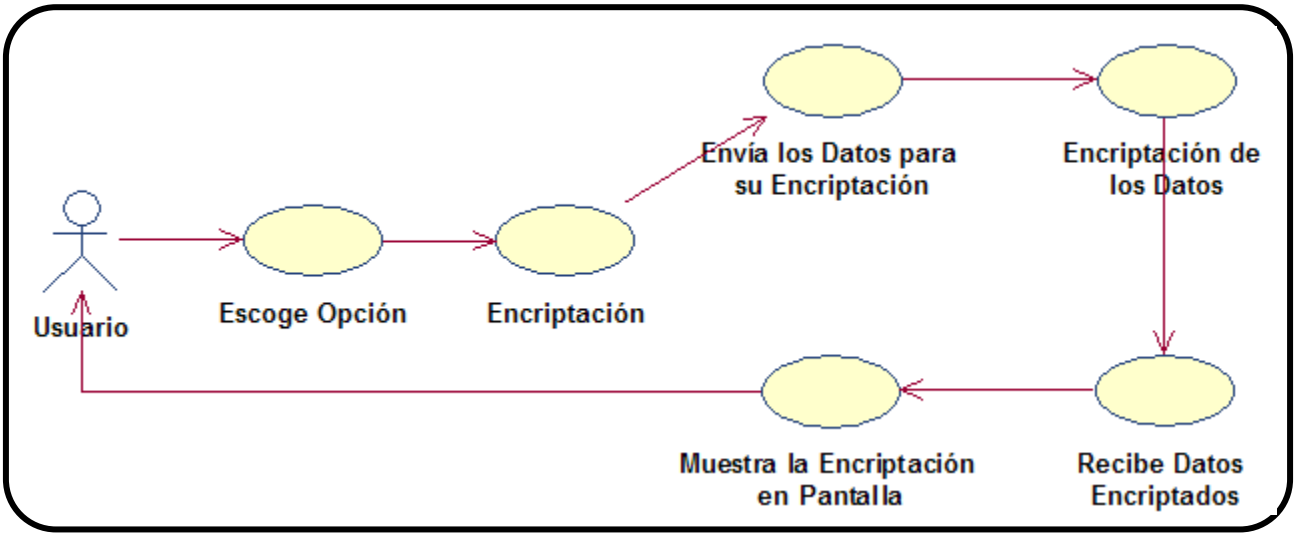


Figura 3.6.3 Caso de uso: Encriptación

Secuencia normal de ejecución:

1. El texto claro se envía para su encriptación.
2. El sistema realiza la encriptación.
3. El sistema envía el texto encriptado.
4. El texto encriptado es mostrado en pantalla.

Excepciones:

1. En caso de haber un error al realizar la encriptación el sistema lo notificará.

3.6.4 Caso de uso: Desencriptación.

El siguiente caso de uso (*Figura 3.6.4*) muestra el comportamiento del proceso *Desencriptación.*

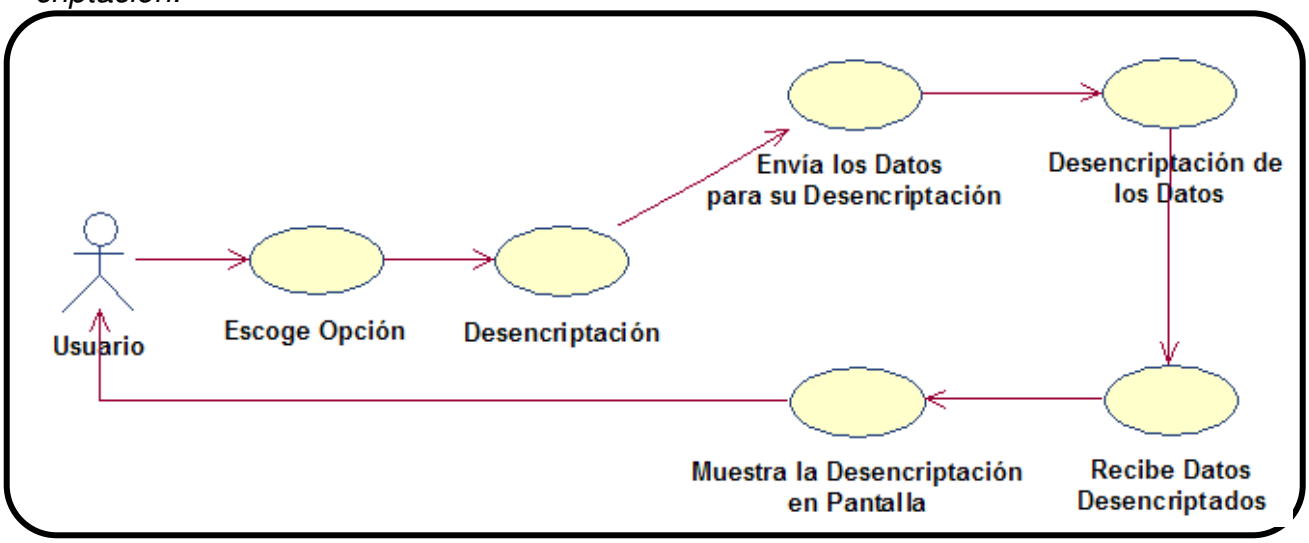


Figura 3.6.4 Caso de uso: Desencriptación

Secuencia normal de ejecución:

1. El texto encriptado se envía para su desencriptación.
2. El sistema realiza la desencriptación.
3. El sistema envía el texto desencriptado.
4. El texto desencriptado es mostrado en pantalla.

Excepciones:

1. En caso de haber un error al realizar la desencriptación el sistema notificará al usuario sobre el error.

3.6.5 Caso de uso: Guardar en Archivo .txt.

El siguiente caso de uso (*Figura 3.6.5*) muestra el comportamiento del proceso *Guardar en Archivo.* El cual solo puede ser aplicado después de la desencriptación y antes de la encriptación, ya que solo guarda el texto encriptado.

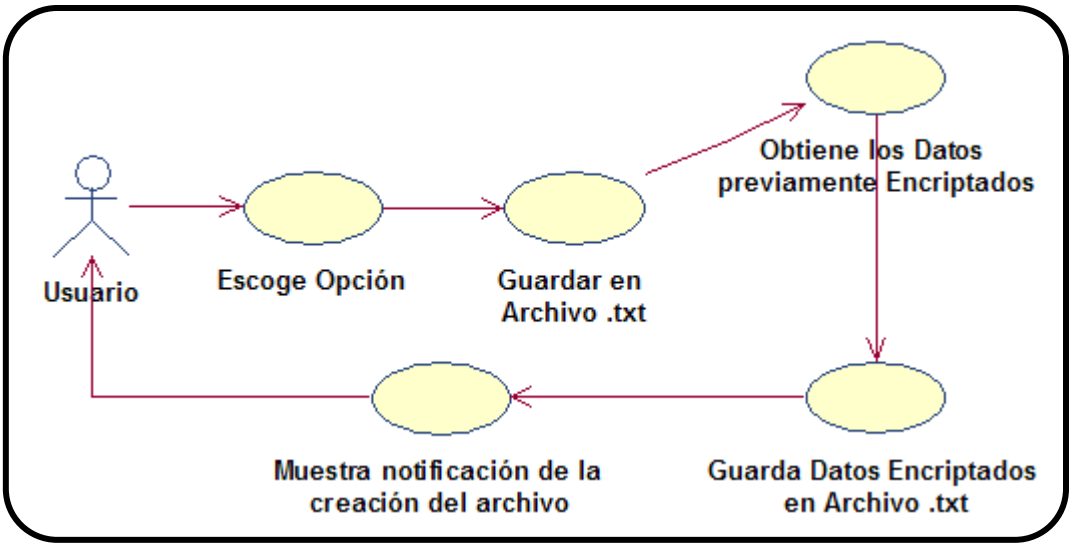


Figura 3.6.5 Caso de uso: Guardar en Archivo

Secuencia normal de ejecución:

1. Se obtiene el texto previamente encriptado.
2. El sistema guarda el texto encriptado en un archivo .txt.
3. El sistema muestra la notificación de que el archivo se creado sin errores.

Excepciones:

1. En caso de haber un erro al guardar el archivo el sistema mostrara una notificación.

3.7 Diagramas de secuencia.

A continuación se mostrarán los diagramas de secuencia utilizados para la realización de esta aplicación.

3.7.1 Diagrama de secuencia: Menú Principal.

El siguiente diagrama de secuencia (*Figura 3.7.1*) muestra el funcionamiento del caso de uso *Menú Principal (Figura 3.6.1)*, donde el usuario puede escoger entre las dos primeras opciones para poder realizar la encriptación, las cuales son: abrir archivo .txt o escribir texto claro, enseguida se realiza el proceso de encriptación y por último el usuario puede escoger entre guardar en archivo .txt con la encriptación realizada o realizar la desencriptación del texto. La opción de guardar en archivo .txt solo se puede realizar antes de la desencriptación y después de la encriptación.

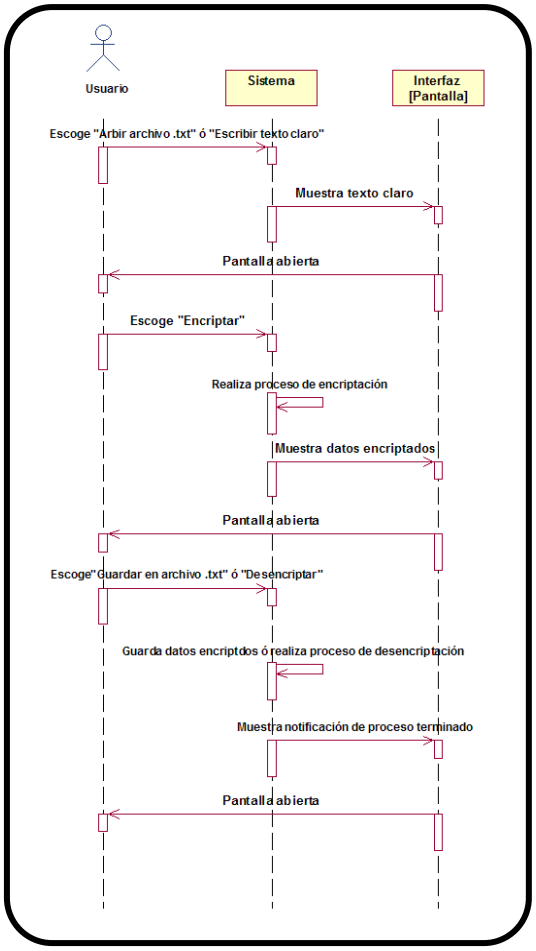


Figura 3.7.1 Diagrama de secuencia: Menú Principal

3.7.2 Diagrama de secuencia: Abrir Archivo .txt.

El siguiente diagrama de secuencia (*Figura 3.7.2*) muestra el funcionamiento del caso de uso *Abrir Archivo .txt (Figura 3.6.2)*, en la cual el usuario abre un archivo .txt escrito y guardado previamente en la máquina, en seguida el sistema carga la información contenida dentro del archivo, la cual debe de ser el texto claro y por último el texto es mostrado en pantalla.

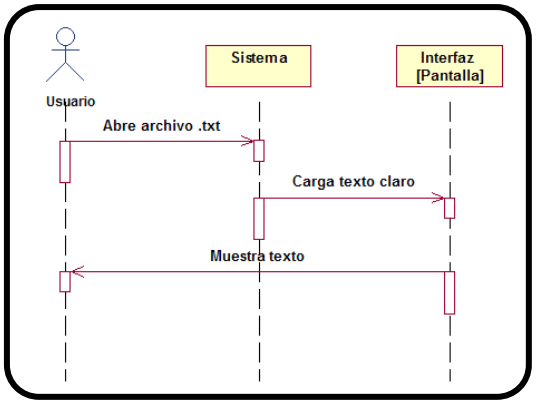


Figura 3.7.2 Diagrama de secuencia: Abrir Archivo .txt

3.7.3 Diagrama de secuencia: Encriptación.

El siguiente diagrama de secuencia (*Figura 3.7.3*) muestra el funcionamiento del caso de uso *Encriptación (Figura 3.6.3)*, en la cual después de que el usuario selecciona esta opción, el texto claro es enviado a Java para que este pueda realizar la encriptación, al final se recibe el texto encriptado y es mostrado en pantalla.

3.7.4 Diagrama de secuencia: Desencriptación.

El siguiente diagrama de secuencia (*Figura 3.7.4*) muestra el funcionamiento del caso de uso *Desencriptación (Figura 3.6.4)*, en la cual después de que el usuario selecciona esta opción, el texto encriptado es enviado a Java para que este pueda realizar la des-encriptación, al final se recibe el texto desencriptado y es mostrado en pantalla.

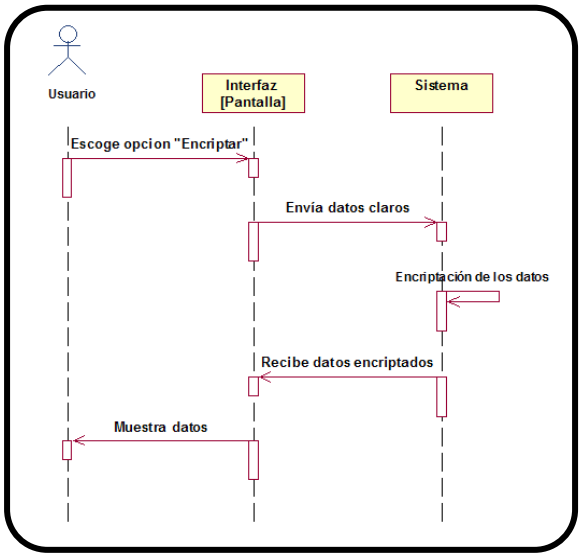


Figura 3.11 Diagrama de secuencia: Encriptación

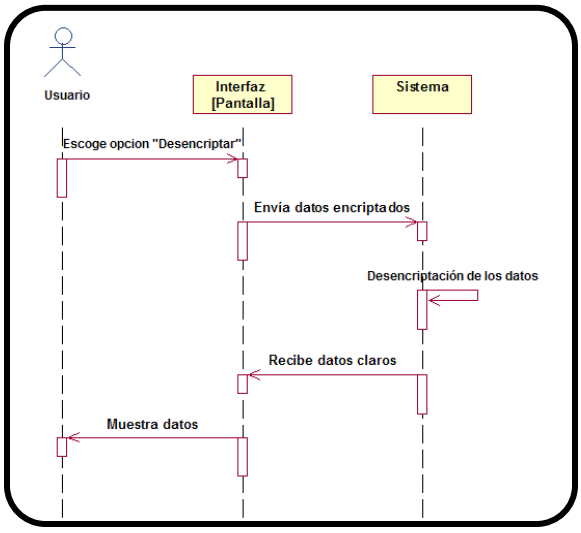


Figura 3.7.4. Diagrama de secuencia: Desencriptación

3.7.5 Diagrama de secuencia: Guardar en Archivo .txt.

El siguiente diagrama de secuencia (*Figura 3.7.5*) muestra el funcionamiento del caso de uso *Guardar en Archivo .txt (Figura 3.6.5)*, en la cual después de que el usuario selecciona esta opción, el sistema lee los datos previamente encriptados, enseguida los guarda en un archivo .txt y por último se muestra una notificación de que el archivo fue creado sin errores.

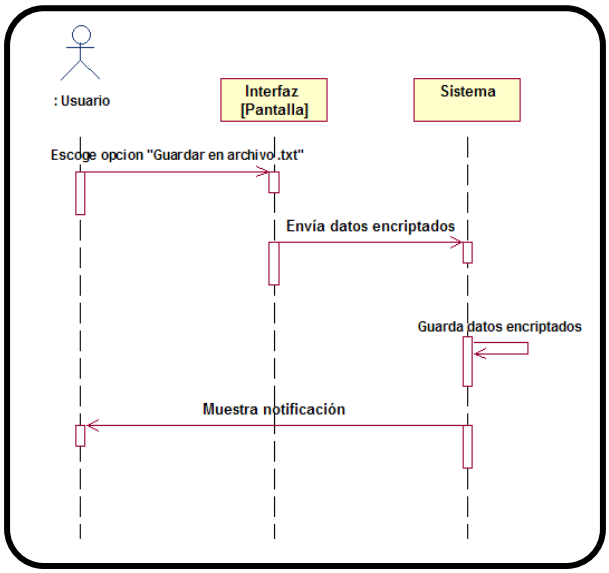


Figura 3.7.5 Diagrama de secuencia: Guardar en archivo .txt

CAPITULO 4

DISEÑO

CAPÍTULO 4

DISEÑO

4.1 Diagrama de Arquitectura.

La *Figura 4.1.1* muestra la jerarquía de cada caso de uso del capítulo anterior. La raíz del árbol será *Encriptación de un texto en español utilizando la forma silábica,* de ahí se despliegan los demás casos de usos.

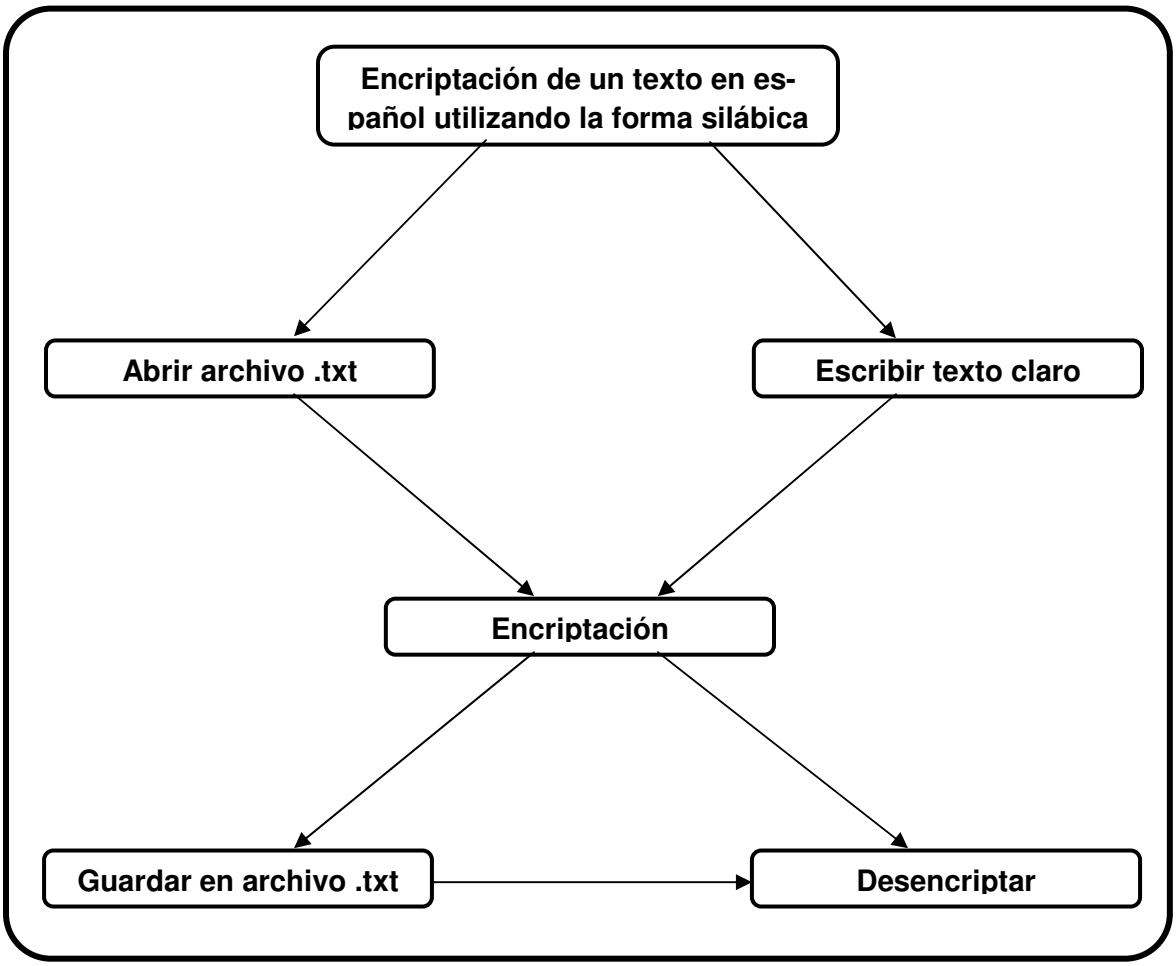


Figura 4.1.1 Diagrama de arquitectura

4.2 Diseño de la Interfaz.

La interfaz será sencilla y fácil de utilizar, contará con una sola pantalla donde se mostrará un menú de botones de selección, el primer botón abrirá archivos .txt, el segundo encriptará la información, el tercero desencriptará la información y finalmente el cuarto guardará el texto encriptado en un archivo .txt. La *Figura 4.2.1.* muestra el bosquejo de la interfaz.

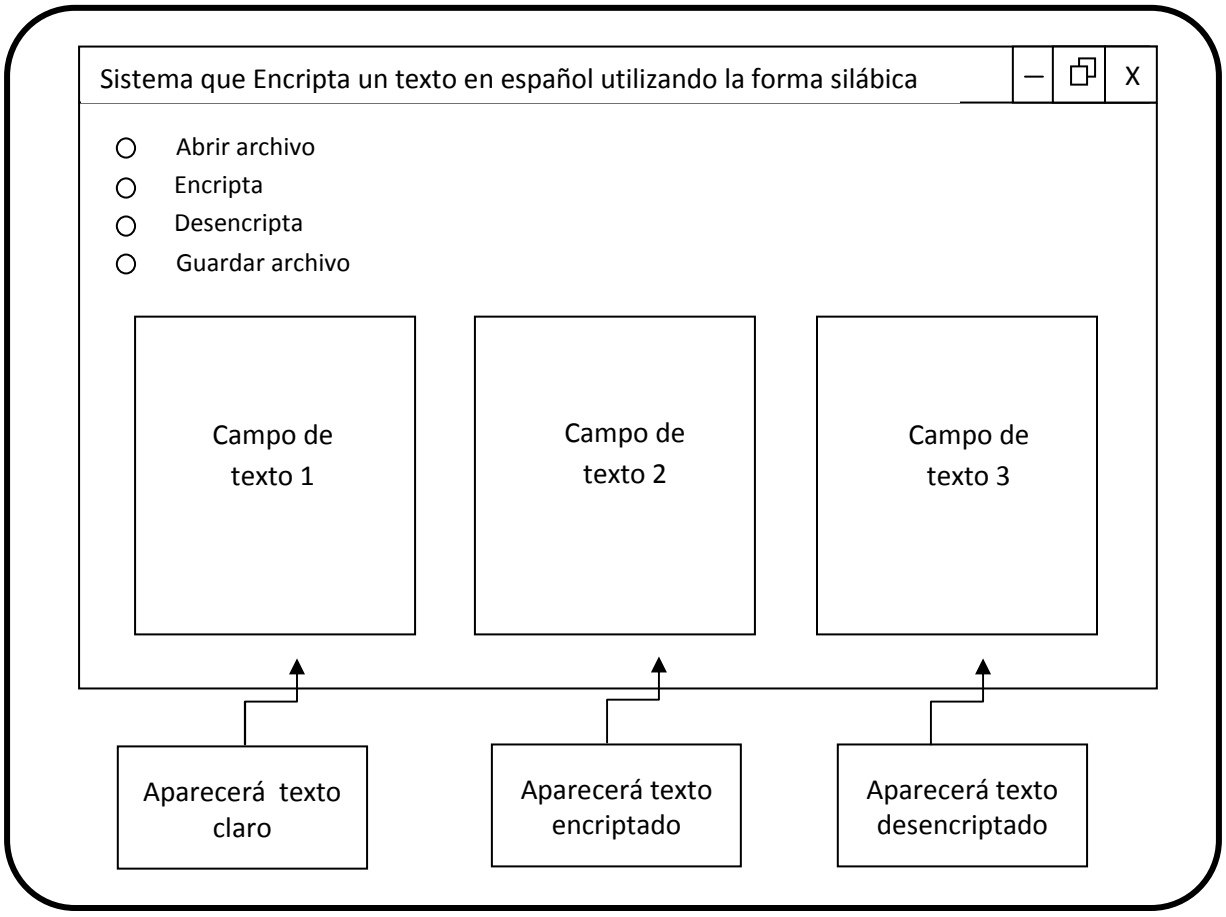


Figura 4.2.1 Diseño de la interfaz

4.3 Estructura de Datos.

A continuación se mostrarán y explicará la identificación de clases y el diagrama de clases para la realización de este proyecto.

4.3.1 Diagrama de identificación de clases.

En la *Figura 4.3.1* se muestran las clases con las que contará este proyecto.

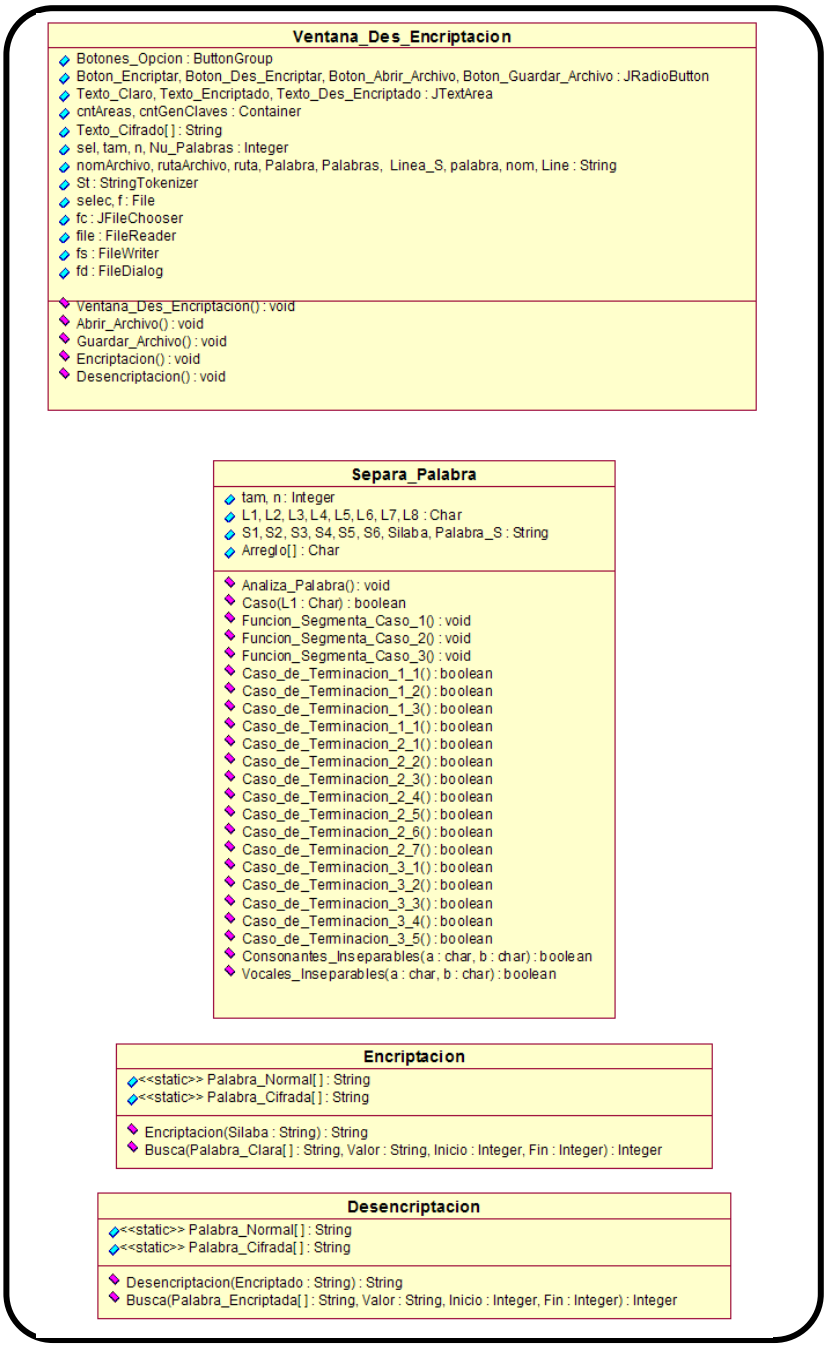


Figura 4.3.1 Identificación de clases

A continuación se explicará la función que realiza cada clase:

Clase Ventana_Des_Encriptacion.- Esta clase es donde se implementará la interfaz de toda la aplicación que encripta un texto en español, así como la ejecución para poder cargar un texto .txt o para poder guardar en un texto .txt la encriptación realizada.

Clase Separa_Palabras.- Esta clase será la encargada de separar todas las palabras en silabas para así poder encriptarlas, la separación en silabas se realizara de acuerdo a las reglas descritas en el *Capitulo 2: Segmentación de las palabras en silabas.*

Clase Encriptación.- esta clase se encarga de encriptar el texto claro; una vez que la palabra ya está en sílabas, encripta cada sílaba, donde se sustituye la sílaba en texto claro por una en texto cifrado.

Clase Desencriptación.- Esta clase realiza el proceso inverso de la encriptación y se realiza para poder comprobar que la encriptación del texto fue correcta.

4.3.2 Diagrama de clases.

El diagrama de clases que tendrá el proyecto se muestra en la *Figura 4.3.2.*

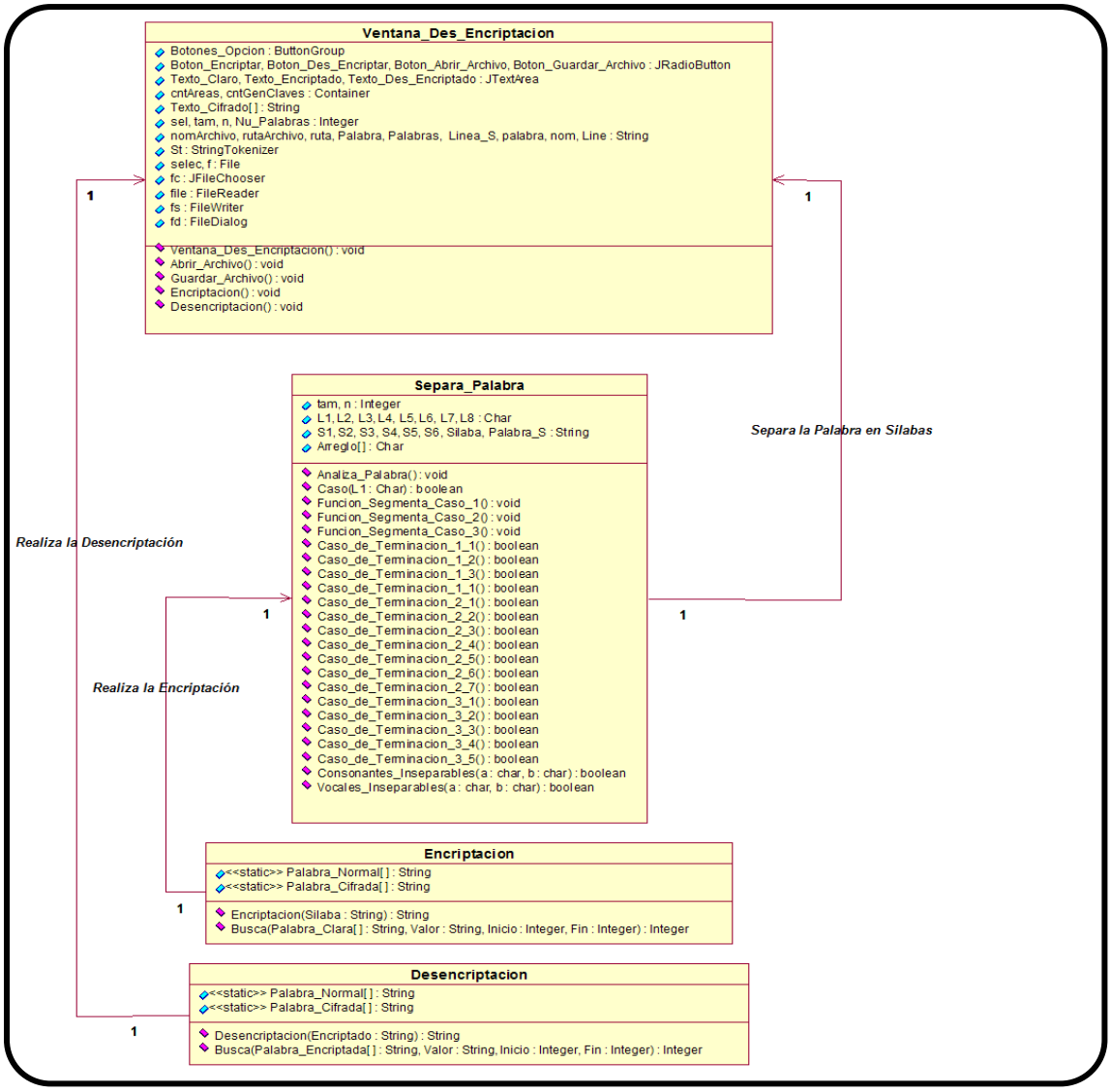


Figura 4.3.2 Diagrama de clases

4.3.3 Funciones

Como se logra ver en el diagrama de clases (*Figura 4.3.2*) mostrada en la sección anterior, esta aplicación utiliza varias funciones para realizar el proceso de encriptación y desencriptación del texto, a continuación se describen:

- **Clase Ventana_Des_Encriptacion.**
  - *Ventana_Des_Encriptacion ()*.- se crean y se inicializan todos los componentes del menú y ventana.
  - *Abrir_Archivo ()*.- sirve para seleccionar y abrir el archivo .txt que contenga el texto claro para poder encriptarlo.
  - *Guardar_Archivo ()*.- función que se encarga de guardar el texto encriptado en un archivo .txt.
- **Clase Separa_Palabra.**
  - *Analiza_Palabra ()*.- analiza el tamaño de cada palabra por medio de arreglos, esto se realiza para saber a que caso corresponde de acuerdo al *Capitulo 2: Segmentación de las palabras en silabas*.
  - *Caso (L1)*.- se determina el caso correspondiente de cada palabra, en donde L1 es la i-ésima letra de la sílaba o palabra.
  - *Funcion_Segmenta_Caso1 (),Funcion_Segmenta_Caso2 () y Funcion_Segmenta_Caso3 ()*.- son funciones que determinan la sílaba del caso 1, 2 o 3 según corresponda, esto se realiza tomando en cuenta su caso de terminación.
  - *Caso_de_Terminación_1_1(), Caso_de_Terminación_1_2(), Caso_de_Terminación_1_3(), Caso_de_Terminación_1_4(), Caso_de_Terminación_2_1(), Caso_de_Terminación_2_2(), Caso_de_Terminación_2_3(), Caso_de_Terminación_2_4(), Caso_de_Terminación_2_5(), Caso_de_Terminación_2_6(), Caso_de_Terminación_2_7(), Caso_de_Terminación_3_1 (), Caso_de_Terminación_3_2(), Caso_de_Terminación_3_3(), Caso_de_Terminación_3_4() y Caso_de_Terminación_3_5()*.- son los posibles casos de terminación que puede tener la sílaba que corresponde al caso 1, 2 o 3 del *Capitulo 2: Segmentación de las palabras en silabas*.

- *Consonantes Inseparables ()*.- contiene las consonantes que no se pueden separar.
- *Vocales Inseparables ()*.- contiene las vocales que no se pueden separar.

- **Clase Encriptacion.**
  - *Encriptacion (String Silaba)*.- se encarga de sustituir la sílaba en texto claro por una cadena en texto cifrado.
  - *Busca (String Palabra Encriptada[], String Valor, Integer Inicio, Integer Fin)*.- busca una cadena dentro de un arreglo de cadena.
- **Clase Desencriptacion.**
  - *Desencriptacion (String Encriptado)*.- se encarga de sustituir la sílaba en texto cifrado por una cadena en texto claro.

4.4 Algoritmos

Para separar todas y cada una de las palabras en sílabas, es necesario analizar la palabra, es decir, determinar por medio de algoritmos su estructura, para así saber a que caso pertenecen de acuerdo a la *Sección 2.3* del *Capitulo 2: Segmentación de las palabras en silabas* y así agrupar cada sílaba conforme a la *Sección 2.4* del *Capitulo 2: Segmentación de las palabras en silabas*, también saber si están acentuadas o no. Para realizar correctamente la separación de cada palabra es necesario tomar en cuenta las reglas ortográficas de la *Sección 2.2* del *Capítulo 2: Segmentación de las palabras en silabas*. El diagrama de flujo mostrado en la figura 4.4.1 describe el algoritmo utilizado para la separación de las sílabas de una palabra.

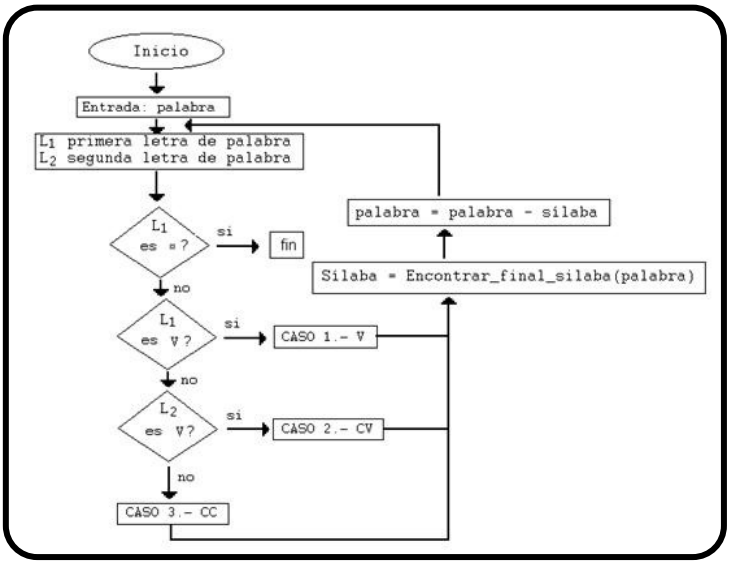


Figura 4.4.1 Algoritmo que determina a que caso pertenece la sílaba

La descripción detallada de este algoritmo se muestra en los siguientes tres pasos:

- *Paso 1*.- Para conocer las sílabas de una palabra, se empieza por tomar las dos primeras letras (L1 y L2 respectivamente) de esa palabra.
- *Paso 2*.- Si la primera letra (L1) es vocal, la sílaba pertenece al caso 1.
- *Paso 3*.- Si la segunda letra (L2) es vocal, es una sílaba del caso 2. De lo contrario la sílaba pertenece al caso 3.

En la *Tabla 4.4.1* muestra la notación utilizada en esta sección.

Símbolo	Descripción
¤	Fin de palabra o carácter diferente a vocal o consonante (‘,’, ‘-’, ‘1’, etc.)
[CV]	Las siguientes dos letras son CV
[CVV]	Las siguientes dos letras pertenecen al grupo de consonantes inseparables
V*	Vocal acentuada
|	Utilizada para establecer posibilidades alternativas
Li	I-ésima letra de la sílaba o palabra
P	Palabra que se va a dividir en sílabas

Tabla 4.4.1 Notación utilizada

4.4.1 Algoritmo para el caso de inicio de sílaba: Vocal.

A continuación se muestra el algoritmo utilizado para definir si la palabra pertenece al caso 1.

Función segmenta_caso_1 (entrada: Palabra, salida: sílaba)
if ($C_{1,1}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{1,1}$
la sílaba es *V*
ir a FIN
else if (L_2 es C? **)**
if($C_{1,2}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{1,2}$
la sílaba es *VC*
ir a FIN
else
ERROR // palabra incorrecta

else if (L_2 es V? **)**
if ($C_{1,3}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{1,3}$
la sílaba es: *VV*
ir a FIN
else if (L_3 es C? **)**
if ($C_{1,4}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{1,4}$
la sílaba es: *VVC*
ir a FIN
else
ERROR // palabra incorrecta
FIN : termina esta función

4.4.2 Algoritmo para el caso de inicio de sílaba: Consonante.

A continuación se muestra el algoritmo utilizado para definir si la palabra pertenece al caso 1. De acuerdo a la regla 1, *Sección 2.2* del *Capitulo 2: Segmentación de las palabras en silabas* se sabe que la siguiente letra es vocal.

Función segmenta_caso_2 (entrada: Palabra salida: sílaba)
If ($C_{2,1}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{2,1}$
la sílaba es *CV*
ir a FIN
else if (L_3 es C ? **)**
if ($C_{2,2}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{2,2}$
la sílaba es: *CVC*
ir a FIN
else if (L_4 es C? **)**
if ($C_{2,3}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{2,3}$
la sílaba es *CVCC*
ir a FIN
else if ($C_{2,4}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{2,4}$
la sílaba es: *CVV*
ir a FIN
else if(L_4 es C ?**)**
if ($C_{2,5}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{2,5}$
la sílaba es: *CVVC*
ir a FIN
else if ($C_{2,6}$ **)** //implementar los casos de terminación de sílaba de la referencia $C_{2,6}$
la sílaba es: *CVVV*
ir a FIN
else if (L_5 es C?**)**
if ($C_{2,7}$ **)** // implementar los casos de terminación de sílaba
// de la referencia $C_{2,7}$
la sílaba es: *CVVVC*
ir a FIN
else
ERROR // palabra incorrecta
FIN : termina esta función

4.4.3 Algoritmo para el caso de inicio de sílaba: Consonante + Consonante.

El siguiente algoritmo es utilizado para definir si la palabra pertenece al caso 3.

Función segmenta_caso_3 (entrada: Palabra, salida: sílaba)

If ($C_{3,1}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{3,1}$
la sílaba es: *CCV*
ir a FIN.
else if (L_4 es C? **)**
if ($C_{3,2}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{3,2}$
la sílaba es: *CCVC*
ir a FIN
else if ($C_{3,3}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{3,3}$
la sílaba es: *CCVCC*
ir a FIN
else if ($C_{3,4}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{3,4}$
la sílaba es: *CCVV*
ir a FIN
else if ($C_{3,5}$ **)** // implementar los casos de terminación de sílaba de la referencia $C_{3,5}$
la sílaba es: *CCVVC*
ir a FIN
else
ERROR // palabra incorrecta
FIN : termina esta función

Si la sílaba pertenece al caso 1 entonces se emplea el *Algoritmo para el caso de inicio de sílaba: Vocal*, donde la *Tabla 4.4.2* muestra los posibles casos para determinar cuando termina una sílaba e inicia otra para este caso.

Referencia	Sílaba	Posibles casos de terminación de la sílaba
$C_{1,1}$	V	¤ | [CV] | [CCV] | V*
$C_{1,2}$	VC	¤ | [CV] | [CCV]
$C_{1,3}$	VV	¤ | [CV] | [CCV]
$C_{1,4}$	VVC	¤ | [CV] | [CCV]

Tabla 4.4.2 Posibles casos que determinan cuando termina una sílaba, Caso 1

Si la sílaba pertenece al caso 2 entonces se emplea el *Algoritmo para el caso de inicio de sílaba: Consonante*, donde la *Tabla 4.4.3* muestra los posibles casos para determinar cuando termina una sílaba e inicia otra para este caso.

Se sabe que la primera letra es consonante, la segunda tiene que ser vocal de acuerdo a la regla 1, *Sección 2.2* del *Capitulo 2: Segmentación de las palabras en silabas*.

Referencia	Sílaba	Posibles casos de terminación de la sílaba
$C_{2,1}$	CV	¤ | [CV] | [CCV]
$C_{2,2}$	CVC	¤ | [CV] | [CCV]
$C_{2,3}$	CVCC	¤ | [CV]
$C_{2,4}$	CVV	¤ | [CV] | [CCV]
$C_{2,5}$	CVVC	¤ | [CV] | [CCV]
$C_{2,6}$	CVVV	¤ | [CV] | [CCV]
$C_{2,7}$	CVVVC	¤ | [CV] | [CCV]

Tabla 4.4.3 Posibles casos que determinan cuando termina una sílaba, Caso 2

Si la sílaba pertenece al caso 3 entonces se emplea el *Algoritmo para el caso inicio de sílaba: Consonante + Consonante*, donde la *Tabla 4.4.4* muestra los posibles casos para determinar cuando termina una sílaba e inicia otra para este caso.

Referencia	Sílaba	Posibles casos de terminación de la sílaba
$C_{3,1}$	CCV	¤ | [CV] | [CCV]
$C_{3,2}$	CCVC	¤ | [CV] | [CCV]
$C_{3,3}$	CCVCC	¤ | [CV]
$C_{3,4}$	CCVV	¤ | [CV] | [CCV]
$C_{3,5}$	CCVVC	¤ | [CV] | [CCV]

Tabla 4.4.4 Posibles casos que determinan cuando termina una sílaba, Caso 3

Una vez que se sabe a que caso pertenece cada sílaba y es separada, ahora se necesita el algoritmo que llevará a cabo la encriptación, el cual se explica a continuación.

4.4.4 Algoritmo para Encriptar las sílabas.

Se está desarrollando un programa que encripta un texto en español utilizando la forma silábica, para generar las llaves definimos dos tipos de arreglos ABC [ ] y NUM [ ], los cuales contienen el abecedario y los números del 0 al 9 respectivamente.

- ABC [ ] = a,b,c,… z donde $0 \leq m \leq 27$
- NUM [ ] = 0,1,… 9 donde $0 \leq n \leq 9$

Se realiza la unión de ambos arreglos:

Para las sílabas de 1 letra que son las vocales, el valor que le corresponde a cada vocal es el de una vocal diferente.

a = y , e = u , i = o , o = i , u = e , y = a

Para las sílabas de 2 letras, donde i = 2

$$\bigcup_{j=1}^{27} \bigcup_{k=0}^{9} \text{ABC [j] NUM [k]} \quad i = 2$$

Se genera:

a0, b0, c0, d0,…, z0

a1, b1, c1, d1,…, z1

a2, b2, c2, d2,…, z2

a3, b3, c3, d3,…, z3

a4, b4, c4, d4,…, z4

a5, b5, c5, d5,…, i5.

Termina en i5 ya que no hay más sílabas con 2 letras.

Para las sílabas de 3 letras, donde i = 3

$$\bigcup_{k=1}^{27} \bigcup_{j=1}^{27} \text{ABC [k] ABC [j] NUM[i]} \quad i = 3$$

Se genera:

aa3, ab3, ac3, ad3,…, az3

ba3, bb3, bc3, bd3,…, bz3

ca3, cb3, cc3, cd3,…, cz3

…

xa3, xb3, xc3, xd3,…, xw3.

Termina en xw3 ya que no hay más sílabas de 3 letras.

Para las sílabas de 4 letras, donde i = 4

$$\bigcup_{k=1}^{27} \bigcup_{l,j=1}^{27} \text{ABC [l] ABC [k] ABC [j] NUM[i]} \quad i = 4$$

Se genera:

aaa4, bab4, cac4, dad4,…, zaz4

aba4, bbb4, cbc4, dbd4,…, zbz4

aca4, bcb4, ccc4, dcd4,…, zcz4

ada4, bdb4, cdc4, ddd4,…, zdz4

…

ana4, bnb4, cnc4, dnd4,…, pnp4.

Termina en pnp4 porque no hay más sílabas con 4 letras.

Para las sílabas de 5 letras, donde i = 5

$$\bigcup_{\substack{k=1 \\ a}}^{27} \bigcup_{l,j=1}^{27} \text{ABC [a] ABC [l] ABC [k] ABC [j] NUM[i]} \quad i = 5$$

Se generan sólo estas sílabas ya que existen muy pocas sílabas con 5 letras:

aaaa5, aaab5

baac5, baad5

cbae5, cbaf5

dbag5, dbah5

ecbi5, ecbj5

fcbk5, fcbl5

gdbm5, gdbn5

hdbo5, hdbp5

iecq5, iecr5

jecs5, ject5

kfcu5, kfcv5

lfcw5, lfcw5

mgdy5, mgdz5

ngda5, ngdb5

Una vez que ya se tiene las llaves, ahora se van a crear 2 arreglos:

- *Normal [ ].*- Contendrá todas las sílabas del español.
- *Cifrado [ ].*- Contendrá las sílabas ya cifradas.

El algoritmo que encripta las sílabas se implementará para que sustituya la silaba en texto claro por una en texto cifrado.

CAPITULO 5

IMPLEMENTACIÓN Y PRUEBAS

CAPÍTULO 5

IMPLEMENTACIÓN. Y PRUEBAS.

5.1 Implementación.

Para realizar la encriptación de un texto en español utilizando la forma silábica, se implementaron los algoritmos del capítulo anterior. En esta sección se explicarán únicamente los módulos más relevantes.

5.1.1 Clase Encripta.

La clase Encripta utiliza dos arreglos, el primero guarda las sílabas antes de ser encriptadas y el segundo guarda las sílabas ya encriptadas, además de que su constructor sustituye la sílaba en texto claro por una cadena en texto cifrado y finalmente el método llamado Busca() realiza la búsqueda de una cadena dentro de un arreglo de cadenas. La *Figura 5.1.1* muestra dicha implementación en Java.

Encripta.java * | Main_P.java | Ventana.java

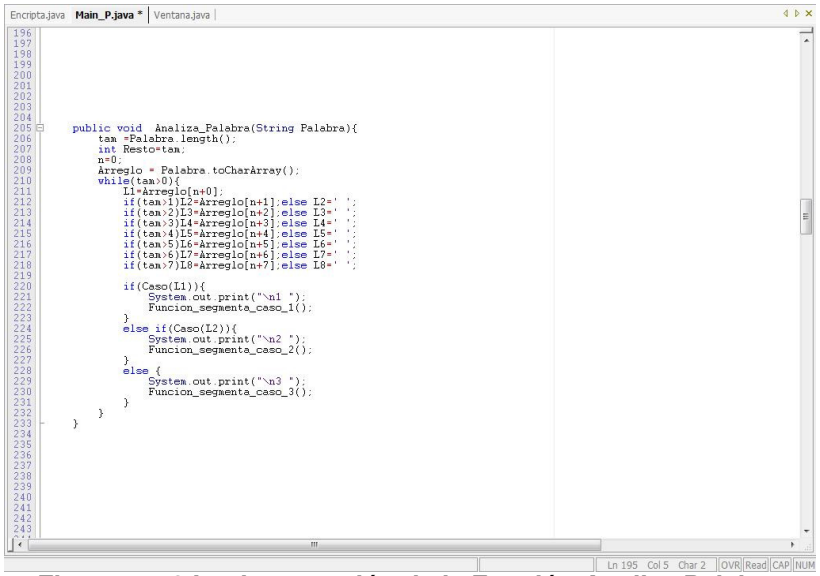
```
import java.io.*;
import java.lang.*;

public class Encripta {


   static String  normal[] = {"a","ab","abs","ac","ad","ah","ai","ais","al","am","an","ar","as","at","au","aun","aus","ay
   static String cifrado[] = {"y","a0","aa3","b0","c0","d0","e0","ab3","f0","g0","h0","i0","j0","k0","l0","ac3","ad3","m0

   /*Sustituye la silaba en texto claro por una cadena en texto cifrado*/
   public static String Encripta(String silaba) {

     int i = Busca(normal,silaba,0,normal.length-1);
     return (i<0) ? silaba : cifrado[i];
   }


   /*Este método busca una cadena dentro de un arreglo de cadenas*/
   public static int Busca(String[] V, String valor, int inicio, int fin) {

       if(fin < inicio)
          return -1;

       int medio = (inicio + fin) / 2;

       if(valor.compareToIgnoreCase(V[medio]) < 0)
          return Busca(V,valor,inicio,medio-1);
       else
          if(valor.compareToIgnoreCase(V[medio]) > 0)
             return Busca(V,valor,medio+1,fin);
          else
             return medio;
   }
}
```

Ln 53 Col 1 Char 1 | OVR | Read | CAP | NUM

Figura 5.1.1 Implementación de la Clase Encripta

5.1.2 Función Analiza Palabra.

Esta función analiza el tamaño de la palabra y determina a que caso pertenece para segmentarla posteriormente. La *Figura 5.1.2* muestra la implementación de la función.

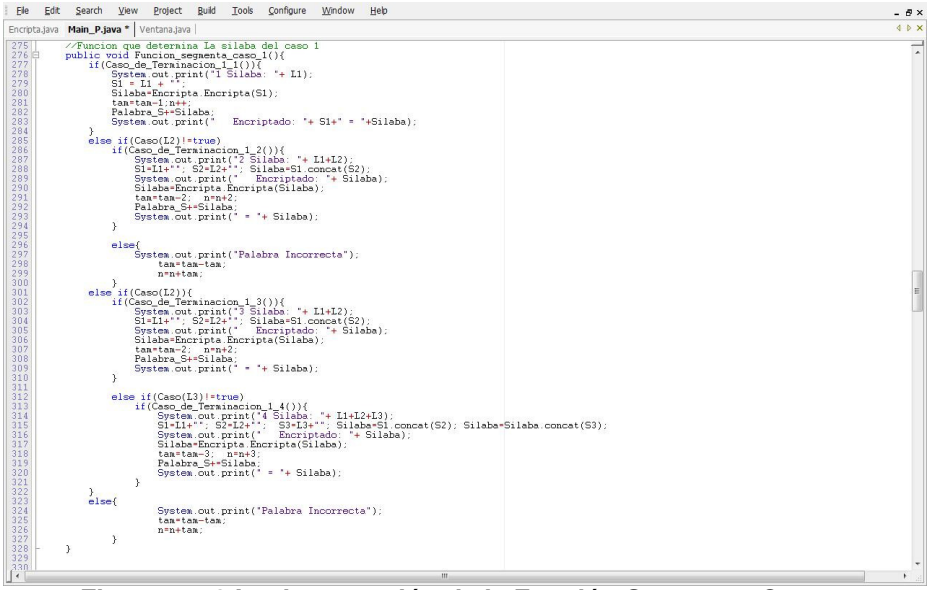


```
196
197
198
199
200
201
202
203
204
205     public void  Analiza_Palabra(String Palabra){
206         tam =Palabra.length();
207         int Resto=tam;
208         n=0;
209         Arreglo = Palabra.toCharArray();
210         while(tam>0){
211             L1=Arreglo[n+0];
212             if(tam>1)L2=Arreglo[n+1];else L2=' ';
213             if(tam>2)L3=Arreglo[n+2];else L3=' ';
214             if(tam>3)L4=Arreglo[n+3];else L4=' ';
215             if(tam>4)L5=Arreglo[n+4];else L5=' ';
216             if(tam>5)L6=Arreglo[n+5];else L6=' ';
217             if(tam>6)L7=Arreglo[n+6];else L7=' ';
218             if(tam>7)L8=Arreglo[n+7];else L8=' ';
219
220             if(Caso(L1)){
221                 System.out.print("\n1 ");
222                 Funcion_segmenta_caso_1();
223             }
224             else if(Caso(L2)){
225                 System.out.print("\n2 ");
226                 Funcion_segmenta_caso_2();
227             }
228             else {
229                 System.out.print("\n3 ");
230                 Funcion_segmenta_caso_3();
231             }
232         }
233     }
234
235
236
237
238
239
240
241
242
243
```

Figura 5.1.2 Implementación de la Función Analiza Palabra

5.1.3 Función Segmenta Caso 1.

Se implementó la función Segmenta Caso 1 de acuerdo a la sección 4.5, esto es para segmentar la palabra en sílabas con *Inicio de sílaba: vocal*, la *Figura 5.1.3* muestra dicha implementación, donde se incluyeron también el caso de terminación correspondiente al caso 1.

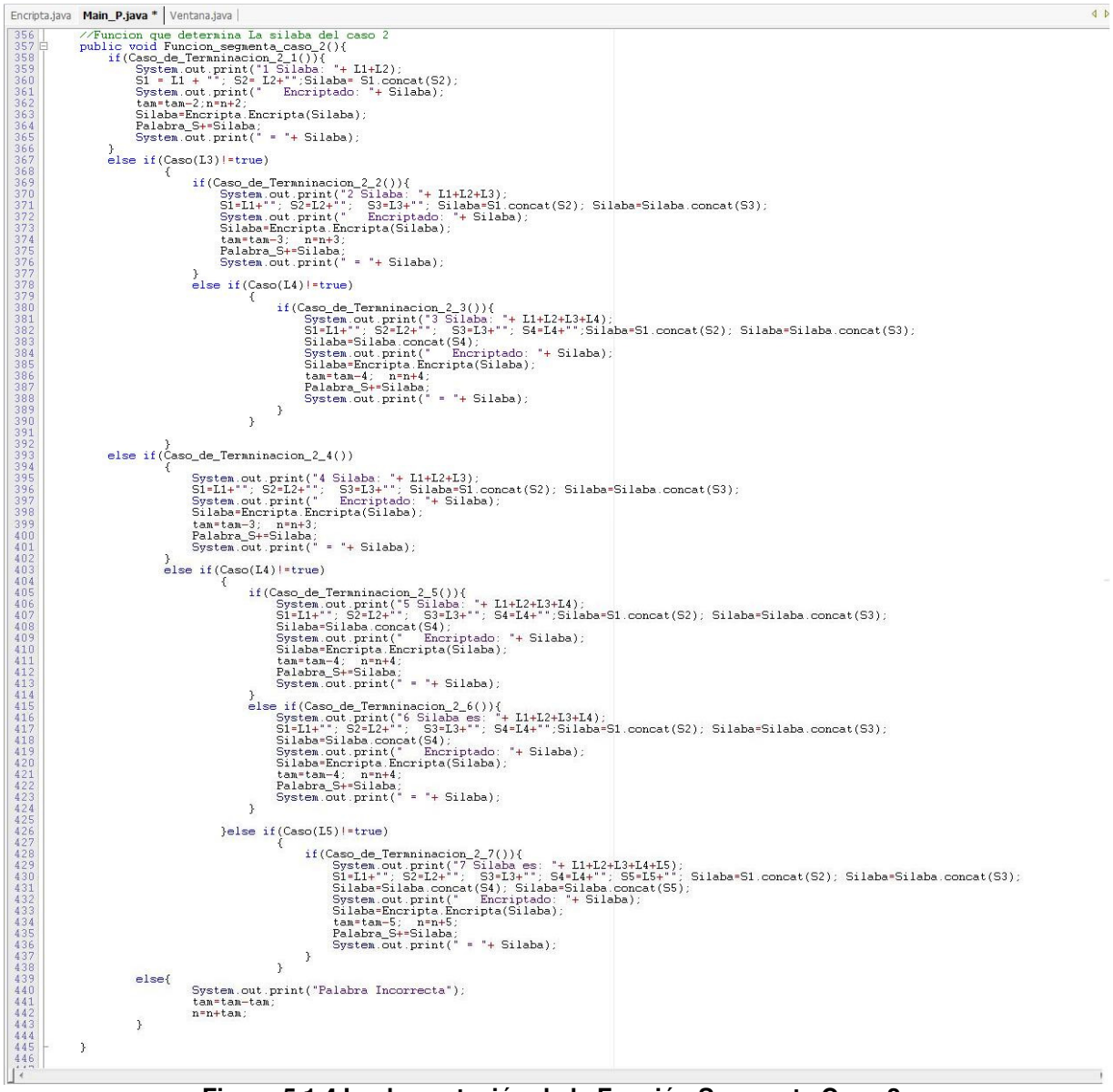


```
File  Edit  Search  View  Project  Build  Tools  Configure  Window  Help
Encripta.java  Main_P.java *  Ventana.java
275     //Funcion que determina La silaba del caso 1
276     public void Funcion_segmenta_caso_1(){
277         if(Caso_de_Terminacion_1_1()){
278             System.out.print("1 Silaba: "+ L1);
279             S1 = L1 + "";
280             Silaba=Encripta.Encripta(S1);
281             tam=tam-1; n++;
282             Palabra_S+=Silaba;
283             System.out.print("   Encriptado: "+ S1+" = "+Silaba);
284         }
285         else if(Caso(L2)!=true)
286             if(Caso_de_Terminacion_1_2()){
287                 System.out.print("2 Silaba: "+ L1+L2);
288                 S1=L1+"";  S2=L2+"";  Silaba=S1.concat(S2);
289                 System.out.print("   Encriptado: "+ Silaba);
290                 Silaba=Encripta.Encripta(Silaba);
291                 tam=tam-2;  n=n+2;
292                 Palabra_S+=Silaba;
293                 System.out.print(" = "+ Silaba);
294             }
295
296             else{
297                 System.out.print("Palabra Incorrecta");
298                     tam=tam-tam;
299                     n=n+tam;
300             }
301         else if(Caso(L2)){
302             if(Caso_de_Terminacion_1_3()){
303                 System.out.print("3 Silaba: "+ L1+L2);
304                 S1=L1+"";  S2=L2+"";  Silaba=S1.concat(S2);
305                 System.out.print("   Encriptado: "+ Silaba);
306                 Silaba=Encripta.Encripta(Silaba);
307                 tam=tam-2;  n=n+2;
308                 Palabra_S+=Silaba;
309                 System.out.print(" = "+ Silaba);
310             }
311
312             else if(Caso(L3)!=true)
313                 if(Caso_de_Terminacion_1_4()){
314                     System.out.print("4 Silaba: "+ L1+L2+L3);
315                     S1=L1+"";  S2=L2+"";  S3=L3+"";  Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
316                     System.out.print("   Encriptado: "+ Silaba);
317                     Silaba=Encripta.Encripta(Silaba);
318                     tam=tam-3;  n=n+3;
319                     Palabra_S+=Silaba;
320                     System.out.print(" = "+ Silaba);
321                 }
322         }
323         else{
324                 System.out.print("Palabra Incorrecta");
325                 tam=tam-tam;
326                 n=n+tam;
327             }
328     }
329
330
```

Figura 5.1.3 Implementación de la Función Segmenta Caso 1

5.1.4 Función Segmenta Caso 2.

Se implementó la función segmenta caso 2 de acuerdo a la sección 4.5, esto es para segmentar la palabra en sílabas con *inicio de sílaba: consonante*, la *Figura 5.1.4* muestra dicha implementación, donde se incluyeron también el caso de terminación correspondiente al caso 2.

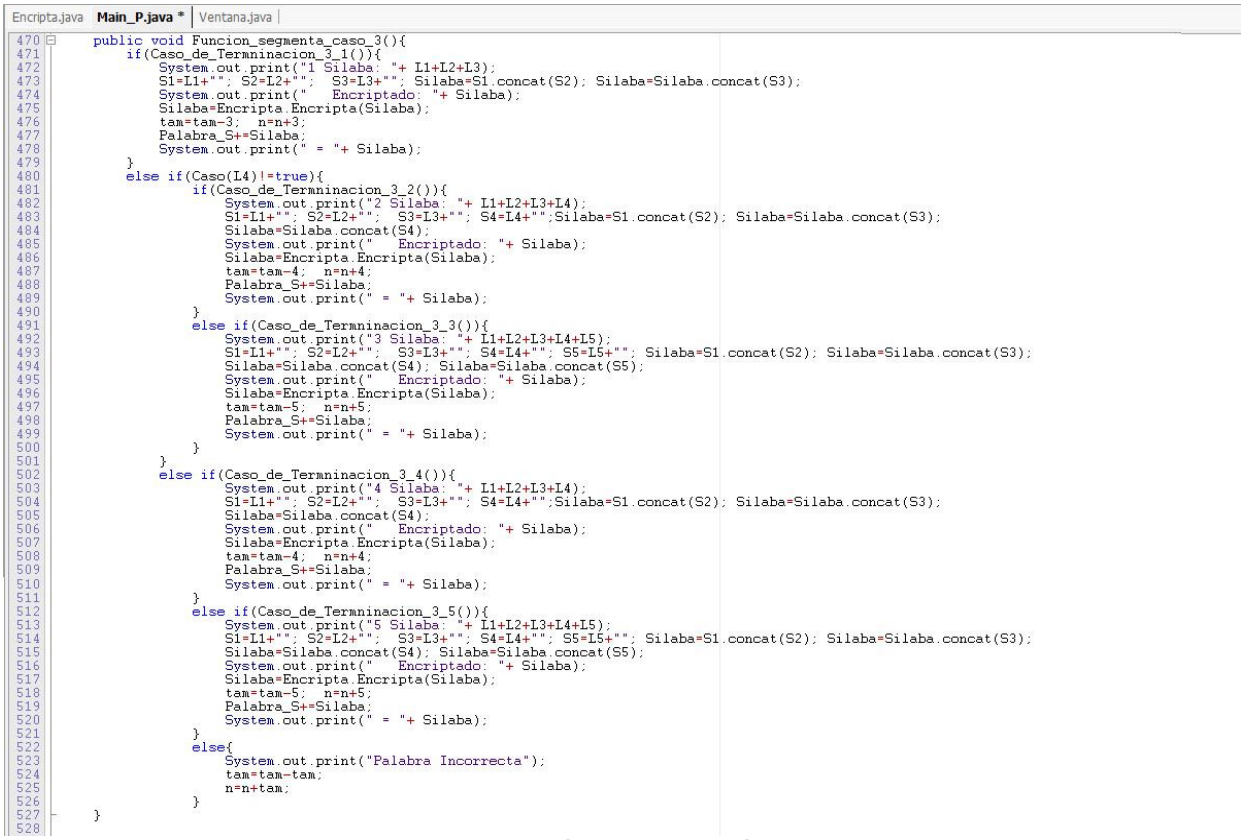


```
Encripta.java | Main_P.java * | Ventana.java
356     //Funcion que determina La silaba del caso 2
357     public void Funcion_segmenta_caso_2(){
358         if(Caso_de_Termninacion_2_1()){
359             System.out.print("1 Silaba: "+ L1+L2);
360             S1 = L1 + ""; S2= L2+"";Silaba= S1.concat(S2);
361             System.out.print("   Encriptado: "+ Silaba);
362             tam=tam-2;n=n+2;
363             Silaba=Encripta.Encripta(Silaba);
364             Palabra_S+=Silaba;
365             System.out.print(" = "+ Silaba);
366         }
367         else if(Caso(L3)!=true)
368                 {
369                     if(Caso_de_Termninacion_2_2()){
370                         System.out.print("2 Silaba: "+ L1+L2+L3);
371                         S1=L1+""; S2=L2+"";  S3=L3+""; Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
372                         System.out.print("   Encriptado: "+ Silaba);
373                         Silaba=Encripta.Encripta(Silaba);
374                         tam=tam-3;   n=n+3;
375                         Palabra_S+=Silaba;
376                         System.out.print(" = "+ Silaba);
377                     }
378                     else if(Caso(L4)!=true)
379                             {
380                                 if(Caso_de_Termninacion_2_3()){
381                                     System.out.print("3 Silaba: "+ L1+L2+L3+L4);
382                                     S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+"";Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
383                                     Silaba=Silaba.concat(S4);
384                                     System.out.print("   Encriptado: "+ Silaba);
385                                     Silaba=Encripta.Encripta(Silaba);
386                                     tam=tam-4;   n=n+4;
387                                     Palabra_S+=Silaba;
388                                     System.out.print(" = "+ Silaba);
389                                 }
390                             }
391
392                 }
393         else if(Caso_de_Termninacion_2_4())
394                 {
395                     System.out.print("4 Silaba: "+ L1+L2+L3);
396                     S1=L1+""; S2=L2+"";  S3=L3+""; Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
397                     System.out.print("   Encriptado: "+ Silaba);
398                     Silaba=Encripta.Encripta(Silaba);
399                     tam=tam-3;   n=n+3;
400                     Palabra_S+=Silaba;
401                     System.out.print(" = "+ Silaba);
402                 }
403                 else if(Caso(L4)!=true)
404                         {
405                             if(Caso_de_Termninacion_2_5()){
406                                 System.out.print("5 Silaba: "+ L1+L2+L3+L4);
407                                 S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+"";Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
408                                 Silaba=Silaba.concat(S4);
409                                 System.out.print("   Encriptado: "+ Silaba);
410                                 Silaba=Encripta.Encripta(Silaba);
411                                 tam=tam-4;   n=n+4;
412                                 Palabra_S+=Silaba;
413                                 System.out.print(" = "+ Silaba);
414                             }
415                             else if(Caso_de_Termninacion_2_6()){
416                                 System.out.print("6 Silaba es: "+ L1+L2+L3+L4);
417                                 S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+"";Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
418                                 Silaba=Silaba.concat(S4);
419                                 System.out.print("   Encriptado: "+ Silaba);
420                                 Silaba=Encripta.Encripta(Silaba);
421                                 tam=tam-4;   n=n+4;
422                                 Palabra_S+=Silaba;
423                                 System.out.print(" = "+ Silaba);
424                             }
425
426                         }else if(Caso(L5)!=true)
427                                 {
428                                     if(Caso_de_Termninacion_2_7()){
429                                         System.out.print("7 Silaba es: "+ L1+L2+L3+L4+L5);
430                                         S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+""; S5=L5+""; Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
431                                         Silaba=Silaba.concat(S4); Silaba=Silaba.concat(S5);
432                                         System.out.print("   Encriptado: "+ Silaba);
433                                         Silaba=Encripta.Encripta(Silaba);
434                                         tam=tam-5;   n=n+5;
435                                         Palabra_S+=Silaba;
436                                         System.out.print(" = "+ Silaba);
437                                     }
438                                 }
439         else{
440                 System.out.print("Palabra Incorrecta");
441                 tam=tam-tam;
442                 n=n+tam;
443         }
444
445     }
446
```

Figura 5.1.4 Implementación de la Función Segmenta Caso 2.

5.1.5 Función Segmenta Caso 3.

La implementación para segmentar la palabra con *inicio de sílaba: Consonante + Consonante* correspondiente al caso 3, la *Figura 5.1.5* muestra dicha implementación, incluyendo su caso de terminación.



```
Encripta.java | Main_P.java * | Ventana.java

    public void Funcion_segmenta_caso_3(){
        if(Caso_de_Termninacion_3_1()){
            System.out.print("1 Silaba: "+ L1+L2+L3);
            S1=L1+""; S2=L2+"";  S3=L3+""; Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
            System.out.print("   Encriptado: "+ Silaba);
            Silaba=Encripta.Encripta(Silaba);
            tam=tam-3;  n=n+3;
            Palabra_S+=Silaba;
            System.out.print(" = "+ Silaba);
        }
        else if(Caso(L4)!=true){
                if(Caso_de_Termninacion_3_2()){
                    System.out.print("2 Silaba: "+ L1+L2+L3+L4);
                    S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+"";Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
                    Silaba=Silaba.concat(S4);
                    System.out.print("   Encriptado: "+ Silaba);
                    Silaba=Encripta.Encripta(Silaba);
                    tam=tam-4;  n=n+4;
                    Palabra_S+=Silaba;
                    System.out.print(" = "+ Silaba);
                }
                else if(Caso_de_Termninacion_3_3()){
                    System.out.print("3 Silaba: "+ L1+L2+L3+L4+L5);
                    S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+""; S5=L5+""; Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
                    Silaba=Silaba.concat(S4); Silaba=Silaba.concat(S5);
                    System.out.print("   Encriptado: "+ Silaba);
                    Silaba=Encripta.Encripta(Silaba);
                    tam=tam-5;  n=n+5;
                    Palabra_S+=Silaba;
                    System.out.print(" = "+ Silaba);
                }
            }
            else if(Caso_de_Termninacion_3_4()){
                    System.out.print("4 Silaba: "+ L1+L2+L3+L4);
                    S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+"";Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
                    Silaba=Silaba.concat(S4);
                    System.out.print("   Encriptado: "+ Silaba);
                    Silaba=Encripta.Encripta(Silaba);
                    tam=tam-4;  n=n+4;
                    Palabra_S+=Silaba;
                    System.out.print(" = "+ Silaba);
                }
                else if(Caso_de_Termninacion_3_5()){
                    System.out.print("5 Silaba: "+ L1+L2+L3+L4+L5);
                    S1=L1+""; S2=L2+"";  S3=L3+""; S4=L4+""; S5=L5+""; Silaba=S1.concat(S2); Silaba=Silaba.concat(S3);
                    Silaba=Silaba.concat(S4); Silaba=Silaba.concat(S5);
                    System.out.print("   Encriptado: "+ Silaba);
                    Silaba=Encripta.Encripta(Silaba);
                    tam=tam-5;  n=n+5;
                    Palabra_S+=Silaba;
                    System.out.print(" = "+ Silaba);
                }
                else{
                    System.out.print("Palabra Incorrecta");
                    tam=tam-tam;
                    n=n+tam;
                }
    }
```

Figura 5.1.5 Implementación de la Función Segmenta Caso 3.

5.1.6 Función Consonantes Inseparables.

Existen consonantes que no se pueden separar, es decir, consonantes combinadas que no se pueden separar, esto de acuerdo a la sección 2.2, regla 2, con lo anterior se implementó la Función Consonantes Inseparables, mostrado en la *Figura 5.1.6*.

```
Encripta.java | Main_P.java * | Ventana.java

        if(Vocales_Inseparables(L1,L2) && (Consonantes_Inseparables(L3,L4) || Caso(L4)))
            return true;
        return false;
    }

    public boolean Caso_de_Terminacion_1_4(){
        if(Vocales_Inseparables(L1,L2) && Consonantes_Inseparables(L3,L4)!=true)
            return true;
        return false;
    }


    public boolean Consonantes_Inseparables(char a, char b){
        if(a=='b' && b=='r') return true;
        if(a=='b' && b=='l') return true;
        if(a=='c' && b=='r') return true;
        if(a=='c' && b=='l') return true;
        if(a=='d' && b=='r') return true;
        if(a=='f' && b=='r') return true;
        if(a=='f' && b=='l') return true;
        if(a=='g' && b=='r') return true;
        if(a=='g' && b=='l') return true;
        if(a=='k' && b=='r') return true;
        if(a=='l' && b=='l') return true;
        if(a=='p' && b=='r') return true;
        if(a=='p' && b=='l') return true;
        if(a=='t' && b=='r') return true;
        if(a=='r' && b=='r') return true;
        if(a=='c' && b=='h') return true;
        return false;
    }
```

Figura 5.1.6 Implementación de la función Consonantes Inseparables.

5.1.7 Función Vocales Inseparables.

Fue necesario implementar esta función, que en sí son los diptongos, esto ayuda a que no se destruyan en la segmentación de la palabra en sílabas (*Figura 5.1.7*).

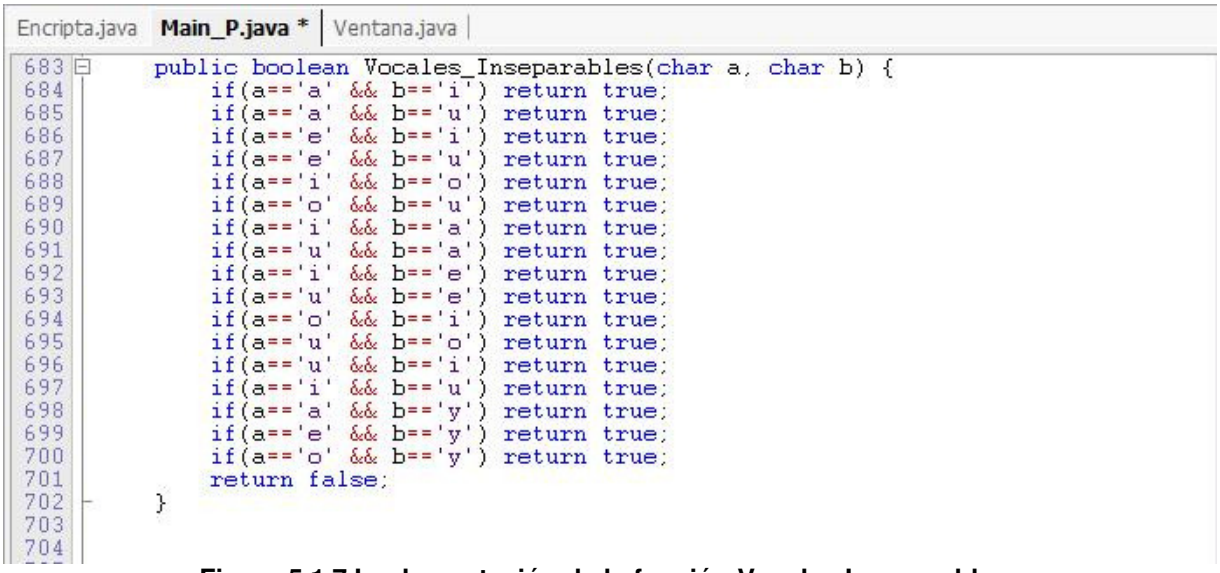


Figura 5.1.7 Implementación de la función Vocales Inseparables.

5.2 Pruebas

Para probar el software desarrollado en esta Tesis, se utilizó la prueba de la Caja Negra.

Para esta prueba se tomaron en cuenta todas las funciones que realiza esta aplicación de encriptación. A continuación se presentan y describen las pruebas realizadas.

5.2.1 Prueba de la Caja Negra.

La prueba de la caja negra no se basa en el conocimiento del código o diseño interno, sino que determina la funcionalidad del sistema.

Para realizar la prueba de la caja negra el usuario probó toda la aplicación, el usuario no tuvo ningún error así pudo probar totalmente este sistema.

En las siguientes imágenes (*Figura 5.2.1, Figura 5.2.2 y Figura 5.2.3*) se observa que el usuario no tuvo ningún problema para utilizar opción de *Abrir Archivo*, el tiempo de respuesta y la facilidad de manejo fueron óptimos.

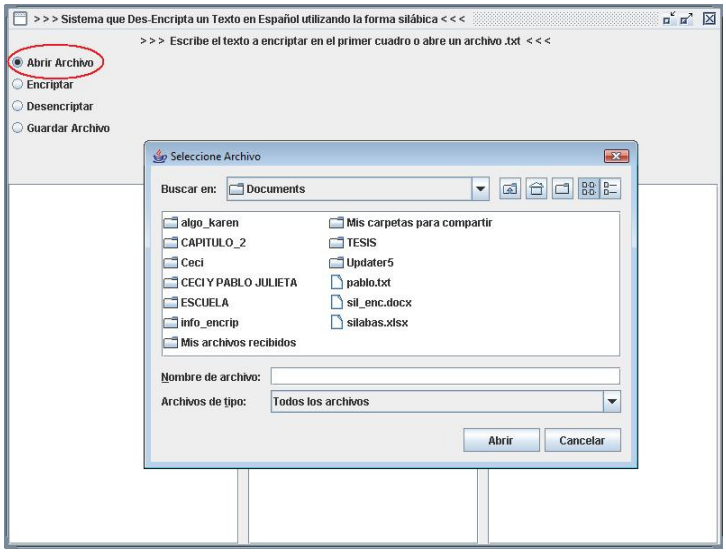


Figura 5.2.1 Prueba de la Caja Negra: Abrir Archivo.

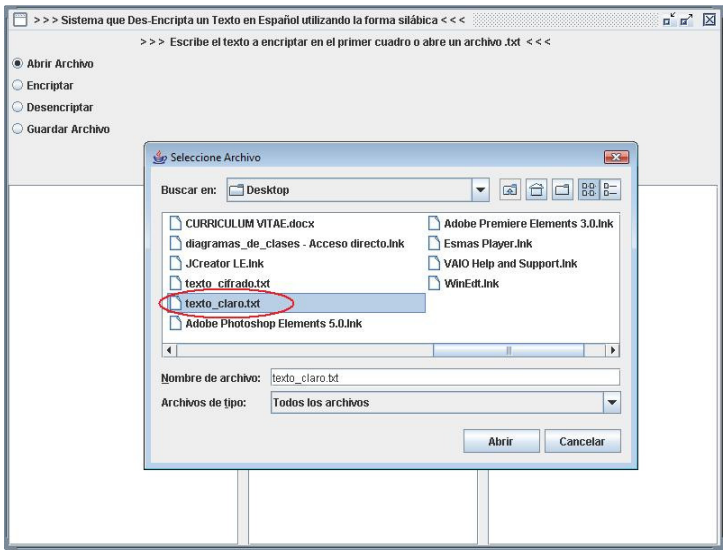


Figura 5.2.2 Prueba de la Caja Negra: Abrir Archivo (Seleccionar archivo).

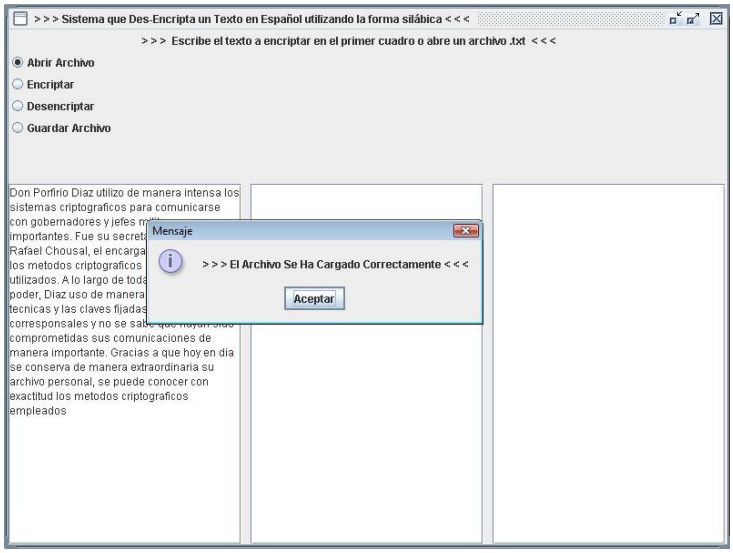


Figura 5.2.3 Prueba de la Caja Negra: Abrir Archivo (Mostrar texto claro en pantalla).

Como se observa cuando el usuario eligió *Abrir Archivo*, se abrió otra ventana donde se tiene que elegir el archivo .txt que contenga el texto claro a ser encriptado (*Figura 5.2.2*).

Una vez que ya se eligió el archivo el programa muestra un mensaje que dice que el archivo se ha cargado correctamente y en el primer campo de texto muestra la información del archivo .txt (*Figura 5.2.3*).

Después de que el archivo .txt se cargo correctamente el usuario dio clic en el botón *Encriptar*, esto se muestra en la *Figura 5.2.4*.

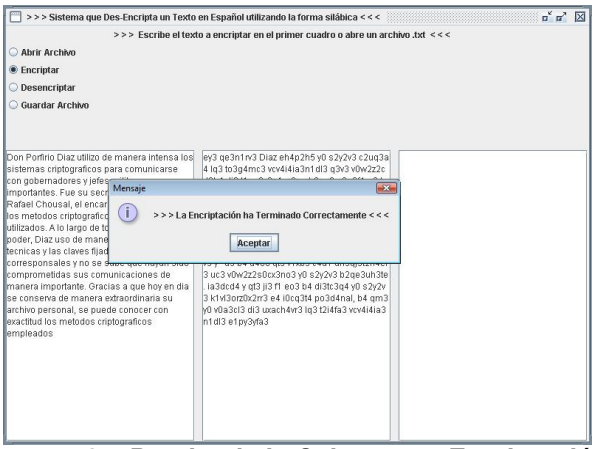


Figura 5.2.4 Prueba de la Caja negra: Encriptación.

La F*igura 5.2.5* muestra la opción de guardar el texto encriptado en un archivo .txt, en esta opción después de que el usuario eligió el botón de *Guardar Archivo* se elige la ubicación de donde se guardará y se escribe el nombre del archivo, después de esto en la *Figura 5.2.6* muestra el mensaje que informa que la encriptación se ha guardado correctamente.

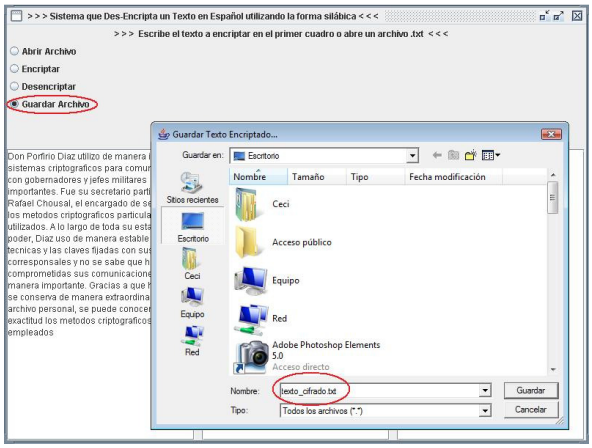


Figura 5.2.5 Prueba de la Caja Negra: Guardar Archivo.

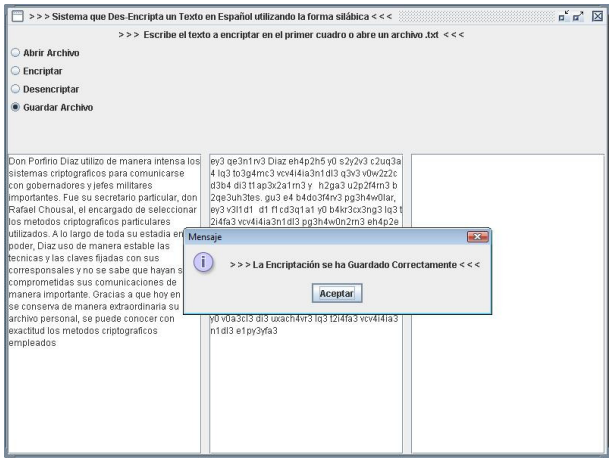


Figura 5.2.6 Prueba de la Caja Negra: Guardar Archivo (Notificación).

Para comprobar que la información haya sido encriptada correctamente, se aplica el proceso inverso: *Desencriptación* (*Figura 5.2.7*).

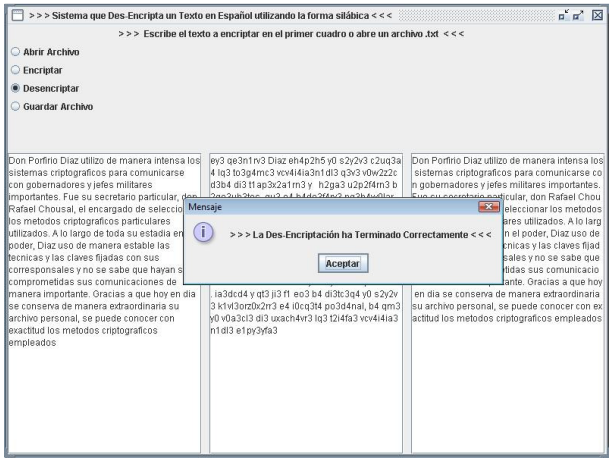


Figura 5.2.7 Prueba de la Caja Negra: Guardar Archivo.

Como se logra ver en todas las imágenes anteriores el usuario no tuvo ningún error al utilizar esta aplicación que encripta un texto con la forma silábica, además el tiempo de respuesta y funcionalidad son óptimos.

CONCLUSIONES Y PERSPECTIVAS

Conclusiones.

La criptografía fue empleada desde épocas antiguas hasta hoy en día, ya que el hombre tiene como una de sus prioridades el proceso de comunicación, la confidencialidad de sus mensajes así como de su información.

A continuación se citan los logros alcanzados con el programa que encripta un texto en español utilizando la forma silábica:

- Los objetivos que se plantearon al inicio de este proyecto se cumplieron, ya que se obtuvo el programa “Encriptación de un texto en español utilizando la forma silábica”.

- Al utilizar este programa la información se cifra y se impide su comprensión.

- El programa es una herramienta con un nivel de utilización alto, rápido, fácil de utilizar y sobretodo duro de romper, porque la manera en que trabaja es diferente a la mayoría de los que ya existen, garantiza total integridad de la información, es decir, no hay pérdida de datos.

- La interfaz del programa es fácil de utilizar además de que como fue creado en Java, puede ser portable por su libre plataforma, utilizándose así en cualquier lugar donde la información requiera estar protegida y sea confiable a la hora de su utilización.

Perspectivas

El programa no encripta palabras con acento, lo cual se puede implementar en un futuro para que el programa este completo. Con esto el usuario podrá utilizar cualquier archivo sin necesidad de quitar el acento a cada palabra.

Adecuar el programa “Encriptación de un texto en español utilizando la forma silábica” para que cada vez que el usuario lo utilice, ingrese una contraseña y así sea más segura su utilización.

Se puede implementar también un paradigma cliente/servidor para proteger los mensajes.

APÉNDICE I

a	bien	bril	cham	cles	cuer	dis
ab	bier	brin	chan	cli	cues	diur
abs	bil	brio	char	clip	cui	diz
ac	bio	brios	chas	clis	cul	do
ad	bios	brir	che	clo	cum	doc
ah	bir	briz	chen	clu	cun	don
ai	bis	bro	ches	club	cuns	dor
ais	biz	bron	chez	co	cuo	dos
al	bla	bros	chi	col	cur	doy
am	blan	bru	chim	com	cus	dra
an	blar	bu	chin	con	cuz	dral
ar	blas	bue	chis	cons	da	dran
as	ble	buen	cho	cop	dac	drar
at	bles	buey	con	cor	dad	dras
au	bli	bui	chos	cos	dal	dre
aun	blia	bul	chu	coz	dan	dren
aus	blio	bum	chue	cra	dap	dres
ay	blo	bun	chum	cran	dar	drez
ba	bloc	bur	chun	cre	das	dri
bac	blon	bus	chus	cres	de	drid
bai	blor	buz	ci	crez	deis	dril
bais	blos	ca	cia	cri	del	drir
bal	blu	cac	cial	cria	den	dro
bam	bo	cai	cian	crian	der	dron
ban	boi	cal	ciar	criar	des	dros
bar	bol	cam	cias	crie	dez	dru
bas	bom	can	cie	crip	di	du
bau	bon	cap	ciem	cris	dia	duc
be	bor	car	cien	cro	dial	due
bei	bos	cas	cier	cros	dian	duen
beig	bot	cau	cies	cru	diar	duer
beis	bra	ce	cil	crus	dias	dul
bel	bral	ced	cin	cruz	dic	dum
ben	bran	cei	cinc	cu	die	dun
ber	brar	ceis	cio	cua	dien	duos

bes	bras	cel	cion	cual	dies	dur
bi	bre	cen	cios	cuan	diez	dus
bia	bren	cep	cir	cuar	dig	e
bian	bres	cer	cis	cuas	din	eh
biar	bri	ces	ciu	cue	dio	eis
bias	brie	ceu	cla	cuel	dios	el
bie	brien	cha	cle	cuen	dir	em
en	fom	ged	grie	haz	je	lia
er	fon	gel	gril	he	jem	lian
es	for	gen	gris	hem	jer	lias
et	fos	ger	gro	her	jes	lie
eu	fra	ges	gros	hi	ji	lien
ex	frac	gi	gru	hia	jien	lies
fa	frag	gia	grue	hie	jil	lim
fac	fran	gian	gu	hier	jin	lin
fai	fras	giar	gua	hin	jir	lio
fal	fray	gias	gual	hip	jis	lios
fan	fraz	gie	guan	his	jo	lip
far	fre	gien	guar	ho	jon	lir
fas	fren	gil	guas	hol	jor	lis
fau	fres	gim	guau	hom	jos	liz
fe	fri	gio	gue	hon	ju	lla
fec	fria	gion	guen	hor	jua	llais
fel	friar	gios	gues	hos	jue	llan
fen	frie	gip	gui	hoy	juez	llar
fer	frir	gir	guia	hoz	jui	llas
fes	fro	gis	guiar	hu	jun	lle
fi	fron	gla	guie	hue	jus	llen
fia	fru	glan	guien	huer	ka	ller
fian	fu	glar	guin	hues	ki	lles
fie	fuc	glas	guio	huir	kin	lli
fiel	fue	gle	guion	hum	kios	llin
fien	fuen	gles	guir	hun	la	llo
fier	fuer	glios	gum	hur	lai	llon
fies	fui	glo	gun	i	lam	llos
fil	fuis	glon	guo	id	lan	llu
fin	fun	glos	guos	im	lar	llue
fir	fur	go	gur	in	las	lo
fla	fut	gol	gus	ins	lau	loj
flan	ga	gon	ge	ir	le	lom
flau	gad	gor	gen	is	lec	lon
fle	gais	gos	gi	iz	len	lor

fles	gal	gra	ha	ja	lep	los
flo	gam	gran	hal	jal	ler	loz
flor	gan	grar	ham	jan	les	lu
flu	gar	gras	han	jar	let	lua
fluen	gas	gre	har	jas	lets	lud
fluo	gaz	gres	has	jau	ley	lue
fo	ge	gri	hay	jaz	li	lum
lus	mues	nor	or	plas	pun	reu
luz	mul	nos	os	plau	pur	rey
ma	mun	noz	pa	ple	pus	rez
mag	mur	un	pac	plen	puz	ri
mal	mus	nua	pai	ples	que	ria
man	mut	nual	pais	pli	quel	rial
mar	muy	nue	pal	plia	quen	rias
mas	na	nues	pam	plie	ques	rie
me	nal	nuez	pan	plio	qui	rien
meis	nam	nuir	par	plios	quia	ries
mem	nan	nul	pas	plo	quial	ril
men	nar	nun	paz	plos	quiar	rin
mer	nas	nuo	pe	pru	quías	rio
mes	nau	nus	pec	po	quid	rion
mez	naz	ña	ped	pol	quie	rior
mi	ne	ñac	pei	pon	quien	ríos
mia	nec	ñal	pel	por	quier	rir
mian	neis	ñan	pen	pos	quin	ris
miau	nel	ñar	per	para	quios	riz
mie	nen	ñas	pes	prac	quis	ro
miel	ner	ñe	pez	pran	ra	roi
mien	nes	ñen	pi	prar	rac	rol
mier	net	ñi	pia	pras	rad	rom
mil	neu	ñil	pian	pre	ral	ron
mim	nez	ñir	piar	pren	ram	ros
min	ni	ñis	pias	pres	ran	roz
mio	nia	ño	pie	pri	rap	rra
mion	nias	ñol	piel	prie	rar	rral
mios	nid	ñon	pien	prin	ras	rran
mir	nie	ñor	pier	pris	rax	rrar
mis	nien	ños	pies	pro	ray	rras
mix	nil	ñue	pin	pron	raz	rre
mo	nin	o	ping	pru	re	rren
moi	nio	ob	pio	prue	rec	rrer
mol	nion	obs	pion	psi	red	rres

mon	nios	oc	pios	pu	rei	rri
mons	nir	oh	pis	pue	reis	rria
mor	nis	oi	piz	puen	rel	rrie
mos	niz	ol	pla	puer	ren	rrien
mu	no	om	plac	pues	rep	rries
mue	noc	on	plan	pug	rer	rril
muer	nom	op	plar	pul	res	rrio
rrion	sie	tau	trai	tum	vios	zam
rrios	siem	taz	tral	tun	vir	zan
rrir	sien	te	tram	tuo	vis	zar
rro	sier	tec	tran	tur	vo	zas
rron	sies	ted	trans	tus	vol	ze
rror	sig	teis	trar	u	vor	zi
rros	sil	tel	tras	ud	vos	zinc
rroz	sim	tem	tre	ul	voy	zo
rru	sin	ten	trein	um	voz	zoi
rrui	sio	ter	tren	un	vu	zon
rrum	sion	tes	tres	ur	vue	zos
rrup	sir	tex	tri	us	vuel	zu
ru	sis	ti	ria	va	vues	zue
rue	so	tia	trias	vai	wa	zul
rui	soil	tial	trio	vais	xa	
rum	sois	tias	trir	val	xe	
rus	sol	tie	tris	van	xi	
sa	som	tiem	triun	var	xia	
sad	son	tien	triz	vas	xias	
sai	sor	tier	tro	ve	xis	
sal	sos	ties	trol	ved	xo	
sam	soy	tig	trom	vein	y	
san	su	til	tron	veis	ya	
sar	sua	tim	tros	vel	yais	
sas	sub	tin	troz	ven	yak	
sau	sue	tio	tru	ver	yan	
se	suel	tion	"truc	ves	yar	
sec	suer	tios	true	vez	yas	
sed	sui	tir	trui	vi	ye	
seis	sul	tis	truir	via	yec	
sel	sun	tiu	truo	vial	yen	
sem	sur	tiun	truos	vías	yer	
sen	sus	tiz	truz	vic	yes	
sep	ta	to	tu	vid	yo	
ser	tac	tol	tua	vie	yon	

ses	tad	tom	tual	viem	yor	
sex	tais	ton	tuar	vien	yos	
sey	tal	tor	tuas	vier	yu	
seys	tam	tos	tud	vil	yun	
si	tan	toy	tuer	vin	yus	
sia	tar	tra	tuir	vio	za	
sias	tas	trac	tul	vion	zal	

BIBLIOGRAFÍA

Referencias a libros.

[2] Knudsen, Jonathan (1998). *Java Cryptography (First Edition)*. Boston, MA Estados Unidos. Editorial O'Rielly Media Inc.

[3] Castañeda Ayala, María Juliana y Morales Quiroga, Mónica (2004). *Seguridad en las Transacciones Electrónicas.* Pontificia Universidad Javeriana, Bogotá D.C.

[4] Figueroa Mora, Karina (1997). *Síntesis de voz en español, un enfoque silábico*. Facultad de Físico Matemáticas de Morelia, Michoacán, Mexico.

[5] Real Academia de la Lengua Española (1999). *Ortografía de la Lengua Española.* Editorial Espasa.

[7] S. Pressman, Roger (1994). *Ingeniería del software: Un enfoque práctico (3ra. Edición Ilustrada).* Editorial McGraw-Hill.

Referencias a fuentes electrónicas.

[1] *Historia de la Criptografía*. (2008, 22 de septiembre). Extraído el 14 de diciembre de 2008, de la World Wide Web: http://es.wikipedia.org/wiki/Historia_de_la_criptograf%C3%ADa

[6] *El castellano sólo tiene poco más de 3000 silabas.* (2008). Extraído el 23 de enero de 2009, de la World Wide Web: http://www.elcastellano.org/ns/edicion/2009/marzo/silabeo.html