Contenido

Creación de Patrones de Criptografía PGP Para Aplicaciones Utilizando Linux

M. en C. Jesús Antonio Álvarez Cedillo
M. en C. Elizabeth Acosta Gonzaga
Ing. Macario García Arregui

Profesores del CIDETEC-IPN

La seguridad informática comprende muchas áreas de interés originadas por la importancia fundamental del valor de la información. El garantizar que la información que es transmitida por cualquier medio de comunicación sea inaccesible para otras personas, no es un proceso fácil, y en algunas ocasiones requiere de recursos computacionales y financieros considerables; existen muchos esquemas de encriptación, sistemas y algoritmos para proteger la información mientras ésta se transmite, de tal manera que sea accesible unicamente por quien tiene la autorización de leerla.

El sistema operativo *Linux* ha servido como una plataforma de arranque importante para el desarrollo de nuevos estándares y sistemas abiertos, donde la colaboración de muchas personas expertas en software permite la creación de protocolos y sistemas nuevos.

El Sistema *PGP (Pretty Good Privacy* ó Muy Buena Privacidad)[1] es un sistema de encriptación por llave pública utilizado en plataformas *Linux* que actualmente se está desarrollando como un estándar, que utiliza dos llaves una pública y otra privada; la llave pública es la que se distribuye a los usuarios autorizados, y sirve para el envio de mensajes codificados que solo pueden decifrarse mediante la llave privada. Esta llave pública también puede servir para firmar un mensaje poniendo una parte de la llave privada en la firma, considerándose esto como un certificado de autenticidad; así, al recibir el mensaje el *PGP* comprueba la firma y texto y lo compara con la llave pública que el destinatario recibe previamente del remitente, mostrando un error si se ha cambiado algo en el texto o la rúbrica electrónica no corresponde a la de la persona que envía el mensaje. Su versión en software libre, *GnuPG*, es una utilidad de línea de comandos que incluye al motor de cifrado, mismo que puede ser utilizado directamente desde dicha línea, desde programas de intérpretes de instrucciones o por otros programas, siendo esta la característica que permite crear servidores de seguridad.

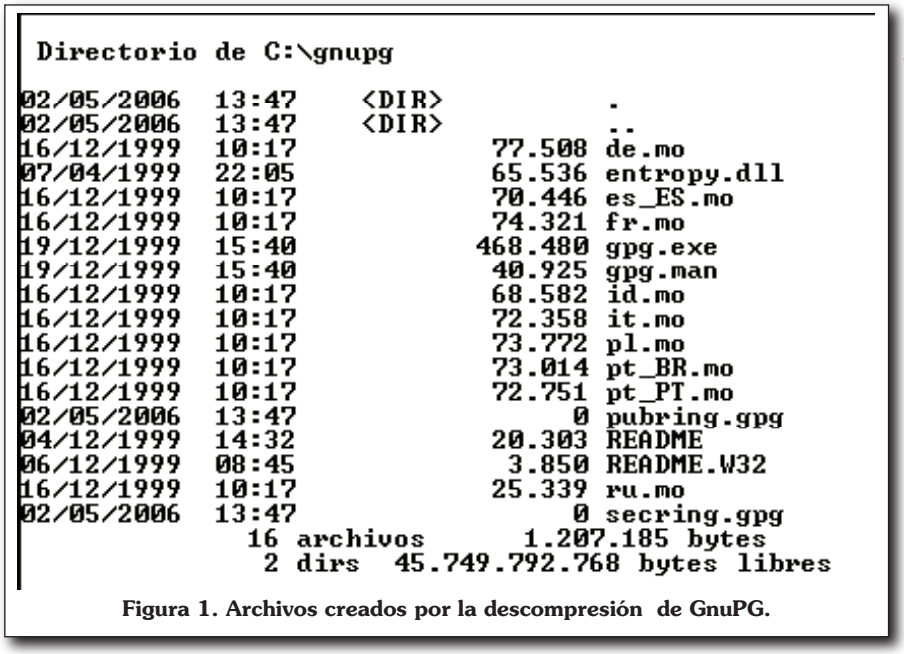
Historia

Las ideas de la innovación del sistema *PGP* eran comunes y generales entre los informáticos y los matemáticos desde hace mucho tiempo, por lo que sus conceptos subyacentes no son en verdad innovadores.

Philip R. Zimmermann es el creador del *Pretty Good Privacy*, para la encriptación de correo electrónico; PGP se publico gratis en la Internet en 1991, y se convirtió en el software de encriptación más utilizado en el mundo. Zimermann es asesor en cuestiones de criptografía para varias compañías y organizaciones industriales, además es socio del Stanford Law School´s Center for Internet and Society (Centro de Enseñanza de Derecho de Stanford para Internet y la Sociedad).

La innovación verdadera de Zimmermann consistía en la creación de estas herramientas para ser utilizadas por cualquier persona con una computadora personal; incluso las primeras versiones del *PGP* dieron acceso a las personas que utilizaban el sistema operativo *MSDOS*. Zimmermann, activista antinuclear, pensó que el *PGP* sería empleado por la mayoría de los disidentes y de los rebeldes; es decir este sistema se usaría fuera y dentro de los Estados Unidos de Norteamérica para el envío de información confidencial activista.

Desde el fin de la segunda guerra mundial, el gobierno de Estados Unidos ha considerado la encriptación resistente como una amenaza seria para la seguridad nacional, por lo que prohíbe exportar este sistema de los Estados Unidos al resto del mundo; la exportación de software del cifrado, incluyendo el PGP, requirió una licencia del Departamento



```
 Directorio de C:\gnupg

02/05/2006  13:47    <DIR>          .
02/05/2006  13:47    <DIR>          ..
16/12/1999  10:17            77.508 de.mo
07/04/1999  22:05            65.536 entropy.dll
16/12/1999  10:17            70.446 es_ES.mo
16/12/1999  10:17            74.321 fr.mo
19/12/1999  15:40           468.480 gpg.exe
19/12/1999  15:40            40.925 gpg.man
16/12/1999  10:17            68.582 id.mo
16/12/1999  10:17            72.358 it.mo
16/12/1999  10:17            73.772 pl.mo
16/12/1999  10:17            73.014 pt_BR.mo
16/12/1999  10:17            72.751 pt_PT.mo
02/05/2006  13:47                 0 pubring.gpg
04/12/1999  14:32            20.303 README
06/12/1999  08:45             3.850 README.W32
16/12/1999  10:17            25.339 ru.mo
02/05/2006  13:47                 0 secring.gpg
              16 archivos      1.207.185 bytes
               2 dirs  45.749.792.768 bytes libres
```

Figura 1. Archivos creados por la descompresión de GnuPG.

del Estado, quedando vedado tanto el software, como la licencia de uso para ciertos países, los cuales no podrían recibir tales exportaciones bajo circunstancia alguna. Estas reglas eran conocidas como *ITAR*, quedando clasificadas las herramientas de cifrado como armas de guerra en el área de las comunicaciones en el ejército de los Estados Unidos de Norteamérica.

El gobierno, con este fundamento, demandó a Zimmermann quien por tres años defendió su proyecto en las cortes. Este pleito dio vuelta al mundo, y Zimmermann fue considerado como un héroe en la comunidad computacional, por lo que mucha gente descargaba el *PGP* para ver cuál era el problema y en que consistía el reclamo hecho por el gobierno, causando así un boom informático. La defensa de Zimmermann separó las noticias del pleito del PGP y las presentó en las audiencias, donde leyó las cartas que había recibido de la gente proveniente de los regímenes opresivos y de las áreas de devastación de guerra, cuyas vidas habían sido salvadas por *PGP*.

Como obtener PGP

El archivo de instalación del *PGP* se llama *pgp263i.zip* y es posible obtenerlo por Internet; el programa, información y auxiliares se encuentran en ***http://www.pgpi.com/***, la cual es la página internacional del *PGP*. El archivo deberá descomprimirse con *PKUNZIP,* de modo que sus componentes queden guardados en el directorio *c:\pgp*, o bien *c:/home/usuario/pgp;* aparecerán algunos archivos, entre ellos otro elemento comprimido llamado *pgp263ii.zip* y un archivo de llave publica denominado *pgp263ii.asc*. Ambos archivos contienen la información para verificar la llave pública de Stale Schumacher que indica si el archivo es auténtico. La llave de Stale resulta al descifrar el archivo keys.asc, que también es resultante de la descompresión del archivo antes mencionado (*pgp263i.zip)*. En la **Figura 1** se muestra la estructura de archivos creados.

Funcionamiento

Para generar el par de llaves, tanto la publica como la privada, es necesario que el programa esté debidamente cargado y configurado. Es importante que esté el nombre del usuario en el config.txt y que se escriba de la misma forma en el par de llaves. El comando para generar dicho par es muy sencillo (**Figura 2**).

gpg –gen –key

Luego el programa preguntará por el tamaño de la llave, (ver **Figura 3**). El nivel de seguridad está en relación con el tamaño de las claves (medido en bits) que se generarán al cifrar cada vez. Cuanto mayor sea el número de bits, los procesos serán más lentos. Con la velocidad de las máquinas actuales, 1024 bits es razonablemente rápido y con mucha seguridad.

Después preguntará por el tiempo de expiración de la llave, (ver **Figura 4**). Se recomienda seleccionar la opción en la que la llave nunca caduque y de esta forma el usuario no tendrá que preocuparse por la vigencia; a continuación será necesario identificar la llave con un usuario, esto se observa en la **Figura 5**.

Después solicitará la frase que irá integrada en la llave, formada por la frase clave o contraseña que servirá para abrir la llave privada. Se recomienda una frase no muy corta y que no se adivine fácilmente, que no sea una frase hecha y que de preferencia tenga algunos signos de puntuación, números o caracteres en código ASCII simple. Es necesario mover el teclado y el ratón con el fin de generar mayor carga computacional que se traduzca en números al azar, (ver **Figura 6**.)

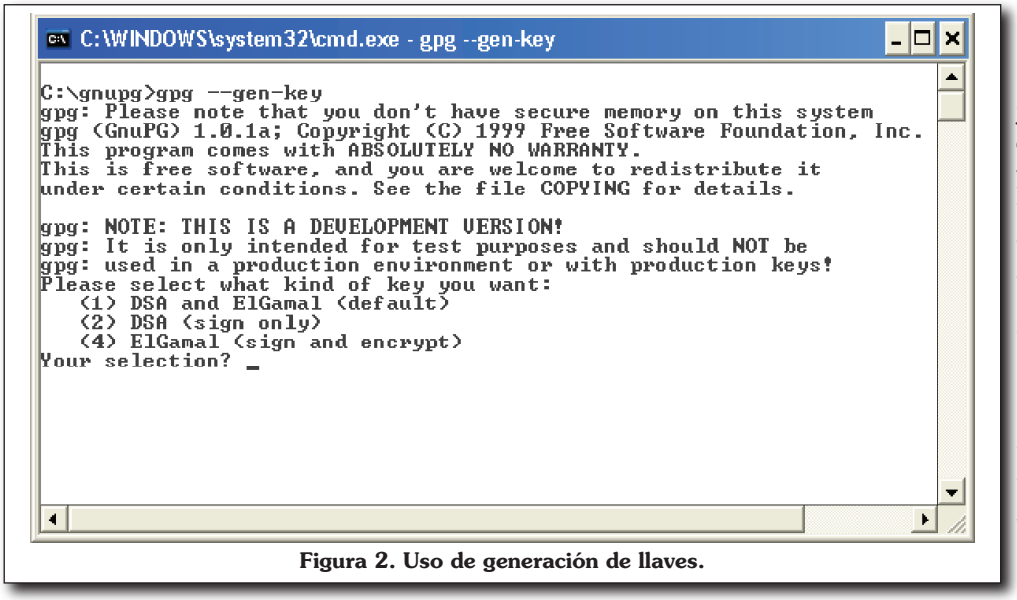


Figura 2. Uso de generación de llaves.

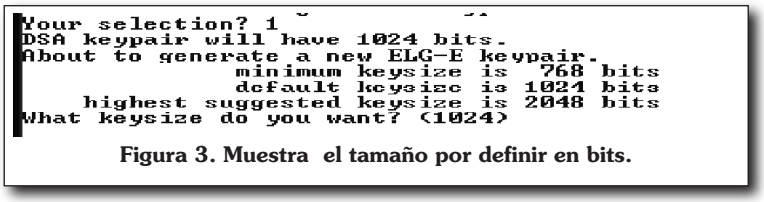


Figura 3. Muestra el tamaño por definir en bits.

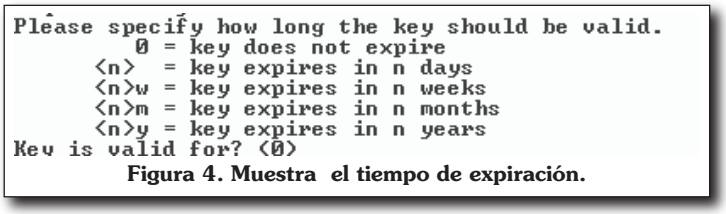


Figura 4. Muestra el tiempo de expiración.

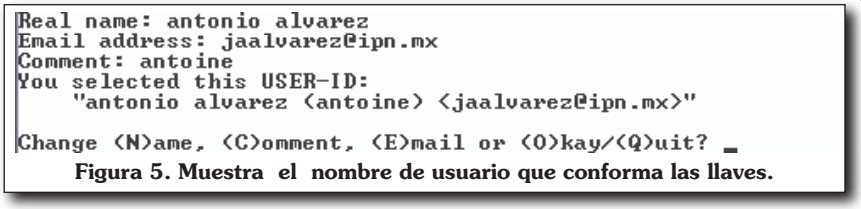


Figura 5. Muestra el nombre de usuario que conforma las llaves.

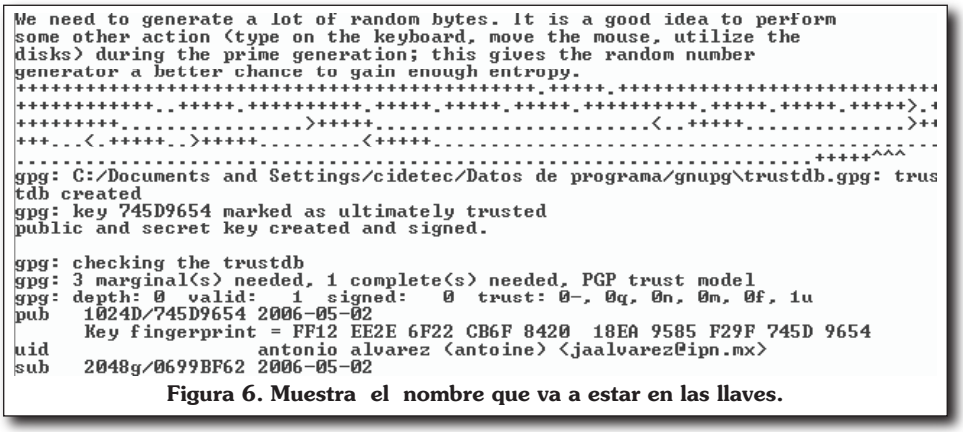


Figura 6. Muestra el nombre que va a estar en las llaves.

Para poder intercambiar mensajes cifrados con otra persona, esta debe conocer nuestra llave pública y nosotros la suya. Cabe considerar que para que el usuario pueda crear su llave, utilizará la instrucción siguiente:

$gpg –export –a –u jesusantonioa@ipn.mx **> pubkey.asc**

Para importar la llave del usuario remoto con el cual se intercambiará información se debe utilizar la siguiente instrucción:

$gpg –import usuario.pubkey.asc

Para validar la llave o darle el visto bueno se hará uso de la siguiente orden:

$gpg –edit-key usuario

Operación

Para la operación de este sistema, se debe usar el archivo de texto llamado tclaro.txt (Texto claro) que contiene la palabra CIDETEC. Primero deberá ser firmado electrónicamente empleando el comando *SIGN* y utilizando la opción –a para que el resultado sea un archivo de 7 bits ASCII y no un archivo binario, (**Figura 7**):

GPG –a –sign tclaro.txt

A continuación se hace la verificación del archivo utilizando el comando (**Figura 8**):

gpg –veryfy TCLARO.TXT.asc

Después sólo es necesario mandarlo al destinatario utilizando cualquier medio: Internet, correo electrónico, etc.

Debe observarse el contenido del archivo, considerándose que ahora es un archivo cifrado y obviamente para descifrarlo es necesario aplicar el siguiente comando (**Figura 9**):

Gpg –o tdescifrado.txt –decrypt TCLARO.TXT.asc

```
C:\ARCHIV~1\GNU\GnuPG>GPG -a --sign TCLARO.TXT

You need a passphrase to unlock the secret key for
user: "antonio alvarez (antoine) <jaalvarez@ipn.mx>"
1024-bit DSA key, ID 745D9654, created 2006-05-02
```

Figura 7. Muestra la manera de convertir y firmar electrónicamente el archivo tclaro.txt (Para realizarlo es necesario la palabra clave).

```
C:\ARCHIV~1\GNU\GnuPG>gpg --verify TCLARO.TXT.asc
gpg: Signature made 05/03/06 13:24:39 using DSA key ID 745D9654
gpg: Good signature from "antonio alvarez (antoine) <jaalvarez@ipn.mx>"
```

Figura 8. Muestra la manera de verificar el archivo tclaro.txt.asc (Para realizarlo se necesita la palabra clave)

```
C:\Archivos de programa\GNU\GnuPG>more TCLARO.TXT.asc
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.4.2 (MingW32)

owGbwMvMwCQ4tfXT/JLYaSGMaySTuEKcfRyD/PUCIkJcIr6mO3u6uIa4OvNyddgz
szKARGCKBZnS9zHMI08/07OqoWLbmzi++Q/dOCIkbyxUYZjNuvNk7gSJT5f+3Gs5
eeJ0zOnqu5V/AA==
=BKol
-----END PGP MESSAGE-----
```

Figura 9. Muestra el archivo encriptado

```
C:\ARCHIV~1\GNU\GnuPG>Gpg  -odtexto.txt --decrypt TCLARO.TXT.asc
gpg: Signature made 05/03/06 13:24:39 using DSA key ID 745D9654
gpg: Good signature from "antonio alvarez (antoine) <jaalvarez@ipn.mx>"

C:\ARCHIV~1\GNU\GnuPG>more dtexto.txt
CIDETEC
```

Figura 10. Muestra la manera de desencriptar el archivo tclaro.txt.asc

Lo anterior se muestra en la **Figura 10**.

Para comprobar el contenido, debe observarse la igualdad con el original, por lo que el proceso de operación se ha terminado, y lo más importante es que fue completamente seguro.

Conclusiones

En el seguimiento de las instrucciones del proceso de encriptación y desencriptación usando esta herramienta, se observa que es sencillo, rápido y dinámico, ya que utilizando solo algunas órdenes de comando simples es posible integrarlo a procesos y aplicaciones complejos.

Por otro lado, el entorno de utilización también es simple, haciendo así posible el integrarlo de manera multiplataforma a proyectos y de manera de multiprogramación en el uso de diferentes entornos que soporten el GnuPG, empleando procesos secuénciales y paralelos y permitiendo inclusive la construcción de servidores distribuidos y servicios autónomos de encriptación y desencriptacion vía conexión de una red de área local o bien a través de la Internet.

Referencias

Pagina Oficial	**www.gnugp.org**
Pagina sobre criptografía	**www.kriptopolis.com**
Pagina de PGP Internacional	**www.pgpi.com**
Phil Zimmermann	**web.mit.edu/~prz**

Bibliografía

[1] Samson Garpike. *PGP: Pretty Good Privacy*. Dec 1, 1994

[2] Michael Lucas. *PGP & GPG: Email for the Practical Paranoid*. April 1, 2006.

Protocolos de Control de Flujo

M. en C. Mauricio Olguín Carbajal,
M. en C. Israel Rivera Zárate,
Ing. Patricia Pérez Romero,
Profesores del CIDETEC-IPN.

A lo largo de la historia de la informática y la computación, de una u otra forma, siempre se ha definido a una computadora como una maquina que procesa datos; sin embargo, este procesamiento requiere de una amplia variedad de aditamentos de soporte.

Toda esta majestuosa orquesta de equilibrios hoy en día está al alcance de una tecla o un clic, manejada por una sencilla idea que dispara un impulso nervioso que viaja hasta la mano, en donde se convierte en energía mecánica y activa un interruptor, enviando a su vez una leve señal eléctrica a un controlador de entrada, que avisa al procesador que «algo» pasó que requiere de su atención. El procesador interpreta esta señal por su origen y decide qué hacer con ella, a dónde enviarla; en esto lo ayuda el sistema operativo, que le dice qué es y a dónde va. En el caso de una tecla, este impulso eléctrico se convierte a su representación codificada y se envia a la memoria de video, en donde el controlador respectivo se encarga de presentarlo en pantalla... resulta fascinante pensar que todo esto puede ocurrir en una millonésima de segundo o menos.

Esta generación de información no siempre es exitosa; para hacerla confiable existen métodos como el control de Flujo, que garantiza que la información después de ser procesada se envié y reciba de una manera íntegra por el Receptor.

Introducción

Primero que nada diremos que el Control de Flujo existe en el intercambio de Datos (Información) solamente entre 2 entidades: *Transmisor* y *Receptor.*

El Control de Flujo es una técnica para que una computadora llamada *Transmisor* (TX) no sobrecargue a otra denominada *Receptor* (RX), al enviarle más información de la que puede procesar, debido a que normalmente tienen velocidades diferentes. Tanto el *Receptor* como el *Transmisor* tienen una zona de memoria temporal llamada *Buffer*, con una cierta capacidad para almacenar la información recibida, procesarla y enviarla.

El *Receptor* reserva generalmente una zona de memoria temporal para éste efecto; el *Receptor* debe realizar una cierta cantidad de procesamiento antes de pasar los datos al software que los utilizará. Si no existieran procedimientos para el control de flujo, la memoria temporal podría llenarse y eventualmente "desbordarse" mientras se estuviera procesando información.

En estos casos, el *Receptor* envía la información de control de flujo al *Transmisor* para que este reduzca la velocidad de transmisión, logrando así el tiempo necesario para poder procesarla. Es aquí donde se deduce la necesidad de una comunicación entre el *Receptor lento* y el *Transmisor rápido,* para que este último se entere de la situación que se está dando al otro lado del enlace; además, el control de flujo se utiliza para que no se sature la red de comunicaciones. De no existir este control de flujo podría suceder que la información se perdiera y los datos no llegaran completos.

Desarrollo

A continuación se explica de manera gráfica el manejo de la información durante la transmisión y recepción, comparando diferentes protocolos, analizando su funcionamiento y control de mecanismos, y el uso para cada situación de transferencia de datos. Estos son los principales protocolos que existen para el control de flujo:

- Protocolo simplex no restringido.
- Protocolo simplex de parada y espera.
- Protocolo simplex para un canal con ruido.

- Protocolo full duplex con información de reconocimiento (piggybacking).
- Protocolo de ventana deslizante con retroceso *n*.
- Protocolo de ventana deslizante con repetición selectiva.

A los tres primeros protocolos se les conoce como simplex por que solamente se puede transmitir información en un solo sentido, ya sea de ida o de regreso, mientras que a los tres restantes se les conoce como Full duplex, ya que se puede estar recibiendo mientras se envía información.

Protocolos Simplex

Los protocolos Simplex transmiten datos en una sola dirección; el regreso es utilizado únicamente para enviar acuses de recibo del receptor (*ACKnowledgements*).

Deben considerarse diversos factores que pueden hacer que la información se pierda durante la transmisión, entre los que podemos mencionar:

- Ruido: cualquier alteración durante la transmisión de datos debida a causas tales como campos de energía eléctrica, fallas en los cables, etc.
- Demasiado tiempo de procesamiento de la información.
- Límite en el tamaño de la información, tomando en cuenta que el buffer no fuera suficientemente grande.

Protocolo Simplex No Restringido

Este es un caso ideal, en el que se supone que la comunicación es perfecta; como su nombre lo dice no existen restricciones: no hay errores, no hay ruido, no hay limite en el buffer, la información no requiere ser procesada, por lo que no es necesario comprobar que los datos hayan llegado bien ni retransmitirlos. El *receptor* está siempre disponible y preparado para recibir datos con un espacio de buffer infinito, por lo que no se requiere control de flujo; el *transmisor* está siempre preparado para transmitir, y en este caso el único evento posible es la llegada de información.

Protocolo Simplex de Parada y Espera (Stop & Wait)

Ahora supongamos que el *receptor* no siempre está disponible para recibir información, por tener ocupado su espacio de buffer, o bien porque el mensaje sea muy grande y tenga demasiadas instrucciones que atender. En este caso, lo más sencillo es que el *transmisor* espere confirmación *ACK* después de enviar cada mensaje (*Data*), de forma que sólo después de recibir la confirmación se envíe el siguiente bloque, garantizando así el no saturar al receptor. Esto se conoce como protocolo de *parada y espera*, mismo que se muestra en la **Figura 1**, donde:

WT *Wait Time* = Tiempo de Espera

EOT *End Of Transmission* =Fin de Transmisión

DATA = Datos, información.

Protocolo Simplex para un canal con ruido

Si el canal de comunicación no es perfecto las tramas (datos) pueden alterarse debido al ruido, o incluso perderse por completo. Utilizando la Comprobación de Redundancia Cíclica, *CRC* (*Check Redundance Cycle*), que es la encargada de verificar que los datos hayan llegado sin alteraciones, el *receptor* podrá detectar la llegada de una trama (datos) defectuosa, en cuyo caso pedirá al transmisor que la reenvíe.

Sin embargo, esto puede generar duplicidad en la información; para evitar esta repetición, lo más sencillo es numerar las tramas, y forzar al transmisor a no enviar un bloque hasta recibir el acuse de recibo del anterior. Incluso, bastaría con numerar las tramas como 0,1,0,1, etc., tal como se muestra en la **Figura 2**, donde:

D-0 =Data0 y D-1 =Data1

reTX=Retransmisión

ACK= (*ACKnowledgement*)=Acuse de Recibo

Nota: El Transmisor tiene un temporizador (Timer), que sirve para detectar si en un periodo de tiempo X no recibe un ACK, para entonces reenviar la trama D-0 o D-1.

Como se muestra en la **Figura 2**, el transmisor envia D-0 y el receptor responde mediante un ACK; si en el momento que se recibe D-1 el receptor pasa mas tiempo en contestar que el especificado, el transmisor asumirá que la trama no llegó y la reenvía. Cuando el recep-

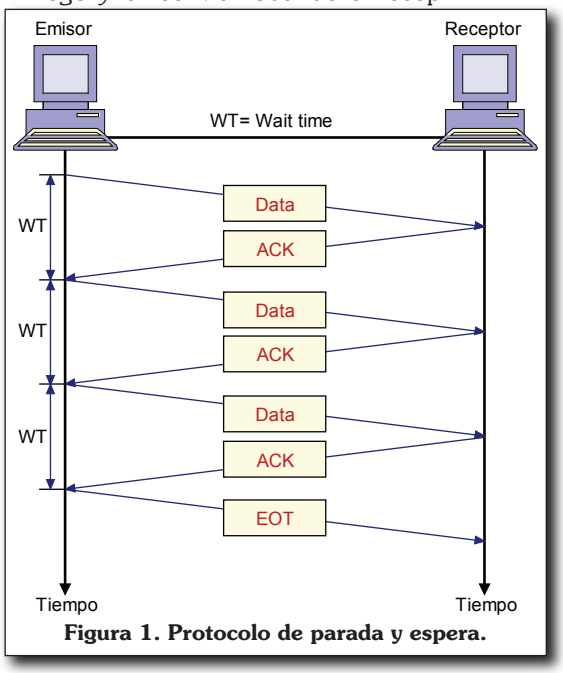


Figura 1. Protocolo de parada y espera.

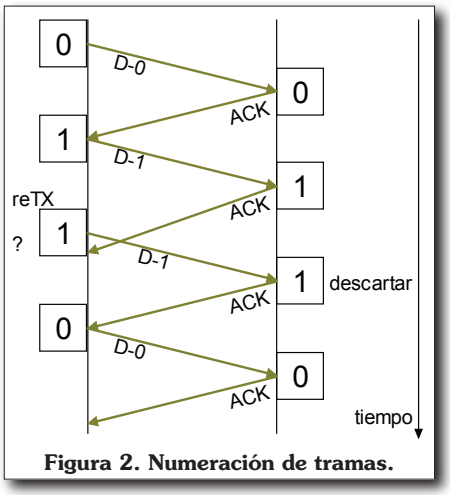


Figura 2. Numeración de tramas.

tor recibe nuevamente a D-1, se da cuenta de que ya lo tiene, por lo que simplemente lo descarta y espera el siguiente envio.

Protocolos de Ventana Deslizante

Los protocolos de ventana deslizante permiten transmitir datos en ambas direcciones utilizando canales Full-dúplex.

Protocolo Full-Duplex con Información de Reconocimiento (Piggybacking)

Recordemos que en el protocolo simplex para un canal con ruido las tramas se numeraban 0,1,0,1,…; en este protocolo el numero 0 o 1 solo servirá para confirmar la recepción de la trama. En este caso se envía un ACK de un numero, no como confirmación de trama correcta, sino para indicar que no llego la trama esperada y le solicita la retransmisión al emisor.

En este caso el ACK se monta en la trama y se ahorra un envío; esta técnica se conoce con el nombre de *piggybacking* (en inglés piggyback significa llevar algo a cuestas).

Protocolo de Retroceso n

Cuando se utiliza un protocolo de ventana deslizante de más de un número (BIT) el emisor no actúa de forma sincronizada con el receptor; por ello, cuando el receptor detecta una trama defectuosa hay varias posteriores ya en camino, que llegarán irremediablemente a él, aún cuando reporte el problema inmediatamente.

Existen dos posibles estrategias en este caso:

1. El receptor ignora las tramas recibidas a partir del error (inclusive) y solicita al emisor retransmisión de todas las tramas subsiguientes. Esta técnica se denomina *retroceso n.*

2. El receptor descarta la trama errónea y pide retransmisión, pero acepta las tramas posteriores que hayan llegado correctamente. Esto se conoce como *repetición selectiva* y corresponde a una ventana deslizante mayor de 1 en el receptor (normalmente de igual tamaño que la ventana del emisor).

En cualquiera de los dos casos el emisor deberá almacenar en su buffer todas las tramas que se encuentren dentro de la ventana, ya que en cualquier momento el receptor puede solicitar la retransmisión de alguna de ellas.

Ejemplo:

a) Caso ideal
b) Caso con retroceso n

Como se muestra en la **Figura 3a**, en un caso ideal el transmisor envía DATA 0123, y el receptor responde con acuses de recibo por cada envio.

En la **Figura 3a** existe una falla, D-2 no llega. El receptor envía un acuse negativo (NACK); sin embargo, el transmisor continua enviando tramas, hasta que recibe el NACK, entonces revisa y empieza a retransmitir a partir de la trama con error, con la desventaja de que se ocupa un ancho de banda considerable.

Protocolo con Repetición Selectiva

La repetición selectiva consiste en aprovechar aquellas tramas correctas que lleguen después de la errónea, evitandose así tráfico en la red al pedir al *emisor* que retransmita únicamente la trama dañada.

Logicamente, la técnica de repetición selectiva da lugar a protocolos más complejos que la de retroceso n, y requiere mayor espacio de buffer en el receptor; a cambio de ello ofrece mayor rendimiento, dado que permite aprovechar todas las tramas correctas.

Conclusiones

El control de flujo nos permite sincronizar el envio de información entre dos entidades, evitando así sobrecargar la red.

Problema: Emisor enviando con mayor velocidad de transmisión que la que el receptor es capaz de procesar.

Solución: Los protocolos incluyen reglas que permiten al transmisor conocer de forma implícita o explícita si puede enviar otra trama al receptor, de manera sincronizada y segura.

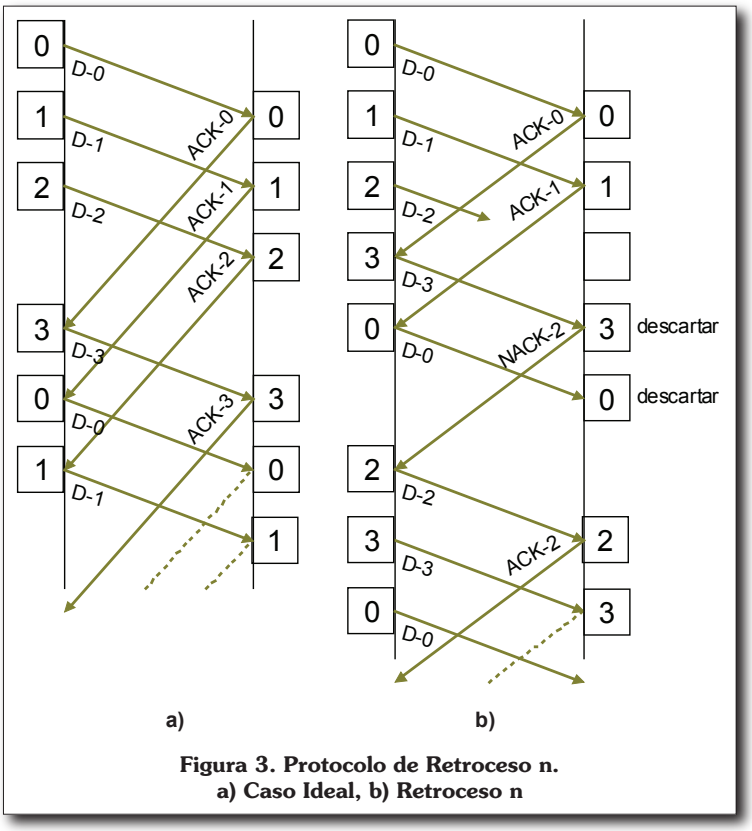


Figura 3. Protocolo de Retroceso n.
a) Caso Ideal, b) Retroceso n

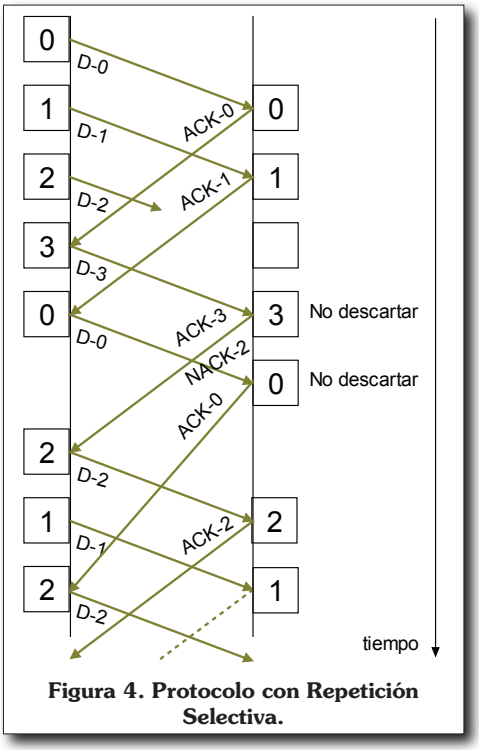


Figura 4. Protocolo con Repetición Selectiva.

Referencias

[1] Tanenbaum, Andrew S., *Redes de Computadoras*. 4ra. ed. Prentice-Hall, 1996.

[2] Forouzan, Behrouz A., *Transmisión de Datos y Redes de Comunicaciones*. 2a ed. McGraw-Hill.

[3] http://gsyc.escet.urjc.es/docencia/asignaturas/itig-transmision_datos/transpas/node5.html

[4] http://www.ilustrados.com/publicaciones/EpVAZVFApFreZSeEml.php

[5] http://deltha.uh.cu/~redes/resources/conf3.htm

[6] http://www.angelfire.com/ga/metalsystem/Redes.txt

[7] http://www.ddominguez.net/teknicas/CURSOS/redes.htm

Análisis Cuaterniónico Real y Complejo: Una Introducción

Dr. Antonio Castañeda Solís
Profesor del CIDETEC-IPN

Este trabajo pretende ser una breve introducción a los aspectos básicos del análisis cuaterniónico y sus aplicaciones. Se maneja la noción de un cuaternión como una generalización de los números complejos y se brinda la definición de los bicuaterniones.

Se consideran únicamente las ideas más importantes del álgebra cuaterniónica, describiendo algunas de sus propiedades y se introduce la representación de los cuaterniones en algunas otras álgebras. Adicionalmente se mencionan algunas aplicaciones del análisis cuaterniónico en los campos de la Física, la Matemática, y la ingeniería.

Introducción

En una gran cantidad de documentos y artículos especializados es posible encontrar aplicaciones del álgebra de cuaterniones reales y complejos.

En el área de las comunicaciones móviles, el álgebra de cuaterniones fue aplicada al estudio de las ecuaciones de Maxwell (descripción de los aspectos de propagación de las ondas electromagnéticas) ya desde los trabajos del mismo Maxwell. Por lo tanto, no resulta extraño que una gran parte de los trabajos científicos modernos utilicen este tipo de técnicas en la resolución del sistema de Maxwell, obteniéndose resultados sorprendentes en algunos casos (ver [5], [6], [10], [11]).

Recientemente se ha utilizado el análisis cuaterniónico en áreas de ingeniería tales como la seguridad en redes inalámbricas (ver [13] y [14]), y en los sistemas de procesamiento de señales e imágenes [15]. Las aplicaciones en la Física son abundantes y se sugiere la consulta de los documentos [1], [2], y [4].

No obstante la gran cantidad de aplicaciones posibles en ingeniería, no podemos dejar de resaltar el interés puramente matemático que por sí mismos aportan los cuaterniones y su álgebra asociada; como referencia es posible apreciar [7] y [12].

Cuaterniones

Consideremos la igualdad siguiente,

$$x^2 - y^2 = (x + y)(x - y), \qquad (1)$$

donde $x, y \in \mathsf{R}$, la cual se verifica directamente. El hecho de poder representar el lado izquierdo de la igualdad (1) como un producto es conocido como factorización [8], y en este caso nos bastan los números reales para poder factorizar la expresión $x^2 - y^2$.

Es posible considerar algunas otras expresiones y preguntarnos acerca de la forma de factorizarlas. Por ejemplo, consideremos la expresión

$$x^2 + y^2,$$

si utilizamos únicamente los números reales no existe una factorización posible para esta expresión. Pero, si utilizamos los números complejos tenemos que

$$x^2 + y^2 = (x + iy)(x - iy),$$

donde i es la unidad imaginaria definida como $i = \sqrt{-1}$

Es decir, si consideramos el número complejo $z = (x + iy)$ y lo multiplicamos por su conjugado complejo $z^* = (x - iy)$ obtenemos la factorización buscada.

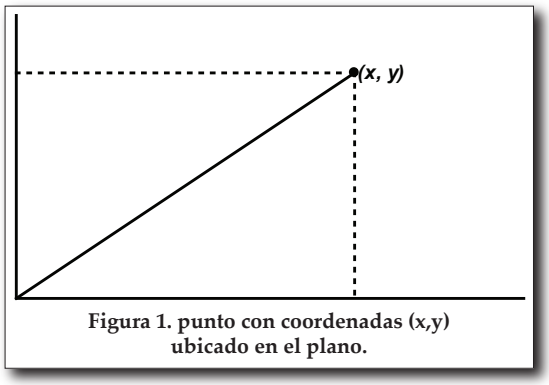


Figura 1. punto con coordenadas (x,y) ubicado en el plano.

Esto es, sea $z \in \mathbb{C}$, entonces

$$z\, z^* = z^*\, z = x^2 + y^2 \qquad (2)$$

La igualdad (2) refleja el hecho de que el álgebra de los números complejos es conmutativa.

Consideremos un punto con coordenadas (x, y) ubicado en el plano (**Figura 1**).

La distancia del punto (x, y) hacia el origen se define como:

$$|z| = \sqrt{x^2 + y^2}. \qquad (3)$$

Comparando (2) y (3) podemos escribir

$$|z| = \sqrt{x^2 + y^2}.$$

Es decir, la distancia entre dos puntos en el plano (y por lo tanto los vectores en dos dimensiones), puede ser expresada mediante el producto de dos números complejos z y z^*.

Consideremos ahora la distancia con respecto al origen en un espacio de cuatro dimensiones, esto es

$$\sqrt{x_0^2 + x_1^2 + x_2^2 + x_3^2}.$$

La pregunta inmediata es ¿cómo podemos factorizar la expresión $x_0^2 + x_1^2 + x_2^2 + x_3^2$?. Desde luego, esto no puede realizarse utilizando los números reales y ni siquiera los números complejos.

En forma análoga al procedimiento usado con los números complejos podemos introducir tres unidades imaginarias definidas en la forma siguiente

$$i_1^2 = i_2^2 = i_3^2 = -1.$$

Si asumimos que las unidades imaginarias introducidas cumplen con las propiedades

$$i_1 i_2 = -i_2 i_1 = i_3,$$
$$i_2 i_3 = -i_3 i_2 = i_1,$$
$$i_3 i_1 = -i_1 i_3 = i_2,$$

es fácil comprobar que la factorización buscada se expresa en la forma siguiente

Al igual que en el caso complejo, la factorización es posible mediante la introducción de nuevos números, los cuales en este caso se llaman cuaterniones. Estos números fueron inventados en 1843 por Sir Lord Hamilton y se definen formalmente como:

Sea $q \in \mathsf{H}(\mathsf{R})$, entonces

$$q = \sum_{k=0}^{3} q_k i_k = q_0 i_0 + q_1 i_1 + q_2 i_2 + q_3 i_3,$$

donde $i_0 = 1$ y q_k son números reales. $\mathbb{H}(\mathbb{R})$ denota el álgebra de los cuaterniones reales.

Un cuaternión tiene representaciones en distintas álgebras. En terminos vectoriales, un cuaternión puede representarse como

$$q = Sc(q) + Vec(q),$$

donde

$$Sc(q) := q_0,$$
$$Vec(q) := \bar{q} = \sum_{k=1}^{3} q_k e_k.$$

Un cuaternión de la forma $q = \bar{q}$ es llamado puramente vectorial. Los cuaterniones puramente vectoriales son identificados con los vectores en $\mathbb{R}^3$.

En el conjunto $\mathbb{H}(\mathbb{R})$ se define la conjugación cuaterniónica en la forma

$$\bar{q} = q_0 - \bar{q},$$

misma que ya habíamos utilizado en la factorización de la expresión $x_0^2 + x_1^2 + x_2^2 + x_3^2$. Por lo tanto, podemos escribir

$$q \cdot \bar{q} = |q|^2,$$

donde

$$|q| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

es la distancia en $\mathbb{R}^4$.

La analogía con los números complejos es total, y el análisis cuaterniónico se considera como la generalización más adecuada del análisis complejo para el espacio de cuatro dimensiones.

Si en la definición de un cuaternión,

$$q = \sum_{k=0}^{3} q_k i_k$$

suponemos que q_k son números complejos y que la unidad imaginaria compleja i conmuta con las unidades imaginarias i_1, i_2, i_3 obtenemos el conjunto de cuaterniones complejos o bicuaterniones denotado como $\mathbb{H}(\mathbb{C})$.

Conclusiones

El hecho de poder representar puntos en el plano como parejas ordenadas abre la posibilidad de utilizar el análisis complejo en aplicaciones de ingeniería tales como el procesamiento de imágenes en dos dimensiones. Pero, dado que el mundo en el cual vivimos está compuesto de más de dos dimensiones, es muy natural intentar representar los problemas planteados en una y dos dimensiones como problemas en más dimensiones.

La importancia del álgebra de cuaterniones se enfatiza a través de aplicaciones en áreas tan diversas como la física relativista y la seguridad en redes de computadoras inalámbricas. Se brindan, además, referencias a distintos trabajos de cada una de las áreas mencionadas.

Se introdujeron los conceptos de cuaterniones reales y bicuaterniones como una generalización de los números complejos, resaltando las ideas principales y más intuitivas y delegando para trabajos subsecuentes una exposición más formal y compleja de los distintos aspectos del álgebra de cuaterniones (reales y complejos) y sus aplicaciones en la ingeniería.

Bibliografía

[1] Bagrov V.; Gitman D.M. *Exact Solutions of Relativistic Wave Equations*. Kluwer Academic Publishers, Dordrecht-Boston-London, 1990.

[2] Berezin A. V.; Kurochkin Yu. A. and Tolkachev E. A., *Quaternions in relativistic physics*. Minsk: Nauka y Tekhnika, 1989.

[3] Bernstein S. Factorization of Solutions of the Schrödinger Equation. *Proceedings of the symposium: Analitical and Numerical Methods in Quaternionic and Clifford Analysis*, Seiffen, 1996, Eds. K. Gürlebeck and W. Sprösig, 1-6.

[4] Bernstein S. *Fundamental Solutions for Dirac-type Operators*. Banach Center Publications, v.37 « Generalizations of Complex Analysis and their Applications in Physics», Warsaw, 1996, J. Lawrynowicz (ed), 159-172.

[5] Castañeda A.; Kravchenko V. Sobre algunas nuevas representaciones integrales para el campo electromagnético en medios no homogéneos. *Memorias del 7o. Congreso de Ingeniería Electromecánica y de Sistemas*, IPN, México, 2003.

[6] Castañeda A.; Kravchenko V. *New applications of pseudoanalytic function theory to de Dirac equation*. J. Phys. A: Math. Gen. 38 (2005) 9207-9219.

[7] Kravchenko V.; Shapiro M. *Integral representations for spatial models of mathematical physics*. Pitman research notes in mathematics series, 1996.

[8] Kravchenko V. *Elementos del análisis moderno y teoría electromagnética*. Instituto Mexicano de Comunicaciones, serie técnica matemáticas aplicadas, 1996.

[10] Kravchenko V. *A new method for obtaining solutions of the Dirac equation*. Zeitschrift für Analysis und ihre Anwendungen, 2000, v. 19, No. 3, 655-676.

[11] Kravchenko V. *Applied quaternionic analysis. Maxwell's system and Dirac's equation*. World Scientific, Functional-analytic and complex methods, their interactions, and applications to partial differential equations, Ed. by W. Tutschke, 143-160, 2001.

[12] Kravchenko V. Quaternionic Reformulation of Maxwell Equations for Inhomogeneous Media and New Solutions. *Journal for Analysis and its Applications*, Vol. 21 (2002), No 1, 21-26.

[13] Kravchenko V. Applied Quaternionic Analysis Heldermann Verlag, Research and Exposition in mathematics Series, v. 28, 2003.

[14] Nagase T. Komata M.; Araki T. Secure signals transmission on quaternion encryption scheme. *Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA'04)*, 2004 IEEE.

[15] Nagase, T. et al. A new quadripartite public-key cryptosystem. *International Symposium on Communications and Information Technologies 2004 (ISCIT 2004)*, Sapporo, Japan, October 26-29, 2004.

[16] Sangwine S. J. *Fourier transforms of colour images using quaternion or hypercomplex numbers*. Elect. Lett., vol. 32, no. 21, p. 1979-1980, Oct. 1996.

La Comunicación Científica y Tecnológica: El Artículo

Alma Bertha León Mejía
Profesora de la UPIICSA-IPN
Fernando García Córdoba
Profesor del CIECAS-IPN.

La culminación de una investigación científica, y de una investigación de innovación y desarrollo tecnológico, es la publicación de sus resultados, con el fin de que puedan ser compartidos y contrastados por la comunidad científica y tecnológica correspondiente al ámbito de su competencia. Algunas personas consideran que la investigación termina cuando se obtienen los resultados y se presenta el informe final, o cuando la investigación se presenta en alguna reunión profesional. Sin embargo, la investigación termina realmente cuando se da a conocer mediante la publicación de una ponencia o de un artículo en una revista especializada. El investigador quizá es el único, entre todos los que desempeñan un oficio o profesión, que está obligado a presentar un informe escrito de lo que hizo, por qué lo hizo, cómo lo hizo y lo que aprendió al hacerlo.

Se le ha denominado artículo científico al texto referido, tanto a trabajos de carácter científico, como de innovación y desarrollo tecnológico. Ambos se sustentan en la investigación; sin embargo, se advierte una diferencia importante en cuanto a sus propósitos. En el artículo de carácter científico se presenta la información completa, clara y detallada, para que cualquier colega del área de la especialidad pueda probarla o demostrar el nuevo conocimiento, mientras que en la difusión del trabajo tecnológico puede haber un manejo restringido de la información, cuando se trata de algún desarrollo susceptible de patentarse.

El artículo es un escrito breve, cuyo propósito es dar a conocer los resultados de una investigación a fin de contribuir a planear, relacionar o descubrir cuestiones técnicas o profesionales como pauta para **investigaciones posteriores.** Por lo general, son avances (capítulos) de una investigación principal, o trabajos hechos específicamente para una presentación con fines de divulgación (congreso y simposio). Se publica en periódicos, semanarios, revistas especializadas, memorias, enciclopedias, páginas Web, CD, etcétera.

Es necesario aclarar que los requisitos exigidos por las revistas varían mucho según las disciplinas, e incluso dentro de una misma disciplina, por lo que no es posible entregar pautas ni hacer recomendaciones que sean universalmente seguidas. Por lo tanto, se tratarán algunos de los principios básicos aceptados en la mayoría de las disciplinas.

El artículo científico tiene una estructura formal y rígida compuesta, generalmente, por estas siete secciones:

1 . Resumen
2. Introducción
3. Desarrollo
4. Resultados
5. Discusión
6. Conclusiones
7. Referencias

1. **Resumen** (*Abstract*). Se presenta en forma condensada el contenido del artículo. El abstract, escrito en inglés, se incluye cuando las publicaciones tienen una cobertura internacional.

Un buen resumen debe permitir al lector identificar, en forma rápida y precisa, el contenido básico del trabajo; no debe tener más de 250 palabras y debe redactarse en pasado, exceptuando el último párrafo o frase concluyente. El resumen no debe aportar información o conclusión que no está presente en el texto, así como tampoco debe citar referencias bibliográficas. Como el resumen es lo primero que el editor y el jurado leen, es muy importante que sea escrito en forma simple y clara.

En general, el resumen debe:

- Plantear los principales objetivos y el alcance de la investigación.
- Describir la metodología empleada.
- Sintetizar los resultados.

- Finalizar con las conclusiones más importantes.

2. **Introducción.** Se expone el propósito y la importancia del trabajo; se limita el tema y se definen los aspectos que se tratarán. El objetivo de esta sección introductoria es presentar muy claramente, la naturaleza y relevancia del problema investigado. Si el problema no se presenta clara y razonadamente, el lector no se interesará por la solución que se propone. Debe indicarse el método usado en la investigación del problema y las razones para su elección. El lector deberá entender el problema y cómo se intentó resolverlo. También deberán incluirse los principales resultados. No es conveniente mantener al lector en suspenso; algunos autores comenten el error de guardar sus resultados más importantes para el final, o en algunos casos omiten mencionarlos en el resumen.

3. **Desarrollo.** Descripción del proceso del diseño y construcción (método y materiales). Se explica cómo se realizó la investigación

4. **Resultados.** Presenta los datos experimentales.

5. **Discusión.** Explica los resultados y los compara con el conocimiento previo del tema. Se presentan los principios, relaciones y generalizaciones demostrados por los resultados. Se deberá mostrar cómo los resultados e interpretaciones están en acuerdo o desacuerdo con trabajos publicados anteriormente (incluir las referencias). Para ello no hará falta abarcar toda la verdad del universo; bastará con que un punto de un área específica de esa gran verdad se aclare.

6. **Conclusiones.** Son las consideraciones finales, donde se resumen y reafirman las aportaciones derivadas de la investigación.

7. **Referencias.** Se enumeran las referencias bibliográficas, hemerográficas, así como las direcciones de las páginas de la Internet, consultadas o citadas en el artículo.

En ocasiones, después del resumen se incluye una lista de palabras clave, que identifican el material a tratar, y que permiten la búsqueda automatizada en bancos de datos computarizados (por Internet, por ejemplo) por tema, o especialidad. También, si el caso lo amerita, es conveniente incluir al final un glosario de algunos términos específicos, para precisar su significado, y facilitar al lector la comprensión del texto.

En el lenguaje científico y técnico los vocablos especializados son absolutamente insustituibles y no pueden ser retirados del texto para colocar otros que actúen como sinónimos o casi sinónimos. El lenguaje especializado exige un significante propio para cada significado; un texto científico donde cada noción especializada no tuviera una palabra (un significante) propia, sería necesariamente un texto confuso. Sólo los especialistas pueden distinguir con precisión los términos propios de su ciencia, ya que frecuentemente éstos tienen la forma de una palabra del léxico general, pero en el texto científico o técnico tienen un significado unívoco para su empleo especializado. Quien pretenda interpretar el sentido de las voces propias de un campo especializado, sin ser especialista, caerá en una confusión total, pues cometerá el error de tratar esos términos como si fueran palabras de la lengua general, y la realidad es que no tienen que ver con ellos.

El desarrollo de las ciencias y de la tecnología va creando una continuada necesidad de crear neologismos. Hay que dotar de nombre a lo que se va inventando y descubriendo, y lo lógico es que eso lo hagan los mismos que inventan o descubren, y normalmente esto ocurre en ambientes de lengua inglesa. Ante el dilema de intentar traducirlos o crear un neologismo, la mayor parte de las veces se opta por una tercera vía, la más cómoda: usar la palabra en su idioma original. Sin embargo, se tendrá que ser cuidadoso en la defensa del buen uso de la lengua, en nuestro caso, del español.

En la Guía para la Redacción de Artículos Científicos publicados por la UNESCO, se señala que la finalidad esencial de un artículo científico es comunicar los resultados de investigaciones, ideas y debates de una manera clara, concisa y fidedigna. Es por ello que para escribir un buen artículo científico hay que aprender y aplicar los tres principios fundamentales de la redacción científica: precisión, claridad y brevedad.

1. **Precisión**. Consiste en usar las palabras y expresiones que correspondan exactamente, sin ambigüedad, al significado que se quiera dar. El lenguaje científico debe ser objetivo, expresar sólo aquello que se relacione con el objeto de estudio. Una de las características de estos textos es precisamente el empleo de tecnicismos, que designan a ciertos objetos o fenómenos, cuya significación requiere de palabras específicas, de uso restringido, propio de campos especializados.

2. **Claridad.** El texto se deberá entender fácilmente. Esto se logra con un lenguaje sencillo, oraciones bien construidas y cuando los párrafos desarrollan el tema con un orden lógico y consistente.

3. **Brevedad**. Significa incluir únicamente la información directamente pertinente al contenido del artículo, evitar los rodeos inútiles, y comunicar dicha información usando el menor número posible de palabras. Cualquier texto innecesario atenta contra la claridad del mensaje, y ocasiona que las ideas importantes se diluyan.

Para facilitar la redacción del artículo se sugiere la elaboración previa de un esquema de contenido, para cada una de las partes que lo integrarán, donde se establezcan las ideas principales y complementarias, y que facilite la estructuración de un desarrollo coherente de lo que se quiere expresar. Posteriormente se desarrollarán las ideas, para conformar un borrador; en esta fase se recomienda escribir como vayan fluyendo las ideas y no detenerse en buscar en ese momento la mejor expresión, pues se corre el riesgo de interrumpir la continuidad del pensamiento. Finalmente, en la revisión y corrección se pondrá especial atención en la calidad expresiva del texto. Se recomienda considerar lo siguiente:

1. Cuidar la claridad. Medir y pesar el empleo de cada palabra; evitar las expresiones que puedan resultar oscuras o ambiguas para el lector.

2. Suprimir la redundancia léxica e ideológica (repetición inútil).

3. Evitar el empleo del «cosismo» (abuso de la palabra cosa). El «queísmo» (repetición de «que»), el «gerundismo» (abusivo empleo de gerundios) y la «extranjerización» inútil o defectuosa.

4. Usar los nexos («luego», «pues», «por otro lado», «ahora bien», etc.) con meditada adecuación, no para llenar espacio. En este tipo de expresiones, diferenciar entre la causa («a causa de ello») y el efecto («como consecuencia de lo anterior»).

5. Utilizar adecuadamente los signos de puntuación.

6. Trate de evitar el uso de anglicismos (vocablos o giros ingleses en distintos idiomas). Ciertos usos gramaticales son más propios del inglés que del español.

Ejemplos:

- Omitir el artículo al principio de la oración («Análisis de los datos sugiere...», en vez de «El análisis de los datos sugiere...»)

- Usar la voz pasiva en vez de la voz activa (« ... fueron analizados.», en vez de «...se analizaron.»)

- Colocar el adjetivo antes del nombre («...lento proceso...», en vez de «...proceso lento...»)

- Colocar el adverbio antes del verbo («.. rápidamente procesando...», en vez de «...procesando rápidamente...»).

- Usar el gerundio excesivamente (las formas verbales que terminan en -ando o en -iendo).

La **Tabla 1** presenta algunos ejemplos de expresiones corregidas.

Formas Originales	Formas Corregidas
Ese resultaba ser un sistema	Ese resultaba un sistema
Sería, sin embargo, erróneo	Sin embargo, sería erróneo
Se hace el propósito de estudiar	Se propone estudiar
Esto quiere decir que	Eso significaba que
Tenía una idea aproximada del tema	Tenía un concepto vago del tema
Quisiéramos explicar a nuestros lectores	Explicaremos a los lectores
Pueden ocurrir cambios irreversibles destruyendo la estructura química	Pueden ocurrir cambios irreversibles que destruyan la estructura química
Entonces pues, como expresamos el camino correcto a seguir	Entonces -como ya expresamos- el adecuado camino por seguir
A base de tan sólo dos	Sólo sobre la base de dos (o con base en sólo dos...)
Quiero que conozcan la página que ha sido criticada por los que no saben	Quiero que conozcan la página criticada por quienes no saben
Eso fue, de hecho, un modelo que en muchos aspectos fue seguido	Este modelo, en muchos aspectos fue seguido
Tuvo gran impacto, por lo que se atacaban unos a otros	Hizo gran impacto, por lo que se atacaban mutuamente
Debemos cumplir con ciertos requisitos para tratar el tema en cuestión	Debemos cumplir ciertos requisitos para tratar el tema establecido

Tabla 1. Ejemplos de expresiones corregidas.

Conclusiones

El artículo científico es una comunicación formal, cuyos objetivos están relacionados directamente con la difusión de resultados de investigaciones científicas y tecnológicas, propuestas de ideas o modificaciones en el seno de una comunidad afectada. El propósito de la difusión es contribuir al desarrollo del conocimiento científico y de las innovaciones tecnológicas *(Know how).*

El investigador, autor de un artículo, tendrá que considerar no solamente la calidad del contenido, sino también la calidad de la expresión del mismo. Podrá tratarse de un trabajo de gran trascendencia, pero si está mal escrito perderá credibilidad. Para escribir un artículo científico es necesario entender y aplicar los requisitos fundamentales de la redacción científica: precisión, claridad y brevedad.

Bibliografía

[1] Basulto, Hilda. *Curso de redacción dinámica.* México: Trillas, 1986.

[2] García Córdoba, Fernando. *La investigación tecnológica, Investigar, idear e innovar en ingenierías y ciencias sociales*. México: LIMUSA, 2005.

[3] León Mejía, Alma Bertha. *Estrategias para el desarrollo de la comunicación profesional,* México: LIMUSA, 2005.

Reconocimiento de Voz en Español Mediante Sílabas

Dr. Sergio Suarez Guerra
Profesor del CIC-IPN
Dr. en C. José Luis Oropeza Rodríguez
Profesor del CIDETEC-IPN

Actualmente, el uso de los fonemas tiene implícitas varias dificultades, debido a que la identificación de las fronteras entre ellos por lo regular es difícil de encontrar en representaciones acústicas de voz. El presente trabajo plantea una alternativa a la forma en la que el reconocimiento de voz se ha estado implementando desde hace tiempo, analizando la forma en la cual el paradigma de la sílaba responde a tal labor dentro del español. Durante los experimentos realizados se examinaron para la tarea de segmentación tres elementos esenciales: a) la Función de Energía Total en Corto Tiempo, b) la Función de Energía de Altas Frecuencias Cepstrales (conocida como Energía del Parámetro RO) y, c) un Sistema Basado en Conocimiento. Tanto el Sistema Basado en Conocimiento como la Función de Energía Total en Corto Tiempo se usaron en un corpus de dígitos en donde los resultados alcanzados usando sólo la Función de Energía, fueron de 90.58%. Cuando se utilizaron los parámetros Función de Energía Total en Corto Tiempo y la Energía del Parámetro RO, se obtuvo un 94.70% de razón de reconocimiento, lo cual causa un incremento del 5% con relación al uso de palabras completas en un corpus de voz dependiente del contexto. Por otro lado, cuando se utilizó un corpus de laboratorio del habla continua, al usar la Función de Energía Total en Corto Tiempo y el Sistema Basado en Conocimiento se alcanzó un 78.5% de razón de reconocimiento y un 80.5% de reconocimiento al usar los tres parámetros anteriores. El modelo del lenguaje utilizado para este caso fue el bigram, y se emplearon Cadenas Ocultas de Markov de densidad continua con tres y cinco estados, con 3 mixturas Gaussianas por estado.

Introducción

Un Sistema de Reconocimiento Automático del Habla (SRAH) es aquel sistema automático capaz de gestionar la señal de voz emitida por un individuo. Dicha señal ha pasado por un proceso de digitalización para obtener elementos de medición (muestras), las cuales permiten denotar su comportamiento e implementar procesos de tratamiento de la señal, enfocados al reconocimiento.

Bajo este esquema, la señal de voz se ve inmersa en dos bloques importantes: entrenamiento y reconocimiento. El entrenamiento es una de las etapas críticas dentro de estos sistemas, y gran parte del éxito de un sistema de reconocimiento de voz recae en esta parte. Como un referente esencial de lo anterior, el presente trabajo presenta la incrustación de un bloque destinado al refuerzo de la obtención de los datos a ser procesados, usando de un Sistema Basado en Conocimiento (SBC, al cual se le denominará también Sistema Experto), capaz de realizar la clasificación de la señal de entrada en unidades silábicas, por medio de la aplicación de un conjunto de reglas lingüísticas del idioma español.

La razón por la cual se pensó en un Sistema Basado en Conocimiento es debido a que en el español, en contraparte del inglés, por ejemplo, la forma en la que se escriben los textos y en la que se lee son muy semejantes; esto se debe a que el español es altamente dependiente del contexto y de la prosodia. Los elementos anteriores justifican la aplicación del experto en esta parte del sistema.

La etapa de entrenamiento puede llevarse a cabo por varios métodos, entre los que destacan:

- Bancos de Filtros.
- Codificación Predictiva Lineal.
- Modelos Ocultos de Markov.
- Redes Neuronales Artificiales.
- Lógica Difusa.
- Sistema de reconocimiento híbrido, etc.

Se han hecho análisis desde fonemas hasta la palabra misma. Esto ha dado origen a una gran cantidad de resultados e implementación de técnicas relacionadas. El presente trabajo se enfoca al área de la sílaba y se analiza su alta sensibilidad al contexto.

a) En una señal de voz, la sílaba es una estructura independiente para cualquier idioma que se ponga de ejemplo, pues no es posible encontrar errores de coarticulación en su estructura interna, como sucede en el caso de los fonemas. Considere la sílaba {*pla*} de las palabras plazo y plato, si no se realiza una división de sus elementos fonéticos y se estudian sus características, se concluye que es exactamente igual en cualquier caso, aunque se use en dos palabras distintas.

Ahora bien, considere al fonema /f/ de las palabras foca, y fofa; en el primer y segundo caso al fonema /f/ le prosiguen dos fonemas totalmente diferentes /o/ y /a/ de las sílabas {fo} y {fa}. Aquí, el problema es que el fonema pierde sus características propias al tener adyacentes dos fonemas totalmente diferentes. A esto se le conoce como el problema de la coarticulación y es la fuente de las grandes dificultades que manifiestan los sistemas de reconocimiento actuales.

b) La sílaba en el caso del español, al contener cierta semejanza entre la forma en que se pronuncia y la que se escribe, puede establecerse como elemento primordial de un SRAH.

c) La separación automática de estructuras fonéticas sigue siendo un problema que no se ha podido resolver. Tal es el caso de que un sistema de reconocimiento de voz que basa sus principios en este esquema, tiene que realizarla en ocasiones, de manera semiautomática para poder incrementar las tasas de reconocimiento. Obviamente, esto no quiere decir que en la sílaba no pueda suceder, pero sus propias características intrínsecas pueden permitir una mejora en los esquemas de segmentación.

Las razones expuestas en los incisos anteriores y las que se presentan en (Wu 1998) son un punto de apoyo en la elaboración de este trabajo.

Este artículo se encuentra dividido en 6 partes, las cuales tienen la siguiente estructura:

a) La parte 2 presenta el conjunto de historia y antecedentes de los sistemas de reconocimiento de voz.
b) La parte 3 presenta las metodologías de evaluación.
c) En la sección 4 se muestran los resultados alcanzados con la metodología propuesta.
d) La sección 5 presenta las conclusiones, y en la 6 las referencias bibliográficas.

Reconocimiento de voz por computadora

Historia

Se considera que el reconocimiento de voz por computadora es una tarea muy compleja, debido a todos sus requerimientos implícitos (Suárez, 2005). Además del alto orden de los conocimientos que en ella se conjugan, deben tenerse nociones de los factores inmersos que propician un evento de análisis individual (estados de ánimo, salud, etc.). Por tanto, en los SARH, ya sea para tareas específicas o generales, la cantidad de aspectos a tener en cuenta es muy grande. La historia esencial de los sistemas de reconocimiento de voz se puede resumir con las siguientes premisas (http://www.gtc.cps.unizar.es):

♦ **Los inicios: años 50's**

- Bell Labs. Reconocimiento de dígitos aislados monolocutor.
- RCA Labs. Reconocimiento de 10 sílabas monolocutor.
- University College England. Reconocedor fonético.
- MIT Lincoln Lab. Reconocedor de vocales independiente del hablante.

♦ **Los fundamentos: años 60's** – Comienzo en Japón (NEC labs)

- Dynamic Time Warping (DTW – Alineación Dinámica en Tiempo -). Vintsyuk (Soviet Union).
- CMU (Carnegie Mellon University). Reconocimiento del Habla Continua. HAL 9000.

♦ **Las primeras soluciones: años 70's** - El mundo probabilístico.

- Reconocimiento de palabras aisladas.
- IBM: desarrollo de proyectos de reconocimiento de grandes vocabularios.
- Gran inversión en los EE. UU.: proyectos DARPA.
- Sistema HARPY (CMU), primer sistema con éxito.

♦ **Reconocimiento del Habla Continua: años 80's** - Expansión, algoritmos para el habla continua y grandes vocabularios.

- Explosión de los métodos estadísticos: Modelos Ocultos de Markov.
- Introducción de las redes neuronales en el reconocimiento de voz.
- Sistema SPHINX.

♦ **Empieza el negocio: años 90's.**

- Primeras aplicaciones: ordenadores y procesadores baratos y rápidos.
- Sistemas de dictado.
- Integración entre reconocimiento de voz y procesamiento del lenguaje natural.

♦ **Una realidad: años 00's**- Integración en el Sistema Operativo.

- Integración de aplicaciones por teléfono y sitios de Internet dedicados a la gestión de reconocimiento de voz (Voice Web Browsers).
- Aparece el estándar VoiceXML.

Generalidades

El reconocimiento de voz por computadora es una tarea compleja de reconocimiento de patrones y de los sistemas biométricos. Por lo regular, la señal de voz se muestrea en un rango entre los 8 y 16 KHz. En el reporte de los experimentos de este trabajo, la frecuencia de muestreo utilizada fue de 11025 Hz. La señal de voz necesita ser analizada para extraer información relevante una vez que ha sido digitalizada.

A manera de resumen, dentro de esta tarea, se aplican las siguientes técnicas de extracción de parámetros característicos de la señal (Kirschning, 1998), (Jackson, 1986), (Kosko, 1992) y (Sydral et al., 1995):

- Análisis de Fourier.
- Codificación Predictiva Lineal.
- Análisis de los Coeficientes Cepstrales.
- Predicción Lineal Perceptiva.

Una de las características esenciales a definir en el proceso de captura de la señal de voz, es la frecuencia de muestreo; este factor es muy importante, pues es la limitante y posible causante de diferenciar entre una buena calidad de señal y los problemas que se pueden presentar si no se respetan las reglas que el procesamiento digital de señales enmarca. Específicamente hablando, y suponiendo que el problema anterior se ha resuelto, quedan aún muchos factores a analizar, dentro de los cuales se encuentran los siguientes (Kirschning, 1998):

- Tamaño del vocabulario y confusión
- Sistemas, tanto dependientes como independientes del locutor
- Voz aislada, discontinua y continua.
- Voz aplicada a tareas o en general
- Voz leída o espontánea
- Condiciones adversas

Metodologías de evaluación

Las reglas de la sílaba

En (Feal, 2000), se menciona que en el español existen 27 letras, las cuales están clasificadas de acuerdo a su pronunciación en dos grupos: vocales y consonantes. El grupo de las vocales está formado por cinco, 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 y cerradas (Oropeza, 2000) y (Rabiner and Biing-Hwang, 1993).

El otro grupo de letras, las consonantes, está formado por veintidós letras, con las cuales se forman tres consonantes compuestas, llamadas así por ser letras simples en su pronunciación y dobles en su escritura; las letras restantes son llamadas consonantes simples, por ser simples en su pronunciación y en su escritura. En el idioma español existen diez reglas, las cuales determinan la separación de las sílabas de una palabra. Estas reglas se listan a continuación mostrando además excepciones a las mismas.

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

Figura 1. Representación de consonantes inseparables.

REGLA 1. En las sílabas, por lo menos, siempre tiene que haber una vocal. Sin vocal no hay sílaba.

Excepción. Esta regla no se cumple cuando se presenta la "y".

REGLA 2. Cada elemento del grupo de consonantes inseparables, mostrado en la **Figura 1**, no puede ser separado al dividir una palabra en sílabas.

REGLA 3. Cuando una consonante se encuentra entre dos vocales, se une a la segunda vocal.

REGLA 4. Cuando hay dos consonantes entre dos vocales, cada vocal se une a una consonante.

Excepción: Esto no ocurre en el grupo de consonantes inseparables (regla 2).

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.

Excepción: Esta regla no se cumple cuando la segunda y tercera consonante forma parte del grupo de consonantes inseparables.

REGLA 6. Las palabras que contienen una h precedida o seguida de otra consonante, se dividen separando ambas letras.

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.

REGLA 8. La h entre dos vocales, no destruye un diptongo.

REGLA 9. La acentuación sobre la vocal cerrada de un diptongo provoca su destrucción.

REGLA 10. La unión de tres vocales forma un triptongo.

Los SRAH actuales son implementados usando el fonema como parte fundamental, gran parte de ellos se encuentran basados en los Hidden Markov Models (HMM), que se concatenan como se muestra en la **Figura 2** (Savage, 1995) para obtener palabras ó frases.

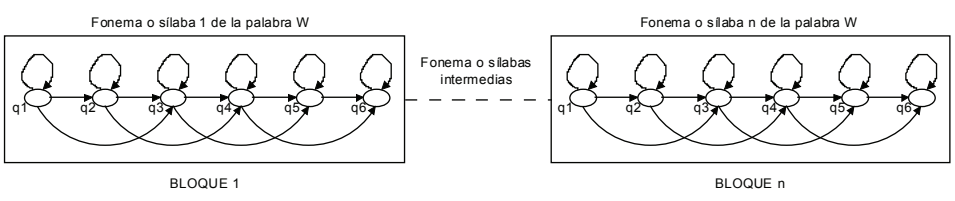


Figura 2. Esquema de representación de Cadenas Ocultas de Markov concatenadas para conformar palabras o frases continúas.

Los tres problemas básicos de las Cadenas Ocultas de Markov (COM)

Por conveniencia, usamos la notación compacta:

$$\lambda = (A, B, \pi) \qquad [1]$$

para indicar el conjunto de parámetros completos del modelo. Este conjunto de parámetros, por supuesto, define una medida de probabilidad para O, por ejemplo, $P(O \mid \lambda)$.

Los Tres Problemas Básicos de las COM

Dados los datos de la HMM anterior, los tres problemas básicos que deben ser resueltos por el modelo son los siguientes:

Problema 1

Dada una secuencia de observación $O = (O_1, O_2, ..., O_T)$, y un modelo $\lambda = (A, B, \pi)$, ¿cómo podemos calcular eficientemente $P(O|\lambda)$, la probabilidad de la secuencia de observación producida por el modelo?

Problema 2

Dada la secuencia de observación $O = (O_1, O_2, ..., O_T)$, y el modelo λ, ¿cómo seleccionamos una secuencia de estados correspondiente $Q = (q_1, q_2, ..., q_T)$ que sea óptima en algún sentido?

Problema 3

¿Cómo ajustar los modelos de los parámetros *l=(A, B, p)* para maximizar *P(O½l)?*. En este problema, intentamos optimizar los parámetros del modelo para describir de mejor forma como se construye una secuencia de observación dada. La secuencia de observación usada para ajustar los parámetros del modelo, se denomina secuencia de entrenamiento porque es usada para entrenar el HMM. El problema del entrenamiento es una de las aplicaciones más cruciales para el HMM, ya que nos permite adaptar de manera óptima los parámetros del modelo, que son observados durante el entrenamiento de datos.

Solución al problema 1

Procedimiento hacia adelante

Considere la variable regresiva $\alpha_t(i)$, definida como se muestra en la ecuación:

$$\alpha_t(i) = P(O_1 O_2 \ldots O_t, q_t = S_i | \lambda) \qquad [2]$$

Esto es, la probabilidad de la secuencia de observación parcial, $O_1 O_2 \ldots O_t$, (mientras el tiempo *t*) y analizada en el estado i, dado el modelo l. En (Zhang, 1999), se menciona que podemos resolver $\alpha_t(i)$ inductivamente, como se muestra en las siguientes ecuaciones:

1. Inicialización

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N \qquad [3]$$

2. Inducción

$$\alpha_{t+1}(j) = b_j(O_{t+1}) \sum_{i=1}^{N} \alpha_t(i) a_j \quad 1 \le j \le N, \quad 1 \le t \le T\text{-}1 \qquad [4]$$

3. Terminación

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad [5]$$

Solución al problema 2

El *algoritmo de Viterbi* encuentra la mejor de las secuencias de estados, *Q={q1, q2, ..., qT}*, para una secuencia de observaciones dada *O={O1, O2, ..., OT)*, como se muestra a continuación:

1. Inicialización

$$\delta_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N \qquad [6]$$

2. Recursión

$$\delta_{t+1}(j) = b_j(O_{t+1}) \left[\max_{1 \le i \le N} \delta_t(i) a_{ij} \right]$$

$$1 \le j \le N, \quad 1 \le t \le T\text{-}1 \qquad [7]$$

3. Terminación

$$p^* = \max[\delta_T(i)] \qquad 1 \le i \le N \qquad [8]$$

$$q^* = \arg\max[\delta_T(i)] \qquad 1 \le i \le N \qquad [9]$$

Solución al problema 3

Para poder dar solución al problema 3, se hace uso del *algoritmo de Baum-Welch*, que al igual que los anteriores realiza por medio de inducción la determinación de valores que optimicen las probabilidades de transición en la malla de posibles cambios de los estados del modelo de Markov. Con el análisis anterior, Baum-Welch logró obtener la siguiente expresión para la implementación de su algoritmo; dicha ecuación, nos permite determinar el número de transiciones del estado s_i al estado s_j.

$$\varepsilon_t(i,j) = \frac{\alpha_t(i) a_{ij} \beta_{t+1}(j) b_j(o_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} \beta_{t+1}(j) b_j(o_{t+1})} \qquad [10]$$

Una manera eficiente de optimizar los valores de las matrices de transición, en el algoritmo de Baum-Welch, es de la forma siguiente (Oropeza, 2000) y (Rabiner and Biing-Hwang, 1993):

$$a_{ij} = \frac{\text{número esperado de transiciones del estado } s_i \text{ al estado } s_j}{\text{número esperado de transiciones del estado } s_i} \qquad [11]$$

$$b_j(k) = \frac{\text{número esperado de veces que estando en j aparece el símbolo } v_k}{\text{número esperado de veces que se analiza el estado j}} \qquad [12]$$

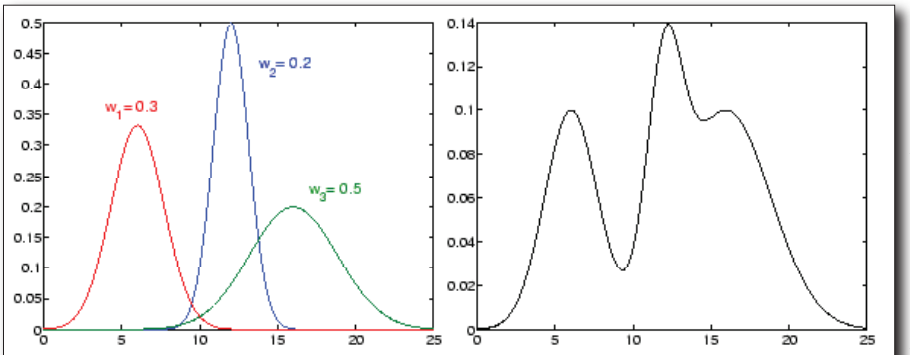


Figura 3. Representación esquemática de una tripleta de mixturas Gaussianas.

Los parámetros de la función de densidad de probabilidad (PDF, *Probability Density Function*), son el número de Gaussianas, sus factores de peso, y los parámetros de cada función de Gaussiana tales como la media μ y la matriz de covarianza Σ. Para encontrar estos parámetros, que de alguna forma describen a una determinada función de probabilidad de un conjunto de datos, se utiliza el algoritmo iterativo de máxima esperanza (*EM*).

Mixturas de Gaussianas

Las mixturas Gaussianas, son combinaciones de distribuciones normales o funciones de Gauss. Una mixtura de k Gaussianas puede ser vista como una suma de densidades de Gaussianas (Resch, 2001a), (Resch, 2001b), (Kamakshi et al., 2002) y (Mermelstein, 1975).

Como es sabido, la función de densidad de probabilidad del tipo Gaussiano es de la forma:

$$g(\mu,\Sigma)(x) = \frac{1}{\sqrt{2\pi^{d}}\sqrt{\det(\Sigma)}} e^{-\frac{1}{2}(x-\mu)^{T}\Sigma^{-1}(x-\mu)} \qquad [13]$$

Una mixtura Gaussianas, con ciertos valores de peso se ve de la forma:

$$gm(x) = \sum_{k=1}^{K} w_k * g(\mu_k, \Sigma_k)(x) \qquad [14]$$

en donde los pesos son todos positivos y la suma de los mismos es igual a 1:

$$\sum_{i=1}^{K} w_i = 1 \quad \forall \quad i \in \{1,.....,K\} \quad : w_i \geq 0 \qquad [15]$$

En la **Figura 3**, se muestra un ejemplo de una mixtura Gaussiana, que consiste de tres Gaussianas sencillas.

Al variar el número de Gaussianas *K*, los pesos w_i, y los parámetros de cada una de las funciones de densidad μ y Σ, las mixturas Gaussianas pueden usarse para describir algunas Funciones de Densidad de Probabilidad Complejas (FDPC).

Implantación del Sistema Basado en Conocimientos

En nuestro caso, la base de conocimientos se encuentra constituida por todas las reglas de clasificación de sílabas del lenguaje español; la tarea es entonces, entender y poner en el lenguaje de programación apropiado tales reglas para cumplir de forma satisfactoria los requerimientos del sistema.

La implantación del Sistema Experto para el presente trabajo, tiene como entrada el conjunto de frases o palabras que conforman un vocabulario determinado a reconocer (corpus de voz). Tras aplicar las reglas pertinentes, la energía en corto tiempo de la señal y la energía en corto tiempo del parámetro RO, se procede a realizar la división en unidades silábicas de cada uno

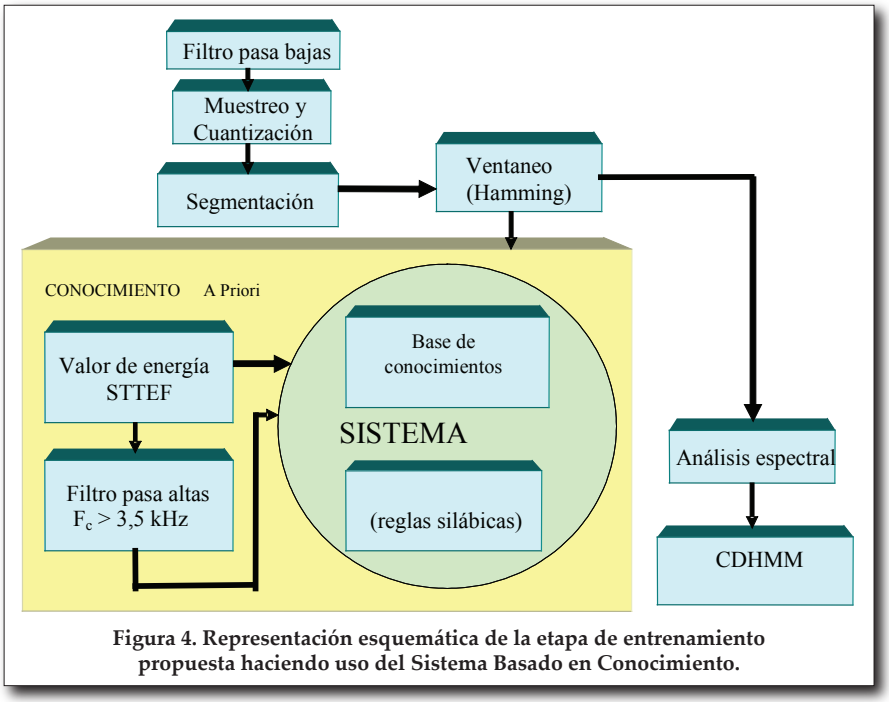


Figura 4. Representación esquemática de la etapa de entrenamiento propuesta haciendo uso del Sistema Basado en Conocimiento.

de los elementos de entrada, con lo cual se logran establecer los inicios y finales de las sílabas (Russell and Norvig, 1996) y (Giarratano and Riley, 2001).La incorporación de un experto a la fase de entrenamiento, para el caso propuesto, se puede resumir con el diagrama a bloques mostrado en la **Figura 4**, el cual demuestra la finalidad y función del mismo.

	TÉCNICA EMPLEADA	# DE UNIDADES DEL CORPUS	% DE RECONOCIMIENTO	% DE RECONOCIMIENTO ACUMULADO
V	LPC	5	98%	98%
VC-GI	LPC	5	96%	97%
VC-GII	LPC	9	96.66%	96.83%
CV-GI	LPC	5	96%	96.41%
CV-GII	LPC	5	100%	98.20%
CV-GIII	LPC	5	100%	99.10%
VC-GII	MARKOV	8	97.5%	98.30%

Tabla 1. Porcentajes de reconocimiento para el caso de sílabas aisladas.

Lo anterior cumple con el objetivo de que en el entrenamiento se extraigan la cantidad y tipos de sílabas que conforman el corpus a estudiar; asimismo, se provee la cantidad de bloques que serán usados en la etapa de etiquetado silábico de las señales de voz.

Los resultados obtenidos en la división silábica, al hacer uso de este sistema sobre sílabas independientes, frases de distintos corpus y textos escritos fue óptima. Uno de los puntos importantes al hacer uso de sílabas, es que existe gran preponderancia de los grupos V, CV, VC, CCV, CVC, sobre otras representaciones (VVV, CVVC, CVVVC, etc.). Una vez realizadas las grabaciones correspondientes, se creo un sistema de reconocimiento para el habla discontinua, para un determinado conjunto de sílabas, generando los resultados de reconocimiento mostrados en la **Tabla 1**. El proceso sigue la lógica de los árboles de inferencia mostrados en la **Figura 5.**

A continuación se muestran las características de cada uno de los grupos analizados:

GRUPO V (vocal). a, e, i, o, u.
GRUPO VC-GI. el, es, ir, os, un.
GRUPO VC-GII. al, am, el, em, es, in, ir, os, un.
GRUPO CV-GI. la, le, li, lo, lu.
GRUPO CV-GII. sa, se, si, so, su.
GRUPO CV-GIII. ba, be, bi, bo, bu.

Efecto del parámetro ERO en una señal de voz

Para poder incrementar la taza de reconocimiento, tanto a corpus de dígitos analizados y al corpus que se empleó al final, se recurrió a utilizar una variante, el parámetro RO (parámetro que permite obtener la respuesta en frecuencia de una señal de voz por encima de los 3,500 Hz), que se obtiene tras la aplicación de un filtro digital a la señal. Cabe hacer la aclaración que el parámetro RO ha sido utilizado en el programa para la extracción y análisis de parámetros de la voz EXPARAM 2.2; con número de registro 03-2004-052510360400-01, del Registro Público de Derechos de Autor a nombre del Dr. Sergio Suárez Guerra. En nuestro caso utilizamos el mismo algoritmo de la energía, pero aplicado a la señal de salida resultante del filtro digital, a lo que se ha denotado como la energía en corto tiempo del parámetro RO, ERO que tiene la representación matemática, mostrada en [16] y es una modificación a RO, lo que representa una contribución del presente trabajo:

$$ERO = \sum_{i=0}^{N-1} ROi^2 \qquad [16]$$

Dado lo anterior, se procedió a crear una nueva forma de segmentación automática, usando el parámetro ERO como artífice para ello.

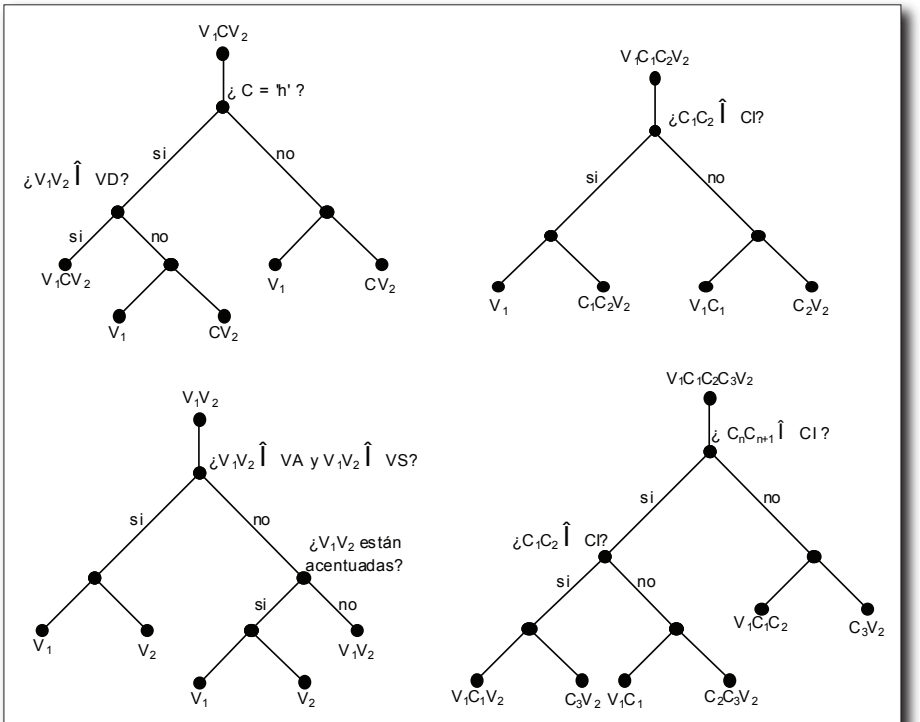


Figura 5. Árboles de inferencia inmersos dentro del Sistema Basado en Conocimiento.

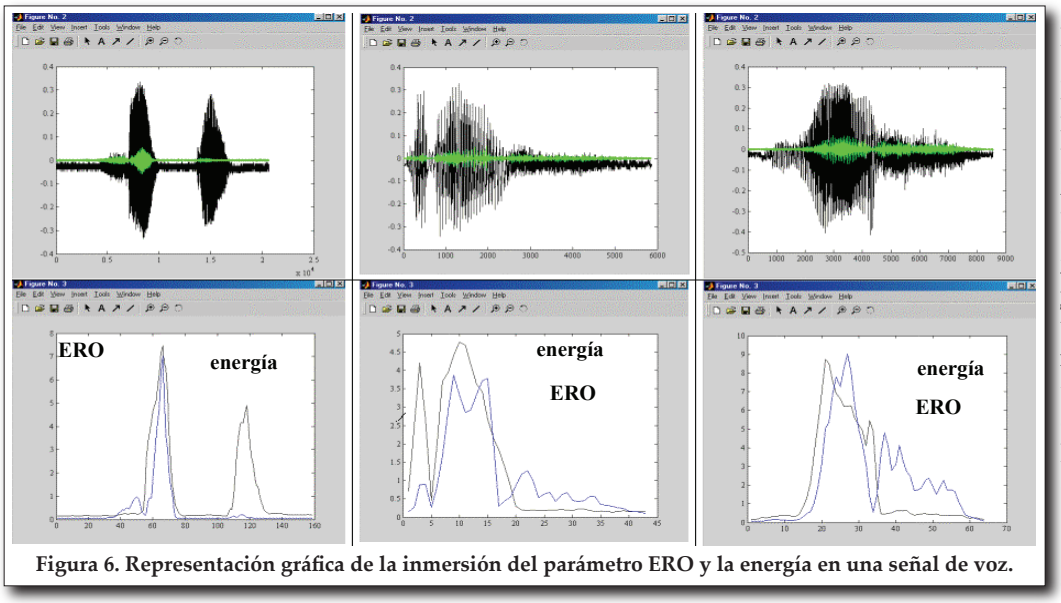


Figura 6. Representación gráfica de la inmersión del parámetro ERO y la energía en una señal de voz.

Los beneficios que conlleva este nuevo recurso radican básicamente en los siguientes puntos:

- Las palabras que comprendan una componente de alta energía se verán beneficiadas, pues el filtro está diseñado para dejarlas pasar; esto se puede observar en las señales de la **Figura 6.**
- Con ayuda del Sistema Experto se identifica el número de sílabas que conforman a las palabras del corpus; posteriormente se obtienen los parámetros de energía para ambos casos (aplicación o no del filtro), con lo cual se identifican tales elementos.
- Dado que el Sistema Experto analiza las componentes de los elementos del corpus, se puede deducir el número de sílabas y como están conformadas. Tales tareas se realizan de forma manual y automática para fines de comparación.
- El uso de estos dos parámetros conlleva a encontrar regiones de duración de las señales de voz, con y sin presencia de tales elementos.
- Con estos parámetros incrustados, se puede verificar que las señales de voz poseen *regiones de transición energía-parámetro ERO*.

La región de transición energía-parámetro ERO

Con los puntos anteriores, se obtiene una segmentación que toma la representación numérica y esquemática de la **Figura 7**. Las líneas de color negro representan las regiones de la señal de voz en donde la energía de la señal sin filtrar produce efectos; las líneas de color gris muestran las regiones en donde la señal de voz ya filtrada genera efectos, es decir, los puntos en donde las componentes de alta frecuencia se encuentran presentes; y, finalmente las líneas de color gris claro indican las *regiones de transición energía-parámetro RO*, que permiten ver las regiones en donde ninguno de los dos elementos tiene su aparición, pero que son necesarias para ligar la aparición o continuación de una sílaba.

Al usar los aspectos mencionados sobre un corpus de dígitos, se reporta un reconocimiento de sílaba individual del 94.7%, para el corpus de voz del habla continua, mientras que las figuras anteriores muestran la comparación entre reconcer con palabras completas y con la concatenación de las sílabas, la cual es más eficiente (91% y 96% respectivamente). El parámetro RO se usa para análisis de señales de voz en nuestro laboratorio; para fines de estudio de su aplicación en sistemas de reconocimiento de voz, se muestran los resultados analizados en el presente trabajo.

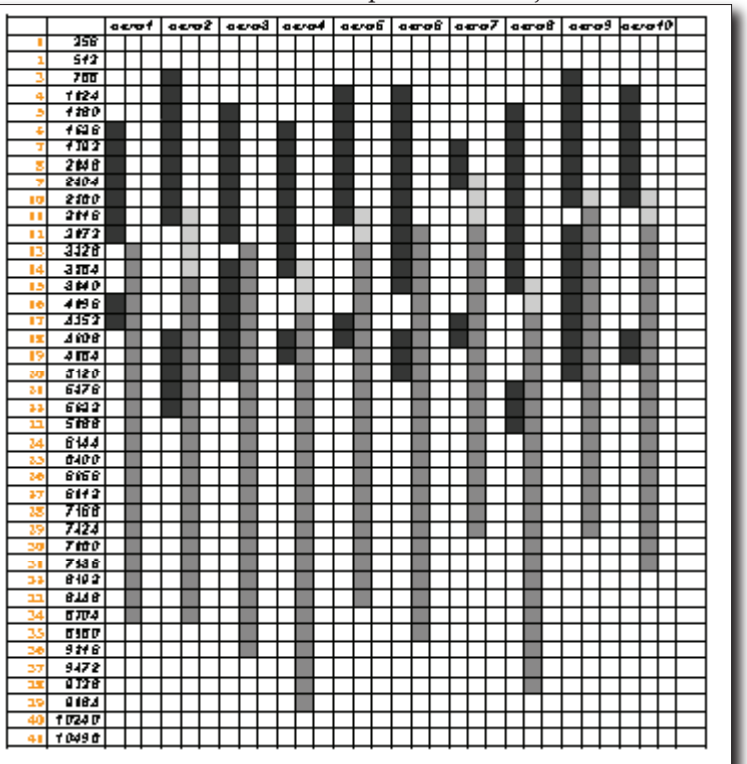


Figura 7. Regiones de manifestación de la energía, parámetro ERO y región de transición energía-parámetro ERO.

1	De Puebla a México
2	Cuauhtémoc y Cuautla
3	Cuautla Morelos
4	Espacio aéreo
5	Ahumado
6	Croacia está en Europa
7	Protozoarios biológicos
8	El trueque marítimo
9	Ella es seria
10	Sería posible desistir

Tabla 2. Corpus de voz sin confusión lingüística.

1	A la mejor ésta es el ala
2	El ala del ave está mejor
3	Alá es el mejor del mejor
4	A la mejor no está
5	Es mejor Alá
6	Es Alá el mejor
7	El ala no la cubre Alá
8	A la mejor, Alá está allá
9	El ala de Alá está allá
10	A la mejor es el ala de Alá

Tabla 3. Corpus de voz con confusión lingüística.

Con el fin de extender la aplicación anterior a un corpus de sílabas más extenso, se usaron las frases de las **Tabla 2** y **3** para el sistema de reconocimiento; la primera tabla muestra un corpus que carece de elementos que se denominan de confusión lingüística, y la segunda es un corpus con un alto índice de confusión.

Se utilizaron Mixturas Gaussianas, 3 de ellas para cada estado, y HMM con 5 y 3 estados; se usaron 12 coeficientes CLPC como componentes de observación, generándose los Modelos independientes por sílaba y realizándose la concatenación de las mismas utilizando las probabilidades obtenidas a través del modelo bi-

Segmentación	Modelos 3 estados	de Markov 5 estados
energía	89.5%	95.5%
ERO	95%	97.5%

Tabla 4. Porcentajes de reconocimiento obtenidos para el corpus de voz 1 usando habla discontinua.

Segmentación	Modelos 3 estados	de Markov 5 estados
energía	77.5%	75.5%
ERO	79%	80.5%

Tabla 5. Porcentajes de reconocimiento para el corpus de voz 1 usando habla continua.

gram del lenguaje, para comenzar con el entrenamiento global de la frase. Estos programas se ejecutaron sobre MATLAB, y los resultados obtenidos se muestran en la **Tabla 4** para el corpus final "1", y en la **Tabla 5** para el corpus final "2".

Se hicieron 20 repeticiones con 5 personas (3 hombres y 2 mujeres), de las cuales 50% se usaron para el entrenamiento y 50% para el reconocimiento. Los resultados de reconocimiento mostrados presentan el acumulado del análisis de las 1000 frases del experimento. Para las sílabas con un orden de aparición de "uno", como es el

Segmentación	Modelos 3 estados	de Markov 5 estados
energía	85.5%	86.3%
ERO	90%	90.8%

Tabla 6. Porcentajes de reconocimiento obtenidos para el corpus de voz 2 usando habla discontinua.

Segmentación	Modelos 3 estados	de Markov 5 estados
energía	65.5%	64.5%
ERO	72%	74.2%

Tabla 7. Porcentajes de reconocimiento para el corpus de voz 2 usando habla continua.

caso de "Pue", el número de muestras en el entrenamiento es de 100; sin embargo, las sílabas como "ti" presentaron complicaciones por el número de muestras tan corto, al ser distribuidas por los estados de la Mixtura Gaussiana. Para evitar esto, en el proceso de inicialización se agrego el doble de elementos. Los resultados para el corpus final "2" se muestran en las **Tablas 6 y 7**.

Para finalizar, se pretendió verificar el efecto que tiene considerar la acentuación de las palabras que conforman al corpus final "2"; el resultado generó un porcentaje de reconocimiento entre el 50 y 60%, lo cual implica que debe utilizarse un análisis de señal de voz que permita identificar la variación que provoca la acentuación. El pitch (Tono fundamental), la amplitud y filtros adaptivos pueden ser herramientas utilizadas para tratar de resolver este problema.

Conclusión

En este trabajo se ha demostrado que la incorporación de la sílaba en un sistema de reconocimiento de voz aplicado a corpus pequeños y medianos, genera buenos resultados en sistemas tanto del habla continua como discontinua, lo cual resulta prometedor para aplicaciones de gran robustez. El reconocimiento orientado en sílabas representa un paradigma diferente al orientado en fonemas, sobre todo cuando se aplica al idioma español. Dicho paradigma conduce a un rendimiento estable y sostenido; los experimentos demuestran tal hecho.

A manera de resumen se tiene que:

1) La introducción de un Sistema Basado en Conocimiento permite, por un lado, agregar conocimiento a priori a la etapa de segmentación, la cual es fundamental

en el esquema propuesto; y por otro, la inmersión de técnicas de Inteligencia Artificial a los procesos de reconocimiento de voz se hace cada vez más necesaria.

2) El esquema propuesto, representa una extensión al planteado por Furui, lo cual implica un aporte importante al área de investigación del reconocimiento de voz.

3) Uno de los puntos esenciales buscados por la comunidad científica dedicada al reconocimiento de voz por computadora, es el incremento en los índices de reconocimiento. La presente investigación basa sus principios en el hecho de que, para poder alcanzar un índice de reconocimiento alto, la etapa de segmentación debe de ser cuidadosamente guiada y realizada. La mayor parte de las investigaciones propuestas se fundamentan en ella; más aún, los resultados obtenidos demuestran que tal aseveración es prácticamente cierta.

La sílaba tiene muchas propiedades que son deseables para la computación vectorial: 1) los modelos basados en sílabas pueden ser conducidos a remover las ramificaciones durante la ejecución y, 2) los modelos basados en sílabas son una unidad de organización natural para reducir la computación redundante y definen el espacio de búsqueda. De la misma forma, aunque este trabajo no explora los beneficios de la programación paralela, algunas de sus conclusiones son aplicables al procesamiento concurrente, a saber, la combinación de información de múltiples cadenas de Markov es una operación obviamente concurrente. El decodificador de dos niveles de Fosler-Lussier, puede ser mapeado cuidadosamente en una máquina de procesador múltiple, dado que las probabilidades de diferentes palabras son calculadas independientemente. Si este es el caso, el uso de máquinas paralelas y concurrentes puede ser ampliamente ventajosa en la investigación del reconocimiento de voz; asimismo, la combinación de la metodología empleada en el trabajo, al unirse con la basada en fonemas, abre un campo de estudio relevante.

Un punto importante que puede incrementar el camino de la investigación, en lo que a la inmersión de las sílabas a los sistemas de reconocimiento se refiere, es el hecho de introducir un conjunto de filtros, que permitan determinar de manera adecuada las manifestaciones de fonemas de mayor ocurrencia en un corpus de voz que conforman a las sílabas; además, la particularidad de mejorar el problema de la entonación, logrará incrementar el alcance que la sílaba tiene dentro del idioma español. Finalmente, los trifonemas pueden ser analizados como unidades de reconocimiento y comparar los resultados que se obtengan con los expuestos en este trabajo, procurando establecer una alternativa de utilización de ambas unidades esenciales. Hay idiomas donde la sílaba es una muy buena alternativa, en otros no tanto, tal es el caso del idioma Español.

Referencias Bibliográficas

[1] Feal (2000). Feal L., "Sobre el uso de la sílaba como unidad de síntesis en el español", Informe Técnico, Departamento de Informática, Universidad de Valladolid, 2000.

[2] Fosler et al. (1999). Fosler-Lussier E., Greenberg S., Morgan N., "Incorporating Contextual Phonetics into Automatic Speech Recognition". XIV International Congress of Phonetic Sciences, pp. 611-614, San Francisco, 1999.

[3] Giarratano and Riley (2001). Giarratano Joseph y Riley Gary, International Thompson Editores, Sistemas expertos, principios y programación 2001.

[4] Hauenstein (1996). Hauenstein A., "The syllable Re-revisited", Technical Report, Siemens AG, Corporate Research and Development, München Alemania, 1996.

[5] Jackson (1986). Jackson L. B. «Digital Filters and Signal Processing». Kluwer Academic Publishers. University of Louisville, Department of Electrical and Computer Engineering, U.S.A., 1986

[6] Jones et al. (1999). Jones R., Downey S., Mason J., "Continuous Speech Recognition Using Syllables", Proceedings of Eurospeech, Vol. 3, pp. 1171-1174, Rhodes, Grecia 1999.

[7] Kamakshi et al. (2002). Kamakshi V. Prasad, Nagarajan T. and Murthy Hema A.. «Continuous Speech Recognition Using Automatically Segmented Data at Syllabic Units». Department of Computer Science and Engineering. Indian Institute of Technology, Madras, Chennai 600-036. 2002.

[8] Kirschning (1998). Kirschning Albers Ingrid, «Automatic Speech Recognition with the Parallel Cascade Neural Network», PhD Thesis, Tokyo Japan, March 1998.

[9] Kosko (1992). Kosko B., «Neural Networks for Signal Processing», Prentice Hall, U.S.A., 1992.

[10] Meneido et al. (1999). Meneido Hugo, Neto Joâo P. and Almeida Luís B., INESC-IST, «Syllable Onset Detection Applied to the Portuguese Language». Sixth European Conference on Speech Communication and Technology (EUROSPEECH'99) Budapest, Hunagry, September 5-9, 1999.

11] Meneido and Neto (2000). Meneido H., Neto J., "Combination of Acoustic Models in Continuous Speech Recognition Hybrid Systems". INESC, Rua Alves Redol, 9, 1000-029 Lisbon, Portugal, 2000.

12] Mermelstein (1975). Mermelstein Paul «Automatic Segmentation of Speech into Syllabic Units». Haskins Laboratories, New Haven, Connecticut 06510, pp. 880-883,58 (4), June 1975.

13] Oropeza (2000). Oropeza Rodríguez José Luis, "Reconocimiento de Comandos Verbales usando HMM". Tesis de maestría, Centro de Investigación en Computación, Noviembre 2000.

14] Rabiner and Biing-Hwang (1993). Lawrence Rabiner and Biing-Hwang Juang, "Fundamentals of Speech Recognition", Prentice Hall, 1993.

15] Resch (2001a). Resch Barbara. «Gaussian Statistics and Unsupervised Learning». A tutorial for the Course Computational Intelligence Signal Processing and Speech Communication Laboratory. www.igi.turgaz.at/lehre/CI, November 15, 2001.

16] Resch (2001b). Resch Barbara. «Hidden Markov Models». A Tutorial for the Course Computational Laboratory. Signal Processing and Speech Communication Laboratory. www.igi.turgaz.at/lehre/CI, November 15, 2001.

17] Russell and Norvig (1996). Russell Stuart and Norvig Peter, Inteligencia Artificial un enfoque moderno, Prentice Hall, 1996.

18] Savage (1995). Savage Carmona Jesus, "A Hybrid Systems with Symbolic AI and Statistical Methods for Speech Recognition". PhD Thesis, University of Washington, 1995.

19] Suárez (2005). Suárez Guerra Sergio, ¿100% de reconocimiento de voz?. Trabajo inédito, no publicado.

20] Sydral et al. (1995). Sydral A., Bennet R., Greenspan S., «Applied Speech Technology», Eds (1995). CRC Press, ISBN 0-8493-9456-2, U.S.A., 1995.

21] Weber (2000). Weber K., "Multiple Timescale Feature Combination Towards Robust Speech Recognition". Konferenz zur Verarbeitung natürlicher Sprache KOVENS2000, Ilmenau, Alemania, 2000.

22] Wu (1998). Wu, S., "Incorporating information from syllable-length time scales into automatic speech recognition", PhD Thesis, Berkeley University, 1998.

23] Wu et al. (1997). Wu S., Shire M., Greenberg S., Morgan N., "Integrating Syllable Boundary Information into Automatic Speech Recognition ". ICASSP-97, Vol. 1, Munich Germany, vol.2 pp. 987-990, 1997.

24] Zhang (1999). Zhang Jialu, "On the syllable structures of Chinese relating to speech recognition", Institute of Acoustics, Academia Sinica Beijing, China, 1999.

Sistemas de Detección de Intrusos (Ids), Seguridad en Internet

M. en C. Mauricio Olguín Carbajal
M. en C. Israel Rivera Zárate
Ing. Patricia Pérez Romero
Profesores del CIDETEC-IPN

El aumento y la gravedad de los ataques en la Internet, hacen que los sistemas de detección de intrusos sean una parte indispensable de la seguridad en las empresas, por lo que se tienen buenas razones comerciales y legales para establecer políticas de seguridad sólidas; por esta razón, es esencial instalar un Sistema de Detección de Intrusos (IDS).

Los sistemas de detección de intrusos no son precisamente nuevos, el primer trabajo sobre esta materia data de 1980; no obstante, este es uno de los campos con mayor auge desde hace algunos años dentro de la seguridad informática. La capacidad para detectar y responder ante los intentos de ataque contra los sistemas es realmente interesante; durante este tiempo, cientos de investigadores de todo el mundo han desarrollado, con mayor o menor éxito, sistemas de detección de todo tipo, desde simples procesadores de archivos históricos (*logs),* hasta complejos sistemas distribuidos, especialmente vigentes con el auge de las redes de computadoras en los últimos años.

Introducción

Se considera a un intruso como cualquier persona que intente interrumpir o hacer mal uso del sistema; los sistemas informáticos se apoyan en los Sistemas de Detección de Intrusos, para prepararse del mal manejo y del uso indebido de la información de una organización. Esta meta se logra recopilando la información de una gran variedad de fuentes del sistema y de la red, analizando la información que contribuye a los síntomas de los problemas de seguridad de la misma, y permitiendo que el usuario especifique las respuestas en tiempo real a las violaciones. Es muy importante que, ante estos posibles ataques, se pueda responder a ellos en tiempo real.

Desarrollo

En general hay 2 tipos de intrusiones

Intrusiones para mal uso.- Son ataques en puntos débiles conocidos que pueden ser detectados haciendo un análisis y revisión en la información de auditoría y reportes del sistema; por ejemplo, un intento de crear un archivo inválido, puede ser descubierto examinando los mensajes en los archivos históricos (bitácoras) que son resultado de las llamadas al sistema.

Intrusiones anómalas.- Se basa en la observación de desviaciones de los patrones de uso normales del sistema. Estas intrusiones son difíciles de detectar, ya que no hay patrones fijos y permanentes; se pueden usar métricas que calculan parámetros del sistema disponibles, tales como el promedio de carga del CPU, número de conexiones de la red por minuto, número de procesos por usuario y cualquier otro tipo de medida que describa un cierto consumo de recursos en un periodo definido; el inconveniente es que si estas métricas cambian, porque así lo exige el sistema por sobrecarga de trabajo, se interpretará erróneamente como una intrusión.

Los objetivos que debe tener un sistema de detección de intrusos son los siguientes:

1. Vigilar y analizar la actividad de los usuarios y del sistema.
2. Revisar las configuraciones del sistema y de las vulnerabilidades.
3. Evaluar la integridad de los archivos críticos del sistema.
4. Reconocimiento de los modelos de la actividad que reflejan ataques conocidos.
5. Análisis estadístico para los modelos anormales de la actividad, aunque esto no siempre es exitoso, debido a que los promedios pueden cambiar.

6. Búsqueda de rastros de intervención al sistema operativo, con el reconocimiento de las violaciones de la actividad del usuario respecto a la política establecida.

7. Vigilar el cumplimiento de políticas y procedimientos de seguridad en la organización.

La meta de un IDS es proporcionar una indicación de un ataque potencial o de uno verdadero; un ataque o una intrusión son acontecimientos transitorios, mientras que una vulnerabilidad representa una exposición, que lleva el potencial para un ataque o una intrusión. La diferencia entre un ataque y una vulnerabilidad es que un ataque existe en un momento determinado, mientras que una vulnerabilidad existe independientemente de la época de la observación. Esto nos conduce a categorizar varios tipos de IDS:

* IDS basado en red.
* IDS basado en anfitrión (host).
* IDS híbridos.
* Inspector de la Integridad del Archivo.
* Explorador de la vulnerabilidad de la red.
* Explorador de la vulnerabilidad del host.

Los tres primeros puntos son tipos de IDS, mientras que los tres siguientes son herramientas de detección de vulnerabilidad.

Los sistemas basados en red actúan como detectores, es decir, ellos observan el tráfico en las capas del TCP/IP y buscan modelos conocidos de ataque, como los que generalmente realizan los hackers; la mayoría de los sistemas basados en red pueden buscar solamente los modelos de abuso que se asemejan al estilo de estos ataques, y esos perfiles están cambiando constantemente. Los sistemas basados en red también son propensos a los positivos falsos; por ejemplo, si el servidor Web tiene sobrecarga de trabajo y no puede manejar todas las peticiones de conexión, los IDS quizá pueden detectar un ataque inexistente. Típicamente, un sistema basado en red no va a buscar otras actividades sospechosas, como podría ser que alguien de su entorno de trabajo intente tener acceso a los datos financieros de la compañía.

Los sistemas basados en anfitrión emplean diversos procedimientos para buscar a los malos usuarios; estos sistemas dependen, generalmente, de los registros del sistema operativo para detectar acontecimientos, y no pueden considerar ataques a la capa de red mientras que estos ocurren. En comparación con los IDS basados en red, estos sistemas obligan a definir lo que se considera como actividades ilícitas, y a traducir la política de la seguridad a reglas del IDS. Los IDS basados en anfitrión se pueden también configurar para buscar tipos específicos de ataques.

Los sistemas relacionados al anfitrión (host) se despliegan de la misma forma que los buscadores de los antivirus o las soluciones de administración de red: se instala algún tipo de agente en todos los servidores y se usa una estación de gestión para generar informes.

Los vendedores observaron las limitaciones que presentan los IDS de red y los basados en anfitrión, y han combinado ambos para mejorar sus capacidades; estos sistemas híbridos reunen las mejores características de los dos en una configuración del sensor del IDS; como ejemplos se tienen Real Secure de Internet Security Systems (ISS) y, CyberCop de Network Associates (NAI).

A continuación se desglosan los errores que puede tener el sistema:

* Los errores positivos falsos conducirán a los usuarios del IDS a no hacer caso de su salida, pues clasifican acciones legítimas como intrusiones. Las ocurrencias de este tipo de error se deben reducir al mínimo; si muchos positivos falsos se generan, los operadores no haran caso de la salida del sistema en un cierto plazo, lo que puede conducir a una intrusión real que es detectada, pero que no es considerada por los usuarios, esto es un ejemplo de lo que se mencionó anteriormente.

* Un error negativo falso ocurre cuando procede una acción, aunque sea una intrusión. Los errores negativos falsos son más serios que los errores positivos falsos, porque dan un sentido engañoso de la seguridad. Permitiendo que todas las acciones procedan, una acción sospechosa no será atraída a la atención del operador. El IDS ahora es un defecto, pues la seguridad del sistema es menor que la anterior a que el IDS fuera instalado.

* Los errores de cambio son más complejos: un intruso podría utilizar conocimientos sobre los mecanismos internos de un IDS para alterar su operación, permitiendo posiblemente que el comportamiento anómalo proceda; el intruso podría entonces violar las necesidades operacionales de la seguridad del sistema. Esto se puede descubrir por un operador humano que examina los registros del detector de la intrusión, pero parecería que el IDS aún trabaja correctamente.

Otro aspecto a considerar al adquirir un sistema de detección de intrusos es que a veces se invierte en un sistema de detección de intrusos difícil de soportar, ya que reporta demasiados falsos positivos, y no puede

responder con la velocidad de la red. Para reducir las posibilidades de que esto suceda, las empresas deben tomar en cuenta los siguientes criterios al evaluar un sistema de detección de intrusos IDS:

- Un Sistema de detección de intrusos debe ir más allá de la simple notificación, proporcionando respuestas automatizadas, basadas en políticas para proteger los sistemas y ofreciendo tiempo y tranquilidad al personal de seguridad. Cuando es necesario localizar el origen de un ataque (frecuentemente se ataca con una dirección falsa), el enfoque tradicional ha sido interrogar manualmente a los enrutadores y encontrar el flujo relevante de los datos; este es un ejercicio agotador que puede llevar muchas horas o días, incluso para un ingeniero de redes experimentado. Una empresa debe a cambio escoger un IDS que pueda rastrear los ataques rápida y automáticamente, incluso aquellos con referencias falsas, o que se reflejan de regreso al punto de ingreso de la red. Esto permitirá a la empresa reaccionar rápida y eficientemente para bloquear los ataques de negación de servicio que pueden afectar la disponibilidad del ancho de banda y del servicio.

- Un sistema IDS debe manejar los escenarios de instalación más grandes y exigentes, incluyendo el monitoreo de los múltiples segmentos de la red.

- Un sistema IDS debe tener un motor de análisis y correlación que analice los numerosos eventos que suceden en la red y los evalúe en su contexto. El tiempo y el conocimiento son críticos para lanzar una respuesta rápida y efectiva a los ataques contra los activos empresariales de misión crítica en el momento en que se produzcan. La acumulación de eventos en tiempo real, la correlación y el análisis pueden reducir dramáticamente el esfuerzo que tradicionalmente se exige del personal de seguridad para darle tiempo a que realice un trabajo de investigación, en lugar de pasar horas examinando los registros de eventos no correlacionados.

- Un sistema IDS debe recoger los datos importantes de detección directamente de los conmutadores de redes, lo que reduce la cantidad de sensores que se necesita instalar y administrar en toda la red, a fin de reducir el Costo Total de Propiedad. Muchas empresas no se dan cuenta de que la ampliación de los sistemas IDS diseñados específicamente para redes conmutadas de alta velocidad cuesta menos que la de los sistemas tradicionales. Una instalación tradicional requiere de un sensor para cada segmento de la red, además del costo del hardware, software y de la administración del sistema.

Cuando una compañía necesita expandirse para tener cobertura de red total, los costos asociados al financiamiento de los sensores, hardware, software y administración del sistema serán comparables a los costos de aquellos componentes de la instalación inicial; por el contrario, un sistema IDS diseñado para funcionar en redes conmutadas de alta velocidad, permite una cobertura máxima sin el costo agregado de la administración del sistema.

Un sistema IDS con la herramienta de localización puede compararse a un grupo de cámaras de seguridad almacenadas convenientemente; sin embargo, a diferencia de dichas cámaras, el sistema IDS debe tener la inteligencia para saber que ha ocurrido un incidente, para continuar recogiendo datos y alertar al administrador del sistema. Con este tipo de sistema, los ahorros empresariales tan sólo en concepto de soporte, instalación y mantenimiento pueden ser significativos.

Conclusiones

Este articulo presenta de manera breve lo que es un IDS, y algunos criterios para su elección, así como una recomendación a las empresas que aún no han invertido en un software de detección de intrusos para considerar la adquisición de uno; aplazar esta inversión supone para la empresa un riesgo no únicamente de modificación, destrucción y robo de la información, sino de ser objeto de demandas legales. El sistema IDS que la empresa adopte en última instancia, debe proporcionar un método muy bien coordinado para administrar los asuntos de seguridad, desde la identificación de robos en la red y obtención de información adicional solicitada, hasta responder rápidamente y tomar las medidas adecuadas.

Referencias

[1] http://www.infosecuritymag.com/newsletter/

[2] http://www.guardiacivil.org/kio/seg/sld001.htm

[3] http://www.idsdetection.com/

[4] www.icsa.net

[5] http://www.cisco.com/warp/public/cc/cisco/mkt/security/nranger/index.shtml

Arquitecturas en n-Capas: Un Sistema Adaptivo

M. en C. Elizabeth Acosta Gonzaga
M. en C. Jesús Antonio Álvarez Cedillo
Profesores del CIDETEC-IPN
M. en C. Abraham Gordillo Mejía
Profesor de la UPIICSA-IPN

Los primeros sistemas de cómputo, tal y como los conocimos, se encontraban aislados unos de otros y contaban con sus propios dispositivos propios de E/S, y los programas tenían acceso a la computadora únicamente a través de dichos dispositivos; con la llegada y auge de las redes de comunicaciones el panorama ha cambiado, y ahora es necesario escribir programas que interactúan y/o dependen de otros programas en computadoras lejanas, en un esquena que se conoce como programación distribuida.

Una aplicación distribuida es un sistema compuesto de un conjunto de programas corriendo en múltiples computadoras (hosts); su arquitectura es un esqueleto de programas diferentes, con una descripción de qué programas están corriendo en qué hosts, cuales son sus responsabilidades y qué protocolos determinan la forma en la cual diferentes partes del sistema se comuinican unas con otras.

El concepto de capas (tiers), proporciona una manera conveniente para agrupar diferentes clases de arquitectura. Así, si una aplicación se ejecuta en una computadora, ésta tiene una arquitectura de una capa (one-tiers), si la aplicación se esta ejecutando dos computadoras, por ejemplo, una aplicación web CGI que corre en un navegador (cliente), y en un servidor web, se dice que es una aplicación de dos capas, ya que se tiene un programa cliente y un programa servidor; la diferencia principal entre éstos es que el servidor da respuesta a solicitudes de varios clientes, mientras que un cliente inicia una solicitud de información a un servidor.

Una aplicación de tres capas suma un tercer programa, comúnmente una base de datos, en la cual el servidor almacena información. Este tipo de aplicaciones es una mejora incremental a la arquitectura de dos capas; así, si una solicitud llega de un cliente a un servidor, éste solicita o almacena información en la base de datos, la base regresa información al servidor, y el servidor regresa la información al cliente.

Una arquitectura de múltiples capas (n-tiers), permite que un número ilimitado de programas corra simultáneamente, enviando información de uno a otro, utilizando diferentes protocolos para comunicarse e interactuar simultáneamente; esto permite crear aplicaciones más poderosas que proporcionen diversos servicios a varios clientes.

Sin embargo, esto también representa desventajas; por ejemplo, existe la posibilidad de que se introduzca código maligno, tal como los gusanos (worms), ocasionando problemas en el diseño, implementación y rendimiento. Se han desarrollado nuevas tecnologías que ayudan a combatir esta anomalía, tales como CORBA, EJB, DCOM y RMI; sin embargo, para saltar de tres a n-capas, así como para saltar de una a dos, o de dos a tres, deben tomarse en cuenta diversas consideraciones de seguridad.

Arquitecturas de una capa

Una arquitectura de una capa es un programa simple que no necesita acceso a la red mientras se ejecuta; entre las aplicaciones más comunes de este tipo están los procesadores de texto y los compiladores. Como ya se ha mencionado, un navegador y un servidor web forman parte de un arquitectura de dos capas, pero ¿qué pasa si el navegador web baja un applet java y éste se ejecuta en la computadora?. El applet no accesa a la red mientras se ejecuta, entonces, desde que llega al cliente se convierte en una aplicación de una capa; en estos términos, un programa escrito con JavaScript o VBScript incrustado en código HTML, también puede decirse que es aplicación de una capa.

Las aplicaciones de una capa tienen una gran ventaja: la simplicidad, pues no necesitan del manejo de protocolos, y no se necesita garantizar la sincronización entre datos lejanos, ni el manejo de rutinas de excepción para soportar las fallas inherentes de la red o de un servidor manejando

diferentes versiones de un protocolo o programa. Estos programas representan una gran ventaja en cuanto al rendimiento, ya que los datos no necesitan atravesar la red, esperar su turno en el servidor y entonces regresar, por lo que no sobrecargan la red ni al servidor con tráfico extra.

Arquitecturas de dos capas

Una arquitectura de dos capas en realidad tiene tres partes: un cliente, un servidor y un protocolo, mismo que hace un puente entre las capas del cliente y del servidor. El diseño de dos capas es muy efectivo para aplicaciones de red, así como para programas con interfaz gráfica de usuario (GUI); comúnmente los programas GUI se almacenan del lado del cliente, y la lógica del negocio del lado del servidor, esto permite regenerar y validar datos del usuario en el cliente, donde se tiene una respuesta más rápida; en el proceso, se conservan los tan apreciados recursos de la red y del servidor. De igual forma, la lógica del negocio vive en el servidor, donde está segura, además de hacer uso los recursos de dicho servidor.

Una aplicación de dos capas es un programa cliente/servidor con una GUI como presentación, escrita en un lenguaje de alto nivel tal como Java, C++, etc. En un programa de dos capas se ve la división entre las capas frontales (front) y las capas posteriores (end). En la primera capa, el cliente no necesita preocuparse acerca del almacenamiento de los datos o acerca del procesamiento de múltiples solicitudes; la segunda capa, el servidor, no necesita preocuparse acerca de cómo alimentar datos, o sobre cuestiones relacionadas con la Interfaz Gráfica (UI). Por ejemplo, una aplicación de conversación en línea (Chat) está compuesta por un cliente que acepta entradas del usuario y despliega mensajes, y un servidor que entrega mensajes de un cliente a otro.

Otro ejemplo sería un programa CGI que calcula una hipoteca (puede ser un servlet Java o script Perl). Los datos de entrada se proporcionan a través del protocolo HTTP y específicamente en un formulario HTML, que el usuario llena; la salida es uno o más archivos HTML, pero en realidad, todos los cálculos ocurren en el servidor, y aunque se incorpora un navegador web para presentar datos de salida y aceptar datos de entrada del usuario, toda la ejecución real del programa ocurre en el servidor.

Así, si se escribe un programa para el servidor y no el código que debe ejecutarse en el cliente, entonces, ¿puede decirse que es una aplicación de dos capas?. Esto es altamente discutible, ya que puede decirse que el código HTML es realmente una forma primitiva de código de un programa, y si además se añade un Javascript u otro código del lado del cliente, entonces en verdad se tiene una aplicación de dos capas. Una arquitectura de una capa combina todas las funciones en un solo proceso, mientras que una arquitectura de dos niveles debe separar diversas funciones. En realidad, elegir entre una o dos capas depende de las necesidades de la aplicación.

Arquitecturas de tres capas

Muy probablemente, una aplicación de dos niveles necesitará almacenar datos en un servidor. Tradicionalmente la información se almacena en un sistema de archivos (FileSystem); sin embargo, el riesgo a la integridad de datos se presenta cuando múltiples clientes solicitan simultáneamente al servidor realizar tareas. El sistema de archivos en el mejor de los casos, maneja controles de concurrencia rudimentarios, por lo cual la solución más común es añadir un tercer programa, una base de datos.

Una base de datos especializada almacena, recupera e indexa datos, tal como una arquitectura de dos capas, separa la capa de lógica del negocio de la capa GUI; una arquitectura de tres capas permite separar la lógica del negocio y el acceso de datos, como se muestra en la **Figura 1**.

Con una base de datos es posible proporcionar índices a los mismos, incluir métodos altamente optimizados para su recuperación, y proporcionar replica, respaldos, redundancia, y procedimientos de carga específicos que balancean según las necesidades de la información.

En la actualidad existen sistemas SQL RDBMS, tales como Oracle, Sybase, etc, y algunos tipos tales como OODB (bases de datos orientadas a objetos), ORDB (bases de datos relacionales), etc.

El procedimiento general para usar una base de datos implica, en primer lugar, realizar un esquema que describa los datos para posteriormente escribir consultas que almacenen y recuperen

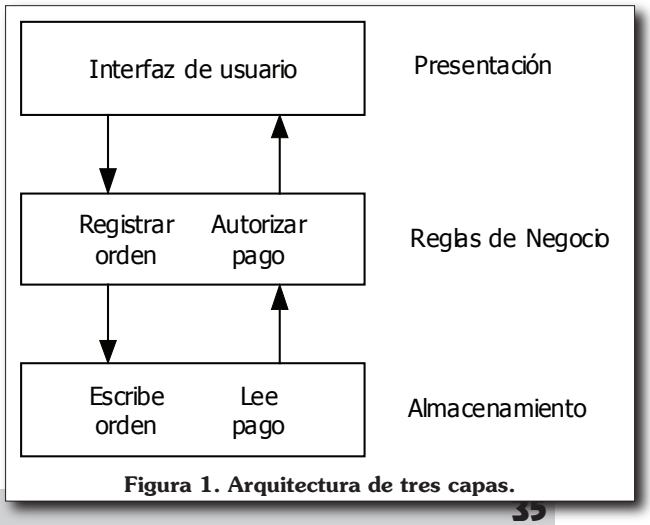


Figura 1. Arquitectura de tres capas.

dichos datos. Existen muchos casos que usan bases de datos; sin embargo, también existen ocasiones en que se puede almacenar una cantidad pequeña de datos en un archivo local simple, en vez de una base de datos relacional. La regla común es: si solo se necesita almacenar datos, y recuperarlos por nombre, un sistema de archivos es suficiente. Pero si se requieren búsquedas sobre la información, entonces es necesaria una base de datos, especialmente si las búsquedas incluyen varios criterios.

Uno de los beneficios de las bases de datos, además del acceso rápido y la fiabilidad, es que múltiples aplicaciones o servicios pueden tener acceso a los mismos datos; éste beneficio está implícito en las aplicaciones de tres o de n-capas. Los procedimientos almacenados (*stored procedure*), son pequeños programas que corren en una base de datos, y puesto que están cerca de los datos, pueden manipularlos; por ejemplo, clasificando, filtrando, transformando, etc. Sin embargo, realizar esto en el lado del servidor es muy costoso.

Existen operaciones que requieren de procedimientos o disparadores almacenados (*triggers*), para funcionar; sin embargo, es fácil utilizarlos de forma incorrecta, es decir si se colocan dentro de la lógica del negocio constituyen un desorden a la estructura de tres capas, y así, en lugar de tener una GUI, lógica y almacenamiento separados, se tiene la lógica del negocio mezclada con el almacenamiento. Además, si el procedimiento se escribe en otro lenguaje diferente al del código en uso, hace que el código del procedimiento sea mucho más difícil de mantener y de eliminar errores.

Por ejemplo, un sitio Web de comercio electrónico incluye un carrito de compras, y al término de las mismas se necesita calcular el costo del envío; entonces, se escribe un procedimiento almacenado que calcule el costo automáticamente por alguna compañía de entrega de paquetería, tal como FedEx, cada vez que se salva un registro de compras (procedimiento conocido como *trigger*); pero si después de seis meses se decide enviar con otra compañía que proporciona un costo más barato dependiendo de la zona del destino, entonces tendrá que escribirse nuevo código, seguramente en SQL. Si se desordenan los datos se puede interferir con la integridad de los mismos, pero si el código original se hubiera escrito directamente en Java, en una capa intermedia, podría extenderse la clase DeliveryMethod.

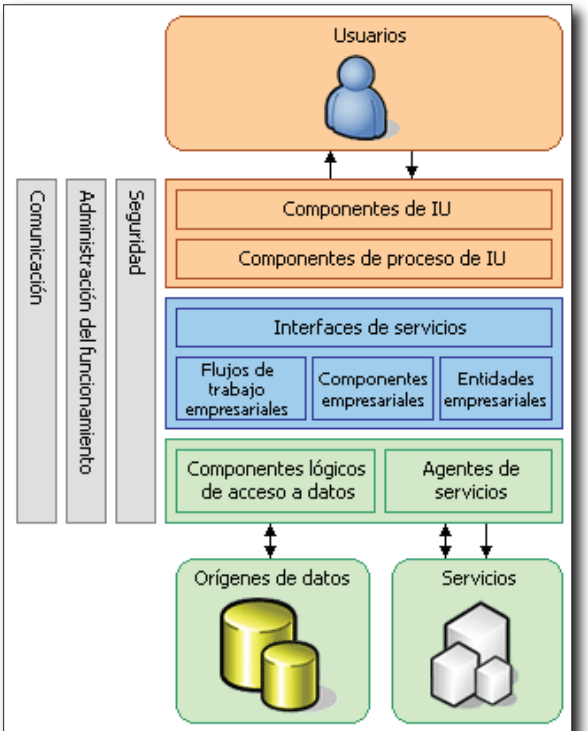


Figura 2. Diseño general de una aplicación con una arquitectura de n-capas.

Arquitecturas de n capas

Cuando se escribe una aplicación, normalmente se diseñan y escogen los objetos comunicándolos a través de mensajes en código de alto nivel. Los protocolos que utilizan los objetos distribuidos manejan lo rudo, los detalles de bajo nivel, tales como los parámetros de ordenamiento, localización de objetos remotos, manejo de transacciones, etc.

Un ejemplo de una aplicación de n-capas es un sistema para almacenamiento de mercancías; en este ambiente múltiples datos alimentan órdenes de compra que llegan de diferentes fuentes, de múltiples bases de datos, y éstas a su vez son consultadas por múltiples clientes que ejecutan aplicaciones especializadas; tiene sentido unir éstos bloques diferentes con el hilo de una arquitectura común de objetos distribuidos, tales como CORBA o EJB, (ver **Figura 2**).

En una arquitectura de n-capas se requiere diseñar objetos realmente reutilizables, que puedan usarse para proyectos futuros. Si los requisitos para un proyecto cambian es necesario reescribir el código; aún más importante es el hecho que, dejando la seguridad que proporciona una arquitectura por capas, se corre el riesgo de diseñar un sistema que sea más complejo que el pensado originalmente. Esto evita el avance, puesto que una decisión descuidada en el diseño puede tener aspectos no considerados. Sin embargo, la transmisión de aplicaciones de objetos distribuidos en CORBA no es tan rápida como aquellas diseñadas con protocolos de socket estándar; CORBA es lento por que necesita ser general, y su gran fortaleza es su mayor debilidad.

Por lo tanto, si lo que se necesita es rapidez el mejor método es hacerlo uno mismo (DIY, *Do It Yourself*). Esto realmente no representa un inconveniente, ya que es aceptable

si el sistema se puede extender añadiendo nuevos módulos, lo que se conoce comúnmente como arquitectura escalable; ésta permite que sus programas se puedan extender sobre múltiples máquinas. Cuando un sistema tiene muchos objetos que interactúan recíprocamente, el número de combinaciones aumenta exponencialmente con el número de objetos, y entonces es recomendable escoger un estándar existente.

En un sistema de tres capas existen millones de accesos a un solo servidor, lo que produce una saturación al mismo; se puede resolver el problema agregando más servidores, a lo que se llama balanceo de carga, (miles de accesos entre varios servidores equivalentes). Cuando se tiene una sola base de datos para todos los servidores, esto no es representa problema si, por ejemplo, un servidor escribe datos y otro servidor necesita leerlos inmediatamente después. Sin embargo, en un arquitectura de n-capas, hay docenas o centenares de objetos que corren simultáneamente, ejecutándose en muchos anfitriones; si el sistema es lento, no es inmediatamente claro que objeto o cual host requiere balancear la carga. Se neesita el análisis a detalle del tráfico de la red y de los archivos de bitácora, para aislar los cuellos de botella; es decir, aumentando el grado de detalle en el diseño del objeto también se limita el rendimiento del sistema.

En un sistema de tres capas prebalanceado, se conoce que hay servidores usando de forma óptima su CPU; si una solicitud llega al sistema, éste se encarga de realizarla. Si es necesario esperar a que la base de datos responda a una consulta (*query*), entonces se generan múltiples hilos o tareas que trabajan sobre otra solicitud; sin embargo, si la cadena de la comunicación tiene que pasar por diferentes máquinas en la red, entonces el servidor original puede estar inactivo, esperando el regreso de una serie de mensajes que están formados en algún objeto remoto sobrecargado en alguna parte de la red. Los CPU de la línea frontal están subutilizados, mientras que el CPU posterior esta sobrecargado, y no existe una forma simple de transferir los ciclos inactivos de una máquina a otra; entonces se debe rediseñar a nivel de objetos para eliminar las ineficiencias.

El surgimiento de la tecnología de componentes distribuidos es la clave de las arquitecturas de n-capas. Estos sistemas de computación utilizan un número variable de componentes individuales que se comunican entre ellos utilizando estándares predefinidos y marcos de comunicación (*frameworks*) como:

CORBA - (*Common Object Request Broker Architecture*) de Object Management Group (OMG).
DNA- (*Distributed interNet Architecture*) de Microsoft (incluye COM/DCOM y COM+ además de MTS, MSMQ, etc.) .
EJB- (*Enterprise Java Beans*) de Sun Microsystems.
XML- (*eXtensible Markup Language*) de World Wide Web Consortium (W3C).

Conclusiónes

La esencia de los modelos de desarrollo por capas es el concepto de *separación*, es decir, mantener cada componente tan separado del contexto global como sea posible. Así, cada capa simplemente es la agrupación de todos los componentes que tienen una funcionalidad común.

Una arquitectura de n-capas forma parte de un proceso revolucionario, actualmente en desarrollo, basado en la aplicación de estas nuevas tecnologías (componentes y estándares de Internet). Estas tecnologías son los bloques para crear Software de Negocio y Sistemas de Información adaptables que ayuden a las empresas a integrar todos sus sistemas de Tecnologías de la Información, mientras que obtienen una ventaja clara en el uso de Internet.

Los componentes distribuidos de una arquitectura de n-capas son una tecnología esencial para crear la siguiente generación de aplicaciones e-business, aplicaciones que son altamente escalables, confiables y que proporcionan un alto rendimiento y una integración sin fallas con los sistemas heredados.

Bibliografía

[1] Jacobson, Ivar, Grady Booch, and James Rumbaugh. *El Proceso Unificado de Desarrollo de Software.* México: Addison-Wesley, 1999.

[2] Kruchten, Philippe. «*Architectural Blueprints—The 4+1 View Model of Software Architecture*». IEEE Software, IEEE. November 1995, pp. 42-50.

[3] Larman, Craig. *UML y Patrones, Introducción al análisis y diseño orientado a objetos.* México: Prentice Hall, 1999.

[4] Wilson, Scott F. *Analyzing Requirements and Defining Solution Architectures*. Redmond: Microsoft Press, 1999.

[5] Fernández Aramayo, David Ricardo. *Arquitectura de Software.* Universidad Tecmilenio, ITESM.