Visualización de datos dinámicos N-dimensionales utilizando métodos de reducción de dimensión y análisis con técnicas de multirresolución.

Beatriz Benítez Mayo
Facultad de Ciencias, UABC

Dr. Fernando Rojas Íñiguez
Centro de Ciencias de la Materia Condensada, UNAM

Ensenada, Baja California. México 2004

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE CIENCIAS

VIAN

VISUALIZACIÓN DE DATOS DINÁMICOS N-DIMENSIONALES UTILIZANDO MÉTODOS DE REDUCCIÓN DE DIMENSIÓN Y ANÁLISIS CON TÉCNICAS DE MULTIRRESOLUCIÓN.

TESIS PROFESIONAL

PRESENTA:

BEATRIZ BENÍTEZ MAYO

APROBADA POR:

DR. FERNANDO ROJAS ÍÑIGUEZ
PRESIDENTE DEL JURADO

M. C. MIGUEL ÁNGEL IBARRA RIVERA
SECRETARIO

M. C. JOSÉ IGNACIO ASCENCIO LÓPEZ
PRIMER VOCAL

Dedicatoria y agradecimientos

Dedicatoria

Dedico este proyecto de tesis a Delfina Mayo Ruiz (mi madre), Hilario Benítez Gómez (mi padre), Alejandro Benítez Mayo (mi hermano) y a Eduardo Herrera Suárez (mi novio) por ser las personas más cercanas a este ciclo de mi vida. Se los dedico agradeciéndoles todo su amor, paciencia, aliento, y apoyo; y expresándoles todo mi amor, entrega y admiración.

Agradecimientos

Primero agradezco al Dr. Fernando Rojas Íñiguez (mi director de tesis) por darme la oportunidad de trabajar con él, y en este proyecto. Además agradezco su fomentación, apoyo, sinceridad, paciencia, prudencia, consejos, impulso y sabiduría. Agradezco que compartiera conmigo su experiencia, la cual fue de más importante es este proyecto de tesis y sin la cual no se hubiese logrado.

Agradezco a la Facultad de Ciencias (FC) de la UABC y al Centro de Ciencias de la Materia Condensada (CCMC) de UNAM por ayudar en mi formación profesional. Agradezco a los que hicieron su labor de profesores y orientadores en mi desarrollo intelectual.

Agradezco a todos y cada uno de los miembros de mi familia y familiares por todo lo brindado; compañeros y en general a todos los que: me tendieron la mano en cualquier aspecto; a los que me acompañaron y me dieron voz de aliento en cualquier momento; a los que creyeron en mi a lo largo de toda mi carrera y durante el desarrollo de mi tesis. No podría mencionarlos a todos, es por ello que omito nombres en esta parte, pero todos ellos conocen mi gratitud sincera.

Agradecimientos para mis amigos muy cercanos por todo el tiempo valioso que compartimos y que contribuyó en mi estancia para la elaboración de mi tesis. Y gracias a mi amigo Oscar Olalde Martínez por su especial apoyo en el diseño gráfico de mi tesis.

Experiencia personal

Hay siete pifias en el mundo que generan violencia: riqueza sin trabajo, placer sin conciencia, sabiduría sin carácter, comercio sin moral, ciencia sin humanidad, culto sin sacrificio y política sin principios.
Mahatma Gandhi

Ni el libro cerrado da sabiduría, ni el título por sí solo da maestría.
Anónimo

La tesis lo que representa para mi, es un trabajo tan poderoso que incluso puede marcar un sello personal en quien la realizó. Además de elaborarse por la motivación de cumplir un requisito en la obtención de un título, es una oportunidad de un amplio aprendizaje, lo que hace que exista muchas más motivaciones, por ejemplo, para abrirte puertas en un campo de estudio más avanzado, o en un trabajo digno lleno de oportunidades para lograr el éxito deseado como profesionista y sobretodo como mujer.

Este espacio quiero aprovecharlo si bien, para remarcar la satisfacción de mi trabajo, también lo deseo para exponer la importancia de mi tesis en mi vida personal. En este tiempo he aprendido muchas cosas, sufrido algunas y logrado algunas otras, la tesis, para mí no solo fue el trabajo que he explicado, sino un reto emocional muy grande, donde el apoyo de mis seres queridos fue fundamental. Aprendí que todo implica sacrificios, pero no solo eso, sino que debes sonreírle a esos sacrificios. También aprendí que debes darle el tiempo y espacio a tus sentimientos, a tu diversión, a tus seres queridos, y nunca perder el enfoque, siempre tener presente la meta, el llegar a tu objetivo. Aprendí que los ciclos deben concluirse a pesar de todos los giros que en tu vida surjan. Aprendí que la comunicación es fundamental, no solo para la realización de un proyecto, sino para la armonía de tu cuerpo, mente, emociones y espíritu, esto es, debes acercarte a quienes te escuchen para hablar acerca de tu entorno.

Mientras desarrollaba este bello proyecto, los sube y baja de mi vida personal no paraban, y esto lo comento porque todo repercutió en mi trabajo, y ahora lo hago notar, no es solo orgullo el que siento, sino una sincera gratitud a los que estuvieron siempre conmigo cerca y, ¿porqué no decirlo? un gran amor a mi carrera, a mi tesis, y a ellos, los que estuvieron siempre conmigo.

Resumen

RESUMEN de la Tesis de **Beatriz Benítez Mayo** presentado como requisito parcial para la obtención de la *Licenciatura en Ciencias Computacionales*. Ensenada, Baja California, México; jueves, 30 de septiembre de 2004.

ViAn

Visualización de datos dinámicos N-dimensionales utilizando métodos de reducción de dimensión y análisis con técnicas de multirresolución.

En este trabajo de tesis presentamos el desarrollo de una aplicación de software científico para la visualización de datos multidimensionales a través de técnicas de reducción de dimensión de datos, que incluye el análisis de los datos reducidos, obteniendo al sistema ViAn. Para el desarrollo del trabajo: planteamos el problema a través de los requerimientos y proponemos una solución con el análisis, establecemos las técnicas de reducción de dimensión de datos, y detallamos las herramientas que utilizamos para la programación de la aplicación ViAn.

Seguimos una metodología de tipo espiral de la ingeniería del software, con las etapas de requerimientos, análisis y diseño de la aplicación como un sistema computacional. Esto con el desarrollo de los diferentes diagramas y su correspondiente documentación. A estas fases le integramos los aspectos metodológicos específicos del análisis y diseño de las técnicas de reducción de dimensión de datos y análisis de datos reducidos.

Para la reducción de dimensión de datos proponemos dos algoritmos: 1. El algoritmo de Kohonen basado en las Redes Neuronales Artificiales, y 2. El algoritmo de Expectación Máxima (EM) para el modelo de Variables Latentes; y para el análisis de datos utilizaremos: 1. La transformada de Fourier, y 2. La transformada de Wavelets (ondeletas). Estas técnicas, junto con los otros requerimientos (interfaces de usuarios, manejo de archivos, visualizaciones, etc.) han sido moduladas en el sistema ViAn, el cual ha sido programado en el lenguaje IDL (Interactive Data Language).

ViAn cumple con los requerimientos especificados y se cuenta con una primera versión de la aplicación. Consideramos que este trabajo muestra la experiencia de desarrollo de software orientado a aplicaciones científicas, además de que proporciona una base sólida en términos metodológicos para el desarrollo de futuras versiones de ViAn, ya que se tiene una documentación consistente y adecuada de las diferentes fases del desarrollo.

Resumen aprobado:

Dr. Fernando Rojas Íñiguez
Director de la tesis

Summary (resumen traducido al idioma inglés)

SUMMARY of the Thesis of Beatriz Benítez Mayo presented like partial requirement for the obtaining of the Degree in Computer Scientist. Ensenada, Baja California, Mexico. Thursday, September 30, 2004.

ViAn

Visualization of N-dimensional dynamic characteristicses using methods of reduction of dimension and analysis with multiresolution techniques.

In this thesis work we introduce the ViAn system, a scientific software application developed for multidimensional data visualization through techniques of dimensional data reduction, which also includes an analysis on data reduction. For the development for this work: we set the problem through the requirements and a solution is proposed through the analysis, we establish the dimensional data reduction techniques, and detail the tools used for the programming of the ViAn application.

We follow the spiral methodology from the Software Engineering discipline. This includes the step of considering the requirements, analysis elaboration and application design as a computational system. This is development with different diagrams and their correspondent documentation. Also in these steps specific methodology aspects of analysis, dimensional data reduction techniques and reduced data analysis have been integrated.

For the dimensional data reduction two algorithms are proposed: 1. The Kohonen algorithm based on Artificial Neural Networks, and 2. The Expectation Maximization algorithm (EM) for the model of latent variables; For the data analysis: 1. The Fourier transformed and 2. The Wavelets transformed. These techniques along with other requirements (user interfaces, file management, visualizations, etc.) have been molded to fit the ViAn system which has been developed under the IDL programming language.

ViAn fulfills certain requirements within this work and has its first release version of the application. Let this work be considered as a sample of the experience of scientific oriented application development, and also as a solid base propose within methodology terms for future development of ViAn versions, since ViAn has a consistent and adequate documentation of all the different development steps.

Índice del contenido

Índice de figuras

Índice de tablas

CAPITULO I.

Introducción

Compartir los sueños con un amigo es
empezar a convertirlos en realidad.
Anónimo

CAPITULO I.
Introducción

I. 1. Planteamiento del Problema

En las áreas científicas y tecnológicas es muy común contar con un número elevado de datos, el manejo de dichas operaciones es de una gran inversión de tiempo y recursos, los fenómenos son muy variados y son trabajos muy complejos. Una manera de manipular los datos de una forma eficiente es con la representación de estos datos y no con los datos mismos, con el fin de realizar operaciones con menor complejidad y evaluación de las características. Para tener claro este problema, supongamos que tenemos datos que se encuentran en un espacio de 7 dimensiones, y como sabemos, no es posible representarlos en un sistema de coordenadas, es decir gráficamente no son visibles, por lo tanto es muy difícil el manejo y el entendimiento de las características.

Una forma de representar un gran número de datos de un fenómeno o experimento es con la **Visualización de Datos Científicos** (VDC) **[Aldrich, 1998]**. Existen varias formas de VDC, como lo son: trazos bidimensionales (2D *plotting*), en líneas, símbolos, barras; figuras geométricas tridimensionales; superficies de objetos 2D y 3D; imágenes en 3D como volúmenes - ésta es una forma muy compleja -; mapeo (trazo) de datos; a todas éstas se les conoce como formas de visualización estándares **[1999]**.

En las Ciencias Computacionales, la VDC, es un campo muy amplio donde hay muchas áreas por desarrollar. Además de los métodos estándares, es necesario contar con métodos alternativos que nos permitan tener el control en el manejo de los datos y sus operaciones, para aplicarlo a problemas complicados. Como ejemplo, la visualización de fenómenos reales de cualquier área científica o de investigación, con las características del experimento muy específicas y donde es necesario procesar la información para el **Análisis de Datos [Burt, 1983]** subsecuente de sus propiedades.

La **Reducción de Dimensión de Datos [Bishof]** es una alternativa para que profesionistas en el área de desarrollo de software, implementen un método innovador para la VDC, y así se tenga mayor control en la información. Una propiedad deseable en la técnica de reducción es la conservación de la topología de los datos. Es decir, si dos puntos son vecinos en el espacio de datos, también lo serían en el espacio reducido.

I. 2. Propuesta del Sistema

Como mencionamos, existen fenómenos reales cuya descripción es compleja, y se necesita de un estudio para la clasificación y/o análisis de sus datos, además de modelos donde los datos de salida (resultantes) no son definidos de manera a priori, forma que ningún modelo de visualización estándar ofrece. Estos fenómenos complejos pueden provenir de diversas áreas científicas y de investigación, por ejemplo. química, matemáticas, biología, física, geología, astronomía, ecología, meteorología, sociología, economía, computación, etc.

Entonces en el presente trabajo proponemos el desarrollo de una herramienta de manera que diseñemos un software para la visualización de datos de fenómenos complejos en grupos de dimensiones grandes. Con este proyecto el usuario tendrá la ventaja de contar con una herramienta computacional que le permita procesar una gran cantidad de información para la visualización de los datos de sus experimentos y obtener una menor pérdida de información al momento de aplicar la reducción de dimensión de datos. La técnica de reducción de dimensión que propondremos es a través de una forma no estándar de la VDC, que cuenta con la ventaja de mantener la topología de los datos.

Las técnicas para la reducción de dimensión de datos que cumplen con las características y ventajas mencionadas, son algoritmos de aprendizaje, que se desarrollaron en las áreas de Redes Neuronales Artificiales (RNA) **[Lippmann, 1987]**, y en el Modelado de Variables Latentes **[Markus, 1998]**. Las RNA es un campo de estudio muy amplio, y el algoritmo que adoptamos para parte de la solución a nuestro problema es el de Kohonen **[Kohonen, 1990]**. El modelado de Variables Latentes también es de tipo neuronal. Este modelado tiene muchas aplicaciones, una de ellas se enfoca en la estadística, para esto, se necesita al concepto de la Maximización del valor Esperado (EM). Es así como surge el algoritmo de EM (*Expectation Maximization*) **[Markus, 1998]**, el cual es nuestro segundo algoritmo para solucionar la reducción de dimensión de datos. Las bases de estos algoritmos las detallaremos más adelante.

Por otra parte, el usuario tendrá en el software propuesto las opciones para realizar un análisis de los datos del espacio reducido con técnicas de multirresolución y espectral, a través de las transformadas de Wavelets **[Rao, 1998]** y Fourier **[Soo-Chang, 2001]** respectivamente. Así, esta aplicación simplificará las tareas del investigador y facilitará el análisis de sus experimentos con operaciones de datos reales, de un espacio de N dimensiones a un espacio reducido de dos dimensiones (M=2).

En la figura 1.1 se muestra un *Esquema general del sistema ViAn para la Visualización y Análisis*. Un posible escenario del uso del software es el caso en que el investigador dará como entrada al sistema una serie de datos **t** (t_1,t_2,..,t_h) del experimento del espacio N-dimensional.

La información que contiene los datos N-dimensionales se almacena como "datos de entrada" en un Sistema de Archivos (SA), se tienen dos algoritmos como opciones para reducir el espacio de datos: uno es el algoritmo de Kohonen y otro es el algoritmo EM de Variable Latente; después de este proceso se generan nuevos datos, que es la información transformada que el usuario introdujo al sistema, ahora en un espacio reducido (espacio M), los datos nuevos se almacenan al SA como "datos reducidos".

Entonces se procede a la opción de visualización de los datos reducidos, los cuales serán presentados en imágenes, que representan nuestro espacio M-dimensional (donde consideraremos la opción de M=2 para nuestro caso); la información a Visualizar es leída del SA y se procesa dicha actividad; para el análisis del espacio reducido se leen los "datos reducidos" del SA nuevamente, y se presentan las posibilidades de utilizar dos técnicas: una de ellas es la Transformada de Wavelets con el método de Burt-Adelson **[Burt, 1983]** y la otra es la Transformada de Fourier; y se almacenan los resultados del experimento en el SA, como "datos analizados".

No está de más hacer énfasis que, si el investigador lo prefiere, puede analizar los datos sin la necesidad de detenerse en el proceso de la visualización; esto es, si el experimento es muy repetitivo y ya se tiene noción de la visualización, entonces, con el fin de agilizar su trabajo de investigación, el usuario analiza directamente los datos reducidos de su experimento.

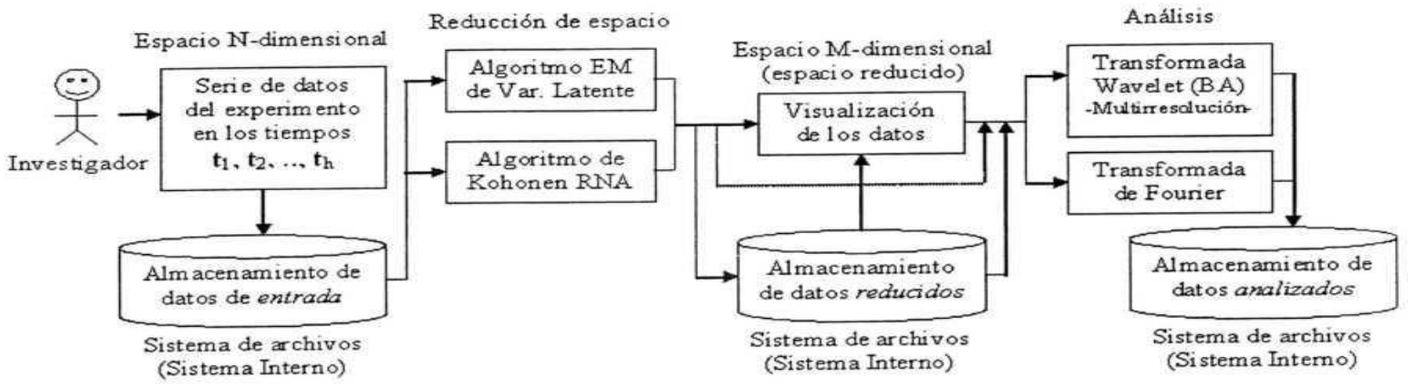


Figura 1.1. Esquema del sistema ViAn para la visualización y análisis

Entonces, para este esquema, existen tres tareas (o actividades) esenciales, para estudiar y manipular sus datos, las cuales se explican a enseguida.

Estas tres actividades las presentaremos como módulos del sistema ViAn, en dos de estas tareas encontraremos a dos técnicas diferentes y a su vez comparativas, una actividad es la reducción de dimensión de datos y la segunda es para el análisis de datos:

1. Un módulo es para la *Reducción* de dimensión de datos N-dimensionales, y cuenta con las siguientes dos técnicas.
    a. Esta se basa en las Redes Neuronales y se trata del algoritmo de *Kohonen*.
    b. Esta otra es basada en el modelo de Variables Latentes, es el algoritmo *EM*.
2. *Un segundo módulo* es para la *Visualización* de datos científicos. Se generan imágenes para poder visualizar las características de los datos reducidos.
3. Y por último tenemos al tercer módulo para el *Análisis* de datos bidimensionales que se procesará con dos técnicas diferentes:
    a. Una técnica de análisis espectral con la transformada de *Fourier*.
    b. Análisis es de multirresolución con la transformada *Wavelets*.

La explicación de la información específica del contexto del dato científico no es de gran importancia para el desarrollo de la tesis, lo es solamente para el científico o investigador que utilizará nuestro sistema ViAn, es el que comprende el experimento y le da la interpretación apropiada a los resultados generados en cada proceso que se realice.

Con el fin de tener una mejor noción de la estructura requerida para ViAn, enseguida se muestra una breve explicación de los tres módulos mencionados, y que serán mejor explicados y desarrollados a lo largo del trabajo de tesis:

Reducción de dimensión de datos: La reducción de dimensión de datos significa que, toda la información que se encuentra en la dimensión N será representada en otro espacio con una dimensión más pequeña que N. Para el objetivo de poder visualizar la información que tiene este proyecto, la dimensión del espacio reducido, será considerado M=2. Esto es con el fin de lograr la generación de datos con un formato de imagen bidimensional.

Visualización de datos reducidos: La visualización de datos consiste en tomar los datos reducidos y su módulo se encarga de convertirlos a imagen; es en sí quien genera la imagen.

Análisis de datos: En el análisis de datos, se analizan las imágenes que se generaron para el espacio reducido con la visualización, y se da como resultado otras imágenes que contienen información que refleja la estructura subyacente de los datos.

Bajo esta visión la aplicación ofrecerá al usuario la opción de realizar operaciones de datos reales de N dimensiones. Para facilitar el desarrollo y la estructura de la aplicación, debemos establecer los objetivos a cumplir para el sistema ViAn.

I. 2. 1. Objetivos

Los objetivos para nuestro trabajo de tesis, los clasificamos en un Objetivo General y en Objetivos Específicos (para el sistema) que ayudan a estructurar la problemática planteada desde un punto de vista de ingeniería de software y algunos aspectos de reto técnico. Los objetivos de nuestro trabajo de tesis son los siguientes.

I. 2.1.1. Objetivo General

Demostrar la efectividad de la metodología utilizada en el desarrollo de una aplicación computacional con enfoque científico. Aplicación para la visualización de datos dinámicos N-dimensionales con técnicas de reducción de dimensión basadas en Redes Neuronales y en el Modelo de Variables Latentes; y para el análisis espectral y de multirresolución de los datos visualizados con las transformadas de Fourier y Wavelets respectivamente.

I. 2.1.2. Objetivos Específicos

- Determinar los requerimientos del usuario y del sistema.
- Realizar un análisis de la aplicación con los requerimientos especificados.
- Realizar un análisis de los modelos para la reducción de datos.
- Diseñar técnicas de reducción de datos en dos dimensiones.
- Diseñar sistemas de archivos para captura y lectura de un gran número de datos.
- Presentar un sistema para comparación de grupo de datos (análisis de datos).
- Implementar:
    a. Algoritmo de Kohonen.
    b. Algoritmo EM de Variable Latente.
    c. Modelo de la Transformada de Wavelets.
    d. Modelo de la Transformada de Fourier.
- Presentar experimentos del sistema.
- Desarrollar la aplicación en el lenguaje de programación IDL **[Fanning, 2000] [Gumley, 2002]**.
- Documentar el sistema completo (la aplicación) en sus distintas etapas de desarrollo.

Estos son los objetivos específicos para el proyecto ViAn. Ahora bien, en todo proyecto existen alcances y limitaciones, respecto a sus objetivos. En el caso de nuestro sistema ViAn, los alcances son satisfactorios, aún cuando nos encontramos con la limitación de presentar módulos preestablecidos. Esto significa que, para el objetivo referente a la implementación de los modelados de las transformadas de Fourier y Wavelets, su desarrollo en la aplicación implica la presentación de módulos preestablecidos y herramientas de aplicación respectivamente, para la primera versión completa de nuestro software.

I. 3. Síntesis

El proyecto se desarrolla con fines científicos, específicamente para el área de investigación en la Física Teórica. Aunque este fin no determina el objetivo de ViAn, quiero decir que no es exclusivo para dicha área, sino para cualquier área en que se manipulen datos de un fenómeno o experimento. El motivo por el cual surge la necesidad de plantear el problema presentado y proponer la solución anterior, es completamente académico. Es decir, en el desarrollo de este proyecto no existe la motivación de obtener un producto de uso comercial. Para un trabajo de Ciencias Computacionales, ViAn ofrece la oportunidad de introducir o ampliar conocimientos acerca de las áreas de Redes neuronales, Estadística aplicada en cómputo, Reconocimiento de patrones y Matemáticas, a través de una aplicación científica implementada en una herramienta con la filosofía de eventos, es decir, con una herramienta que trata de interactuar de forma flexible y amigable con el usuario. Además, en el aspecto de la ingeniería del software, ofrece la oportunidad de organizar nuestras ideas, problemas y soluciones en forma gráfica y de interacción con el usuario, y hacerlo bajo un enfoque diferente al orientado a objetos, hacerlo bajo el enfoque (filosofía) orientado a eventos.

En el presente proyecto se tratan diversos puntos (como los temas que mencionamos anteriormente) con un mismo fin: Agilizar de forma eficaz y eficiente las tareas que tiene un investigador para cualquier área científica. Los temas que se podrán encontrar son acerca de la visualización de datos científicos, y un tema muy allegado es la reducción de dimensión de datos N-dimensionales, como técnicas para la reducción tenemos un algoritmo que trata de Redes Neuronales Artificiales y uno más de Maximización del valor Esperado para del modelado de Variables Latente. Un área muy importante que también trataremos es el análisis de datos bidimensionales, este análisis puede ser logrado a través de las Transformadas de Fourier y Wavelets. Estas áreas que mencionamos solo cubren una parte de la tesis, también trataremos la filosofía de los eventos a través de un lenguaje específico, y veremos metodologías y sus herramientas para el desarrollo de la aplicación ViAn y de la documentación requerida para la tesis.

CAPITULO II.

Antecedentes

Si pudiéramos saber primero en donde
estamos y a donde nos dirigimos, podríamos
juzgar mejor que hacer y como hacerlo.
Abraham Lincoln

CAPITULO II.
Antecedentes

II. 1. Introducción

En este capítulo presentamos los Antecedentes, que son bases necesarias para describir las herramientas que se utilizarán para el desarrollo del sistema ViAn, y así contar con un respaldo (o un soporte) para el resto de la tesis. Conoceremos conceptos básicos de los diferentes procesos que se realizarán, aprenderemos acerca del lenguaje en que se implementará ViAn, estableceremos las herramientas para la metodología que se aplicará en el desarrollo del sistema y la documentación para las fases de análisis, diseño e implementación de ViAn.

Primero veremos los conceptos de visualización, reducción y análisis, para comprender los módulos que contendrá el sistema. Después veremos las herramientas para comprender los capítulos III y IV, que corresponden al Análisis del Sistema y al Diseño del Sistema respectivamente. Describiremos el lenguaje en que se implementará el sistema ViAn, el lenguaje es IDL. Y también presentaremos algunas referencias de otros trabajos afines al sistema.

II. 2. Conceptos

Los conceptos básicos que se requieren para una mejor explicación del resto de la tesis son los conceptos de las técnicas que nos servirán para procesar tareas específicas dentro del sistema ViAn, estas tareas son los procesos principales que se desarrollarán para obtener la aplicación. Los procesos que necesitamos son para la visualización de datos, reducción de dimensión de datos y análisis de datos.

II. 2. 1. La Visualización y la Reducción de datos

Un concepto claro y sencillo de visualización nos lo presentan Horst Bishof, Axel Pinz y Walter G. Kropatsh. y la re-escribimos a continuación: "La visualización se define como la representación gráfica orientada a la facultad de la percepción humana" **[Bishof]**. La Visualización Científica de Datos es muy

importante para facilitar procesos y cálculos, como también lo mencionan Christopher M. Bishop y Michael E. Tipping, quienes explican que: "La visualización ha probado que es una herramienta poderosa y altamente aplicable para el análisis e interpretación de datos multivariables. La mayoría de los algoritmos tienen como propósito encontrar una proyección para un espacio de datos reduciéndolos a un espacio bidimensional" **[Bishop, Tipping, 1998]**.

La visualización es la técnica que ayuda al científico, ingeniero y en general a cualquier investigador a alcanzar dimensiones que parecen inexistentes, es decir, la visualización permite trabajar con información que se encuentra en un espacio N-dimensional, en algunos casos por la naturaleza de sus características, otros por la simulación de algún evento, además de tener la ventaja de que N pueda ser muy grande. Para lograr la tarea de la VDC, esta información debe estar representada de 1 a 3 dimensiones para que pueda ser interpretada, analizada, procesada y documentada fácilmente por el ojo humano.

Para lograr la visualización de datos, es necesario aplicar el proceso de reducción de dimensión de los datos. Este proceso de reducción se conceptualiza como sigue: supongamos una matriz **X** (matriz de entrada) que representa a nuestro conjunto de datos reales en un espacio de N dimensiones; y la matriz **Y** (matriz de salida) que representa a nuestra información en un espacio reducido de M dimensiones ($M \leq N$). El problema de reducción de dimensiones y la visualización correspondiente se lleva a cabo a través de una función **F** o transformación **T** que se representa por:

$$\{T, F\} : \quad R^N \rightarrow R^M \qquad \text{donde:} \quad M \leq N; \qquad y \qquad M \in \{1,2,3\}$$

Para el propósito de la visualización de datos, M representa al espacio 1D, 2D o 3D, debido a que son las dimensiones que son vistas por el ojo humano. Además el espacio reducido M es menor o igual al espacio original N.

Sea $\mathbf{P_i} = (P_{i1}, P_{i2}, .. , P_{iN})$ el vector que representa a todo un conjunto de datos que los ubica en un lugar único del espacio original N-dimensional de la matriz - de entrada - **X** (por ejemplo, entiéndase en el caso del espacio de 3 dimensiones al "lugar único" como el PUNTO). Cada $\mathbf{P_i}$ contiene un total de N datos y la matriz **X** se conforma de Q número de vectores **P**, entonces el índice i puede variar desde 1 hasta Q ($1<i<Q$) representando cada renglón de la matriz de entrada a un conjunto de datos diferentes.

Si se desea representar a todos los datos de un solo eje o a lo largo de una dimensión, se tiene al *vector-dimensión* $\mathbf{x_j} = (P_{1j}, P_{2j}, .. , P_{Qj})$, donde cada $\mathbf{x_j}$ representa al conjunto de datos para cada dirección

(cada columna de la matriz **X**), el valor máximo que puede tomar la **j** es el valor de Q. Para aclarar la definición del *vector-dimensión*, consideremos por ejemplo, el caso del espacio de 3 dimensiones donde se representa un punto como **V** = (x, y, z), donde **x**=**V**, P_1=x, P_2=y, P_3=z y N=3. Regresando a nuestra definición, entonces a la matriz de entrada **X** en general la podemos presentar como sigue:

$$\mathbf{X} = \begin{bmatrix} P_{11} & P_{12} & \cdot\ \cdot & P_{1N} \\ P_{21} & P_{22} & \cdot\ \cdot & P_{2N} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ P_{Q1} & P_{Q2} & \cdot\ \cdot & P_{QN} \end{bmatrix}$$

Donde los vectores $\mathbf{P_i}$ (renglones) representan un punto en el espacio de datos y los mostramos enseguida:

$$P_1=(P_{11}, P_{12}, \ldots, P_{1N}),\ P_2=(P_{21}, P_{22}, \ldots, P_{2N}),\ \ldots,\ P_i=(P_{i1}, P_{i2}, \ldots, P_{iN}),\ \ldots,\ P_Q=(P_{Q1}, P_{Q2}, \ldots, P_{QN})$$

Y los vectores $\mathbf{x_j}$ (columnas) son los j-ésimos componentes en el espacio de datos:

$$x_1=(P_{11}, P_{21}, \ldots, P_{Q1}),\ x_2=(P_{12}, P_{22}, \ldots, P_{Q2}),\ \ldots,\ x_j=(P_{1j}, P_{2j}, \ldots, P_{Qj}),\ \ldots,\ x_N=(P_{1N}, P_{2N}, \ldots, P_{QN})$$

Al aplicar la transformación de reducción **T**, se generan nuevos datos $\mathbf{D_i} = (D_{i1}, D_{i2}, \ldots, D_{iM})$, donde M es la nueva dimensión de la matriz **Y** de salida. El valor máximo que puede tomar i es Q que representa al total de vectores **D**. Análogamente se tiene un conjunto de vectores $\mathbf{y_j} = (y_{1j}, y_{2j}, \ldots, y_{Qj})$ que es el *vector-dimensión*, el cual corresponde a nuestro espacio reducido. La matriz de salida **Y** se puede representar entonces:

$$\mathbf{Y} = \begin{bmatrix} D_{11} & D_{12} & \cdot\ \cdot & D_{1M} \\ D_{21} & D_{22} & \cdot\ \cdot & D_{2M} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ D_{Q1} & D_{Q2} & \cdot\ \cdot & D_{QM} \end{bmatrix}$$

La transformación de datos se puede representar, ya sea a través de forma lineal: $\underline{Y} = \mathbf{T}\ \underline{X}$, o de forma no lineal: $\underline{Y} = \mathbf{F}(\underline{X})$. Sin embargo, los procesos anteriores no son únicos, existen otras alternativas dentro de la categoría de la transformación no lineal que se basa en el uso de algoritmos inteligentes, las cuales proporcionan una mejor representación topológica del espacio de datos, como explicaremos más adelante.

La matriz **Y** resultante son los datos reducidos, y estos datos reducidos son los que se visualizarán con alguna técnica. Los datos de la matriz **Y**, para propósitos de nuestro trabajo, serán representados, es decir, serán visualizados en el espacio bidimensional, y serán visualizados en forma de imagen (espacio bidimensional). Estos son los resultados de los datos procesados a través del procedimiento de reducción de dimensión, y están listos para su análisis.

II. 2. 2. El Análisis

El análisis de los datos reducidos nos da información específica del fenómeno, la cual depende de la interpretación y del contexto científico de los datos. El análisis representa los cambios en la imagen que se está tratando. Se puede aplicar varios procesos que auxilian al investigador a comprender su problemática y/o a mejorar la representación, por ende la explicación de la misma. Una interpretación del análisis consiste en comparar la información de los datos ya reducidos y visualizados para observar: 1. los cambios que sufre el fenómeno en sus diferentes tiempos; 2. el efecto de cambiar algún parámetro, o alguna condición del modelo o del experimento que genere los datos.

II. 3. Notaciones para el Análisis del Sistema ViAn

La metodología que utilizaremos para desarrollar el análisis del sistema está basada en los pasos de la Ingeniería del Software. Además, para el desarrollo de algunos diagramas nos apoyaremos en un modelado estructurado, el Lenguaje de Modelado Unificado, UML (*Unified Modeling Language*) [Jacobson, 2000]. El análisis del sistema lo haremos desde varias perspectivas que se reflejan con los siguientes diagramas: los casos de uso, el modelado de procesos, los estereotipos y los requerimientos de interfaz. Todos estos diagramas son generados en base a los requerimientos del sistema.

II. 3. 1. Requerimientos

Según los pasos de la ingeniería del software, existen tres tipos de requisitos o requerimientos, que son los siguientes [Pressman, 1997]:

1. Los requerimientos normales,
2. los requerimientos esperados; y
3. los requerimientos innovadores.

II. 3. 1. 1. Requerimientos normales

En los requerimientos normales se declaran objetivos y metas para un producto o sistema durante las reuniones con el cliente. Si estos requerimientos están presentes, el cliente quedará satisfecho. Ejemplo de los requerimientos normales podrían ser peticiones de tipos de presentación gráfica, funciones específicas del sistema y niveles definidos de rendimiento. Para nuestro trabajo, los detalles de estos requerimientos los encontraremos en el capítulo III, como "*Requerimientos del usuario*".

II. 3. 1. 2. Requerimientos esperados

Los requerimientos esperados son implícitos al producto o sistema y pueden ser tan fundamentales que el cliente no los declara explícitamente. Sin embargo, la ausencia de estos requerimientos sería motivo de una insatisfacción significativa. Ejemplos de los requerimientos esperados, son la facilidad de interacción entre el hombre y la máquina, el buen funcionamiento y fiabilidad general, otro ejemplo sería la facilidad de instalación del software.

Para nuestro contexto, los requerimientos esperados los encontraremos en el capítulo III como "*Requerimientos técnicos*", y no solo eso, sino que se encontrará dentro de los "*Requerimientos del usuario*". esto es debido a que las peticiones del usuario que son implícitas, se aterrizan con el análisis del problema del desarrollador y se vuelven en Requerimientos del usuario explícitos.

II. 3. 1. 3. Requerimientos innovadores

Los requerimientos innovadores son características que van más allá de las expectativas del cliente y suelen ser muy satisfactorias. Para comprender mejor esto, digamos que por ejemplo, se pide un software procesador de textos con las características estándares, y el producto entregado contiene ciertas capacidades de diseño de página que resultan muy válidas y que no eran esperadas.

A lo largo del desarrollo del sistema, se espera que estos requerimientos se presenten, y serán notorios para el cliente en el momento de la conclusión del proyecto. Por ende, no se especifica una sección para ver el contenido de los requerimientos innovadores. Sin embargo, se reflejarán en el proyecto, por ejemplo, en el diseño del sistema, nos encontraremos con el almacenamiento de datos que no se especificaron y que el usuario podrá encontrarle utilidad importante. Otro ejemplo, sería el diseño gráfico en su última versión, que se presentará en la implementación del sistema.

II. 3. 2. Casos de uso

Los casos de uso pueden tener diferentes definiciones, lo importante es que son quienes ayudan a generar el análisis de una forma fácil, estos nos proporcionan la perspectiva del usuario. Expliquemos lo que son los casos de uso: Los casos de uso son una técnica que permite mejorar la comprensión de los requerimientos de un sistema, son descripciones narrativas (de alguna forma) de los procesos del sistema. Un caso de uso también se puede ver como una descripción de un *proceso* de principio a fin relativamente amplia, descripción que suele abarcar muchos pasos o transacciones; normalmente no es un paso ni una actividad individual del proceso.

Los casos de uso en ocasiones se consideran el *pre-análisis* de un sistema. Esto explica, que los casos de uso representan los procesos o actividades que el usuario desea realizar en el sistema, e indica cómo desea interactuar con ellos. Al analizar como es que el usuario realizará las actividades que indican sus requerimientos, no estamos influyendo en la decisión de qué filosofía manejará el sistema. Lo que significa que, los casos de uso pueden modelarse para sistemas Orientados a Objetos (OO) [Booch], y para sistemas orientados a eventos (como nuestro sistema ViAn).

Para los casos de uso, y en general para el resto del desarrollo del sistema, manipularemos el concepto de proceso. Nuestro sistema no es OO, ViAn está orientado a eventos, por lo que los procesos serán representativos en los modelados para el análisis del sistema y para el diseño del sistema.

II. 3. 2. 1. Concepto de proceso

Presentamos el concepto de proceso para tener una idea completa de lo que estamos tratando: Un proceso describe de principio a fin, una secuencia de actividades (eventos), de las acciones y las transacciones que se requieren para lograr un objetivo (producir u obtener algo de valor para un actor).

II. 3. 2. 2. Diagramas de casos de uso

Estos casos de uso se representan gráficamente a través de los *diagramas de casos de uso*, y así es como permiten mejorar la explicación de los requerimientos del sistema. Estos diagramas se conforman con los procesos, los actores y sus relaciones. Los elementos gráficos de los diagramas de casos de uso, según la notación estándares en UML son los que presentamos en la tabla II.I.

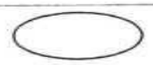
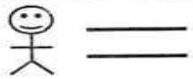
	*Caso de uso*	Representa a un caso de uso (a la funcionalidad o servicios provenientes de un sistema para el usuario).
	*Actores*	Un actor representa a un usuario que interactúa con el sistema, el actor puede ser una persona (figura humana) u otro sistema externo, por ejemplo, un sistema de archivos (dos líneas horizontales paralelas).
	*Relaciones*	Es la asociación que existe entre los casos de uso y los actores. O las asociaciones de los casos de uso entre sí (la secuencia de los casos).
 <<uses>>	*Relación USES*	Indica que una instancia (parte) del caso A[1] incluirá especificaciones del caso B[2]. También se conoce como Relación *INCLUDE*.
 <<extends>>	*Relación EXTENDS*	Indica que el caso B puede formar parte del caso A, a través de una condición, es decir, B puede ser opcional para el usuario o para el proceso.
 <<realize>>	*Relación REALIZE*	Indica que el caso A es parte del caso B, es decir, el caso B es la forma en que se realiza el caso A, puede ser una técnica o el procedimiento clave.

Tabla II.I. Elementos de los diagramas para los casos de uso

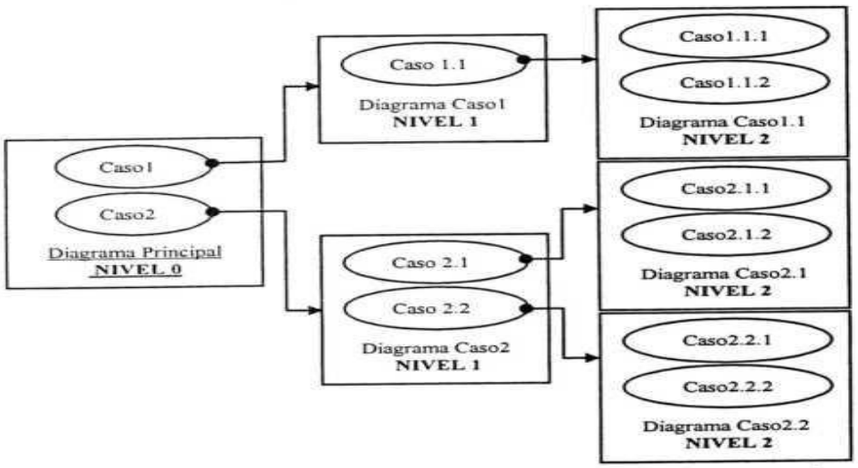


Figura 2.1. Niveles en los diagramas de casos de uso

Los diagramas de casos de uso se clasifican en niveles (observe la figura 2.1), que va desde el 0 (cero) hasta cualquier número, aunque lo recomendado es llegar hasta el nivel 3. El nivel 0 es el diagrama que presenta a los casos de uso más genéricos del sistema, puede que exista solamente un caso de uso y sus actores. El siguiente nivel es un diagrama que representa a un caso de uso del diagrama con el nivel anterior, y así sucesivamente. Pueden existir varios diagramas con el mismo nivel, esto es, si en un diagrama existen más de un caso de uso y estos necesitan ser representados en otros diagramas, estos diagramas se encuentran en el mismo nivel, ya que vienen del mismo diagrama con el nivel anterior.

1 Caso de uso A, es el caso de uso que inicia la relación o asociación entre los casos (se encuentra en el inicio de la flecha).

2 Caso de uso B, es el caso de uso que se encuentra al final de la asociación de los casos (se encuentra al final de la flecha, es decir, a quien apunta).

II. 3. 2. 3. Documentación de casos de uso

Los diagramas de casos de uso son acompañados de una breve documentación, que describen o narran lo que está sucediendo entre los procesos y los actores. Esta documentación consta de: 1. El nombre del caso de uso, 2. El actor o actores que contiene el diagrama, 3. El propósito del caso de uso; y 4. la descripción del caso de uso que se encuentra representado en el diagrama.

II. 3. 2. 4. Ejemplo de un diagrama de casos de uso

En la figura 2.2 presentamos un diagrama de casos de uso, en el que se encuentra un actor que es un usuario que se relaciona con el caso de uso *"Buscar una palabra"*, este caso de uso (llamémosle A) se encuentra relacionado con varios casos de uso, y con otro actor de tipo sistema externo, uno de *"Archivo de palabras"*. El caso A **usa** (relación <<uses>>) a otro caso llamado "Selecciona un tipo de palabra", a su vez, A se relaciona con "Detecta error en la palabra" **incluyéndolo** (relación <<include>>) a su procedimiento. El caso *"Buscador de palabras"* es una forma de **realizar** (relación <<realize>>) el caso A.

El caso A tiene una asociación con el caso de uso *"Despliega la palabra"* (caso B), esto significa que al concluir el procedimiento del caso A, se precede al caso B. El caso B a su vez tiene una relación <<include>> donde incluye un caso opcional para el usuario.

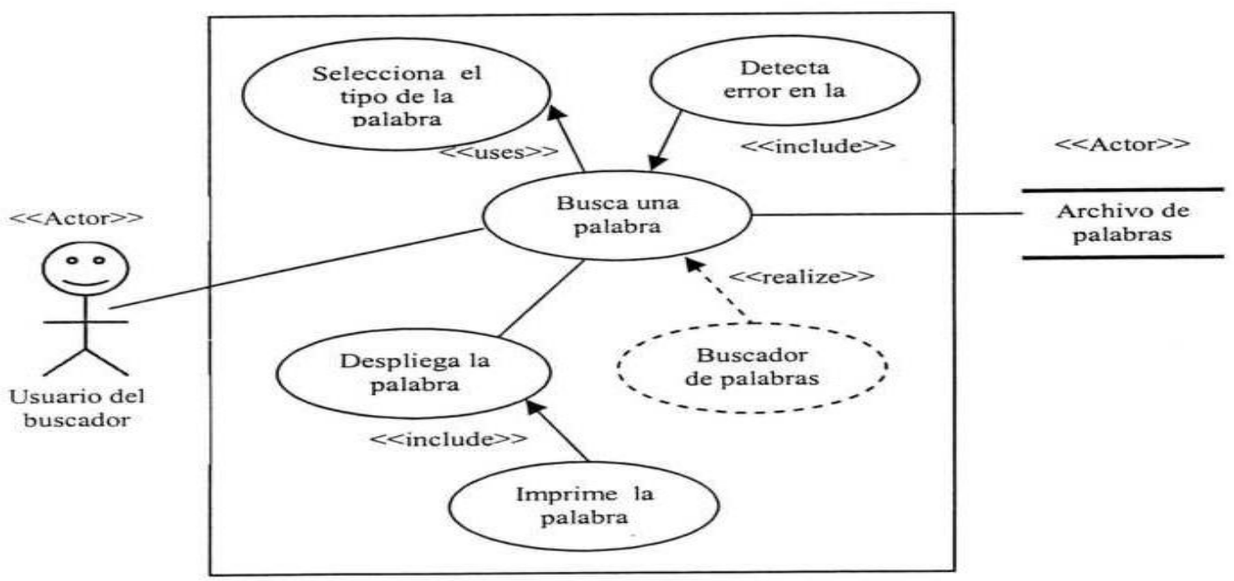


Figura 2.2. Ejemplo de un diagrama de casos de uso

II. 3. 3. Modelado de procesos

El modelado de procesos consiste en organizar a nuestro sistema ViAn en procesos específicos y estudiar la relación que existe entre ellos y el comportamiento de esta relación. Para tener un modelado de procesos completo, es necesario representar a los procesos y elementos con dos diferentes modelados y con dos diferentes diagramas, estos son los siguientes:

- Modelado del flujo de procesos y Modelado del flujo de información.
- Diagramas de estereotipos y Diagramas de secuencia (a nivel análisis)

El modelado de procesos que establecemos, es la combinación de la Ingeniería del software y el UML. Para nuestras necesidades, ampliamos un poco más el panorama del modelado modificando la notación del modelado de flujo de procesos y el modelado de flujo de información, que están establecidos.

El modelado de procesos tiene muchas aplicaciones y literatura enfocada a los sistemas Orientados a Objetos (OO), y también son de utilidad para los sistemas que manejan procesos, actividades y eventos, sin la necesidad de orientarse a objetos, como lo presentaremos en este modelo de procesos para nuestro sistema orientado a eventos, ViAn.

II. 3. 3. 1. Modelado de flujo de procesos

El modelado de flujo de procesos es un conjunto de funciones o actividades que debe proporcionar el sistema y se encuentran relacionados, además se detallan con otros sub-procesos del sistema. La relación de las funciones se muestra en forma de secuencia de procesos, con lo que hace más fácil comprender la relación de las actividades del sistema.

Conozcamos las figuras de la notación (tabla II.II) del modelado de flujo de procesos. Un proceso se representa con un óvalo, el flujo de datos de un proceso a otro con una flecha, y las flechas que sólo entran al proceso indican la participación del actor para el inicio de proceso, que puede ser opcional, y la flecha que sólo sale indica en fin del proceso, o en dado caso una opción para finalizar el proceso principal. Un diagrama del modelado tiene un proceso principal, el cual se forma de varios subprocesos.

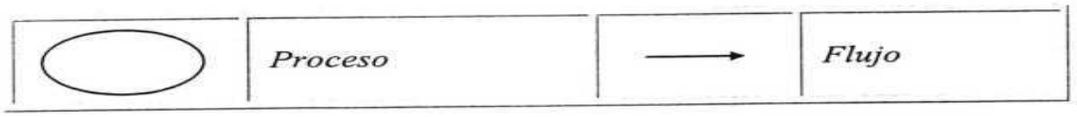


Tabla II.II. Elementos del modelado de flujo de procesos

Por ejemplo, representamos en la figura 2.3 el modelado de flujo de procesos para calcular el promedio de X números: El proceso principal es "Calcular el promedio de X datos" y tiene una entrada (flecha izquierda) y una salida (flecha derecha), este proceso se forma por otros dos procesos, y estos a su vez podrían formarse de más procesos. El sub-proceso "Suma de X datos", cuenta con la entrada del sistema, después de realizar sus cálculos, el flujo indica que (flecha entre los dos sub-procesos) el siguiente sub-proceso es "División entre X", y este último tiene la salida del sistema.

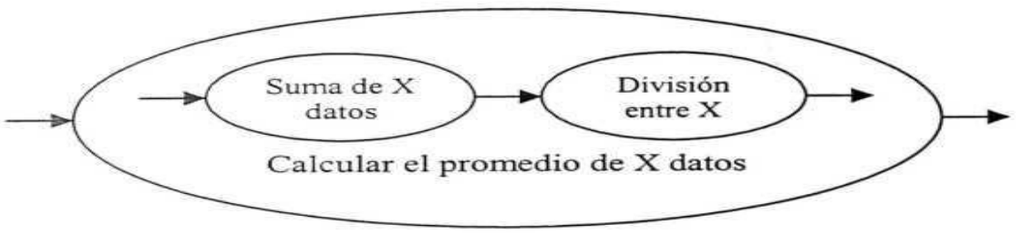


Figura 2.3. Ejemplo del modelado de flujo de procesos

II. 3. 3. 2. Modelado de flujo de información

El modelado de flujo de información nos proporciona más elementos sobre las estructuras de los datos, que el "Modelado de flujo de procesos", ya que éste último se encuentra integrado con el "Modelo de datos" con el objetivo de proporcionar una indicación de cómo es que fluye la información a través del sistema. Se muestran los parámetros de entrada y salida para cada proceso, proporcionando una indicación de cómo el proceso transforma la información para facilitar la definición de una función. El *modelo de datos* es una actividad que se concentra en los objetos de datos, sus atributos y relaciones para el flujo de datos. Para sistemas no orientados a objetos, el modelado de datos se presenta como un auxiliar para el flujo de información entre las entidades, procesos y respuesta de los eventos a través de sus parámetros entrada-salida. Es así como se refleja el modelado de datos en nuestro sistema ViAn.

Los elementos que utilizaremos para el diagrama del modelado de flujo de información, serán los mismos elementos que utilizaremos para el diagrama del modelado de flujo de procesos, y que presentamos en la tabla II. II., anexándoles un elemento llamado "objeto". Este elemento envía y recibe la información procesada, además de que se observa el tipo (característica) de información que esta fluyendo a través de los procesos. El elemento objeto, se representa como lo mostramos en la tabla II.III:

	Objeto

Tabla II.III. Elemento objeto

Para nuestro ejemplo, el objeto es una base de datos donde se almacenan resultados. Observemos en la figura 2.4 al modelado de flujo de información de nuestro del modelado de flujo de procesos que presentamos en la figura 2.3 de la suma de números.

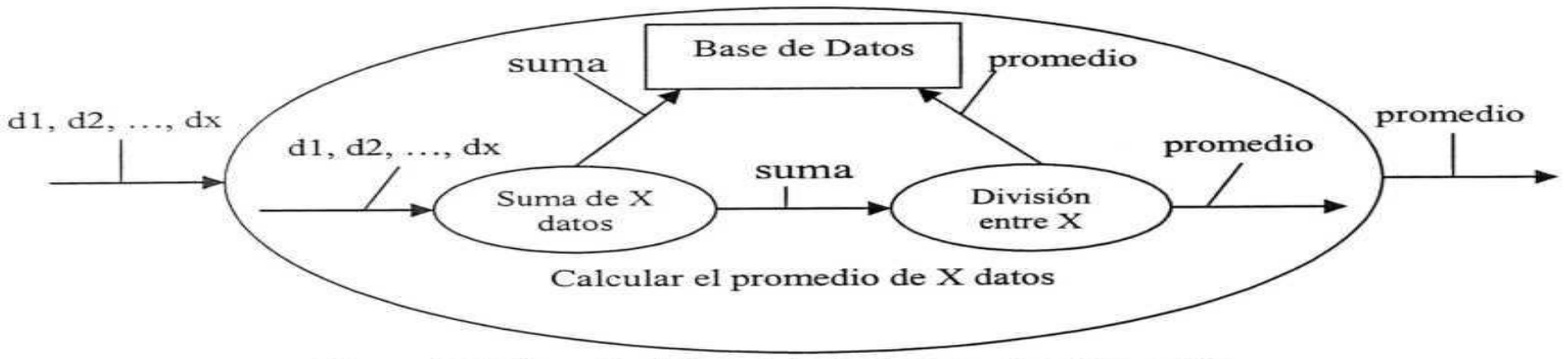


Figura 2.4. Ejemplo del modelado de flujo de información

En la figura anterior presentamos que los datos de entrada al proceso principal son los datos: d1, d2, .., dx y la salida es el promedio de los datos de entrada. En los subprocesos vemos que los datos de entrada mencionados son los mismos datos de entrada del primer subproceso. El sub-proceso de la suma de los datos dados (Suma de X datos) arroja como dato de salida a "suma" y este dato a su vez es un dato de entrada para el segundo subproceso, la división de Suma entre el número de datos dados (División entre X). El dato "suma", además de ser enviado al segundo subproceso, se envía al objeto "Base de Datos" que lo almacena en memoria. El segundo subproceso tiene como dato de salida a "promedio", mismo dato de salida del proceso principal. Este dato de salida también es enviado al objeto "Base de Datos" para su almacenamiento en memoria.

II. 3. 3. 3. Diagramas de estereotipos

Los diagramas de estereotipos son una herramienta para el análisis del sistema y ayuda a hacerlo robusto. El análisis de robustez auxilia a redefinir los casos de uso, y es una buena base para el diseño del sistema. El análisis del sistema cuenta con estereotipos para clasificar a los procesos, interfaces y elementos pasivos, como un sistema de archivo que el sistema contiene en funciones (u objetos, para sistemas OO) muy específicos. Los actores pueden ser representados en los diagramas de estereotipos.

Aunque nuestro sistema ViAn no es OO, los estereotipos son útiles para representar al sistema bajo la filosofía de los eventos. Se ajustan los mismos conceptos y el enfoque del tiempo en el sistema. Existen tres estereotipos, además del actor, y se representan con los íconos descritos en la tabla II.IV.

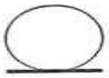
	*Interfaz*	La <<*interfaz*>> (*boundary* en inglés) es la función de interfaz a través del cual el usuario (actor) interactúa con el sistema. Y es con el único con quien la función de interfaz puede asociarse.
	*Entidad*	La <<*entidad*>> (*entity* en inglés) es un elemento pasivo, que no inicia interacciones por su propia cuenta. El elemento entidad, es de dominio del modelo. Ejemplos del elemento entidad son: sistemas de archivos y estructuras de datos.
	*Control*	Los elementos de <<*control*>> representan a los procesos (funciones o procedimientos), y son los que comunican con la interfaz y a las entidades del sistema. Este elemento puede asociarse con los elementos de interfaces, elementos de entidades e incluso con otros elementos de procesos.

Tabla II.IV. Elementos para los diagramas de estereotipos

Seguiremos con el mismo ejemplo de Calcular el promedio de X datos, presentado en el modelado de flujo de procesos y en el modelado de flujo de información (figuras 2.3 y 2.4). Ahora representaremos con la figura 2.5 a este sistema con estereotipos:

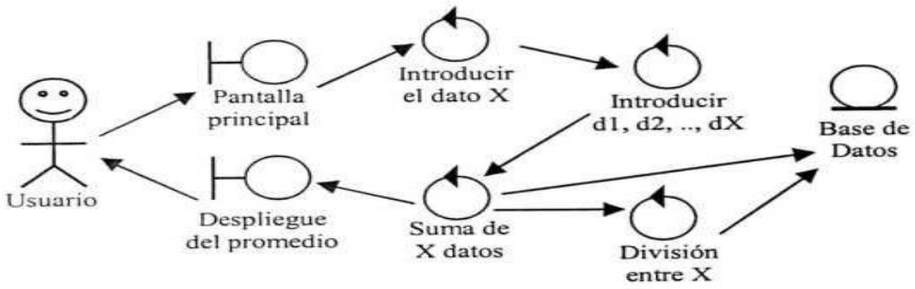


Figura 2.5. Ejemplo de un diagrama de estereotipos

En el diagrama de estereotipos vemos que el usuario interactúa solamente con los elementos de interfaz, la primer interfaz es la pantalla principal, esta llama al proceso de introducir el dato X y los datos d1, d2, .. , dX. Estos procesos se representan con el elemento de control, y los otros procesos se encargan de la suma y división correspondiente. En la figura 2.5 se observa como se comunican los elementos de control y los elementos de entidad, esto para almacenar datos que los procesos indiquen.

II. 3. 3. 4. Diagramas de secuencias

Los diagramas de secuencias muestran los escenarios más importantes que el usuario puede realizar en el sistema, y son extraídos del diagrama de estereotipos. Son una representación más detallada de los diagramas de procesos.

El tiempo se representa por líneas verticales punteadas, y los eventos (o actividades) por flechas horizontales, con las cuales los estereotipos interactúan. La secuencia puede tener diferentes opciones en el mismo proceso, se representan con ":" (dos puntos) al cambio de flujo de procesos. Los eventos son enumerados para facilitar la explicación de la secuencia, si existe una opción puede ser que algún(os) número(s) se repitan con el fin de poder seguir la secuencia a través de estos números. Al lado izquierdo del diagrama, se explican los eventos brevemente, para aclarar la notación y evento que se está realizando.

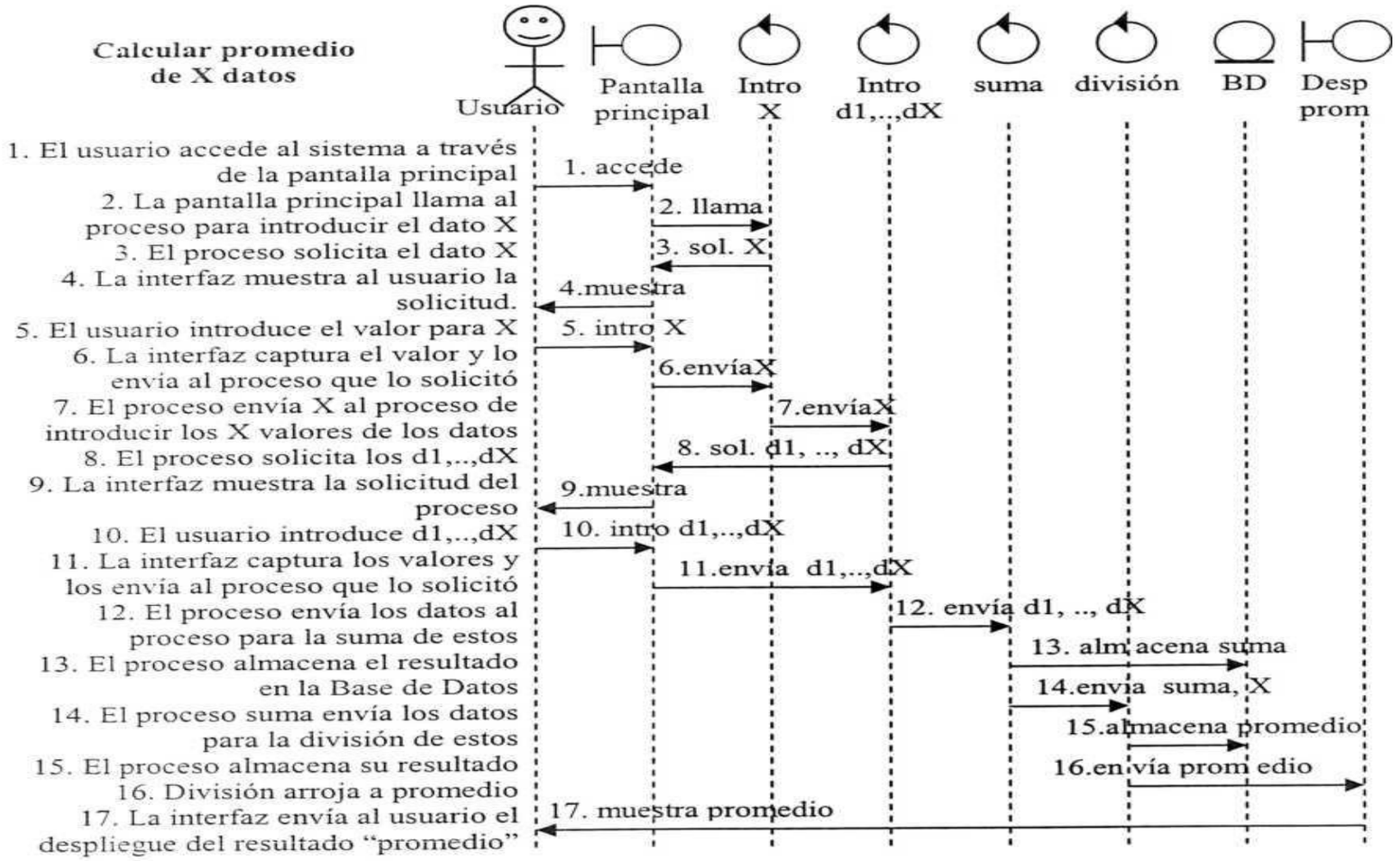


Figura 2.6. Ejemplo de un diagrama de secuencias

En la figura 2.6 presentamos a un ejemplo de un diagrama de secuencias, basándonos en el caso de la figura 2.5. Aquí presentamos cómo a través del tiempo se desarrolla la interacción con el sistema, y como va la secuencia de los procesos y las actividades entre los elementos que se especificaron para realizar el caso de uso del cual tiene origen dicho proceso o actividad.

II. 3. 4. Requerimientos de interfaz

Estos requerimientos de interfaz, especifican la forma en que el usuario va a interactuar con el software. Determinan los procesos a los que se tienen acceso, según las necesidades del proyecto a estudiar definidos a través de los casos de uso. Para especificar estos requerimientos es necesario haber definido primero los requerimientos del sistema y los casos de uso, esto es para que se definan las tareas orientadas al usuario y las tareas orientadas al sistema (procesos). En el análisis del sistema (capítulo III) presentaremos entonces a las pantallas que representan los elementos gráficos principales para interactuar con la aplicación y que concuerdan con las definiciones de los casos de uso. Aquí la característica de los estereotipos de interfaz, serán la base para definir adecuadamente estos requerimientos.

II. 4. Notaciones para el Diseño del Sistema ViAn

La metodología que utilizaremos para desarrollar nuestro sistema ViAn está basada en los pasos de la Ingeniería del Software. Debido a que nuestra filosofía está orientada a eventos, módulos o funciones, muchos modelados los encontraremos similares a los que la Ingeniería del Software nos ofrece, con esto quiero decir que utilizaremos bajo nuestras necesidades, los tipos de modelados y diagramas que más les den forma a la representación deseada para el diseño del sistema ViAn.

El diseño del sistema cuenta con varias perspectivas que se verán reflejadas con los diagramas, para el flujo de datos, el flujo de eventos, la secuencia de un evento, para la estructura del sistema y modelos para el diseño de interfaz. Además de que estableceremos desde un principio como es que se organiza la información del sistema.

II. 4. 1. Organización de la información

La organización de la información que utilizaremos, la podemos encontrar como "Diseño de los datos" **[Jacobson, 2000]** en la literatura. En el diseño de los datos, se seleccionan representaciones lógicas de los objetos de datos y se convierten en estructuras de datos. Para la información de las estructuras, se toman en cuenta los atributos y las relaciones entre los objetos o funciones. Al igual, vemos que una actividad la interpretaremos como módulos del programa que operan directamente sobre las estructuras. Es por esto que, solo nos basaremos en el diseño de datos para realizar la organización de la información.

En esta fase presentamos a la estructura de datos y su relación con el sistema. Definimos los atributos y relaciones de los datos con respecto a los eventos y procesos. Establecemos la estructura del sistema de archivos, que es donde se almacenan los datos resultantes de los procesos con formato propio de nuestro sistema. También presentamos la forma de organizar los datos que se encuentran en el sistema de archivos, desde el punto de vista del usuario, de cómo se definen y estructuran los proyectos del sistema ViAn.

Para representar un archivo con formato texto y su contenido utilizaremos la representación de una tabla sencilla con una sola celda y líneas dobles como se muestra en la figura 2.7. Un ejemplo simple de cómo se representa un archivo de texto y su contenido se muestra en la figura 2.8.

Figura 2.7. Notación para representar un archivo con formato texto

Este es el contenido de un archivo texto. 1201.2730 1210.0719

Figura 2.8. Ejemplo de la representación de un archivo con formato texto

II. 4. 2. Estructura del sistema

En la estructura del sistema se modelan las funciones y procedimientos del programa que cumplirán con las tareas de los procesos definidos a través de los casos de uso. Esta estructura muestra el esqueleto del sistema, es la forma en que el programa se está implementando. Es importante que el desarrollador conozca esta perspectiva antes de ahondar con los procesos, datos, flujos, etcétera. Esto debido a que es un plano general del sistema, y de ahí entonces es que podemos visualizar y canalizar mejor el motor del sistema, y la estructura global del sistema.

Esta parte del diseño del sistema puede ser un equivalente al diseño arquitectónico que ofrece la ingeniería del software. El objetivo primario del diseño arquitectónico es desarrollar una estructura del programa modular y representar las relaciones de control entre los módulos. Este mismo objetivo es el que tenemos al realizar la estructura del sistema.

Los elementos gráficos para la notación del diagrama que representa la estructura del sistema son los siguientes (tabla II.V) con sus respectivos significados:

	*Módulo del programa*	Dentro de él se encuentra el nombre del módulo en letras remarcadas (negritas o bold), e indica el tipo de módulo entre paréntesis donde puede ser: *proc* = procedimiento, o *func* = función.
	Conexión de módulos	Conecta a los módulos que se comunica directamente. Esta línea en ocasiones se ve acompañada de parámetros los cuales pueden ser datos o eventos. Si son eventos se especifica su valor.

Tabla II.V. Elementos para la estructura del programa

Un ejemplo de un diagrama para representar la estructura de un sistema es el que presentamos en la figura 2.9. Tiene un procedimiento principal llamado *Conversión_event* y a este le antecede su ventana de entrada *Conversión*, esta ventana consta de elementos gráficos. Existen tres módulos con diferentes opciones de conversión de datos, según la opción llama a la función que contiene la fórmula que le corresponda y un procedimiento despliega el resultado de la fórmula indicada.

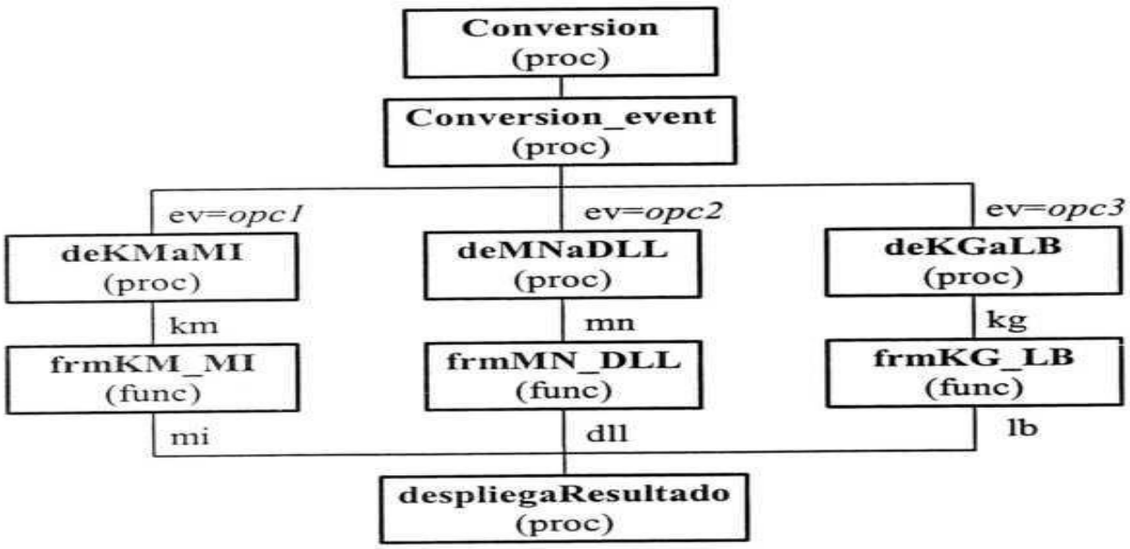


Figura 2.9. Ejemplo de una estructura de sistema

II. 4. 3. Flujo de datos

Para modelar al flujo de datos, la ingeniería del software ofrece a los Diagramas de Flujo de Datos (DFD), los cuales se enfocan a una parte de la estructura del sistema para asignar y dirigir los datos entre los módulos, determinando cuales datos sean de entrada, de proceso y de salida. Entonces nuestro flujo de datos se realiza basándonos en la notación estándar de los DFD.

El flujo de datos indica a que procesos se dirigen los datos de entrada del sistema, y que datos son los de salida para estos mismos, y a su vez los datos de salida se convierten en datos de entrada para otros procesos. El flujo tendrá en ocasiones alternativas para dirigirse a distintos procesos, por lo tanto, los datos de entrada también podrán ser distintos en cada proceso. La notación para formar a los diagramas que representan nuestro flujo de datos son los que presentamos en la tabla II.VI.

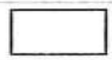
	*Módulo del programa*	Dentro de él se encuentra el nombre del módulo. Este módulo debe encontrarse en el código del programa.
	Proceso que respalda al módulo	Dentro de él se encuentra el nombre del proceso. Este proceso debe estar modelado en un previo análisis.
	*Elemento que contiene datos*	Puede ser un arreglo, una estructura de datos, archivos. Dentro de él se encuentra el nombre del elemento.
	*Flecha*	Indica hacia donde va el flujo de datos, y la acompaña los datos que se envían.

Tabla II.VI. Elementos para los diagramas para el flujo de datos

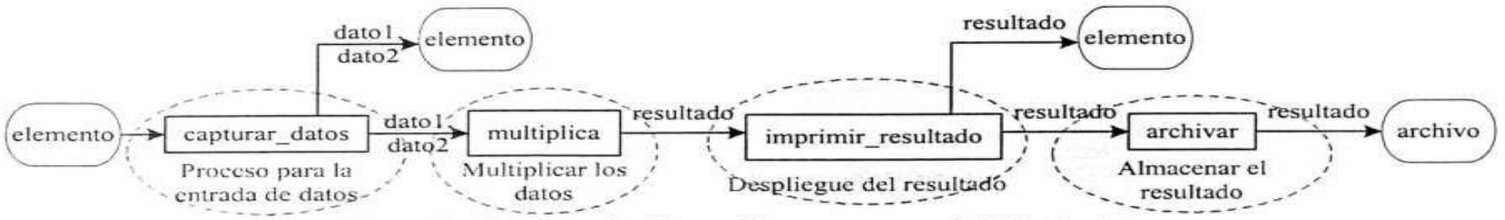


Figura 2.10. Ejemplo de un diagrama para el flujo de datos

Un ejemplo de un diagrama para el flujo de datos podría ser la multiplicación de dos datos, y lo representamos en la figura 2.10. Aquí tenemos a dos elementos, "*elemento*" que es una estructura de datos con tres campos: *dato1*, *dato2*, y *resultado*, y "*archivo*" que es el que se almacena en memoria secundaria. Cada vez que un módulo le envía datos a *elemento*, significa que los va a actualizar por haber sido procesados. Para este ejemplo suponemos la existencia de eventos, por lo que en un principio, elemento no envía dato alguno, significa que se generó un evento, además de que todos los módulos se ejecutan con este mismo evento.

II. 4. 4. Flujo de eventos

El modelado de flujo de eventos, es una perspectiva más, de gran importancia, para el diseño del sistema ViAn. En este se modelan a los eventos que se manejan en el programa, eventos que indican qué módulos del sistema son los que van a ejecutarse, obedeciendo al proceso deseado por el usuario. Un diagrama de flujo de eventos nos muestra como a partir de la generación de un evento, se activan o desactivan otros eventos, especificando así el control que se ejerce en el manejo de los procesos y datos.

Con los diagramas de flujo de eventos presentaremos cómo el flujo de eventos depende de los eventos mismos, aunque en ocasiones será necesario el auxilio de los datos, es decir, el flujo de eventos no depende solo de los eventos mismos sino también de los datos que se estén procesando. Enseguida presentamos la tabla II.VII que contiene a los elementos gráficos que se utilizan en la formación de estos diagramas de flujo de eventos.

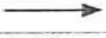
	Denota al evento que se genera, dentro del círculo doble se encuentra en nombre o los nombres de los eventos, cada uno encerrado entre apóstrofes (''). De este círculo parten las flechas del flujo. Si existe la dependencia de algún dato para el flujo de eventos, se escribe el nombre del dato en negritas, en la parte inferior dentro del círculo externo.
	Eventos que se activan a partir de la generación del evento especificado. Dentro de él se encuentra el nombre del evento a activar entre apóstrofes ('').
	Flecha que indica el flujo para la activación de un evento.
	Eventos que se desactivan a partir de la generación del evento especificado. Dentro de él se encuentra el nombre del evento a desactivar entre apóstrofes ('').
	Flecha que indica el flujo para la desactivación de un evento.

Tabla II.VII. Elementos para los diagramas de flujo de eventos

Cuando son dos eventos o más en un mismo círculo (en cualquiera de los dos círculos), es debido a que los eventos que se encuentran en éstos, cumplen los mismos procesos, dependen de los mismos datos (si existe dependencia de datos) y cumplen con el mismo flujo de eventos. Existen situaciones en que los diagramas para flujo de eventos sean muy sencillos, en la cual existe la posibilidad de que un evento se desactive a si mismo y se active, o sólo se desactive. Cuando esta situación sucede, generalmente es para desactivarse durante el proceso que se esté desarrollando para no ejecutarlo más de dos veces simultáneamente. En éste caso, la flecha del flujo, ya sea para activarse o desactivarse, iniciará del evento generado e irá hacia él mismo.

Y cuando un evento no active ni desactive a ningún otro evento, entonces no es necesario representarlo, pero si se desea así, entonces solo se presenta el evento generado en el círculo doble con ninguna flecha y con ningún otro evento que lo acompañe, véase el ejemplo en la figura 2.11(a).

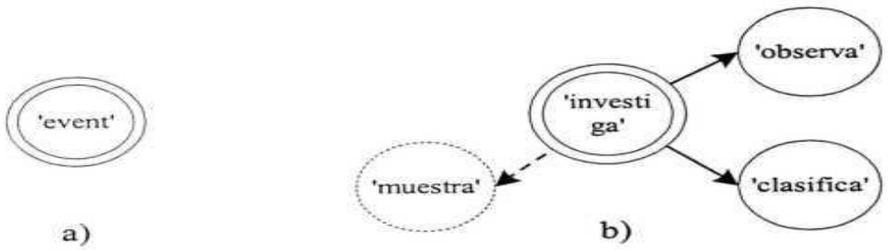


Figura 2.11. Los diagramas de flujo de eventos más sencillos:
(a) Diagrama del flujo de eventos para *event* que no activa ni desactiva a ningún otro evento.
(b) Ejemplo de un diagrama de flujo de eventos para el evento *investiga*.

Ahora presentamos al ejemplo de un diagrama de flujo de eventos con activación y desactivación de eventos, donde solo depende el evento generado. La figura 2.11(b) muestra el diagrama para el evento *investiga*, donde activa a dos eventos, *observa* y *clasifica*, y desactiva al evento *muestra*.

II. 4. 5. Secuencia de un evento

La secuencia de un evento representa al sistema ViAn desde la perspectiva de un programa con el manejo de elementos de interfaz y explica la forma en la que son ejecutados los módulos principales (procedimientos ViAn y ViAn_event) a causa de un evento generado por el usuario o internamente. La secuencia de un evento muestra el enlace que existe entre los módulos del sistema ViAn y módulos propios del lenguaje IDL, y como se comportan con la intervención del usuario. Para realizar esta parte del diseño, nos basamos en la notación de los diagramas de secuencia, parte del análisis del sistema, y la encontramos en "Estereotipos". Presentaremos más adelante de que trata un programa con el manejo de elementos de interfaz, y la correspondiente estructura de datos.

II. 4. 6. Diseño de interfaz

El diseño de interfaz detalla la forma en que el usuario va a interactuar con el software. Determina la estructura de las pantallas cumpliendo con los requerimientos de interfaz vistos en el "Análisis del sistema", además especifica la interfaz teniendo presente todas las perspectivas del "Diseño del sistema". En el diseño de interfaz presentaremos las relaciones entre las pantallas, cómo se comunican y cómo se encuentran ordenadas a través de los casos de uso, la estructura de datos y de los estereotipos de interfaz.

II. 5. Lenguaje para la implementación de ViAn

La implementación del sistema ViAn se desarrollará en IDL, llamado así por sus siglas en inglés de "*Interactive Data Language*" que significa Lenguaje para Datos Interactivos. Presentaremos las características del lenguaje, sus elementos, y describiremos la justificación en la que nos apoyamos para desarrollar nuestro proyecto en este ambiente. Además, como se describirá., el desarrollo de la aplicación en este ambiente es un requerimiento del sistema

II. 5. 1. Definición y justificación

IDL es una aplicación desarrollada por el *Research System Inc* con el propósito de realizar animaciones, visualizaciones, gráficas, contornos, mapas, etc. En particular, cuenta con un lenguaje de programación propio con la capacidad de definir módulos y así estructurar mejor los programas. Además cuenta con módulos propios que permiten facilitar la programación para crear interfaces, establecer eventos, realizar lectura y escritura de diferentes formatos, realizar contornos, superficies, gráficas, animaciones, etcétera. **[Gumley, 2002]**

A nivel de aplicación el IDL permite realizar varias tareas como visualización, análisis de datos, desarrollo de aplicaciones donde se requiere de mucho análisis matemático y técnicas de despliegue gráfico, ahorrando líneas de código si se compara por ejemplo, con lenguajes como C o FORTRAN. Se pueden entonces desarrollar programas sofisticados para la visualización de datos, como en nuestro caso, que establecimos desarrollar al sistema ViAn con técnicas de reducción de dimensión de datos y con técnicas para el análisis de datos visualizados.

II. 5. 2. Filosofía de eventos (*Widgets*)

Un evento, es un suceso generado por algo externo al sistema como lo es el usuario. El usuario realiza su petición al sistema alterando al elemento a través del cual se comunica con la aplicación, la interfaz gráfica. La filosofía del lenguaje IDL en las interfaces, es el manejo de eventos, los cuales son manipulados a través de *Widgets*. El nombre de *Widgets* se refiere a un elemento gráfico donde el usuario interactúa con el traspaso de información del programa. Es decir, permite la definición y manipulación de botones, barras deslizadoras y campos de texto.

El paquete de herramientas *Widgets* es un conjunto de módulos de IDL que forman las interfaces. Un API, que significa "Interfaz de Programación de Aplicación" (*Application Programming Interface*), es útil para construir programas que manejan eventos con interfaces gráficas (programas *Widgets*).

Un proyecto se desarrolla a través de la definición de módulos (procedimientos o funciones) especificando un problema a resolver. El lenguaje IDL está orientado a definir módulos, de forma que, la aplicación se realice con un enfoque visual y flexible al usuario. Estos módulos forman a los programas *Widgets*, que funcionan a través de dos módulos específicos, uno donde los *Widgets* son definidos, y el otro donde se encuentran los procedimientos y funciones que rigen a los *Widgets*. Es decir, la parte de la interfaz (despliegue visual) del programa Widget, es generada en el *módulo de la definición del Widget*, y la "acción" de un programa Widget es generada en el *módulo del evento handler* (más cercano).

II. 5. 2. 1. Estructura de un programa Widget

Para explicar un poco más al concepto de Widget, en la figura 2.12 se muestra la estructura de un programa Widget, donde se observan las relaciones entre elementos para el manejo de eventos. Antes de describir el proceso, describiremos los elementos de la estructura de un programa Widget:

a) *Módulo de la definición del Widget:* En este módulo se encuentran definidos los *Widgets*, sus atributos y la forma en que se desplegarán en la interfaz gráfica. Este módulo se ejecuta una sola vez durante la vida del programa Widget. Este módulo es como cualquier procedimiento normal, en el cual se puede definir parámetros de posición o de llave.

b) *Módulo del evento handler:* En este módulo se encuentran los procedimientos y/o funciones que rigen las acciones que se realizarán según el evento que generó el usuario y según el Widget con quien fue generado dicho evento. Este módulo se ejecuta las veces que sea necesario durante la vida del programa Widget.

c) *Interfaz gráfica:* La interfaz gráfica es el despliegue de los Widget y es quien interactúa directamente con el usuario.

d) *Ventana administradora:* Imaginemos a un doble fondo para la interfaz gráfica [1]. Esta se encuentra en estado de espera hasta que el usuario genera un evento a través de la interfaz gráfica. Esta ventana es responsable de saber el tipo de evento que se generó y a través de qué Widget, y es responsable de asociar correctamente al evento y a la acción que debe ocurrir como consecuencia de éste. También es quien se encarga del paso de información de los eventos, formar a la estructura del evento y que sea recibida por el módulo del evento *handler*.

---

1 La ventana administradora es imaginaria, ya que ha sido creada solo para la mejor explicación de los programas *Widgets*, no existe de forma física, ni existe un módulo para esta ventana, sino que son otros módulos quienes realizan su trabajo.

e) *Widget_control:* El Widget_Control es parte del elemento al que nombramos ventana administradora, es el procedimiento que se encarga del despliegue de la interfaz y de manipular la información de la estructura del evento.

f) *XManager:* El XManager es otra parte de la ventana administradora, es el comando que se encarga del registro de las etiquetas, esto es, de relacionar a los dos módulos del programa entre ellos para saber que pertenecen al mismo programa, además esta parte se encarga de formar correctamente a la estructura de evento que será enviada al módulo del evento *handler*.

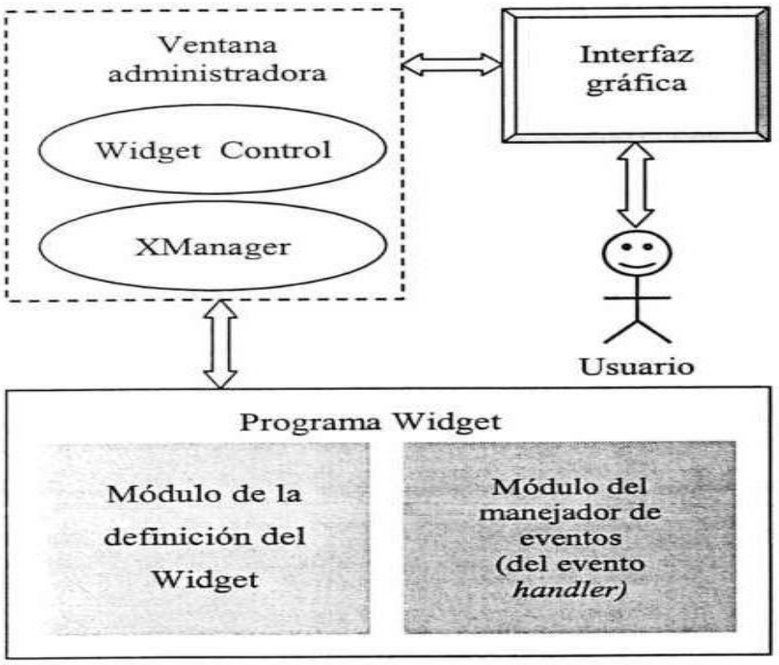


Figura 2.12. Estructura de un programa widget

g) *Estructura del evento:* Esta estructura se forma con la información que envía la ventana administradora y es enviada al módulo del evento *handler* para determinar cual es el evento que se realizará. La estructura del evento contiene varios campos según la información recibida por el XManager, pero siempre contará con tres campos importantes: ID, TOP, y HANDLER.

1. **ID**: Identifica al Widget que causó el evento.
2. **Top**: Identifica al Widget en el nivel más alto de la jerarquía a la que pertenece la estructura del evento.
3. **Handler**: Identifica la asociación de los *Widgets* con el manejador de eventos.

Basándonos en la figura 2.12 describimos el proceso o secuencia interactiva que se realiza al ejecutarse el programa *Widget*.

1) La ventana administradora hace lectura de las descripciones de los *Widgets* que se encuentran en el módulo de la definición del widget.
2) La ventana administradora se encarga del despliegue de la interfaz hecha por dichos *Widgets* y registra la relación de los módulos que forman al programa widget.
3) La ventana administradora se encuentra en estado de espera.
4) El usuario genera un evento a través de la interfaz.
5) La ventana administradora identifica al evento y al widget que lo generó y forma a la estructura del evento.
6) La ventana administradota envía la estructura del evento al módulo del evento *handler*.
7) El módulo del evento *handler* realiza la acción correspondiente al evento.
8) La ventana administradora actualiza a la interfaz que se encuentra en estado de espera nuevamente.

II. 5. 2. 2. Secuencia de eventos para la estructura de un programa Widget

Para ampliar aún más en la filosofía de IDL, ahora presentamos una descripción de la estructura del programa Widget desde otra perspectiva, auxiliándonos en los diagramas de secuencias vistos en el tema "Estereotipos" de la sección II.3. En la figura 2.8 presentamos el diagrama de secuencia de eventos para la estructura de un programa *Widget*. Los objetos o módulos, con sus respectivos nombres y significados que se encuentran en el diagrama de secuencia de un programa widget son los módulos que presentamos en la tabla II.VIII.

Módulo	Nombre	Significado	Observaciones
Interfaz	Interfaz	Interfaz gráfica	Interfaz con la que el usuario interactúa.
Entidad	EstEve	Estructura del evento	Estructura creada al generarse el evento.
Control	VA	Ventana Administradora	Establece la comunicación entre el usuario y el programa y entre los módulos de éste.
Control	Mod1	Módulo de la definición del Widget	Módulo que crea a la interfaz gráfica.
Control	Mod2	Módulo del evento *handler*	Módulo que ejecuta los procesos correspondientes al evento generado.

Tabla II.VIII. Los módulos del diagrama de secuencias

En el diagrama de eventos de la figura 2.13 presentamos la siguiente secuencia de datos, la cual es apta para todos los eventos posibles de acuerdo a la opción elegida por el usuario.

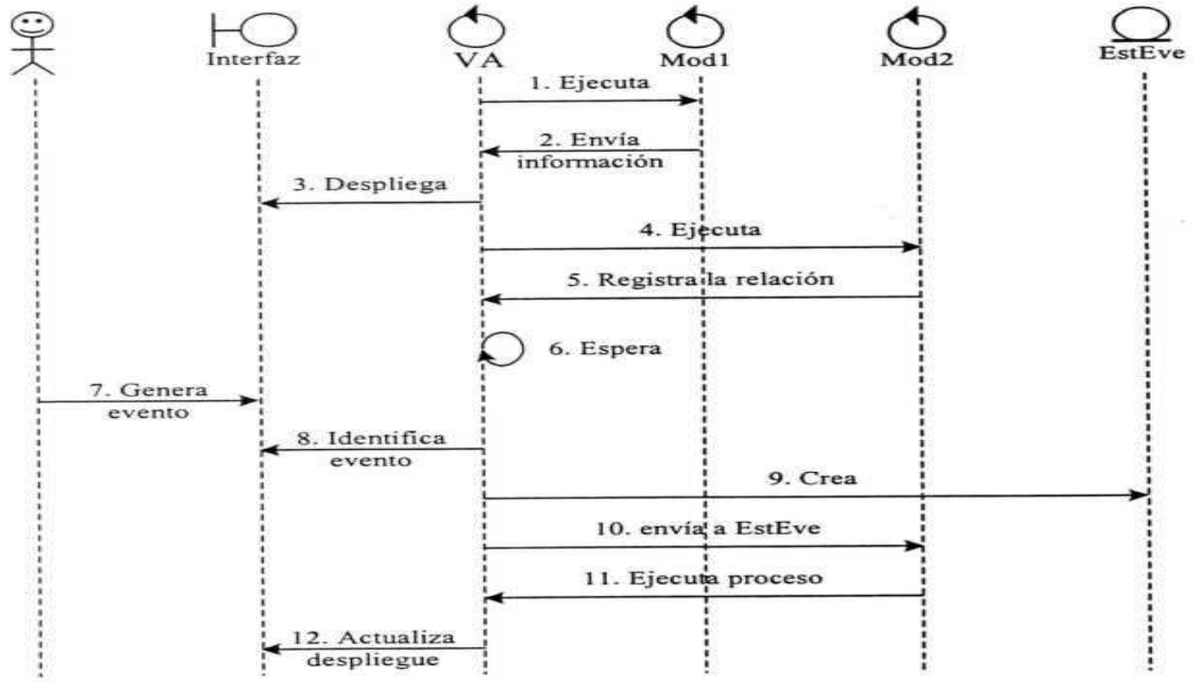


Figura 2.13. Diagrama de secuencia de un programa widget

1. La VA ejecuta primero al mod1.
2. El Mod1 le envía la información de los *Widgets*, sus atributos y valores.
3. La VA despliega la interfaz gráfica con los *Widgets* recibidos.
4. La VA ejecuta ahora a su segundo módulo, el Mod2.
5. El Mod2 registra la relación que tiene con el Mod1, para poder ejecutar los procesos que tienen relación con los *Widgets* que están en el Mod1.
6. La VA se encuentra en el estado de espera.
7. El usuario genera a través de la interfaz a un evento.
8. La VA identifica al evento generado.
9. La VA crea a la estructura del evento EstEve con la información recibida del eventos generado.
10. La VA envía a la EstEve al Mod2 para encontrar al proceso correcto.
11. El Mod2 ejecuta el proceso deseado por el usuario.
12. La VA actualiza a la interfaz.
13. Y el paso trece es regresar al estado de espera de la VA, que es equivalente al paso 6.

II. 5. 3. Sintaxis del IDL para módulos que definen la interfaz

En esta sección mostraremos códigos de programación en IDL muy sencillos, se trata de ejemplos con los cuales se logra asimilar como es la creación de un *widget* y como se forman jerarquías de estos. En el primer ejemplo, presentamos un programa (procedimiento o módulo) que realiza la creación de un *widget*, y en el segundo presentamos un programa que genera jerarquía de *Widgets*.

- **Creación de un *widget***: Para crear un *widget*, es necesario generar una base, y de ahí generar el *widget* especificando a que base pertenece y sus características, en el ejemplo, el *widget* es una etiqueta que dice: "ViAn con Widgets".

```
pro W_creación
        base = widget_base()
        label = widget_label(base, value='V i A n   c o n   W i d g e t s')
        widget_control, base, /realize
end
```

- **Jerarquía de los *Widgets:*** Las bases se generan por jerarquías, si una pertenece a la otra se dice que la primera esta debajo de ella. Si se destruye una base y contiene a otras bases, estas se destruyen.

```
pro W_jerarquia

        ; Jerarquia mas alta
        base1 = widget_base(title='Jerarquía de Widgets - A', xoffset=400, yoffset=100)

        base2 = widget_base(group_Leader=base1, title='B', xoffset=200, yoffset=200)
        base3 = widget_base(group_Leader=base1, title='C', xoffset=650, yoffset=200)
        base4 = widget_base(group_Leader=base2, title='D', xoffset=75,  yoffset=300)
        base5 = widget_base(group_Leader=base2, title='E', xoffset=325, yoffset=300)
        base6 = widget_base(group_Leader=base3, title='F', xoffset=650, yoffset=300)

        label1 = widget_label(base1, value='A: J e r a r q u i a  m a s  a l t a')

        label2 = widget_label(base2, value='B: M i  p a d r e  e s  A')
        label3 = widget_label(base3, value='C: M i  p a d r e  e s  A')
        label4 = widget_label(base4, value='D: M i  p a d r e  e s  B')
        label5 = widget_label(base5, value='E: M i  p a d r e  e s  B')
        label6 = widget_label(base6, value='F: M i  p a d r e  e s  C')

        widget_control, base1, /realize
        widget_control, base2, /realize
        widget_control, base3, /realize
        widget_control, base4, /realize
        widget_control, base5, /realize
        widget_control, base6, /realize

end
```

II. 6. Proyectos afines al Sistema ViAn

Se han desarrollado diferentes técnicas de VDC, algunas son: diagramas de visualización de Pesos (*Weight-Visualization Diagrams*) que pueden mostrar unidades ocultas de pixeles, solamente se necesita un peso por pixel y se puede aprovechar para poder Visualizar pesos de manera simultánea. Curvas de visualización de Pesos (*Weight-Visualization Curves*), se utilizan para Visualizar los pesos de una red completa, la cual está entrenada para clasificar imágenes **[Bishof]**.

Solución de tareas arbitrarias booleanas por redes de *feed-forward* (tipo de comunicación de datos entre los nodos de una red) de multicapas con solo dos unidades ocultas, se determina por el algoritmo de Retro-propagación (*backpropagation algorithm*) **[Munro, 1992]**. Estas tres son técnicas bidimensionales.

Técnicas para la visualización de 3D son: a) para conversión directa se utiliza el trazo de rayos para volumen (*Volume ray-casting*) y b) para la conversión indirecta se utiliza un método común para extraer altas resoluciones de isosuperficies en datos de volumen llamado *Marching Cubes*. El trazo de rayos para superficies (*Surface ray-tracing*) es un método de conversión para escenas en 3D compuestas de armazones de polígonos o superficies formadas por curvas ordenadas **[Newman, 2000]**. Dichas técnicas no se desarrollarán en el proyecto que propongo, solo se mencionan como ejemplos.

El reconocimiento de voz se ha desarrollado dentro del área de las RNA (Redes Neuronales Artificiales), donde la VDC se ha utilizado para la clasificación de fonemas **[Sitio 20]**. Al igual, la VDC clasifica palabras y ayuda con problemas referentes a la gramática para lenguaje natural **[Sitio 2]**, que es una rama de inteligencia artificial. La VDC ha mejorado técnicas para lograr la integración de datos sísmicos y de la perforación de superficies con mayor eficiencia **[Carr, 2001]**. Una parte importante en la que se ha desarrollado la VDC es en el área de la Inteligencia Artificial, y en RNA se pueden encontrar muchos proyectos afines.

Una red neuronal que clasifica datos y se visualizan con facilidad, es un ejemplo de estos trabajos; la clasificación visual neuronal proporciona diseños que visualicen la relación que existe en los datos entre sí, y la relación que existe entre estos datos y regiones definidas. La red de reducción de dimensiones combinada con decisión de los expertos locales (forma en que recibe los datos la red) para formar la clasificación final es llamada "Visualización de Redes" **[Ornes, 1997]**.

Los métodos de visualización proporcionan los principales enfoques para analizar simultáneamente una gran cantidad de información oculta en las redes **[Bishof]**. La visualización de una transformada de conjunto de datos multivariables, es una estrategia basada en la transformada de datos a priori para utilizar RNA autoasociativas donde varias perspectivas de espacios N-dimensionales pueden ser exploradas por mapeos dinámicos con respecto a puntos definidos por el usuario **[Aldrich, 1998]**.

El algoritmo de Kohonen y sus mapas de auto-organización, han sido utilizados en aplicaciones médicas, donde se requiere de visualización de imágenes **[Manduca, 1994]**, **[Myklebust, 1995]**. Existe una herramienta de visualización para RNA de la Teoría Adaptativa de la Resonancia **[Sitio 16]** (*Adaptive Resonance Theory*, ART) llamada SOTA **[Bartfai, 1993]**. ART abarca una variedad amplia de RNA basadas explícitamente en neurofisiología **[Sitio 8]**.

En el problema de Reconocimiento de Patrones es de gran interés la tarea de reducir dimensiones, para manejo de costos y exactitud en la clasificación de datos. Debido a que las RNA tienen la habilidad de aprendizaje complejo entre datos de entrada/salida no lineales, utilizando procedimientos de entrenamiento secuencial y se actualizan los datos por sí mismos, el método de Reconocimiento de Patrones, utiliza algoritmo de Kohonen, principalmente para agrupar datos y asociar (mapear) sus características. Y así la red puede realizar eficientemente una clasificación o grupo específico de tareas.

El algoritmo EM se aplica en el problema de Reconocimiento de Patrones para estimar la definición de los parámetros de manera formal, para la descomposición de la mezcla (*Mixture decomposition*) en la clasificación - y aprendizaje - del modelo sin supervisión **[Jain, 2000]**.

El modelo de las variables latentes, realiza una construcción de una hipótesis desarrollada con el fin de entender cierto campo de interés de investigación. El modelo explica la estructura de dicha hipótesis con un conjunto de variables que se correlacionan. Ejemplos de los campos que tienen interés de investigación son: clase social, opinión pública, personalidad extrovertida, prejuicios raciales, y en general lo que concierne a las ciencias sociales y de comportamiento **[2001]**.

La aplicación ViAn que proponemos, presentará al modelado de variables latentes, basándonos en una función de verosimilitud (*likelihood*) del área de la estadística, utilizando un algoritmo llamado EM que significa la Maximización del valor Esperado, por sus siglas en inglés *Expectation Maximization*. Presentará además, otra técnica que también es una red de aprendizaje, utilizando al algoritmo de Kohonen de las RNA. Ambos algoritmos se utilizarán para la técnica de reducción de dimensión y así, visualizar los datos, generando imágenes. ViAn en un futuro, pretende analizar los datos reducidos a través de dichas técnicas con una técnica de multirresolución, y espectral, con las transformadas de Wavelets y Fourier respectivamente.

CAPITULO III.

Análisis del Sistema ViAn

Nada ocurre porque si. Todo en la vida es una
sucesión de hechos que, bajo la lupa del análisis,
responden perfectamente a causa y efecto.

Richmad de Berr

CAPITULO III.
Análisis del Sistema ViAn

III. 1. Introducción

El análisis es la parte más importante de cualquier software a desarrollar, ya que son nuestros planos y cimientos que nos ayudan a definir nuestro problema y solución del mismo. Con el análisis aterrizamos toda idea del cliente y del desarrollador, también nos ayuda a especificar lo que se va a hacer, qué se va a lograr, y cuáles son nuestras limitaciones, además de que nos ayuda a determinar las herramientas de trabajo.

En este capítulo presentamos al análisis del sistema ViAn con el objetivo de establecer las bases para el problema a resolver, planteando los diversos puntos de vista en una solución. En la figura 3.1(a) presentamos los diferentes aspectos que deberá cubrir el análisis para dar de base a la estructura de ViAn. Así mismo será de utilidad como referencia a los aspectos que se presentarán en el resto del capítulo.

En dicha figura presentamos la necesidad de un extenso trabajo para determinar los requerimientos del sistema, que son una parte muy importante y base de un buen análisis. Esta fase consiste en varias entrevistas entre mi director de tesis (mi cliente) y yo, en un cuestionario (En la sección "1.Requerimientos" del apéndice A se puede observar el cuestionario de 24 preguntas que contestó mi cliente, al igual que yo como la desarrolladora de la aplicación) y en el análisis que realizo para interpretar lo que propone mi cliente en forma de requerimientos.

En base a estos requerimientos se generan los *casos de uso* que representan al sistema desde el punto de vista del usuario. A partir de los casos de uso y los requerimientos se generan entonces los *requerimientos de interfaz*, además de que estos mismos casos de uso y requerimientos son la base para el desarrollo del *modelado de procesos*.

El modelado de procesos consiste en 4 elementos importantes como lo presentamos en la figura 3.1(b) Estos son: 1. *El modelado de flujo de datos*, 2. *el modelado de flujo de información*, 3. *diagramas de estereotipos* y 4. *diagramas de secuencia*.

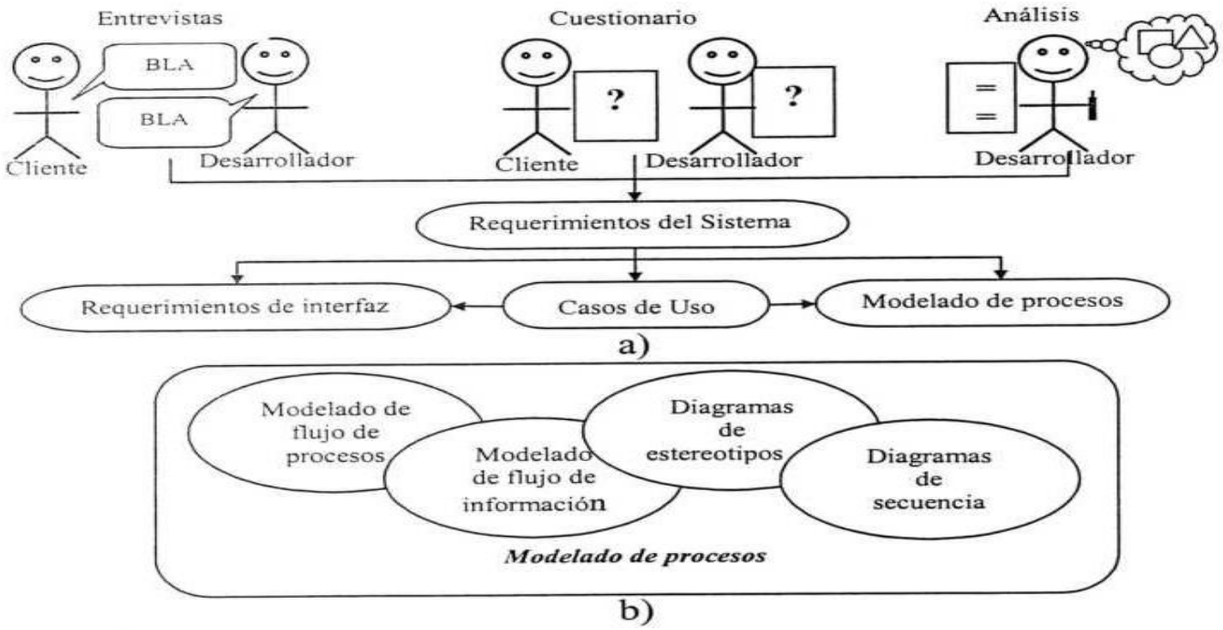


Figura 3.1. El análisis del sistema ViAn:
(a) Esquema general del análisis del sistema ViAn. (b) Elementos del Modelado de procesos.

Otra parte igualmente importante del análisis del sistema ViAn es el análisis y el diseño de las técnicas de reducción de dimensión, y el análisis y el diseño para las técnicas de análisis de datos, que se presentará en el capítulo V. El análisis y diseño de los algoritmos consisten en la explicación de las técnicas de reducción y de las técnicas del análisis de datos, todas estas con sus respectivos algoritmos y su enfoque particular de desarrollo. En la figura 3.2 se estructura el análisis y diseño de los algoritmos, y así completar al análisis global del sistema. Las técnicas que se proponen según los requerimientos son las siguientes: 1. El algoritmo de Kohonen de Redes Neuronales, 2. Algoritmo EM del modelo de Variables Latentes, 3. Análisis Espectral con la Transformada de Fourier y 4. Análisis de Multirresolución con Wavelets (ondeletas).

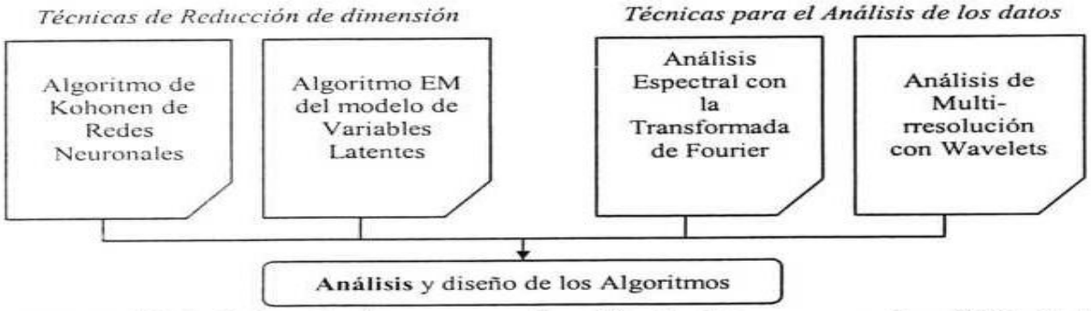


Figura 3.2. Análisis de las técnicas para reducción de datos y para el análisis de datos

III. 2. Requerimientos del Sistema

Los requerimientos son la principal fuente de información que ayudan a: 1. determinar las necesidades que propone cubrir el usuario para tener una herramienta eficiente con que cumplir su trabajo; y 2. determinar los objetivos, metas, alcance y forma que tendrá el proyecto. Los requerimientos se formalizan a través de una serie de actividades (figura 3.1) y se convierten en la base para el desarrollo de los diagramas del análisis.

Los requerimientos de un sistema pueden clasificarse de varias formas, y esto nos ayuda a tener un panorama más amplio del sistema a desarrollar. Los requerimientos de nuestro sistema ViAn, los especificaremos en tres grupos, que varían según la visión del problema, se cuentan con los requerimientos: 1. de usuario, 2. técnicos y 3. de Hardware. Los requerimientos de usuario son los que se determinan a través de entrevistas y pláticas entre el desarrollador y el cliente, el resto con necesidades claras del cliente y generalmente se concretan con el análisis del desarrollador.

Los *requerimientos normales* se encuentran estructurados en los *requerimientos de usuario*. Los *requerimientos esperados* se encuentran clasificados en los *requerimientos de usuario* y *requerimientos técnicos*. En cuanto a los *requerimientos innovadores*, no los encontraremos dentro de ninguno de estos tres tipos de requerimientos especificados. Estos últimos requerimientos dependerán del desarrollador, según su creatividad, habilidad y perspectiva del software a desarrollar.

III. 2. 1. Requerimientos del usuario

Después de varias entrevistas, discusiones del problema a resolver y determinar el tipo de aplicación a desarrollar, además de extraer la información del cuestionario mencionado, se logró obtener explícitamente los siguientes requerimientos de usuario. Esto no significa que sean los únicos y definitivos, ya que a lo largo del desarrollo del sistema surge la visión de nuevas características.

III. 2.1.1. Requerimientos para el proceso del sistema.

Los requerimientos para el proceso del sistema se dividieron en tres partes importantes: Entrada, proceso y salida. Estas son partes que todo sistema, por simple, que parezca debe cumplirlas.

Entrada:

1a. Se acepta una serie de datos t de entrada como mínimo para un experimento.

1b. La captura de los datos del experimento se requiere en forma de matriz para cada serie de datos t. Donde cada serie tiene un vector de dimensión N. Esto lo representamos con la matriz X:

$$X = \begin{bmatrix} X_1 & X_2 & \cdots & X_N \\ X_1 & X_2 & \cdots & X_N \\ \vdots & \vdots & & \vdots \\ X_1 & X_2 & \cdots & X_N \end{bmatrix} \begin{matrix} t = t_1 \\ t = t_2 \\ \\ t = t_N \end{matrix}$$

Proceso:

1c. Las técnicas específicas para la reducción de dimensión de datos son:

- Algoritmo de Kohonen basado en Redes Neuronales Artificiales
- Algoritmo de Variables Latentes con EM

1d. Las técnicas específicas para el análisis de datos son:

- Análisis de multirresolución con Wavelets (ondeletas)
- Análisis espectral de Fourier

1e. Se propone comparar técnicas durante el procesamiento del experimento. Por ejemplo: poder usar el análisis de Fourier y el análisis de Wavelets.

Salida:

1f. La información que se propone ver como resultado de la reducción de datos es:

- Una matriz (para almacenar) y una imagen (para desplegar) que muestren una mejor clasificación de los datos y ver la relación entre estructura del espacio reducido con la estructura o característica del espacio de datos.

1g. En la reducción de datos, se propone que se visualicen los resultados en dos dimensiones, pero también ver la posibilidad de realizarlo en tres dimensiones.

1h. La información que se propone ver en el proceso de visualización es la siguiente:

- Gráficas de contornos, curvas de nivel o de superficie.
- Datos generales del experimento, como su nombre, cantidad de series t.

1i. La información que se propone ver como resultado del análisis de los datos es la siguiente:

- Detalles del espacio reducido, como ejemplos: regiones constantes, regiones variables, etc. En forma de matriz para almacenar e imagen para desplegar.

III. 2.1.2. Requerimientos en herramientas

Estos requerimientos en herramientas son muy específicos por parte del cliente, y pueden ser mejorados con las sugerencias del desarrollador.

1j. Que el sistema sea implementado en IDL.

1k. Que sea implementado para plataforma Windows.

III. 2.1.3. Requerimientos de soporte

Los requerimientos de soporte son los que ayudan a la comprensión del sistema, es decir, la funcionalidad que tiene el sistema.

1l. Un manual de usuario.

1m. Ejemplos del experimento en el menú del sistema para cada posible uso de la aplicación.

III. 2. 2. Requerimientos técnicos

Los requerimientos técnicos son determinados generalmente con el análisis del desarrollador, estos surgen implícitamente en muchas ocasiones pero no son vistos en un principio, y conforme avance el ciclo de vida de desarrollo de la aplicación, van surgiendo nuevos requerimientos técnicos. Estos son algunos de los que han sido determinados:

2a. Almacenar y leer archivos en formato propio del sistema.

2b. Almacenar y leer archivos en formato de imagen.

2c. Convertir datos a formato de imagen.

III. 2. 3. Requerimientos del software para el lenguaje IDL

Nuestro sistema ViAn será implementado en el lenguaje IDL de acuerdo al requerimiento de usuario **1j**. Específicamente para nuestro sistema ViAn, el requisito del software es desarrollarlo para la plataforma Windows (requerimiento de usuario **1k**.), aunque cabe mencionar que el sistema ViAn podría ser ejecutado en cualquier plataforma en que se ejecute la aplicación IDL. Estos son los requerimientos mínimos de hardware para ejecutar a IDL, por lo tanto, también para ejecutar la aplicación ViAn.

- Para los sistemas Unix y VMS, el mínimo recomendado para la configuración del sistema IDL es de 100 Mb espacio en disco duro y 24 Mb en memoria RAM.
- Para el sistema Macintosh, el mínimo recomendado para la configuración del sistema IDL es de 100 Mb de espacio en disco duro y 8 Mb en memoria RAM.
- Para el sistema Windows, 100 Mb de espacio en disco duro y 8 Mb en memoria RAM. Además que tenga por lo menos capacidad de 256 colores y tarjeta de video de alta resolución.

III. 3. Casos de uso

Un caso de uso, como mencionamos en los antecedentes, determina la relación que tiene el usuario – según sus objetivos – y el sistema a desarrollar. Se considera a un caso de uso un patrón de comportamiento que el sistema presenta, una secuencia de transacciones desarrolladas por un actor (usuario o factor externo al sistema que interactúa o se interrelaciona con él). Los diagramas de casos de uso describen la funcionalidad del sistema y sus actores.

En base a los requerimientos de usuario se obtuvieron los siguientes diagramas de casos de uso para el sistema de ViAn. Presentaremos a cada diagrama con su descripción, estos son del nivel 0 y del nivel 1, los casos de uso de nivel 2 se encuentran localizados en el apéndice A, sección "2. Casos de uso". Los casos de uso de nivel 2, describen las características, completan y detallan a los procesos principales del sistema ViAn, establecidos a través de los requerimientos del sistema. Enseguida mostramos la tabla III.I que contiene a los casos de uso utilizados en el presente análisis del sistema.

Nivel	Nombre del caso de uso	Figura
0	*ViAn - Visualización y análisis*	3.3
1	*Visualización y análisis de datos del espacio N-dimensional*	3.4
2	*Reducción de dimensión de datos N*	A1
2	*Visualización de datos M*	A2
2	*Análisis de datos G*	A3

Tabla III.I. Diagramas de los casos de uso utilizados en el análisis de sistema ViAn

Esta tabla contiene en sus columnas (de izquierda a derecha) el nivel del caso de uso, el nombre del caso de uso, y la figura en la que localizará el diagrama del caso de uso. Ahora presentamos en la figura 3.3 el primer diagrama del caso de uso que mostramos en la tabla, con su respectiva descripción. El segundo diagrama de caso de uso y su descripción que presentamos en esta sección, corresponde al segundo caso de uso, que mostramos en la tabla III.I.

Figura 3.3. Diagrama de caso de uso general (nivel 0): "***ViAn* - Visualización y análisis**"

Nombre: ViAn – Visualización y análisis.

Actores: Investigador (usuario).

Propósito: Visualizar y analizar datos N-dimensionales de un experimento.

Descripción: El investigador entra al sistema y pre-procesa los datos de su experimento, el sistema prepara los datos con cálculos específicos para lograr el proceso para la reducción de dimensión de los datos; los datos reducidos son entonces procesados para ejecutar su visualización; y se aplican métodos para el análisis de los datos reducidos.

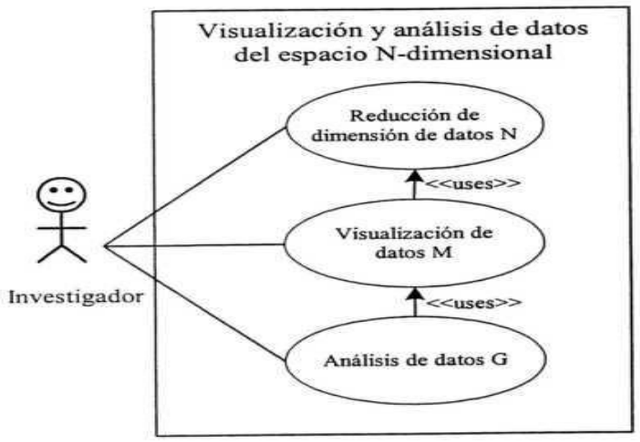


Figura 3.4. Diagrama de caso de uso de nivel 1:
"Visualización y análisis de datos del espacio N-dimensional"

Nombre: Visualización y análisis de datos del espacio N-dimensional.

Actores: Investigador (usuario).

Propósito: Reducir, visualizar y analizar datos del espacio N-dimensional de un experimento.

Descripción: En forma general tenemos que el investigador entra al sistema e indica la reducción de los datos N-dimensionales de su experimento; los datos reducidos M se procesan para su visualización, y los datos visualizados G se preparan para su análisis; Y se despliega los datos analizados A. Esto es, primero, el investigador entra al proceso de *Reducción de dimensión de los datos N*, aquí los datos brutos (originales) del experimento se pre-procesan (se archivan como datos de entrada), y se les aplica una de dos (o ambas) técnicas de reducción de datos: 1. el algoritmo de Kohonen, y 2. el algoritmo EM de variables latentes, una vez reducidos los datos, ahora en el espacio M, se pueden archivar, y se encuentran listos para el siguiente proceso. El segundo proceso que se observa en la figura es la *Visualización de datos M*, los datos que se redujeron con alguna técnica de reducción son visualizados como datos bidimensionales (formato de imagen), y estos datos G están listos para ser archivados, y listos también para ser procesados por la tercera tarea de la aplicación. Como tercer y último caso de uso que vemos en el diagrama 3.4, el *Análisis de datos G* se encarga de aplicar una de las dos técnicas (o ambas) para el análisis de los datos reducidos y visualizados, estas técnicas son: 1. la transformada de Fourier y 2. la transformada Wavelets; Después de ser analizados, los datos están listos para ser archivados como datos A (analizados).

Más adelante, en los requerimientos de interfaz se observarán las pantallas en las que se reflejan los diferentes casos de uso. Por ejemplo, en la figura 3.4 se encuentra un diagrama que está formado por tres casos de uso a nivel 1, y a cada caso de uso le corresponde una pantalla para la interfaz, pero esta relación quedará más clara y será descrita en la sección de "Requerimientos de interfaz" de este capítulo.

III. 4. Modelado de procesos

Resulta necesario realizar el análisis de nuestro sistema ViAn desde el punto de vista de las necesidades del propio sistema, sin perder el punto de vista del usuario. Es necesario tener presentes los objetivos del usuario (requerimientos) y los "casos de uso" anteriores, ambos son base para la formación de procesos y el analizar el flujo información entre dichos procesos. Este análisis se logra con dos clases de modelados: 1. el de flujo de procesos y 2. el de flujo de información; y dicho análisis se complementa con dos tipos de diagramas: 1. el de estereotipos y 2. el de secuencias.

III. 4. 1. Modelado de flujo de procesos

El modelado de flujo de procesos es un conjunto de funciones del sistema que se refinan en más procesos del sistema, se muestra este conjunto en forma de secuencia de procesos. La tabla III.II presenta a los modelados que presentamos a lo largo de esta fase, el análisis del sistema.

***ViAn* – Visualización y análisis** *(figura 3.5)*		
Reducción de dimensión de datos N *(figura A4)*	Visualización de datos M *(figura 3.6)*	Análisis de datos G *(figura A6)*
Pre-procesamiento de datos N *(figura A5)*		

Tabla III.II. Diagramas de los modelados de flujo de procesos para el análisis del sistema

La figura 3.5 muestra el flujo de procesos de nuestro sistema ViAn. El proceso principal corresponde a la "Visualización y análisis de datos del espacio N-dimensional". Se puede observar que cada proceso corresponde a cada caso de uso de la figura 3.4. Se forma por tres procesos: 1. el de reducción, 2. el de visualización y 3. el de análisis, los cuales están formados por un conjunto de sub-procesos.

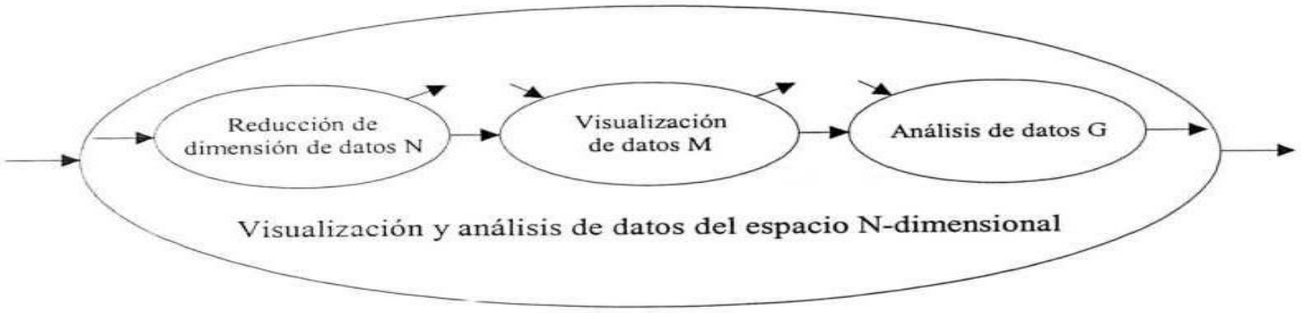


Figura 3.5. Modelo de flujo de procesos: "***ViAn* – Visualización y análisis**"

En la figura 3.6 se presenta el Modelado de visualización de los datos reducidos M. El refinamiento del resto de los modelados se encuentra en la sección "3. Modelado de Procesos" del apéndice A. El resto del modelado en mención, complementa y completa la estructura de los procesos del sistema ViAn. Así, todos los apéndices presentados en el trabajo de tesis son complementos (parte de los capítulos correspondientes) y no estructuras aisladas que consultar.

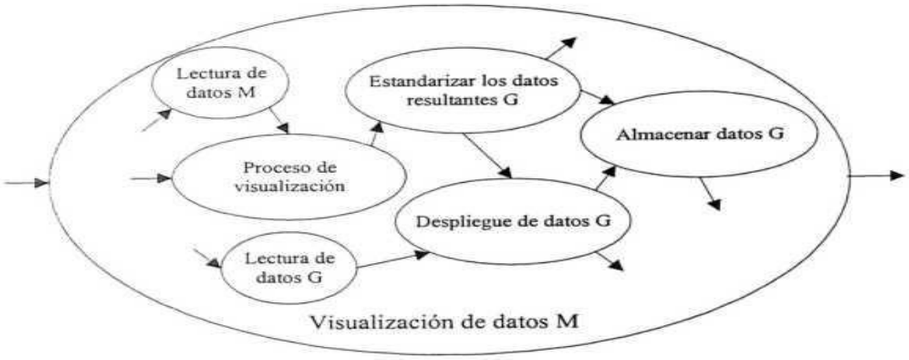


Figura 3.6. Modelado de flujo de procesos: **"Visualización de datos M"**

En la figura 3.6 presentamos el modelado de flujo de procesos para la visualización de datos reducidos M, donde se muestra que existen tres formas de iniciar el proceso. Primero mostramos en el sub-proceso superior izquierdo la lectura de los datos M que se realiza desde nuestro Sistema de Archivos (SA), y de ahí se procesa a la visualización. En segundo término observamos directamente al proceso de visualización, esto sucede cuando la salida del proceso anterior solicita la visualización inmediata de sus resultados. Tercero, presentamos en el sub-proceso inferior la lectura de los datos G, que son datos ya visualizados con anterioridad y almacenados como tales, estos datos pasan directamente al despliegue en pantalla de la imagen resultante a la visualización. En cuanto a la primera y segunda forma de iniciar el proceso, se precede a estandarizar los datos resultantes G para tener entonces tres opciones: 1. Salir del proceso, 2. Almacenar los datos Visualizados en nuestro SA o 3. Desplegar los datos en la pantalla, después de elegir cualquiera de las dos últimas opciones, tenemos la salida del proceso.

III. 4. 2. Modelado de flujo de información

Recordemos lo que presentamos en los antecedentes (Capítulo II): que el modelado de flujo de procesos está integrado con el modelo de datos para proporcionar una indicación de cómo es que fluye la información a través del sistema. Aunque el modelo de datos no es necesario desarrollarlo para nuestro sistema, debido a que no está orientado a objetos, se presentan algunos aspectos en el modelado de procesos, ya que nuestro Sistema de Archivos (SA) son nuestras entidades de información. Además de que ViAn tendrá una estructura de datos en la que se manipularán los datos para poder ser almacenados.

En la figura 3.7 se muestra el Modelado de Flujo de Información del Sistema ViAn, presentamos lo que los archivos leen en cada proceso y los archivos que arrojan datos resultantes, también los datos intermedios que presentan resultados, y todos estos se pueden almacenar en el SA. La salida de los datos se proporcionará en forma de matriz (datos M, datos G y datos A), y algunos de ellos en formato gráfico también (datos G y datos A).

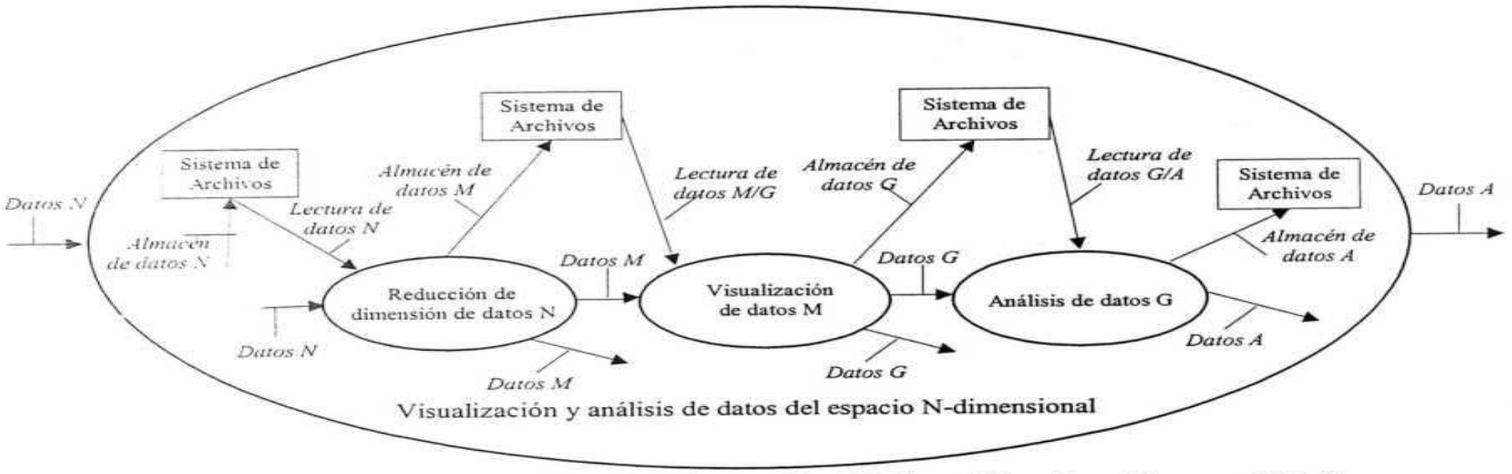


Figura 3.7. Modelado de flujo de información: "***ViAn* – Visualización y análisis**"

Las matrices de los datos de salida serán muy similares a la matriz de datos N de entrada, esto se estableció en **1a** y **1b** de los requerimientos de usuario, para los datos de entrada, dichas matrices se observarán de la siguiente forma:

$$\begin{bmatrix} X_1 & X_2 & \cdot\ \cdot & X_N \\ X_1 & X_2 & \cdot\ \cdot & X_N \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ X_1 & X_2 & \cdot\ \cdot & X_N \end{bmatrix}$$

Esto justifica el cumplimiento de **1f** a **1i**, de los requerimientos de usuario para los datos de salida. En especifico el formato cumple con el requerimiento **1f** y para cubrir con los requerimientos **1g**, **1h**, y **1i** es necesario contemplar un formato de imagen pre-establecido que pueda ser generado por el lenguaje IDL especificado. Este aspecto será retomado en la etapa del diseño del sistema ViAn y en la etapa de su implementación. Además, para cubrir los requerimientos técnicos de usuario **2a**, **2b** y **2c**, es necesario llegar al diseño y a la implementación del sistema, al igual que los requerimientos antes mencionados.

III. 4. 3. Diagramas de estereotipos

El diagrama de estereotipos de nuestro sistema ViAn, basándonos en los diagramas de procesos anteriores modela la estructura de las funciones en elementos o módulos específicos. Tenemos con estos, una amplia visión de cómo es que los procesos fluyen y se comunican entre ellos, y como interactúa el sistema de archivos y con la interfaz gráfica, que es el intermediario entre el usuario y el sistema ViAn.

La figura 3.8 presenta un diagrama de estereotipos donde se identifican algunos objetos (elementos o módulos) de interfaz, de control, de entidad y actores. En ocasiones presentamos algunos de estos objetos repetidos, por ejemplo: "Menú principal", pero no significa que sean elementos diferentes, hablamos del mismo elementos, se duplicó para comodidad del diagrama.

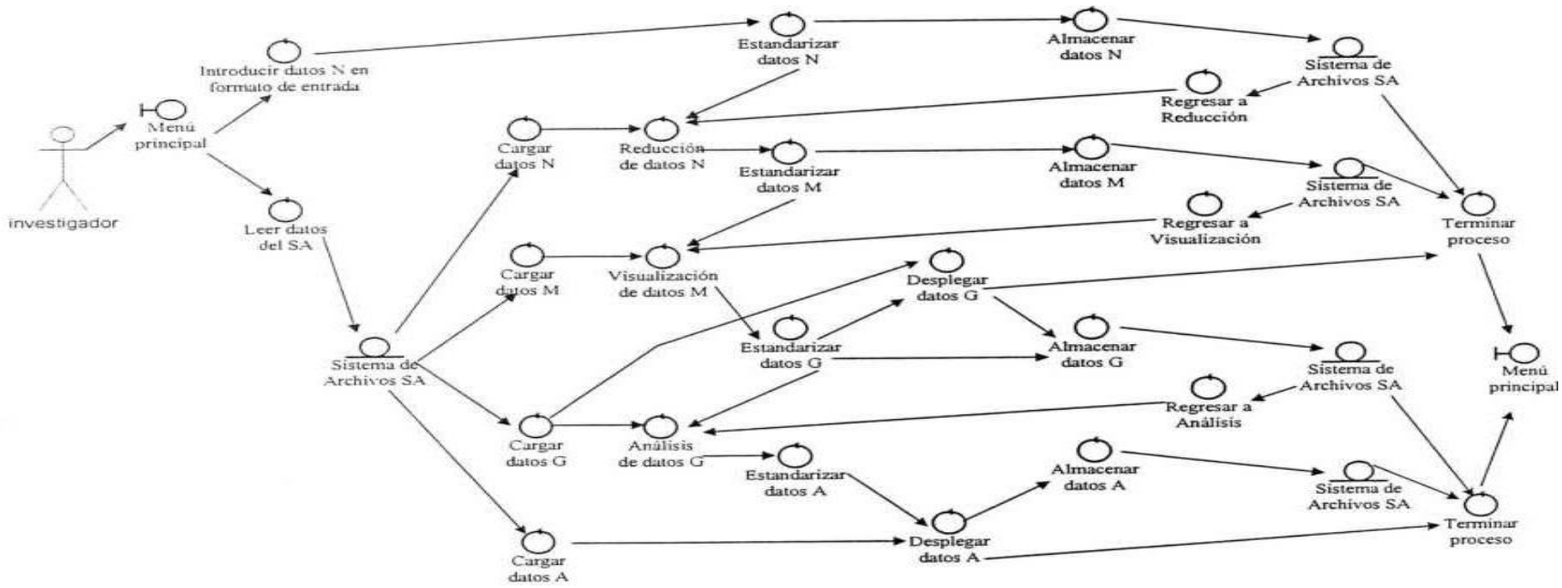


Figura 3.8. Diagrama de estereotipos para el análisis de robustez

En la figura 3.8 podemos observar las diferentes opciones por las que el sistema ViAn nos puede llevar, realizando sus procesos definidos en los casos de uso. El hecho de que toda alternativa empiece en el Menú principal y termine en el mismo menú, nos indica que estamos dentro de un ciclo (así trabajan los sistemas orientados a eventos), y es así como se comportan los eventos. Cabe mencionar que el Sistema de Archivos se encuentra varias veces, esto es, 1. por comodidad y 2. porque es más fácil comprender que son diferentes tipos de archivos los que almacenaremos y leeremos.

Cualquier ruta que se siga en el diagrama de estereotipos, tiene la salida adecuada, es decir, no existen procesos aislados. Cada objeto de control indica un proceso diferente, que tiene sus tareas específicas, alguno de ellos son más complicados que otros, pero se presentaron los que tienen trascendencia para explicar el flujo o la secuencia de procesos para el análisis completo del sistema.

Para ejemplificar una ruta que se seguiría, suponiendo que el usuario (investigador) desea reducir datos y almacenar el resultado en el sistema de archivo, iniciaríamos con el *investigador,* quien genera un eventos en el *menú principal* que le muestra el sistema a través de la interfaz, el usuario *introduce datos N en el formato de entrada*, enseguida, se *estandarizan los datos N* y se aplica el proceso para la *reducción de datos N*, se *estandarizan los datos M*, que son los datos reducidos, y se *almacenan* en el *sistema de archivos*, lograda la actividad deseada *termina el proceso* y se regresa al *menú principal*. Los otros escenarios son igualmente representables a través de trayectorias y serán la base de los diagramas de secuencia que presentamos a continuación.

III. 4. 4. Diagramas de secuencia

Los diagramas de secuencia, nos especifican rutas del diagrama de estereotipos (figura 3.8), se enfocan en un procedimiento, e indican como se seguiría la ejecución de los procesos a través del tiempo. Los diagramas de secuencia, desglosan al diagrama de estereotipos para estudiar a la representación de los casos de uso de una forma más detalla.

Enseguida construiremos y describiremos el diagrama de secuencia para el caso de uso, en que se desee Visualizar a los datos reducidos. El resto de los diagramas se pueden ver en la sección "3. Estereotipos" en el apéndice A. En esta figura se da una explicación breve de lo que está sucediendo. En realidad estamos detallando una ruta del diagrama de estereotipos como lo mencionábamos.

En general, tenemos que el usuario solicita el proceso para Visualizar los datos reducidos. Para obtener los datos que se van a visualizar, tenemos dos opciones, 1. que los datos se tomen como entrada de otro proceso que los arrojó como datos de salida, y 2. que los datos estén en el SA (Sistema de Archivos) y un proceso se encargue de su lectura. En la primera opción, se procesan los datos (se estandarizan) para lograr la actividad de la "Visualización". Ya que los datos fueron visualizados, pueden ser almacenados en nuestro SA, o bien se puede solicitar la siguiente tarea que es el análisis de los datos visualizados. Esto si el usuario no eligió salirse del sistema. Si lo que hizo el usuario fue almacenar los datos visualizados, entonces también cuenta con la opción de continuar con el análisis de los datos, o de salirse del sistema. Si en un principio la opción de que los datos a visualizar se leyeran del SA, entonces el proceso es más sencillo, porque se leen los datos del SA, se procede a visualizarlos y entonces se tiene la opción de almacenar nuevamente los datos visualizados en el SA o se tiene la opción de salirse del proceso. La secuencia se observa con más a detalle en la figura 3.9, de una forma breve y entendible.

Con la construcción del diagrama de la figura 3.9 y el resto de los diagramas de secuencia que se encuentran en el apéndice A, logramos organizar y definir a) los procedimientos según el tiempo, b) las relaciones que se forma entre los eventos, los módulos del sistema, el SA y la estructura de datos.

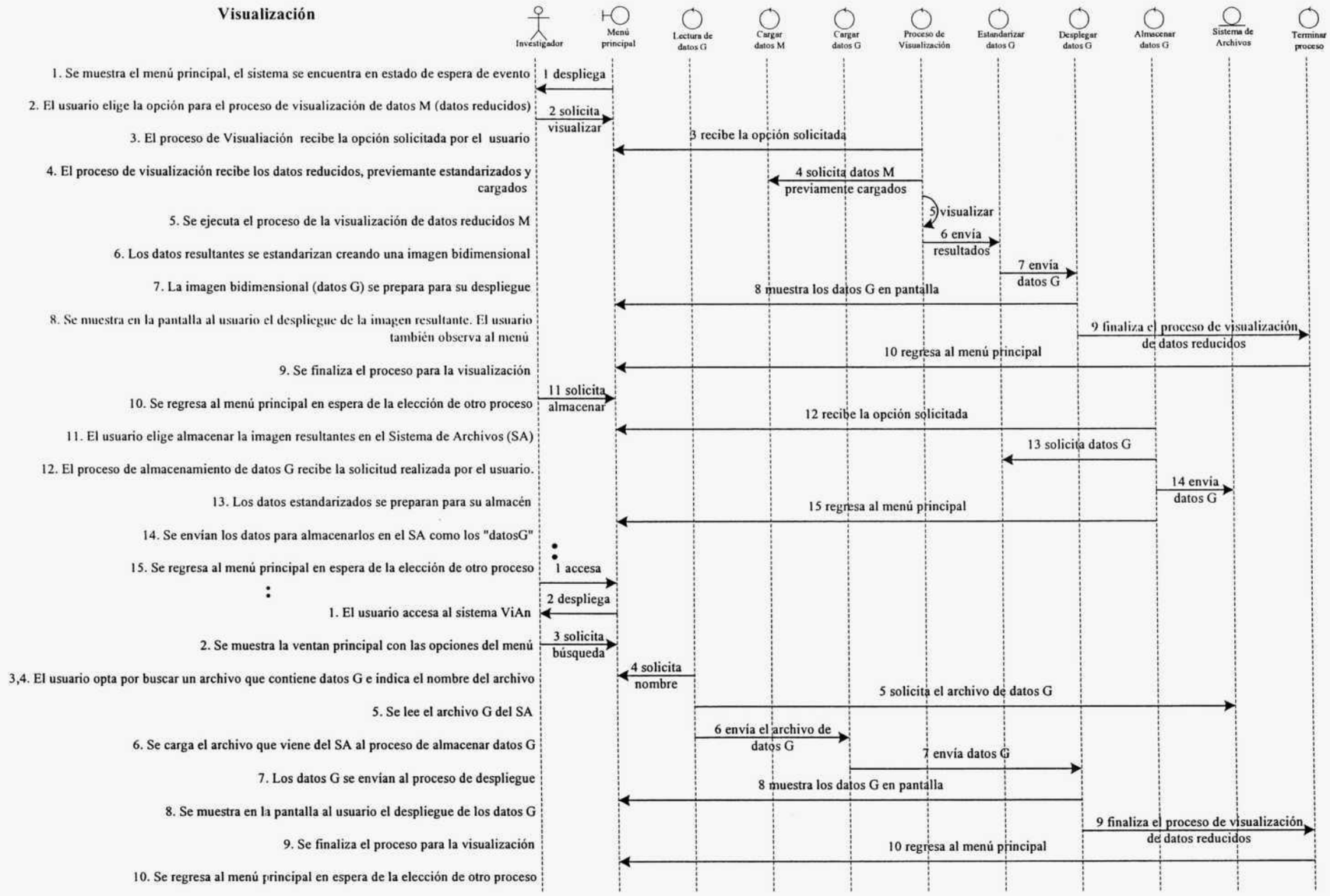


Figura 3.9. El investigador elige los procesos para *Visualizar* los datos reducidos

III. 5. Requerimientos de interfaz

Después de formalizar los requerimientos, de continuar con el desarrollo de los casos de uso, de concluir el modelado de procesos y estereotipos, tenemos un panorama más amplio de cómo será la interacción del usuario y la aplicación. Estas pantallas contienen los elementos requeridos, detallados por el usuario en complemento con el desarrollador, para estructurar los requerimientos de interfaz y con los que se dará forma a los módulos de interfaz y eventualmente a la estructura de *Widgets* en IDL.

Las pantallas que presentan estos requerimientos, son los elementos de interfaz que especificamos con estereotipos en el modelado de procesos. Es a través de estas pantallas que el usuario genera los eventos que regirán el procesamiento de sus datos, es decir, a los procesos o actividades que de igual forma especificamos con estereotipos en los modelados presentados anteriormente. Los requerimientos de interfaz, son la base para formar a las ventanas de nuestra interfaz gráfica, por lo que se le considera el pre-diseño de interfaz. Enseguida mostramos las pantallas que conforman los requerimientos de interfaz y que representan la formación de los requerimientos especificados para el diseño de interfaz. En estas pantallas podemos identificar elementos o regiones (algunas en forma de botones por ejemplo) que son nuestros generadores de eventos, son los que permiten las opciones de los procesos a ejecutar o bien, con los que interactuamos directamente.

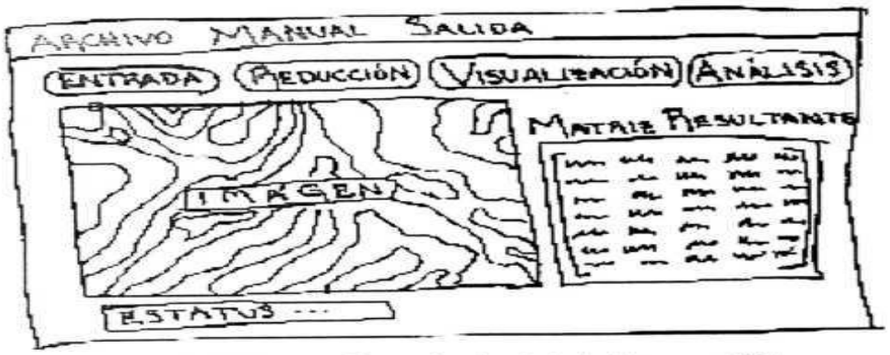


Figura 3.10. Pantalla principal del sistema ViAn

En la figura 3.10 presentamos a la pantalla principal del sistema ViAn, donde mostramos un menú que contiene a la opción de *Archivo*, en la cual se tendrán las opciones para grabar los datos originales del experimento y los datos procesados, también en el menú se encuentra el *Manual* del usuario (Requerimientos de usuario **1l** y **1m** para soporte), y como en todo sistema, debe existir la opción de *Salir* del sistema, como se ve igualmente en el menú. La pantalla presenta a cuatro componentes en forma de opciones (*Entrada*, *Reducción*, *Visualización* y *Análisis*), son los procesos principales del sistema y se detallan más adelante.

La aplicación contará con un área donde se desplieguen las imágenes generadas con los datos visualizados y analizados, además un área donde se desplieguen los resultados de los procesos de la reducción y análisis de los datos. La interfaz mostrará el estatus del sistema, que indica el proceso en el que estaría trabajando la aplicación. Las opciones de los procesos se encuentran en botones (un elemento de interfaz gráfica) para la fácil interacción entre el usuario y el sistema ViAn.

La siguiente pantalla corresponde a la función que solicita los datos originales del experimento, que es otro estado para la interfaz de la figura 3.10. Y encontraremos en el apéndice A, a cuatro más que detallan las interfaces para los procesos de reducción, visualización, análisis y para funciones que realicen procesos comparativos en el caso de la reducción y análisis, esto se especificará más adelante.

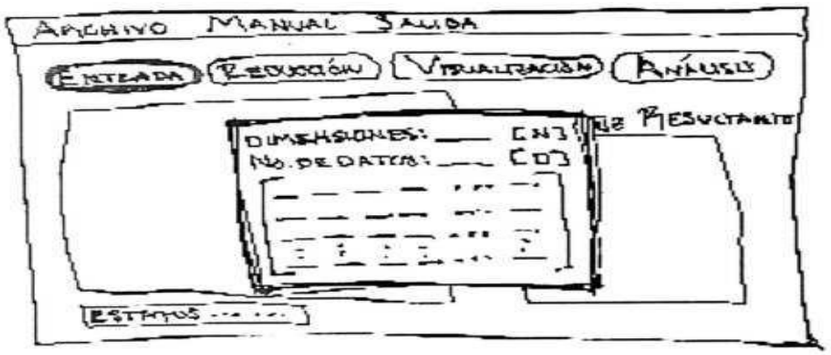


Figura 3.11. Pantalla para la entrada de datos originales

La figura 3.11 muestra como al elegir el usuario la opción *Entrada*, la cual abre una región (o área) donde se solicita el número de dimensiones y el número de datos de la información original, y posteriormente se solicitan los datos del experimento, en forma matricial.

En los requerimientos de interfaz se observan las pantallas que reflejan a los casos de uso, por ejemplo, en la figura 3.4 se observan tres casos de uso, el primero es el caso de uso *Reducción de dimensión de datos N* y se observa en la figura A10 la pantalla con la que se procesará la reducción de dimensión de datos; así mismo el caso de uso *Visualización de datos M* se identifica en la figura A11, y en la figura A12 se identifica el caso de uso *Análisis de datos G*. Las figuras A10, A11 y A12 se encuentran en el apéndice A.

Con los requerimientos de interfaz, finalizamos el análisis del sistema ViAn, y nos encontramos en posición de iniciar el diseño, y el desarrollo de ViAn en el ambiente de programación del IDL.

CAPITULO IV.

Diseño del Sistema ViAn

Los libros no se hacen como los niños, sino como las pirámides,
con un diseño premeditado y añadiendo grandes bloques,
uno sobre otro, a fuerza de riñones, tiempo y sudor.

Gustave Flaubert

CAPITULO IV.
Diseño del Sistema ViAn

IV. 1. Introducción

El diseño de un sistema computacional, nos indica "cómo" se realizará y transformará lo modelado en la fase de análisis; en esta etapa indicaremos de qué forma y con qué herramientas desarrollaremos el sistema ViAn. En esta fase de diseño, nos concentraremos en los detalles para los elementos de interfaz, el manejo de los datos, su almacenamiento, para el flujo de eventos e información, y aplicaremos la filosofía del lenguaje de programación para lograr los objetivos especificados.

La siguiente descripción muestra lo que involucra la fase del diseño de acuerdo a Freeman [Pressman, 1997]: "El diseño es una actividad en la que se toman decisiones importantes, frecuentemente de naturaleza estructural. Comparte con la programación un interés por la abstracción de la representación de la información y secuencias de procesamiento, pero el nivel de detalle es muy diferente en ambos casos. El diseño construye representaciones coherentes y bien planificadas de los programas concentrándose en las interrelaciones de los componentes al mayor nivel y en las operaciones lógicas implicadas en los niveles inferiores."

En este capítulo presentamos entonces, el diseño del sistema ViAn con el objetivo de establecer de qué forma desarrollaremos la solución técnica a nuestro software, con modelos en distintos niveles como lo menciona Freeman. Presentaremos diferentes puntos de vista a nivel de proceso e implementación, basándonos en nuestros planos y cimientos, que son nuestro análisis del sistema. En el diseño del sistema, presentaremos cómo es que se entrelazan las ideas y los hechos, ésto con las técnicas que el lenguaje nos permite manipular, así como las limitaciones que el sistema pudiera tener, ya sea momentáneamente (sólo en caso de nuestra primera versión) o de forma definitiva.

Para la fase del diseño, es necesario auxiliarnos de modelados y diagramas que permitan visualizar, comprender y completar las ideas que se desarrollaron en nuestro análisis. Los modelados y diagramas que mostramos en este capítulo, representan los siguientes aspectos a considerar en la fase de diseño: 1.*Estructura de datos*, 2.*Estructura del sistema*, 3.*Flujo de datos*, 4.*Flujo de eventos*, 5.*Secuencia de un evento*, y 6.*Diseño de interfaz*. Unas partes con mayor complejidad que otras, y que en conjunto forman al diseño completo de nuestro sistema ViAn para dar forma a la implementación y a las pruebas.

En la figura 4.1 presentamos el esquema general del diseño que desarrollaremos a lo largo de este capítulo. En dicha figura, se enfatiza cómo el análisis del sistema junto con los requerimientos del sistema (presentados en el capítulo III), y el análisis de los algoritmos, son los cimientos (o base) para empezar a construir a nuestro sistema ViAn. Presentamos las diferentes partes del diseño, con sus diferentes perspectivas o puntos de vista, pero que tratan de desarrollar a un nivel más cercano a la forma de la plataforma de programación, y así darle forma a la aplicación ViAn.

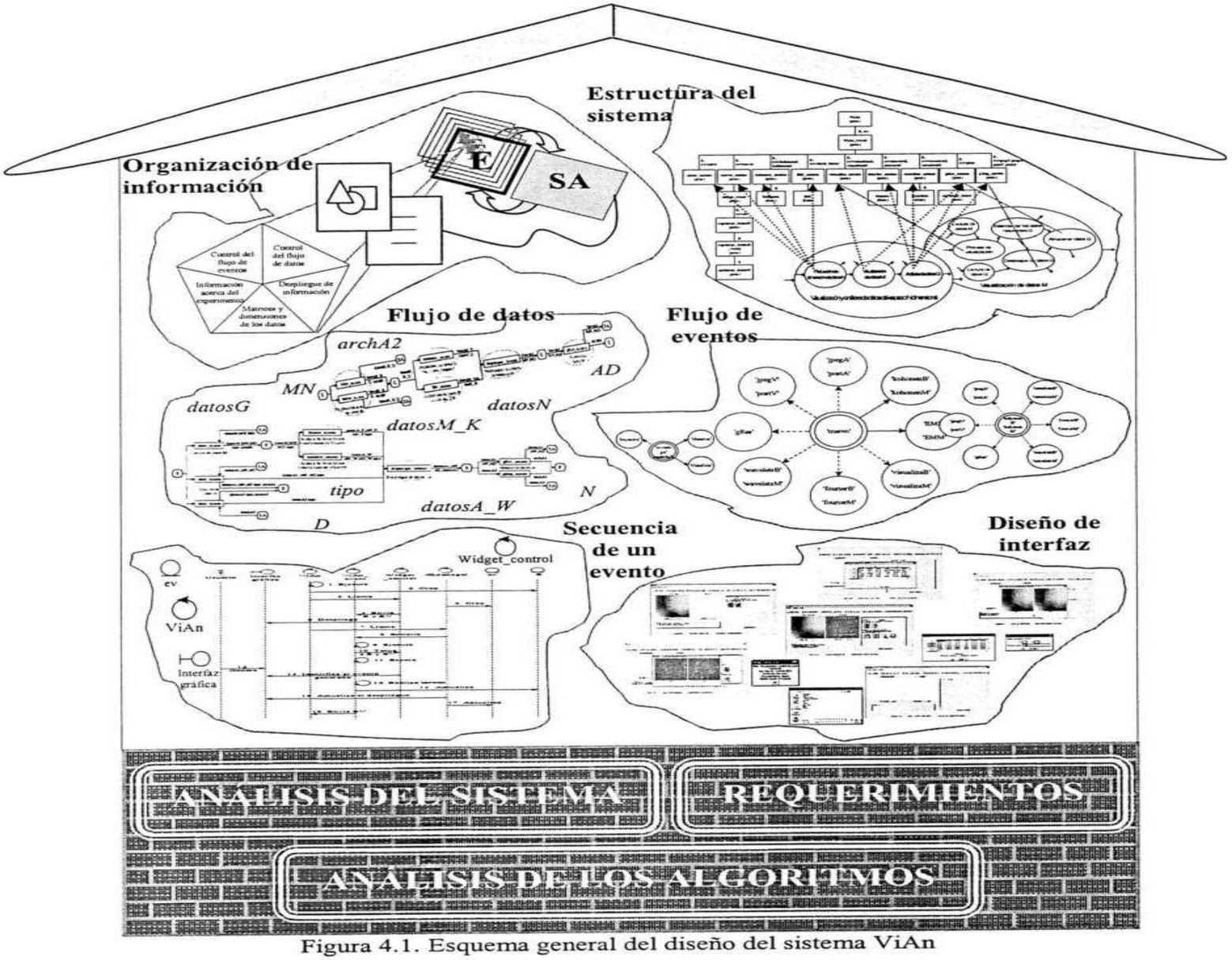


Figura 4.1. Esquema general del diseño del sistema ViAn

Las partes del diseño del sistema ViAn (como ya mencionamos) son: 1. *Estructura del sistema*, nos permitirá explicar cómo es que las funciones y procedimientos (módulos), se encuentran estructurados para realizar los procesos y tareas deseadas dentro de nuestro sistema, 2. *Estructura de datos*, es un elemento fundamental, aquí se encuentran los datos y variables necesarias con un orden para ser manipuladas en el tiempo y en el procedimiento o función que le corresponda, 3. *El flujo de datos*, que modela la forma en la que la información y los datos en general fluyen a lo largo de los procesos y objetos, 4. *El flujo de eventos*, esta parte indica cómo es que los eventos generados por el usuario a través de los elementos de interfaz permiten el acceso a otro u otros eventos, así como también los prohíbe, 5. *La secuencia de un evento*, aquí figuramos a los eventos, sus roles y la secuencia que siguen, según la filosofía del lenguaje y la tarea que desempeñamos, y finalmente 6. *El diseño de interfaz*, el cual es el que nos muestra las pantallas y sus eventos (botones, menús, etc.) que verá el usuario y que cubre las necesidades de lo establecido en el diseño del sistema en sí. Así entonces representamos a nuestro diseño del sistema ViAn con el esquema de la figura 4.1. Todos estos elementos se desarrollarán y presentarán en las siguientes secciones.

Una parte importante del diseño del sistema, que se expande con una parte del análisis del sistema, es el "análisis y diseño de los algoritmos" que los discutiremos en el capítulo V. En la figura 4.2 estructuramos el análisis y diseño de los algoritmos, y así completamos al diseño global del sistema. El análisis y diseño de los algoritmos consisten en la explicación de las técnicas de reducción y de las técnicas del análisis de datos, todas estas con sus respectivos algoritmos. Las técnicas que se desarrollarán, y que cumplen con requerimientos del sistema son las siguientes: 1. El algoritmo de Kohonen de Redes Neuronales, 2. Algoritmo EM del modelo de Variables Latentes, 3. Análisis Espectral con la Transformada de Fourier y 4. Análisis de Multirresolución con Wavelets (ondeletas).

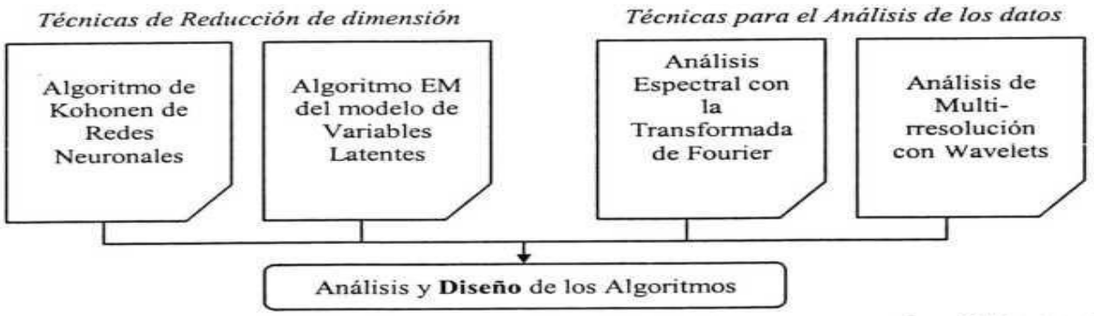


Figura 4.2. Diseño de las técnicas para reducción de datos y para el análisis de datos

IV. 2. Organización de información

En el desarrollo de un sistema se manejan muchos datos con diferentes objetivos, con diferentes formatos y estructuras para lograrlos. Para el diseño del sistema ViAn, es necesario organizar todos estos datos y demás información según dichas características, y según el control que se desee tener sobre ellos.

Para el diseño de nuestro sistema ViAn presentaremos tres elementos esenciales que tienen organizada la información con diferentes fines. El primer elemento es la *estructura de datos*, que nos auxilia a organizar los datos y variables necesarias para ejecutar el sistema. El segundo elemento es el *sistema de archivos*, que almacena para el usuario los datos resultantes de los procesos del sistema. Y el tercer elemento son los nombrados *proyectos del sistema* que es simplemente una forma de organizar a los archivos que se encuentran en nuestro sistema de archivos, organizarlos en grupos para el mejor entendimiento y manejo del experimento del usuario.

IV. 2. 1. Estructura de datos (E)

La estructura de datos es el espacio que contiene a las variables (con sus características definidas) que representan algunos de los datos que necesitamos para realizar las tareas y procesos de nuestro sistema. La ventaja que se tiene al utilizar una estructura de datos es que organizamos la información con diferentes alternativas y accedemos con mayor facilidad y seguridad a los datos que se deseen manipular. Las estructuras de datos varían en cuanto a su complejidad, para nuestro sistema ViAn, todos y cada uno de los datos que se requieren para ejecutar nuestros procesos, se encuentra definida una sola estructura de datos, a la cual nos referiremos con la abreviación "E", por lo tanto es una estructura de datos compleja que necesita de una explicación más amplia. Se definió una sola estructura de datos, debido a que el lenguaje de programación que utilizaremos en el desarrollo de ViAn (IDL), permite el paso de un solo dato entre la interfaz gráfica (definición de botones, menús, etc.) y la ejecución de los procesos que corresponden a los eventos generados de forma interna o por el usuario.

a) b)

Figura 4.3. Relaciones entre la E y el SA: (a) Diferencia entre la actualización de E, que se comunica con SA que permanece estable. (b) Comunicación entre una parte de la E y el SA.

La E es una parte muy delicada (por su complejidad) e importante de nuestro sistema, ésta se actualiza cada vez que un evento es generado a lo largo de la vida del programa[1]. Esto se debe a que en el manejo de eventos, siempre se está monitoreando a los elementos de interfaz para detectar una señal de entrada al sistema, como se explicó en los "Antecedentes". Cada vez que la interfaz se actualiza, la E se actualiza también, el SA no se actualiza. El SA se actualiza cada vez que el usuario lo solicite y el sistema lo realice, y no cada vez que se genere un evento. La figura 4.3(a) nos ayuda a ver esta diferencia.

La E es el enlace entre el sistema ViAn y el Sistema de Archivos (SA), ya que para almacenar datos en el SA, estos se toman directamente de la E, o más específicamente de una parte de la E que más adelante se explicará; y si se desea leer del SA la información que contiene algún archivo, se carga directamente en él o los datos correspondientes a la E. Para imaginarnos este enlace, la figura 4.3(b) nos muestra la comunicación entre la E y el SA.

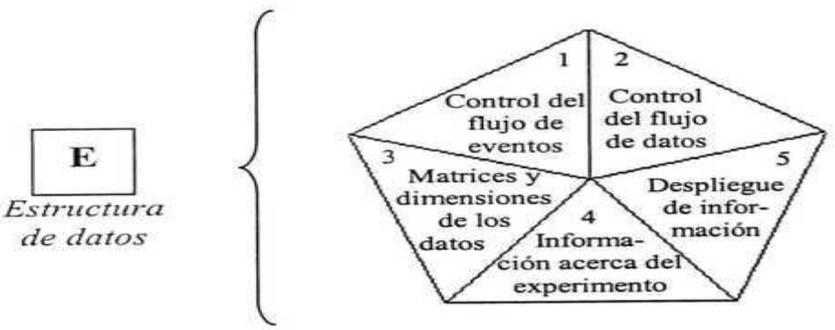


Figura 4.4. Los elementos de la estructura de datos E de nuestro sistema ViAn

La E está constituida por varias partes que contienen conjuntos de datos integrados por una característica, es decir, una parte la conforman variables que contienen los datos que se manipulan con un fin común. La forma en que se encuentra la E se muestra en la figura 4.4, está dividida en forma conceptual por cinco elementos. Estos elementos son los siguientes:

1. *Control del flujo de eventos*: Esta parte de la E permite el control del paso o flujo de los eventos según los que se vayan generando. Al generarse un evento, lleva consigo un permiso para activar la generación o desactivación de otro u otros eventos. Las variables de esta parte representan los widgets (véase los "Antecedentes", capítulo II), los que se pueden acceder a través de su identificador para modificar sus valores y también sus características.

[1] Al decir: "A lo largo de la vida del programa", nos referimos al tiempo de ejecución del sistema ViAn, desde que da inicio, hasta que se ejecuta la opción de salida del programa o se interrumpe la ejecución por cualquier error propio o externo.

2. *Control del flujo de datos*: Las variables que pertenecen a este grupo nos auxilian en el flujo de los datos de entrada, los datos reducidos y los datos analizados dentro de los procesos requeridos. Estas variables lo que hacen es formar una etiqueta para seguirle la pista a los datos para el mejor control sobre ellos, y así indican que datos son los que se llevan a los procesos solicitados para su actualización, despliegue o almacén de ellos. Un ejemplo es una variable que permite registrar si los datos resultantes fueron procesados por un algoritmo de reducción, por el otro o por ambos.

3. *Matrices y dimensiones de los datos*: Aquí se concentra la información más importante para nuestro sistema ViAn, puede decirse que es el núcleo de la E. Esta parte de E contiene a las matrices que se procesarán, y a las matrices resultantes de los procesos, incluyendo a la matriz de los datos originales del experimento nombrados datos N. Además, en este grupo se localizan las dimensiones de estas matrices. Otro factor para la importancia de este grupo, es que dichas matrices y dimensiones son las que se almacenan en el SA, y si se desea leer del SA, la información se convierte en estas matrices y datos.

4. *Información acerca del experimento:* Estos datos son lo que nos indican principalmente el nombre del o los archivos que se están utilizando del SA. Esta parte de la E es poco compleja, sin embargo es esencial para la comunicación con nuestro SA, lo que la hace muy importante. También se incluyen datos como el nombre del autor, fecha y hora de creación, toda la información que pertenece a un proyecto, el cual se explicará con mayor detalle más adelante. Entonces, esta parte de la E, no solo se comunica con un elemento de la organización de información, el SA, sino con otro elemento más, los proyectos. Lo que significa que aunque no es muy compleja es muy delicada y de gran valor.

5. *Despliegue de información*: Esta parte de la estructura contiene a los identificadores que conforman a la interfaz, con el fin de auxiliar en el despliegue de información. Significa que es otra parte de la E donde representamos a los widgets, esta vez con el fin de identificar en cuales se desplegará la información que nos incumba, ya sea en formato de texto o en formato imagen Por ejemplo, qué widget desplegará la matriz de datos resultante de la reducción de dimensión.

Cabe dar énfasis que la E y el SA se comunican, como lo presentamos en la figura 4.3a), solamente con las partes 3 y 4 de la E. Esta comunicación es posible con procesos determinados en el sistema ViAn como quedará claro con la presentación de los modelados del diseño del sistema. En el aspecto de los proyectos, a su vez son estructurados por una parte de esta E, con la *información acerca del experimento*, misma que se comunica con SA, ya que los proyectos dependen fuertemente de nuestro SA como lo explicaremos.

IV. 2. 2. Sistema de Archivos (SA)

Necesitamos cinco tipos diferentes de archivos en total para ejecutar todo proceso en el sistema. El SA puede contener desde cero hasta X número de archivos de un solo tipo. Estos diferentes tipos de archivos se encuentran en la figura 4.5(a), aquí presentamos cómo es que forman parte del SA. El tipo de datos depende del rol que tengan en los procesos y algoritmos, y se distinguen a través de sus extensiones.

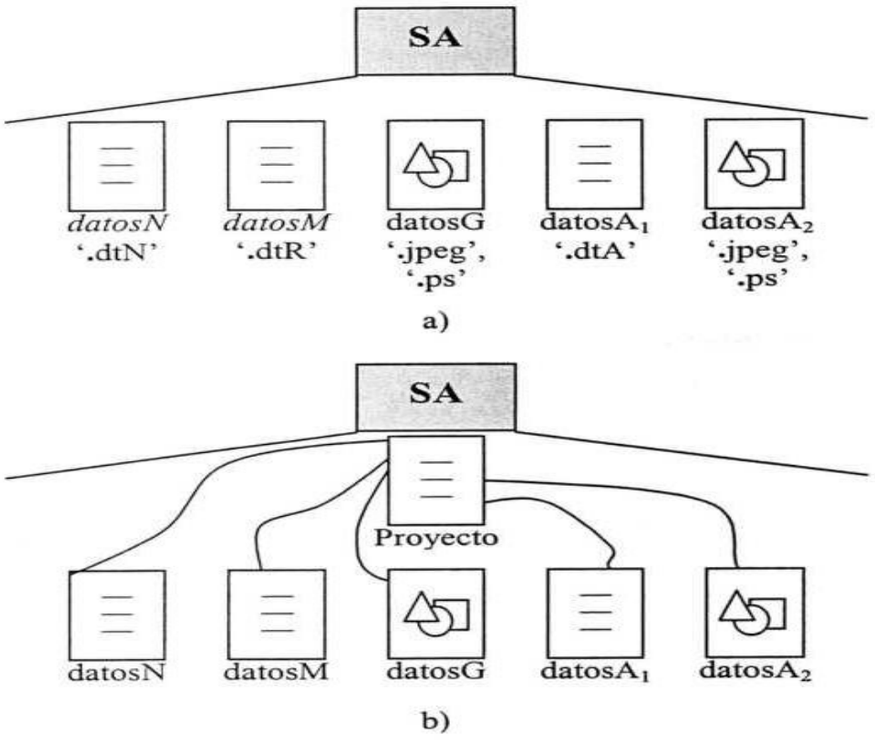


Figura 4.5. El SA y sus diferentes tipos de archivos: (a) SA y los cinco tipos de archivos para los procesos. (b) SA y los seis tipos de archivos, "Proyecto" hace referencia al resto.

Mencionábamos que necesitamos cinco tipos de archivos, sin embargo, contamos con un sexto tipo de archivo, que no son para ejecutar los procesos directamente, son para la administración de los proyectos del sistema que veremos adelante con más detalle. En la figura 4.5(b) se muestra como los proyectos pertenecen al SA para hacer referencia al resto de los archivos.

Ahora proporcionaremos una explicación de cinco de los seis archivos que se generan y pertenecen al SA, y le dan funcionalidad de manejo de datos a ViAn. El archivo para proyectos será un tema para desarrollar aparte, se presentará más adelante una explicación sencilla de lo que éste trata.

- *datosN.* Estos son los datos originales del experimento, datos de entrada al sistema, considerados así porque no han sido procesados por ninguna tarea. Dentro del sistema, estos datos además de almacenados y leídos como el resto de los archivos, pueden ser capturados, actualizados o solamente estudiados por el usuario. El formato que tienen es el de texto, que se especificará más adelante. La extensión para reconocer a este archivo es: '.dtN', que significa: "**datos N**".

- *datosM.* Estos datos son el resultado del proceso de reducción de dimensión de datos por cualquiera de nuestros dos algoritmos (me refiero al de Kohonen y al EM para variables latentes). Estos datos también los encontraremos en formato de texto. La extensión para reconocer a este archivo es: '.dtR', que significa: "**datos Reducidos**". La extensión no es modificada por el algoritmo con que se redujo, ya que al ser estos datos procesados y/o almacenados, el interés que se tiene sobre ellos en el sistema es como datos de entrada.

- *datosG.* Así les llamamos a los datos que ya fueron reducidos y que además han sido visualizados por el usuario. Este tipo de archivo se encuentra en formato de imagen, y los datos de imagen tienen a su vez extensiones que los hacen diferenciarse, éstas son '.ps' y '.jpeg' que son formatos de imagen establecidos.

- *datosA*$_1$. Los datosA$_1$ indican que son datos que han sido analizados por cualquiera de los dos algoritmos para el análisis de los datos (éstos son: la transformada de Fourier y Wavelets), se almacenan en formato de texto. La extensión para reconocer a este archivo es: '.dtA', que significa: "**datos Analizados**". Al igual que los datosM, no importa con que algoritmo se analizaron estos datos para determinar su extensión, una vez que han sido procesados.

- *datosA*$_2$. Al igual que los datosA$_1$, son aquellos que han sido procesados por los algoritmos de análisis de datos, pero a diferencia de los datosA$_1$, los datosA$_2$ son archivos que se encuentran en formato de imagen, dicha imagen vista por el usuario. Las extensiones al igual que en los datosG son las correspondientes a los formatos de imagen '.ps' y '.jpeg'.

IV. 2. 3. Formato de los archivos de datos

Los archivos que contienen a los datos N (datos originales del experimento), deben contener el número de datos (N) y el número de dimensiones de los datos (D) en el primer renglón, y los datos en forma de matriz, donde el número de renglones sea equivalente a N y el número de columnas equivalente a D. Deben mantener el formato que presentamos en la figura 4.6.

Cada dato debe estar separado por una coma como se muestra en la figura, los datos del primer renglón del archivo deben ser número enteros mayores de cero, y los datos pueden ser números entero y/o números flotantes, sin importar el signo. Un ejemplo de un archivo de datos originales N, es el que presentamos en la figura 4.7, donde N=7 y D=19.

N, D							
$Dato_{11}$,	$Dato_{12}$,	$Dato_{13}$,	$Dato_{14}$,	$Dato_{15}$,	$Dato_{16}$,	$Dato_{17}$, .. ,	$Dato_{1D}$
$Dato_{21}$,	$Dato_{22}$,	$Dato_{23}$,	$Dato_{24}$,	$Dato_{25}$,	$Dato_{26}$,	$Dato_{27}$, .. ,	$Dato_{2D}$
$Dato_{31}$,	$Dato_{32}$,	$Dato_{33}$,	$Dato_{34}$,	$Dato_{35}$,	$Dato_{36}$,	$Dato_{37}$, .. ,	$Dato_{3D}$
$Dato_{41}$,	$Dato_{42}$,	$Dato_{43}$,	$Dato_{44}$,	$Dato_{45}$,	$Dato_{46}$,	$Dato_{47}$, .. ,	$Dato_{4D}$
$Dato_{51}$,	$Dato_{52}$,	$Dato_{53}$,	$Dato_{54}$,	$Dato_{55}$,	$Dato_{56}$,	$Dato_{57}$, .. ,	$Dato_{5D}$
:	:	:	:	:	:	:	:
$Dato_{N1}$,	$Dato_{N2}$,	$Dato_{N3}$,	$Dato_{N4}$,	$Dato_{N5}$,	$Dato_{N6}$,	$Dato_{N7}$, .. ,	$Dato_{ND}$

Figura 4.6. Formato del archivo que almacena los datos N

```
7, 19
1.0, 2.0, 3.0, -4.0, 5.0, 6.0, 7.0, -8.0, -9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0, 16.0, 17.0, -18.0, 19.0
2.1, 4.1, 6.1, 8.1, 10.1, 12.1, 14.1, 16.1, 18.1, 20.1, 22.1, 24.1, 26.1, -28.1, 30.1, 32.1, -34.1, 36.1, -38.1
3.2, 6.2, 9.2, 12.2, -15.2, -18.2, 21.2, 24.2, 27.2, -30.2, 33.2, 36.2, 39.2, 42.3, 45.2, 48.2, 51.2, 54.2, 57.2
-4.3, 8.3, 12.3, 16.3, 20.3, 24.3, 28.3, 32.3, 36.3, 40.3, 44.3, -48.3, -52.3, -56.3, 60.3, 64.3, 68.3, 72.3, 76.3
5.4, 10.4, 15.4, -20.4, 25.4, 30.4, 35.4, -40.4, 45.4, 50.4, 55.4, 60.4, 65.4, 70.4, 75.4, 80.4, -85.4, 90.4, 95.4
6, 12, 18, -24, 30, 36, -42, 48, -54, 60, 66, 72, -78, 84, -90, -96, 102, 108, -114
7.0, 14.1, 21.2, 28.3, 35.4, 42.5, 49.6, 56.7, 63.8, 70.9, -77, 84.0, 91, 98.0, 105, 112.0, 119, -126.0, 133
```

Figura 4.7. Ejemplo de un archivo de datos N

De igual forma, los datos resultantes del proceso de reducción de dimensión de datos y del proceso de análisis de datos, se archivarán con el mismo formato, específicamente los datosM y $datosA_1$ (explicados anteriormente) pero con una ligera diferencia: en lugar de que el número de datos este regido por la variable N, estos serán guardados con las variables M para los datos reducidos y A para los datos analizados. No mostraremos ejemplos de archivos de estos dos últimos tipos de archivos, ya que con lo mostrado (ejemplos para datos N) el usuario puede realizar su propio archivo en un editor de textos, grabarlo con la extensión correspondiente, y el sistema estará capacitado para leerlo como tal, donde los datos M y A son reconocidos y manipulados internamente por el sistema ViAn.

Los otros tipos de archivos con formato de imagen son: datosG y $datosA_2$ y son una serie de bits que no podrán ser leídos del SA desde nuestro sistema, solo podrán almacenarse. En estos dos casos muy en particular (datosG y $datosA_2$), si se desean observar las imágenes dentro del sistema, para los datosG, deben leerse los datosM que le corresponda y ejecutar el proceso que permite la visualización de estos

datos, y para observar los datosA_2 en el sistema, deben ser leídos los datos que les corresponda del SA y ejecutar entonces el proceso que permite el análisis de los datos con el algoritmo que se desea, y que le corresponda. Se le deja la responsabilidad al usuario para saber que archivos con exactitud les corresponde a dichos datos. El formato de imagen que se manejarán para estos archivos, serán dos: postcript ('.ps') y jpeg ('.jpeg'), el usuario tendrá la oportunidad de elegir en cual de los dos (o en ambos) almacenará sus imágenes resultantes.

IV. 2. 4. Proyectos del sistema ViAn

Los proyectos serán diseñados como un requerimiento innovador, esto es porque sin la consideración de formar proyectos, los requerimientos (normales) serían cumplidos, y además de forma satisfactoria. Esto es, no se encuentran en los requerimientos de usuario, ni técnicos del sistema. Es por esto mismo, que deseamos estén bien explicados los conceptos de nuestra E y nuestro SA, antes de introducir y comprender esta parte del sistema, los proyectos. Si bien, los proyectos son diseñados en la primera versión de ViAn, su desarrollo en la implementación se propone para una siguiente versión.

Los proyectos son una herramienta muy útil para el usuario, ya que puede organizar de una forma estándar los resultados que nuestro sistema ViAn le proporciona, agrupándolos en diferentes áreas de trabajo si así lo desea. Los proyectos se pueden considerar como una llave para empezar a ejecutar los procesos del sistema.

Técnicamente hablando, un proyecto del sistema, es un archivo de datos, que será leído, almacenado y modificado a través de nuestro sistema ViAn. Este archivo también pertenece a nuestro SA (como se mostró en la figura 4.5b) y se comunica con una parte de nuestra estructura de datos E.

Se menciona que el archivo de *proyecto* hace referencia al resto de los archivos, donde se enlistan los archivos que contiene el proyecto. Los proyectos agrupan de uno a X número de archivos (que explicamos anteriormente), y son los datos de entrada y/o salida de algún proceso, y también pueden ser imágenes resultantes de algún proceso. Un proyecto debe contar con al menos un archivo de tipo texto con datos N.

Los proyectos serán leídos del SA cuando el usuario elija abrirlos, estará abierto solamente uno a la vez durante la vida del programa, esto quiere decir que si el usuario opta por abrir un proyecto mientras se encuentra otro abierto, al momento de abrirlo, se cerrará el que se encontraba abierto. Estas medidas de control deben ser tomadas por el investigador, principalmente si no ha archivado sus resultados.

El formato para el archivo de proyectos del sistema, es diferente a los presentados anteriormente. Ya que no son datos para los procesos, sino información para el mismo sistema, que ayudan a manipular los datos para los procesos. Para conocer el formato de archivo de un proyecto, presentamos la figura 4.8, en ella encontramos la información necesaria para agrupar y acceder a diferentes archivos de diferentes tipos. Los datos que contiene el archivo para proyectos son.

```
Nombre_proyecto
Nombre_autor
Fecha_hora_creación
Archivo1, Tipo_archivo1
Archivo2, Tipo_archivo2
    :          :
ArchivoX, Tipo_archivoX
```

Figura 4.8. Formato del archivo que almacena a un proyecto

1. Nombre_proyecto: *Nombre del proyecto*. Indica el nombre con el que será reconocido el proyecto. Su extensión es: '.via', parecido al nombre del sistema: ViAn, que significa "**V**isualización y **A**nálisis". Aunque ViAn desarrolla un proceso importante y básico para la Visualización y Análisis, me refiero a la *Reducción de dimensión de datos*.

2. Nombre_autor: *Nombre del autor del proyecto*. Es el nombre del usuario que ha creado el proyecto para procesar los datos de su experimento o sus experimentos.

3. Fecha_hora_creación: *Fecha y hora de creación del proyecto*. Es el registro de la fecha y la hora en que el usuario creó el proyecto.

4. ArchivoX, Tipo_archivoX: *Nombres y tipos de los archivos que contiene el proyecto*. En esta parte, se encuentran enlistado los archivos y el tipo de estos que contiene el proyecto. Con ésto se hace referencia a los archivos que se encuentran en el SA para su se acceso a través de la E.

Enseguida presentamos un ejemplo de un archivo de proyecto en la figura 4.9, donde existen tres archivos en total, donde uno forzosamente debe ser con datos N, y con este debe iniciarse la lista, le sigue un archivo de datosM, datos reducidos, y por último un archivo con formato de imagen, el tipo de dato nos indica que es una imagen visualizada, y como aclaración, no analizada.

```
H2OMexico.via
Alberto Quezada Franco
Tue Apr 08 14:39:50 2003
LagRiosBC.dtN, datosN
Lagunas_Rios.dtR, datosM
LagunaRioBC.ps, datosG
```

Figura 4.9. Ejemplo de un archivo de proyecto

Tendremos dos formas de acceder a los proyectos, una de ellas, es *para leer un proyecto previamente almacenado*, y la segunda es *para crear un proyecto*. Las secuencias en estos dos casos y dos sub-secuencias son las que siguen:

- Para abrir un proyecto previamente almacenado:
  1. Se elige la opción de abrir un proyecto del SA.
  2. Seleccionar el proyecto.
  3. Leer el archivo que contiene al proyecto.
  4. Cargar la información del archivo a la E.
  5. Desplegar el nombre del archivo, nombre del autor, fecha de creación y los nombres de los archivos que pertenecen al proyecto.
  6. Solicitar la selección de un archivo para iniciar el proceso correspondiente, tiene dos opciones para seleccionar un archivo que pertenezca al proyecto, estas son:
     a. Seleccionar un archivo que se encuentra en una lista desplegada. Ver tercer caso.
     b. Crear un archivo de datos originales de un experimento (datos N). Ver cuarto caso.

- Para crear un proyecto:
  1. Se solicitan los datos del proyecto (nombre del proyecto, nombre del autor).
  2. Se solicitan los datos N, estos pueden ser accedidos de dos formas:
     c. Abrir un archivo con datos N previamente almacenados en el SA y que no pertenezca al proyecto del que estamos tratando. Ver quinto caso.
     d. Crear un archivo de datos originales de un experimento (datos N). Ver cuarto caso.

- Seleccionar un archivo que se encuentra en una lista desplegada (punto a.):
  1. Solicitar la selección de un archivo para su proceso.
  2. Cargar el archivo elegido en la matriz o dato correspondiente que se encuentra en la E, y dar acceso al flujo de datos indicado.

3. Actualizar el proyecto en el SA.
4. El sistema se encuentra en la espera de un nuevo evento.

- Crear un archivo de datos originales de un experimento (punto b. y d.):
  1. Se solicitan las dimensiones de la matriz de datos N.
  2. Se despliega el formato de entrada.
  3. Se espera hasta la introducción de los todos los datos.
  4. Se cargan los datos N en la matriz correspondiente que se encuentra la E.
  5. Se agrega a la lista del proyecto el nombre del archivo y las dimensiones de sus datos.
  6. Se actualiza el proyecto en el SA.
  7. Se almacena los datos en el SA.
  8. El sistema se encuentra en la espera de un nuevo evento.

- Abrir un archivo con datos N previamente almacenados en el SA (punto c.):
  1. Se muestran los archivos del SA que sean datos N, y que no pertenezcan al proyecto.
  2. Se solicita la selección de un archivo.
  3. Se carga el archivo en la matriz o dato correspondiente de nuestra E.
  4. Se agrega a la lista del proyecto el nombre del archivo y las dimensiones de sus datos.
  5. Se actualiza el proyecto en el SA.
  6. El sistema se encuentra en la espera de un nuevo evento.

Existe la posibilidad de que los proyectos contengan archivos con formato de imagen, al pertenecer éstos a un proyecto solo podrán desplegarse en el lugar que le corresponda dentro del sistema. Solo puede desplegarse debido a que un archivo con formato de imagen no puede ser procesado por el sistema ViAn, para realizar un proceso se necesita de los datos numéricos. Significa que, si la imagen fue hecha por la visualización de datos y esa imagen se desea analizar, no se podrá, ya que no se introdujo esta función (o procedimiento) para procesar el análisis de datos. Para este análisis de datos se necesaria la matriz de datos que contiene el resultado de la reducción de dimensión que origina esta imagen. Y si se trata de una imagen previamente analizada, no puede desplegarse el valor numérico de la operación a partir de la imagen.

A partir de estos elementos que ayudan a organizar la información, en las siguientes secciones, continuamos con los modelados de nuestro diseño, aquí describiremos ampliamente el archivo de los proyectos, y cuales son los roles de los tres elementos presentados (E, SA y proyectos).

IV. 3. Estructura del sistema

La estructura del sistema muestra como los procedimientos y funciones se comunican como módulos de un programa para cubrir los requerimientos del sistema ViAn. Se presenta en el diagrama de la figura 4.10 la estructura básica de ViAn, no son todos los módulos del programa. Para cubrir las necesidades del diseño y la implementación en IDL, la estructura es más amplia y complicada, con más ramas de módulos. Entonces los que presentamos en dicha figura son las principales tareas (procesos o actividades) y son las que hay que fortalecer con los demás modelados del diseño y por supuesto, con la implementación del sistema.

Para elaborar a la estructura del sistema, nos basamos en el modelado del flujo de procesos a más detalle. En la estructura del sistema detallamos como es que la E (estructura de datos de ViAn) se pasa de un módulo a otro, sin especificar qué datos son los que se necesitan para procesarlos. Por ello, necesitamos del *diagrama de flujo de datos*, para que nos detalle que datos son los que son manipulados por los módulos. La E es transmitida a los módulos, debido a que se actualiza completa y no solo parte de ella. En la estructura del sistema, se modela a través de qué eventos (se especifican con *ev*) es que se comunican los módulos. pero no indican cual es el flujo de los eventos, cuales están activos y cuales inactivos, por lo que se especificará esto más adelante con el *diagrama de flujo de eventos*.

En la estructura del sistema representamos al módulo *ViAn* encabezando a nuestro diagrama, este módulo es un conjunto de widgets, significa que es quien ejecuta el despliegue de la interfaz gráfica (botones, menús, etiquetas, campos de texto, etc.) para que el usuario interactúe con el sistema. El usuario genera un evento (por ejemplo: presiona un botón) e inicia la ejecución del módulo *ViAn_event*, el módulo principal de nuestro programa. *ViAn_event* ejecuta el procedimiento que obedece al evento generado.

Cada vez que se genera una nueva interfaz gráfica (una nueva ventana), la acompaña su módulo *handler* (módulo que obedece a los eventos generados a través de la interfaz gráfica). Esta forma para el manejo de módulos del ambiente IDL, se explicó en los "Antecedentes". Para identificar a estos dos tipos de módulos dentro de nuestra estructura del sistema, identifiquemos primero a un módulo que tenga la terminación "_event", éste es el que ejecuta los procesos. Y el módulo que le antecede (que está arriba de él) y que lleva su mismo nombre omitiendo "_event", es el módulo que ejecuta la interfaz gráfica. Esto nos indica que el módulo de la interfaz gráfica llama a su módulo con terminación "_event".

Al generarse un evento, estaremos iniciando siempre en el módulo principal *ViAn_event* y llamará a uno de los 9 módulos con la terminación "*_accion*", que se conectan directamente como lo representamos en el diagrama de la estructura del sistema (figura 4.10). En la conexión del módulo principal y los 9 módulos mencionados, mostramos al elemento *ev* y a nuestra estructura de datos E.

El elemento *ev* es una estructura que especifica a través de qué widget (elemento gráfico, el cual podría ser un botón por ejemplo) se generó el evento. Para más detalle, consulte a los "Antecedentes". La E se envía en todos los casos, debido que, como mencionamos, ésta contiene todos los datos, y cada módulo se encargará de acceder y actualizar los datos necesarios en el proceso correspondiente.

Presentamos un ejemplo de relación entre el modelado de procesos de acuerdo a los casos de uso y como los diferentes módulos de la estructura del sistema lo lleva a cabo. El caso más genérico lo describimos en el modelado de flujo de procesos que presentamos en el "Análisis del sistema", y es el modelado para "ViAn – Visualización y análisis", figura 3.5: El primer proceso *Reducción de dimensión de datos N* lo conforman los módulos abrir_accion, nuevo_accion, kohonen_accion, EM_accion y gRes_accion mientras el proceso *Visualización de datos M* es realizado por el módulo visualiza_accion y gImg_accion. por último observamos al proceso *Análisis de los datos G* para lograr las tareas de este proceso recurrimos a los módulos fourier_accion, wavelets_accion, gRes_accion y gImg_accion.

Con esto determinamos la relación que tienen los módulos del sistema y el modelado de procesos que mostramos en el capítulo III. Esta relación la presentaremos en la tabla IV.I. Presentamos que, para cada proceso del modelado (2da. columna) que pertenece al modelado que se indica en la primera columna, se muestran los módulos con lo que se logrará la ejecución del proceso (tercer columna) y se describen pequeñas observaciones para la mejor comprensión de esta relación.

Con la tabla IV.I, establecemos cómo es que el sistema desarrollado con los procesos, especificado en su análisis, se estructura en un programa con los módulos necesarios para realizar el sistema ViAn. Encontraremos dentro de la tabla los términos: a) *ES* que significa Estructura del Sistema, b) *alg.*, significa algoritmo, y c) *transfor.*, significa transformada. Estos significados se especifican en la misma tabla.

Modelado	Procesos del modelado.	Módulo(s) de la ES (Estructura del Sistema)	Observaciones
	Pre-procesamiento de datos N	•abrir_accion •nuevo_accion	Hay dos opciones para preparar los datos de entrada.
Reducción de dimensión de datos N Figura A4 (Apéndice A)	Inicializar datos para reducción	•abrir_accion •nuevo_accion	El proceso no requiere módulo propio por lo que se realizará en el módulo del proceso que le antecede.
	Reducción de datos N con el algoritmo de Kohonen	•kohonen_accion	Se realiza las operaciones necesarias para desarrollar al alg. (algoritmo) de Kohonen.
	Reducción de datos N con el algoritmo EM	•EM_accion	Se realiza las operaciones para desarrollar al alg. EM para Variables Latente.
	Estandarizar los datos resultantes M	•despliegue_datosM	Se estandarizan los datos para desplegarlos sin importar que alg. los procesó.
	Almacenar datos M	•gRes_accion	Almacena datos de tipo texto (matriz).
Pre-procesamiento de datos N Figura A5 (Apéndice A)	Lectura de datos N	•abrir_accion	Abre un archivo del SA, el usuario decide el tipo de archivo.
	Mostrar formato de entrada	•nuevo_accion	Solicita la información para un archivo que contenga datos N (de entrada).
	Introducir datos del experimento	•capturar_datosN	Se muestra una tabla para la introducción de los datos originales N.
	Estandarizar datos N para reducción	•capturar_datosN	El proceso no requiere módulo propio por lo que se realizará en el módulo del proceso que le antecede.
	Almacenar datos N	•archivar_datosN	Almacena datos de tipo texto (matriz).
Visualización de datos M Figura 3.6 (Capítulo III)	Lectura de datos M	•abrir_accion	Abre un archivo del SA, el usuario decide el tipo de archivo.
	Lectura de datos G	•abrir_accion	Abre un archivo el SA, el usuario decide el tipo de archivo.
	Proceso de visualización	•visualiza_accion	La matriz resultante de la reducción la convierte en imagen para su visualización.
	Estandarizar los datos resultantes G	•visualiza_accion	El proceso no requiere módulo propio por lo que se realizará en el módulo del proceso que le antecede.
	Despliegue de datos G	•despliegue_datosG	Se estandarizan los datos para desplegarlos, ya sean leídos del SA o hayan sido procesados solamente.
	Almacenar datos G	•gImg_accion	Almacena datos de imagen.
Análisis de datos G Figura A6 (Apéndice A)	Lectura de datos M	•abrir_accion	Abre un archivo del SA, el usuario decide el tipo de archivo.
	Lectura de datos A	•abrir_accion	Abre un archivo del SA, el usuario decide el tipo de archivo.
	Inicializar datos para análisis	•abrir_accion	Abre un archivo del SA, el usuario decide el tipo de archivo.
	Análisis de datos G con la transformada de Fourier	•fourier_accion	Se realiza las operaciones necesarias para desarrollar la transfor. (transformada) de Fourier.
	Análisis de datos G con la transformada de Wavelets	•wavelets_accion	Se realiza las operaciones necesarias para desarrollar la transfor. Wavelets.
	Estandarizar los datos resultantes A	•despliegue_datosA	El proceso no requiere módulo propio por lo que se realizará en el módulo del proceso que le sigue.
	Despliegue de datos A	•despligue_datosA	Se estandarizan los datos para desplegarlos sin importar que alg.los procesó, ni si son leídos del SA o no.
	Almacenar datos A	•gRes _accion •gImg_accion	Almacena datos de tipo texto (matriz) y almacena datos de tipo imagen.

Tabla IV.I. Relación de la estructura del sistema y el modelado de procesos

La figura 4.10 representa entonces, a la estructura del sistema ViAn. El módulo principal *Vian_event* se encuentra en estado de espera hasta que el usuario genere un evento, éstos pueden ser cualquiera de los que presentamos en la estructura del sistema. Al generarse el evento se ejecutan los módulos con los que se conectan directamente en el diagrama. Estos módulos realizan el procedimiento correspondiente, y al finalizar, la E se actualiza y volvemos al módulo *ViAn_event* en estado de espera.

Si se genera el evento *nuevo* el sistema ejecuta al procedimiento *nuevo_accion* que llama *nuevo_event* y a su vez llama a *nuevo_accion* para que ahora éste último espere la generación de un evento, si el evento fue *ok*, entonces se procede a la *capturar_datosN* a través del procedimiento que llama; ya que han sido capturados, entonces debe indicarse con un evento más (*ok*) y el módulo que se ejecutará es el de *archivar_datosN*.

En el caso de que el evento que recibe *Vian_event* fuese *abrir* entonces el módulo *abrir_accion* se ejecuta, y dependiendo de la solicitud del usuario, se llamará a uno de los dos procedimientos con los que se ve que tiene unión, es decir, en este caso tener dos conexiones directas significa que tiene dos opciones de proceso.

El caso anterior es diferente a los de *kohonen_accion* y *EM_accion*, ya que éstos tienen a dos módulos conectados, pero éstos llaman a los respectivos de forma secuencial, primero mandan llamar a la función correspondiente a cada uno, y después al *despliegue_datosM*. Para solicitar la ejecución de *kohonen_accion*, se debe generar uno de dos eventos: *kohonenB* o *kohonenM*, la diferencia de estos es que el primero es llamado desde un botón que se encuentra en la interfaz principal, y el segundo es llamado desde el menú principal, que estará en la parte superior de la interfaz principal.

El mismo caso tenemos para EM_*accion*, *EMB* significa que es llamado desde un botón y *EMM* que es llamado desde el menú. Ahora bien, tenemos a otros casos donde un módulo tiene conectados a dos módulos, éstos son los procedimientos *fourier_accion* y *wavelets_accion*, que llaman a las funciones que les corresponde y después llaman al procedimiento *despliegue_accion*. Estas dos las agregamos para completar el diseño, dado las características del proyecto ésto estará limitado al nivel de implementación.

También se encuentra el caso en que se solicita a un procedimiento no solo con dos opciones de evento sino con cuatro: *jpegV*, *jpegA*, *postV*, y *postA*, es para la ejecución del procedimiento *gImg_accion*. Todos estos eventos llevan consigo características que le sirven al módulo a determinar qué es lo que hará en su proceso con precisión.

Por último explicamos a los casos más sencillos. Uno, es la ejecución del procedimiento *visualiza_accion* con cualquiera de los eventos *visualizaB* (botón de la interfaz) o *visualizaM* (opción del menú principal). Dos, es para el evento *gRes* para llamar al procedimiento *gRes_accion*, que a su vez llama al procedimiento *archivar_datos*. Para conocer más detalles de cada uno de los módulos, presentamos la información completa en la tabla B1, en el apéndice B.

Supongamos el caso en el que el usuario desea reducir y visualizar sus datos, y supongamos que estos datos que ya se encuentran en la E (Estructura de datos). El usuario genera el evento *kohonenB*, correspondiente a un botón en la interfaz gráfica, el proceso k*ohonen_accion* inicia su ejecución, solicita a la función *Kohonen* que contiene el desarrollo del algoritmo, esta función le regresa los datos M, que corresponden a los datos reducidos.

El sistema regresa al estado de espera después de actualizar la E, donde alteró al dato que contiene a los datos reducidos. El usuario ahora genera al evento *visualizaB*, lo hizo a través de un botón en la interfaz, y el procedimiento *visualiza_accion* inicia su ejecución, para procesar los datos M, que los toma de la E, este mismo procedimiento despliega la imagen generada con los datos reducidos, y el sistema regresa nuevamente al estado de espera de la generación de un evento.

Enseguida presentamos a la figura 4.10, es el diagrama que muestra la estructura del sistema con los módulos principales de ViAn.

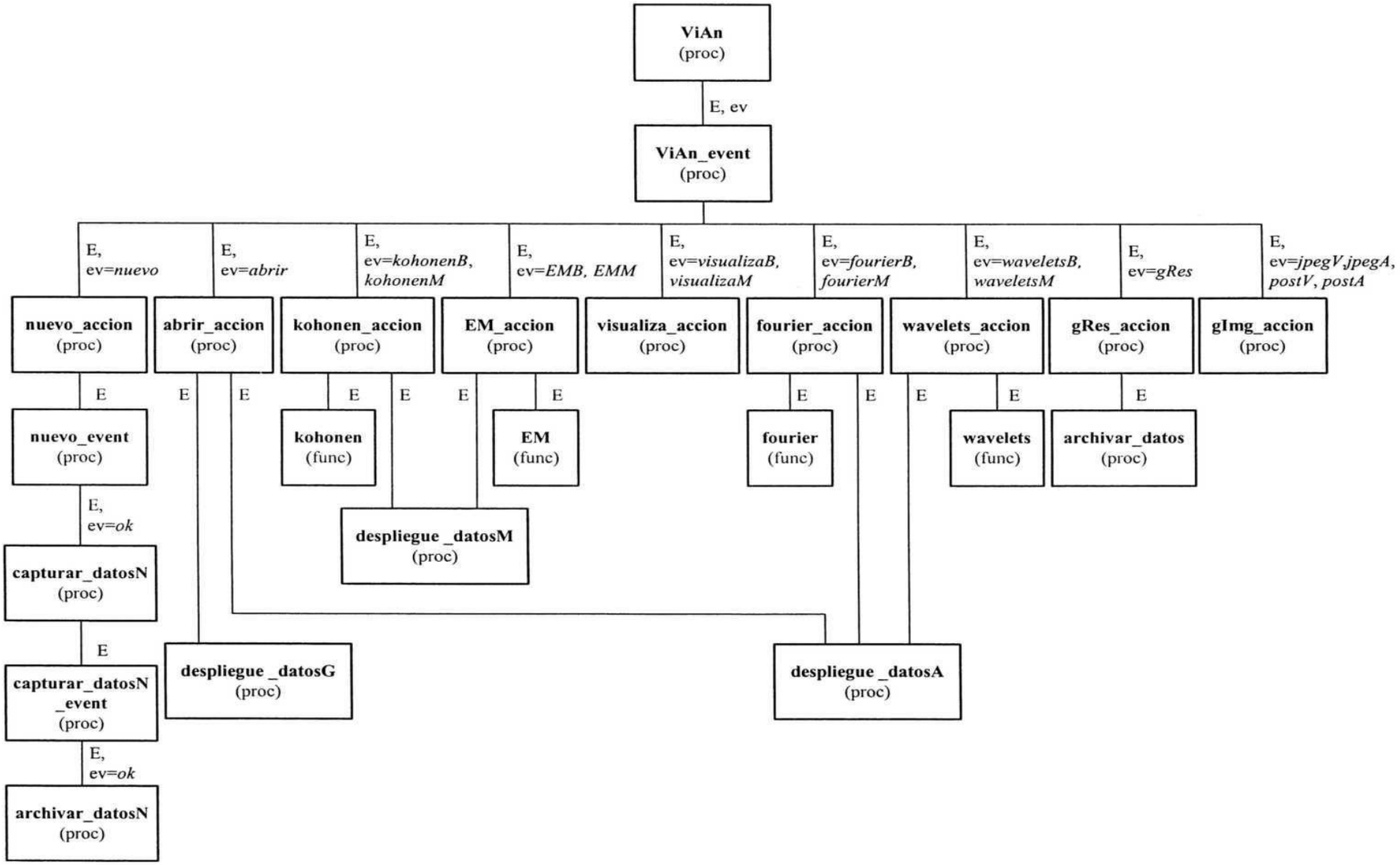


Figura 4.10. Diagrama que muestra la estructura del sistema con los módulos principales

IV. 4. Flujo de datos

Una vez definidos nuestros datos (estructura de datos), se modela el flujo de éstos a través de los procesos del sistema. A partir del modelado de procesos (capítulo III) y de la estructura de datos (presentada anteriormente), formamos a los diagramas para representar al flujo de datos. Presentaremos cómo se relacionan los módulos descritos en la estructura del sistema, con la E (Estructura de datos) y con el SA (Sistema de Archivos), para proporcionar a un modelado de procesos y caso de uso, su correspondiente flujo de datos.

Un proceso se realiza en uno o más eventos, y un evento puede ejecutar a uno o más módulos de un programa. Para realizar el flujo de datos necesitamos conocer con qué módulos vamos a trabajar para lograr el proceso deseado. Para el sistema ViAn, definimos la relación de los procesos y los módulos a través de eventos en la tabla IV.I, esta relación ahora la utilizaremos para el flujo de datos.

En los diagramas para el flujo de datos, la E representa a nuestra estructura de datos básica y única, y que sabemos contiene a todos los datos que necesitamos para modelar en estos diagramas. Cada módulo recibe los datos de E y lo envía también a ella, y es así como la estructura de datos se actualiza.

Cada vez que la E recibe datos y que a su vez envía otros, significa que terminó un evento y se generó otro evento. Cuando observemos que la E solamente recibe datos y ella no envía ninguno, significa que los datos enviados se actualizan en la E y el mismo evento generado continúa.

Los módulos arrojan todos sus datos de salida a la E sin importar qué proceso o qué módulo debe seguirle (ya que tratamos con eventos), puede ser que el módulo siguiente no tenga dependencia de alguno de los datos que arrojó, pero puede darse el caso de que sí necesite algún módulo más adelante. Es por ello que los procesos toman como entrada a los datos que se encuentran en la E y no datos que vienen del proceso anterior.

En caso que, en los diagramas para el flujo de datos se presenta una E que no envía datos a los módulos, es señal de que se generó un evento y los datos están inicializados o actualizados por el evento generado anteriormente, el cual ignoramos quien haya sido. Los procesos que presentamos no son todos los que se encuentran en el flujo de procesos, esto es debido a que los procesos faltantes se desarrollan dentro del proceso anterior o posterior de los que presentamos, para mayor descripción consultar la tabla IV.I.

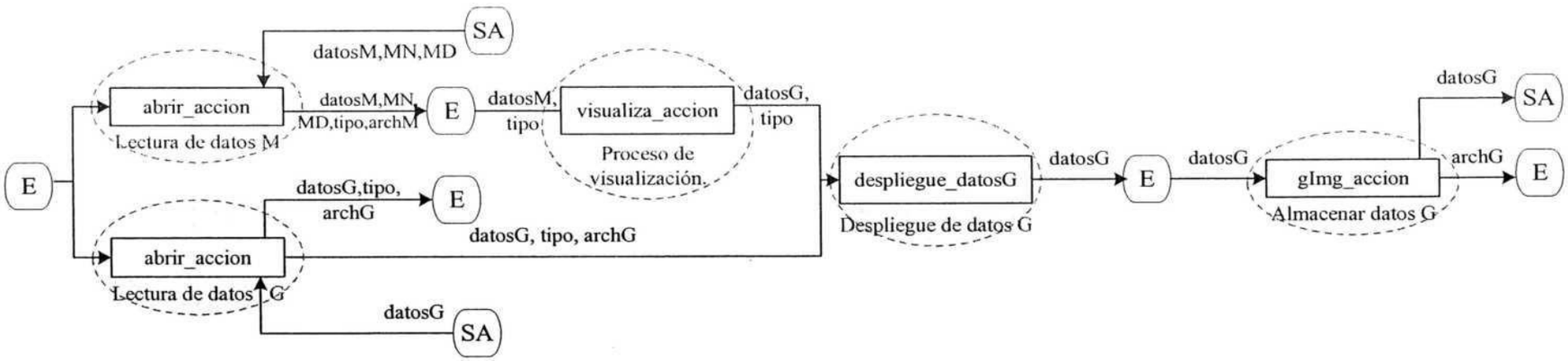


Figura 4.11. Diagrama para el flujo de datos reducidos: **"Visualización de datos M"**

El SA se comunica con un proceso para que tome los datos y para que envíe los datos con el fin de leerlos y almacenarlos respectivamente. Cabe mencionar que la E y el SA no se pueden comunicar entre ellos directamente, ya que no actúan por si mismos, necesitan de los módulos para su comunicación y actualización. Esto es, la E y el SA son las entidades que representamos en el análisis del sistema. Mientras que los elementos de control son los procesos y módulos del diagrama para el flujo de datos. Las interfaces son los elementos a través de los cuales se actualizan los datos para enviarlos o recibirlos desde el sistema. Los datos son representados con sus correspondientes diagramas para el flujo de datos.

Aunque mencionamos anteriormente la existencia de más módulos (en la sección de la estructura del sistema específicamente), solo modelamos a los que tienen como respaldo a un proceso. Y para mayor detalle (explicación) del flujo de datos, en ocasiones se verán repetidos los módulos, incluso en un mismo diagrama.

Los títulos que asignamos a los diagramas para flujo de datos corresponden a los títulos de los modelados de flujo de procesos, por ejemplo, para el modelado de flujo de procesos que presentamos en la figura 3.6 del capítulo III, que lleva como título "Visualización de datos M", corresponde al diagrama para flujo de datos que está en la figura 4.11 y que lleva el mismo título.

En el diagrama para el flujo de datos de la figura 4.11 se encuentran en principio dos procesos que se desarrollan con el mismo módulo *abrir_accion*, para el proceso de *Lectura de datos M* solicita al SA los datos reducidos *datosM* y sus dimensiones *MN* y *MD*, estos datos además del *tipo* de archivo (que determina si se trata de texto o imagen), y del nombre del archivo *archM* que contiene todos estos datos, los actualiza dentro de la E.

Para el *proceso de visualizacion* la E le facilita la matriz de *datosM* y le facilita *tipo* al módulo *visualiza_accion* para que éste arroje a los *datosG* que es una imagen y al *tipo*. Enseguida se llama al módulo para el *despligue de datos G*, el módulo *despligue_datosG* se encarga de enviarle a la E los *datosG*. Si el usuario decide grabar la imagen, entonces el módulo *gImg_accion* del proceso *Almacenar datos G* toma de la E a los *datosG* y los envía al SA, y a la E le indica el nombre del archivo *archG* donde se almacenó la imagen.

Los diagramas para el flujo de datos correspondientes a los procesos de la reducción de dimensión de datos y para el análisis de datos los presentamos en el apéndice B, donde también se muestra una tabla que nos detalla el significado de los datos de todos los diagramas.

IV. 5. Flujo de Eventos

En la vida real, los procesos no tienen la continuidad precisa presentada en los diagramas para el flujo de datos, es entonces que la aplicación ViAn proporciona una mayor flexibilidad al usuario, al ejecutar los procesos deseados a través de la generación de eventos. Los diagramas anteriores representan un escenario para la realización de una tarea específica, y el usuario puede tomarlos de guía para lograr sus objetivos.

El flujo de eventos se representa a través de diagramas que contienen solamente a un tipo de elementos, los eventos. Esto es, en el flujo de eventos no detallamos datos, ni procesos, ni interfaces, solo a los eventos y su comportamiento según su flujo. En dichos diagramas presentamos cómo es que fluyen los eventos en el sistema, y así observar cuáles procesos se permiten realizar y cuáles no, según la generación de un evento. Por lo tanto, los diagramas de flujo de eventos, también permiten conocer qué evento puede generarse en secuencia de otro, además de saber qué datos se manipulan en ese momento.

Dos de los diagramas de flujo de eventos del sistema ViAn se presentan en la figura 4.12. En el diagrama 4.12(a) mostramos el flujo de eventos a partir de la generación del evento llamado *nuevo*, éste permite seguir con los eventos para la reducción de dimensión de datos activándolos, estos son: *kohonenB*, *kohonenM*, *EMB* y *EMM*, y desactiva a los eventos para realizar cualquier otro proceso a excepción de él mismo y al proceso *abrir*. El evento para abrir cualquier archivo es *abrir*, y como no lo activa ni desactiva no se muestra modelado en este diagrama.

En el diagrama 4.12(b) presentamos ahora el flujo de eventos para la generación del evento *abrir*, pero como el flujo de eventos no depende solamente del proceso exitoso que se desarrolla al generar este evento, sino también de los datos que se hayan manipulado, presentaremos el diagrama del caso en que el proceso haya arrojado al dato *archN*. El resto de los casos de haber manipulado datos diferentes, los presentamos en el apéndice B, junto con el resto de los diagramas para los flujos de datos. También presentaremos en la tabla BIII una descripción explícita de los eventos. Siguiendo con la explicación de la figura 4.12(b), el evento (*abrir*) activa a los eventos: kohonenB, kohonenM, EMB y EMM, desactivando al resto de los eventos, con excepción de él mismo y del evento *nuevo*.

Como mostramos en estos dos diagramas, los eventos: *nuevo* y *abrir*, no se activan ni desactivan en ningún momento (en ningún diagrama de flujo de eventos).

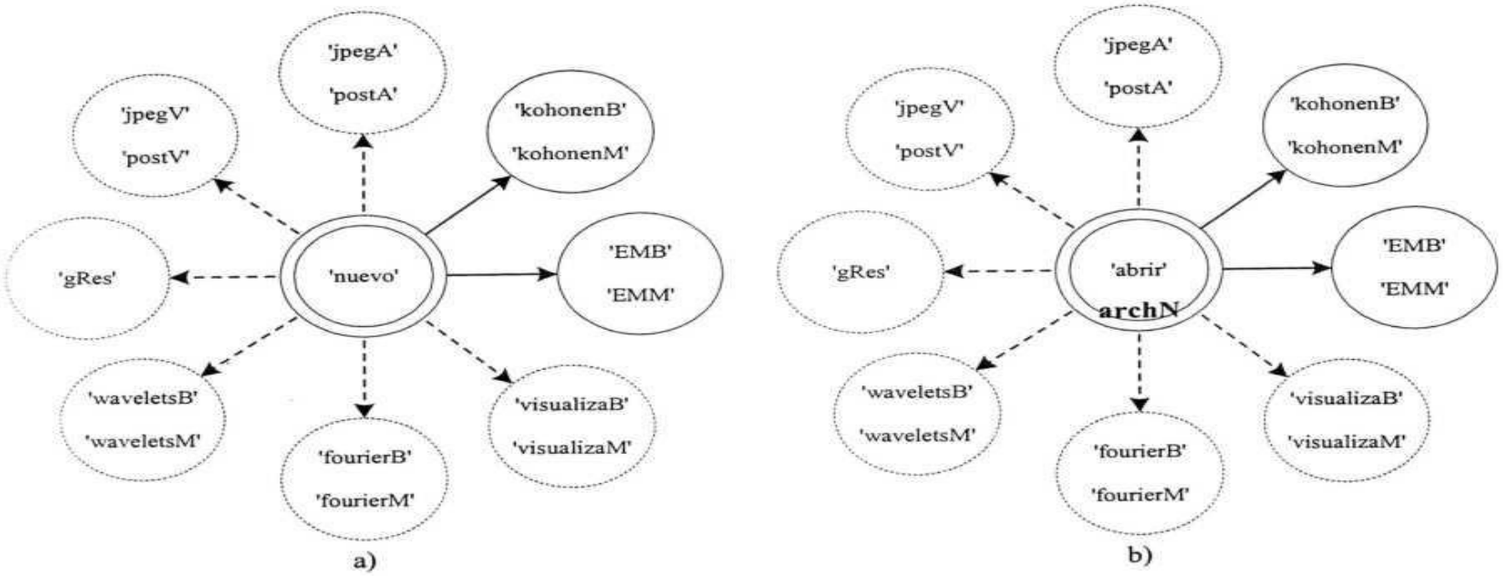


Figura 4.12. Diagramas de flujo de eventos:
(a) Para el evento *nuevo*. (b) Para el evento *abrir* con proceso del dato *archN*.

Modelamos los eventos de utilidad para el diseño del sistema ViAn. En la etapa de implementación nos encontraremos con más eventos que no afectarán las actividades (o tareas) y procesos que han sido representados hasta este momento. Con respecto a los eventos que sí se presentarán como parte de los diagramas de flujo de eventos, y que no generen activaciones o desactivaciones de otros eventos son los siguientes: *gRes* (corresponde al proceso de grabar los resultados), *jpegV*, *postV* (estos dos corresponden al almacenamiento de imágenes visualizadas), *jpegA* y *postA* (y por último, estos dos corresponden al almacenamiento de imágenes analizadas).

Un punto importante que debe hacerse notar es que, si al generarse un evento, el proceso que debe realizarse no se cumple, entonces no procede el flujo de eventos. Por ejemplo, si se genera el evento *nuevo* pero en el proceso al que ha solicitado, no se abre ningún archivo nuevo, entonces se produce un error y el sistema entiende que no se generó ese evento, por lo tanto no se alteró el flujo de eventos.

El sistema ViAn cuenta con una limitación, que a este nivel del desarrollo del diseño será más fácil de visualizar. Primero expliquemos que si se leen del SA a imágenes, éstas se desplegarán en el lugar correspondiente de la interfaz y podrá almacenarse nuevamente si el usuario así lo desea. Pero la limitación a la que me refiero, la detectamos en el caso de la lectura de datos tipo texto, sean el resultado de algún proceso (con excepción de los datos N), estos datos no podrán almacenarse nuevamente, solo podrán ser procesados. Para reconocer estas características en los diagramas de flujo de eventos, consulte a los diagramas que presentamos en el apéndice B.

IV. 6. Secuencia de un evento

En la secuencia de un evento se detalla la unión del flujo de eventos y del manejo de tiempo. A diferencia del flujo de eventos, en la secuencia de un evento, detallamos cómo es su comportamiento a través del tiempo, esto implica la representación de módulos, datos e interfaces. Recordemos que en el flujo de eventos, solo determinamos la activación y desactivación de eventos, que es lo que permite acceder a los procesos determinados.

Además, la secuencia de un evento implica procesos propios del lenguaje de programación que manipulan los eventos, así, el diagrama para la secuencia de un evento nos explica como la filosofía del lenguaje IDL y nuestros procesos se coordinan para lograr un objetivo (es decir, lograr un caso de uso) en el desarrollo del sistema ViAn.

De manera natural, la secuencia de un evento se puede desarrollar con los diagramas de secuencia, estos diagramas son presentados en el análisis del sistema. Debido a que estereotipamos nuestros elementos y así definimos la comunicación entre los procesos del sistema y los del lenguaje con las interfaces y los elementos de entidad. Además, los cambios a través del tiempo están muy bien representados con estos diagramas de secuencia.

En el diagrama de secuencias, para la secuencia de un evento que presentamos en la figura 4.13 modelamos a los elementos de control que existen en los módulos de nuestro sistema. Sólo que en esta etapa del diseño, mostramos a estos módulos como elementos que se encuentran desarrollados para el ambiente de programación que utilizamos. Así es como también se encuentran representados los módulos propios del IDL, como elementos de su ambiente de programación.

Existen fuertes dependencias entre los elementos de control que presentamos en el diagrama de secuencia de un evento: Al momento de ejecutarse uno de los módulos del sistema, los módulos del IDL Widget_control (controlador de los elementos de interfaz) y Xmanager (administrador de los eventos) son llamados desde ese módulo. Si se detecta la ejecución de algún otro módulo del sistema, los módulos del lenguaje mencionados ahora serían llamados por este último módulo del sistema. En la explicación de la secuencia de un evento que presentamos enseguida, lo detallamos.

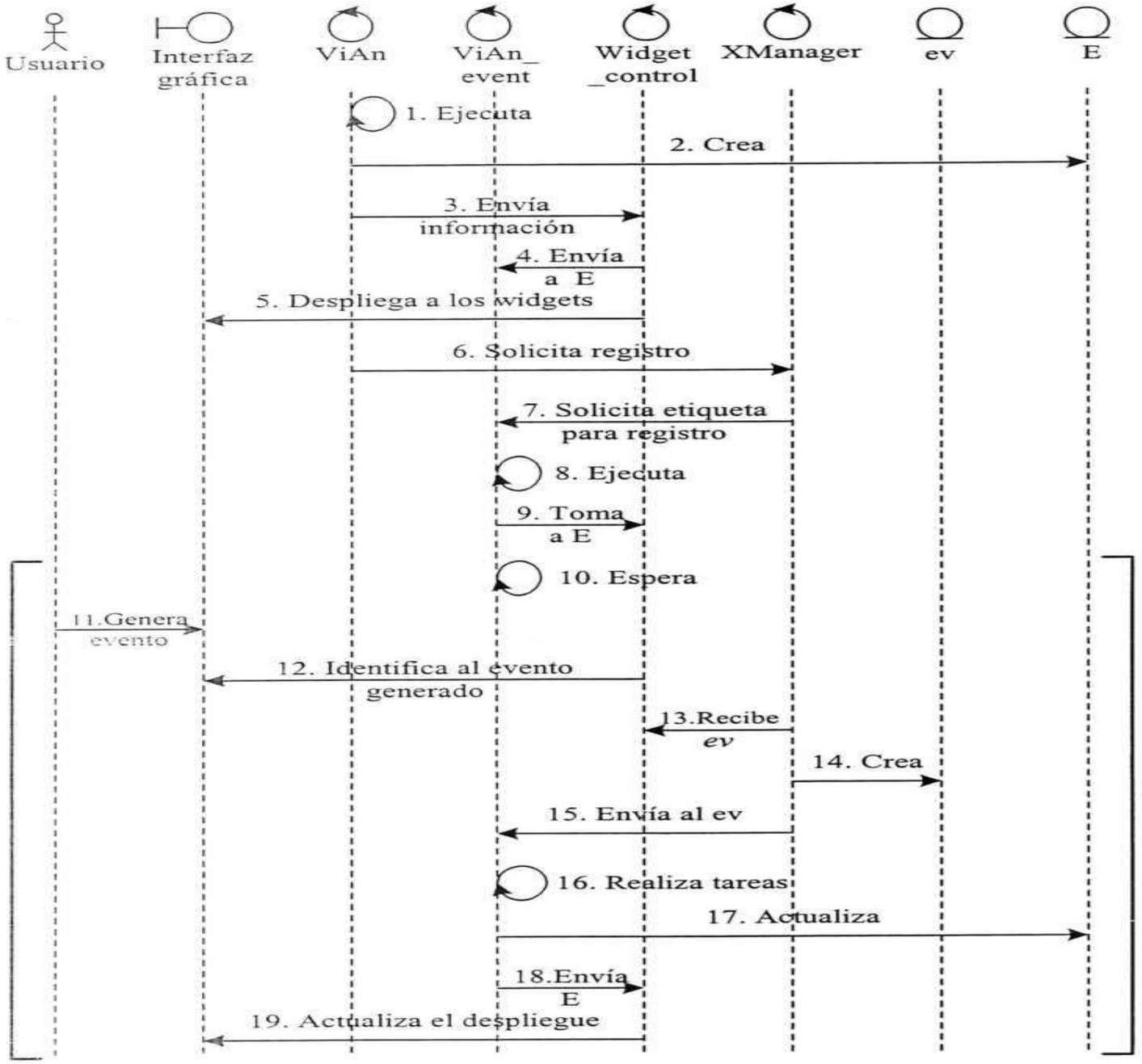


Figura 4.13. Diagrama de Secuencia, para la secuencia de un evento del sistema ViAn

En el diagrama de secuencia de la figura 4.13 presentamos al usuario como el generador de los eventos en el sistema. Presentamos unos corchetes que abarcan una parte del diagrama, estos corchetes indican que existe un ciclo, el cual inicia en el paso 10 y termina en el paso 19. En dicho diagrama, mostramos a dos estructuras de datos: una de ellas es la estructura del sistema (E), contiene todos los datos que se necesitan a lo largo del ciclo de vida de la aplicación ViAn; la otra es ev, esta estructura se genera cada vez que se genera un evento, es la estructura del evento, que contiene los datos del evento generado.

1. Se ejecuta el módulo para la creación de Widgets (elementos de interfaz), el procedimiento ViAn, que contiene la información de los elementos de interfaz gráfica con los cuales el usuario interactuará con el sistema.

2. ViAn crea a la estructura de datos del sistema E, que contiene a todos los datos necesarios para todos los procesos que se utilizarán durante la vida del sistema.

3. ViAn envia la información (atributos y valores) de los Widgets y la estructura de datos E, al módulo establecido del IDL Widget_control (módulo que controla los elementos de interfaz).

4. El módulo controlador de Widgets, Widget_control, se encarga de enviar la E al módulo ViAn_event (módulo que contiene la referencia a los procesos), que en ese momento representa para él solamente una etiqueta, ignora exactamente a que módulo dirigirse, ya que ViAn_event aún no ha sido ejecutado, así es que, solo tiene en puerta la tarea mencionada.

5. El módulo Widget_control despliega en forma de interfaz gráfica los widgets (por ejemplo: botones, etiquetas, cuadros de imagen, cuadros para texto, etc.) y su información (tamaño, color, etc.).

6. El módulo principal ViAn (quien despliega a la interfaz gráfica ya que contiene la información de los widgets), llama al módulo propio de IDL XManager (administrador de eventos y procesos) para registrar la relación que existe entre ViAn y ViAn_event.

7. El administrador XManager (el cual tiene tareas muy rígidas) solicita la etiqueta del módulo ViAn_event (módulo que contiene la referencia a los procesos según el evento que se genere) para relacionarla con ViAn (módulo que despliega la interfaz gráfica), y para saber que a ese módulo es al que se debe ejecutar para buscar los procesos que se deseen realizar a través de los widgets especificados con ViAn.

8. El módulo del sistema ViAn_event se ejecuta por la localización del módulo del IDL XManager.

9. El módulo ViAn_event toma la estructura de datos del sistema ViAn E, que ha sido enviada por el módulo controlador de Widgets Widget_control (paso 4).

10. El módulo del sistema ViAn_event se encuentra en estado de espera de la generación de un evento. Aquí es donde inicia el ciclo para cada evento, se repiten los pasos del 10 al 19, hasta que el evento generado sea la salida del sistema, o que bien, la vida del sistema llegue a su fin.

11. El usuario genera un evento a través de la interfaz creada por ViAn (presionando un botón por ejemplo, o seleccionando una opción del menú principal).

12. El controlador de Widgets, Widget_control, identifica al evento generado y al widget (junto con sus atributos y valores) a través del cual fue generado.

13. El módulo administrador XManager obtiene la información del evento generado del controlador de Widgets Widget_control.

14. El módulo administrador de eventos XManager, crea la estructura del evento ev, con la información del evento generado.

15. El módulo del IDL XManager envía la estructura del evento ev al módulo del sistema ViAn_event para localizar al proceso que se debe ejecutar según la correspondencia con el evento generado.

16. El módulo ViAn_event realiza las tareas necesarias para cumplir con el proceso deseado, ejecutando módulos que se encuentran como referencia en el módulo del sistema ViAn_event.

17. El módulo ViAn_event actualiza la estructura de datos del sistema ViAn, E, con los resultados de los procesos realizados.

18. El módulo del sistema ViAn_event envía la estructura de datos del sistema E, al módulo del IDL Widget_control, para que actualice a los Widgets en la interfaz gráfica.

19. El módulo del IDL controlador de Widgets Widget_control, actualiza la interfaz gráfica, sus widgets junto con sus atributos y valores, éstos los contiene el módulo del sistema ViAn.

IV. 7. Diseño de interfaz

Para completar el diseño del sistema ViAn, presentamos las ventanas y elementos que visualizará el usuario para interactuar con la aplicación. Las ventanas contienen los elementos necesarios que el usuario observará en su despliegue, estos elementos son botones, espacios para texto y para imágenes. Estas ventanas son las que forman al diseño de interfaz y cubren los requerimientos presentados en el análisis del sistema. Además, estas ventanas le darán forma a los casos de uso descritos para el sistema ViAn.

El diseño de interfaz modela así las pantallas a través de las cuales el usuario va a interactuar con el sistema ViAn, y cómo es que tendrá acceso a los procesos. Se explicará la funcionalidad de las pantallas y justificaremos sus formas basándonos en los requerimientos de interfaz que presentamos en el análisis del sistema, y en los casos de uso. Parte del diseño de la interfaz lo mostramos en el apéndice B.

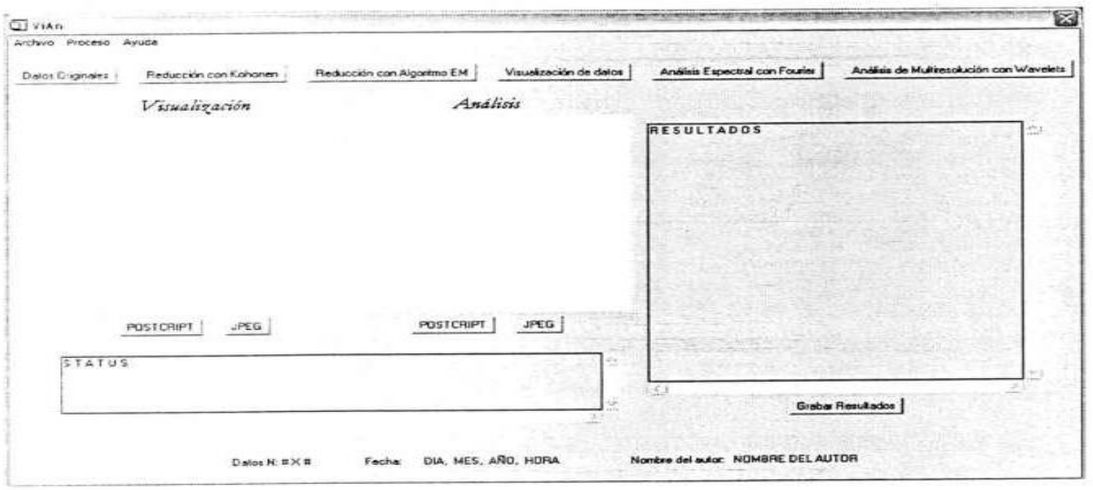


Figura 4.14. Ventana principal del sistema ViAn

Realizamos un prototipo en el IDL, de los elementos requeridos. Este prototipo lo presentamos como la ventana principal del sistema ViAn, la mostramos en la figura 4.14, esta ventana está lista para empezar a interactuar con el software. En el pre-diseño de interfaz, que lo desarrollamos con el nombre de "requerimientos de interfaz" (capítulo III), observamos que los botones que se requieren (ver figura 3.10) los diseñamos de una forma más específica en esta nueva versión del diseño de interfaz.

Esto es, para la opción de *entrada* en los requerimientos de interfaz, le corresponde una opción del menú superior (de *Archivos*); para *reducción*, les corresponden dos botones, *Reducción con algoritmo Kohonen* y *Reducción con algoritmo EM*; para *visualización* le corresponde el botón de *Visualización de datos*, y para el análisis, los botones *Análisis espectral con Fourier* y *Análisis de multirresolución con Wavelets*.

El botón *Datos Originales* del diseño de interfaz, no se especifica en los requerimientos de interfaz. Este aspecto lo detallamos más adelante en esta misma sección. El espacio para la imagen que presenta el requerimiento de interfaz, lo diseñamos en dos espacios para las imágenes resultantes de la *Visualización* y el *Análisis*, además se agregaron botones para poder almacenar las imágenes en dos diferentes formatos cada una: *postcript* y *jpeg*. El área de *matriz resultante* le corresponde a *Resultados*, junto con el botón *Grabar Resultados* y el espacio para el *Estatus* corresponde al *Status*.

El menú superior cambió un poco en esta etapa, se mantuvo *Archivo*, la opción de *Manual* la encontraremos como una opción de la *Ayuda* en el diseño de interfaz, la *Salida* se introdujo en la lista de opciones del *Archivo* y se agregó *Procesos* al diseño de interfaz. Esta última opción contiene los procesos para la reducción de dimensión de datos y para el análisis de datos, los botones que corresponden a estos procesos mencionados, también los encontraremos como opciones dentro de la lista de opciones de *Procesos*.

Cada uno de los botones que se especifican en las ventanas, representan un proceso que realizar, y son el medio por el cuál el usuario genera un evento (presionando el botón). También puede generar un evento, seleccionando del menú superior alguna opción. Los procesos se encargan de desplegar en las áreas de imágenes, datos que se encuentran en el elemento entidad E (la estructura de datos de ViAn), resultantes de los procesos que generan datos bidimensionales en formato de imagen. El mismo suceso ocurre para las áreas de texto, por ejemplo, los procesos envían a desplegar datos resultantes de procesos en la región correspondiente a los resultados, e indican el estado del sistema en la región de status.

Los botones que acceden a los procesos para almacenar datos en el sistema de archivo (SA), se encuentran bajo las áreas en que los datos resultantes de algún proceso son desplegados. Al presionar alguno de estos botones y generarse eventos, el proceso correspondiente solicita los datos indicados de la E, y los envía al SA. De igual forma, si se desea leer datos del SA, el usuario genera el evento indicado, que se encuentra en el menú superior, y el proceso correspondiente solicitará el archivo del SA y lo enviará directamente a la E, la cual será actualizada, para su próximo proceso, según el evento generado.

Hemos diseñado la ventana para el proceso correspondiente a la entrada de datos N al sistema. La ventana que presentamos en la figura 4.15 modela una segunda ventana dependiente de la ventana principal presentada en la figura 4.14. Para que esta ventana sea creada, el usuario debe elegir la opción del menú superior *Archivo*, y seleccionar la opción que corresponda a la introducción de datos nuevos al sistema.

Así, el usuario genera el evento para solicitar las dimensiones de la matriz de datos N y es entonces que se crea la ventana para solicitar los datos originales en forma matricial. El área que solicita la dimensión de los datos N no la presentamos en el diseño de interfaz. La ventana de la figura 4.15 se basa en la pantalla 3.11 del pre-diseño de interfaz (requerimientos de interfaz).

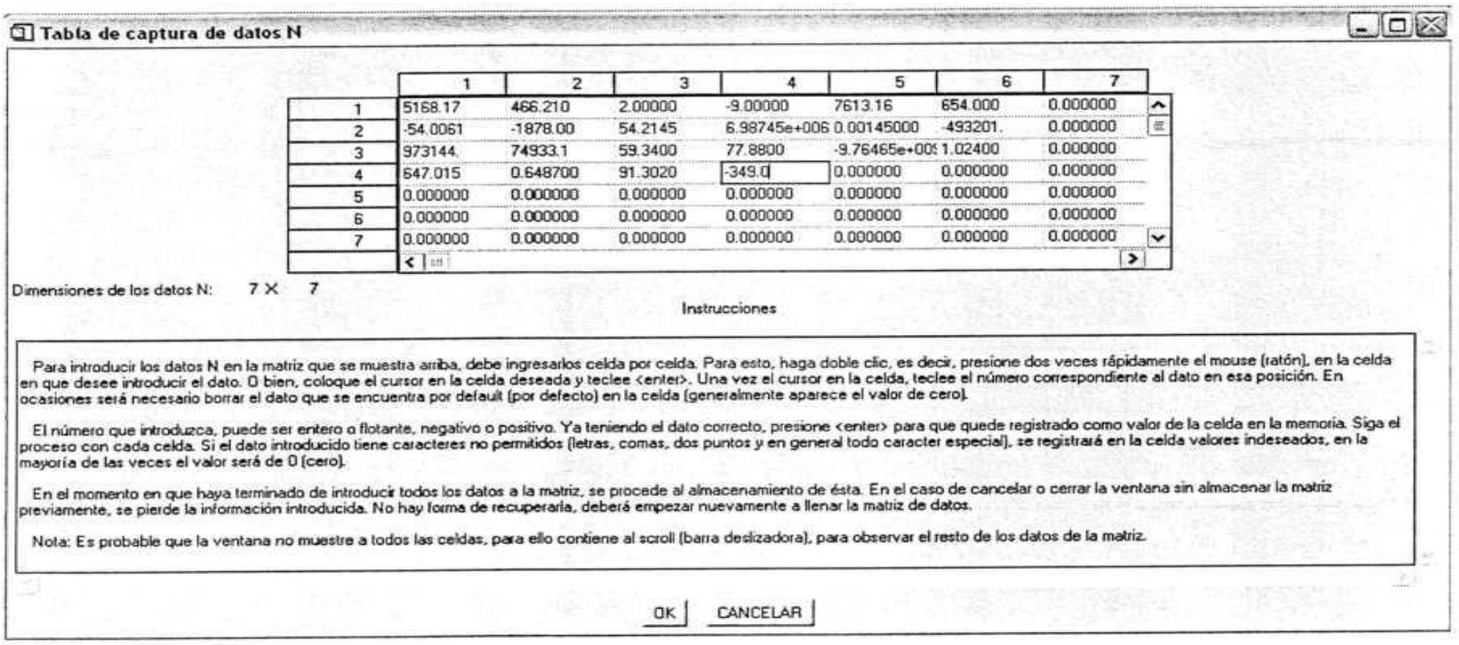


Figura 4.15. Ventana para la entrada de datos originales

Para el proceso de la entrada de datos, el usuario puede leer del SA, los datos N previamente almacenados. Generando el evento al elegir la opción para abrir datos N, de la opción de *archivos* del menú superior, se despliega una ventana para la solicitud del archivo. Esta ventana no se presenta en el diseño de interfaz.

En la figura 4.16 mostramos el diseño de la ventana para la modificación de los datos originales, los cuales también pueden ser almacenados al SA desde esta misma ventana. Se pueden almacenar con un nombre diferente, o sobrescribirlos. Esta ventana depende de la ventana principal, y es creada después de generar el evento a través del botón *Datos Originales*. El diseño y desarrollo de este proceso, son requerimientos innovadores para el sistema ViAn, ya que el requerimiento especificado, es la introducción de datos originales y su almacén (consultar los requerimientos del sistema, en el capítulo III). Las actividades de acceso a los datos N y actualización de los datos N, los reconocemos como requerimientos innovadores.

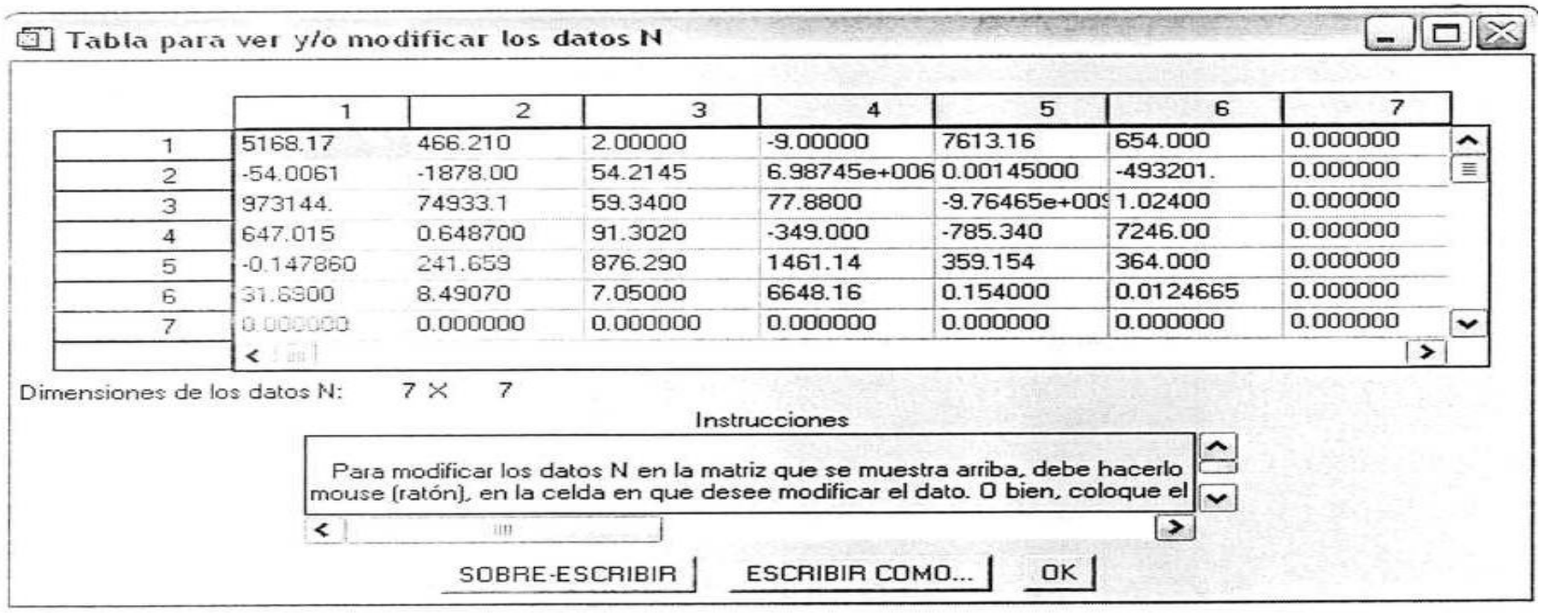


Figura 4.16. Ventana para la modificación de los datos originales

En el apéndice B, presentamos el resto de las ventanas que complementan esta parte del diseño de interfaz. En ellas, detallamos las ventanas que corresponden a los procesos de la reducción de dimensión de datos (con cada una de las dos técnicas), al proceso de visualización de datos reducidos, a los procesos para el análisis de los datos (con cada una de las dos técnicas), y para los procesos de la reducción o análisis en forma comparativa.

Finalizamos el diseño del sistema con el diseño de interfaz. Estamos ahora en posición de desarrollar el análisis de los algoritmos. También hemos alcanzado un nivel de desarrollo, en el que nos concentraremos en la implementación de la aplicación.

CAPITULO V.

Análisis y Diseño de los Algoritmos

Toda nuestra ciencia, comparada con la realidad, es primitiva e infantil... y sin embargo es lo más preciado que tenemos.

Albert Einstein

CAPITULO V.

Análisis y diseño de los algoritmos

V. 1. Introducción

En los capítulos III y IV presentamos al análisis y al diseño (respectivamente) de la aplicación como un sistema computacional, donde se analiza y diseña los módulos y funcionalidades, además de que se cubren los requerimientos del sistema. Para tener un análisis y un diseño completo presentamos este capítulo, que es parte de los capítulos III y IV, correspondiente al análisis y diseño de los algoritmos.

Al decir algoritmos, me refiero que esta parte del análisis y diseño se conforma de cuatro técnicas, dos para la reducción de dimensión y dos para el análisis de los datos. Para tener una mejor visión del análisis y diseño de los algoritmos, presentamos la figura 5.1. En ésta figura mostramos al análisis de sistema con un óvalo, mismo que al diseño del sistema, y hay una parte donde ambos óvalos se unen. En esa unión se forma el análisis y diseño de los algoritmos, siendo la parte integral del desarrollo del sistema.

Figura 5.1. Análisis y diseño de los algoritmos

Presentamos las bases de las técnicas para obtener una base del conocimiento y comprensión de los algoritmos que utilizaremos. Más adelante, desarrollaremos los algoritmos con el objetivo de estudiarlos en forma detallada. Cada técnica es diferente, por lo tanto su desarrollo y estructura lo son, de acuerdo al contexto. Es por esto que no se seguirá un mismo formato para describir cada técnica o algoritmo. Además, por esta misma razón, en algunas técnicas desarrollaremos el algoritmo y su pseudocódigo, para proseguir con la implementación y pruebas de la técnica. Enseguida detallamos entonces, 1. los algoritmos Kohonen y EM, y 2. las transformadas de Fourier y de Wavelets. En las siguientes secciones presentaremos el desarrollo de éstos.

V. 1. 1. Técnicas de reducción de dimensión

Las técnicas de reducción de dimensión se especificaron a través del requerimiento de usuario para el proceso. 1c. Una técnica se basa en el concepto de las Redes Neuronales, y la otra en el modelo de Variables Ocultas (o Latentes), también considerada una Red Neuronal Estadística. La primera técnica se trata del algoritmo de Kohonen y la segunda es el algoritmo de la Maximización del valor Esperado (EM). La reducción de dimensión de datos, es un método que se utiliza para manipular datos que se encuentran en dimensiones muy altas. Por lo que son datos que no se pueden graficar para la visualización a través del ojo humano, además de que es muy complicado y tedioso hacer cálculos con ellos.

El trabajo de la reducción de dimensión de datos lo presentamos a través de los algoritmos que mencionamos, en una transformación no lineal y no explícita, donde su solución son algoritmos complejos y en ocasiones denominados como inteligentes. Esta transformación la hemos llamado **F(x)**, para su mejor comprensión consulte el capítulo II. Sin embargo, esto es simbólico, ya que no se puede identificar explícitamente la función **F(x)**. Además de ser una transformación no lineal, se requiere conservar la topología de los datos. Estas características las cubren los siguientes algoritmos.

V. 1.1.1. Algoritmo de Kohonen para redes neuronales

Kohonen es un algoritmo basado en las Redes Neuronales Artificiales (RNA), con el cual se realizan mapas de auto-organización y se caracteriza por mantener la topología de los datos. El algoritmo de Kohonen tiene muchas aplicaciones en diferentes áreas computacionales y científicas. En el sistema ViAn, adaptamos al algoritmo de Kohonen como una técnica para realizar nuestra propuesta de la reducción de dimensión de datos, y así lograr nuestro objetivo de la visualización de datos científica.

V. 1.1.2. Algoritmo de EM para variables latentes

Otra alternativa para lograr la VDC con un enfoque diferente de nuestra transformación **F(x)**, aplicando la misma técnica de reducción de dimensiones de datos, es un algoritmo que se basa en el modelo de Variables Latentes u Ocultas. Este algoritmo es el de la Maximización del valor Esperado (EM por (*Expectation Maximization*), del área de estadística y probabilidad. El algoritmo EM es muy complicado, y para la utilidad que en ViAn le estamos dando, abarca muchas características dignas de un extenso estudio y explicación amplia. Además esta diseñado a través de la determinación de los parámetros de una función de probabilidad llamada función de máxima verosimilitud (*likelihood*).

V. 1. 2. Técnicas para el análisis de datos

Las técnicas para el análisis de datos se especificaron a través del requerimiento de usuario para el proceso, **1d**. Una de estas dos técnicas, realiza el análisis espectral de datos, y la otra el análisis de multiresolución. La primera técnica se trata de la Transformada de Fourier y la segunda técnica es la Transformada Wavelets (ondeletas) con el método de Burt-Adelson.

El análisis de los datos, nos permite identificar cambios o variantes en los datos, con respecto a distintos factores específicos del fenómeno o experimento a estudiar. Los métodos más utilizados por todo científico e ingeniero (y en general cualquier investigador) para el análisis de datos, son las gráficas, donde observan el comportamiento de frecuencias, amplitudes, etc. a través de tiempos, datos, etc.

Presentamos estas técnicas a un nivel básico del conocimiento, y para su justificación dentro de la aplicación. En la primera versión de ViAn, la transformada de Fourier se encuentra como un módulo del lenguaje IDL, el cual es llamado para su ejecución directamente. Y encontramos referencia de lo que se encuentra implementado en el lenguaje IDL, acerca de la transformada Wavelets. Ambas trasformadas están integradas en el análisis y diseño del sistema ViAn.

V. 1.2.1. Análisis espectral con la transformada de Fourier

La transformada de Fourier es de los métodos más utilizados para el análisis de datos. La transformada de Fourier analiza una señal dentro de diferentes frecuencias, es decir, esta técnica determina la frecuencia espectral de señales digitales. El análisis espectral presenta una gráfica de: los datos por sus frecuencias. Para realizar este análisis de datos, utilizaremos al algoritmo computacional que se desarrolló para la Transformada de Fourier Discreta (DFT, *Discrete Fourier Transform*), este se conoce como FFT (*Fast Fourier Transform*), la Rápida Transformada de Fourier. **[Sitio 1]**

V. 1.2.2. Análisis de multirresolución con la transformada Wavelets

La transformada de Wavelets es la herramienta para el análisis de datos bidimensionales más popular, la cual representa el tiempo-escala de los datos. Específicamente la transformada de Wavelets es Continua (CWT, por sus siglas en inglés). La transformada Wavelets permite representar los datos en distintos niveles de resolución, lo que hace posible manipularlos de manera más eficiente. La transformada Wavelet consta de la técnica de multirresolución de patrones y la pirámide Laplaciana. Para una mayor explicación, consulte a: **[Rao, 1998]**.

V. 2. Algoritmo de Kohonen

Presentamos en esta sección al primer algoritmo que se propone para el problema de la visualización de datos, a través de la técnica de reducción de dimensión de datos. Basado en las redes neuronales artificiales, es el algoritmo de Kohonen. Ésta es una red neuronal de aprendizaje.

V. 2. 1. Introducción

El Mapa de Auto-Organización (SOM, por sus siglas en inglés: *The Self-Organizing Map*), es una técnica de Visualización de datos creada por el Profesor Teuvo Kohonen, la cual interpreta grandes cantidades de datos que se encuentran en un espacio de altas dimensiones y este mapa es resuelto por el llamado "algoritmo de Kohonen" **[Fausett, 1994]**.

Recordemos que el problema que la Visualización de datos procura resolver es la forma de que el ser humano distinga la información que se encuentra en dimensiones fuera de su alcance (más de tres), logrando su interpretación en dimensiones con las que se puede trabajar. Y para esto, SOM nos presenta los datos en dos dimensiones, es decir, en imágenes planas.

El algoritmo de Kohonen se basa en el concepto de Redes Neuronales Artificiales (RNA), para reducir los datos y lograr el objetivo de la Visualización. El algoritmo es una red de aprendizaje sin supervisión, esto es, se desea encontrar la estructura natural inherente en los datos de entrada de la información, los datos de entrada son no binarios. Los resultados no son definidos de manera a priori, es decir, que no se puede predecir los datos de salida de la RNA estadísticamente.

El SOM se organiza a sí mismo a través de la competencia entre las neuronas de la red para ganar la representación de los datos de entrada. Las neuronas se actualizan cambiando sus pesos para lograr mayor similitud con la entrada más reciente, la cual ya se le asignó su representante (neurona ganadora). Para actualizar los pesos, se crea un vector cuantitativo que ajusta los pesos en nodos de entrada en común para una matriz **W** (matriz de pesos).

La forma en que el algoritmo de Kohonen reduce las dimensiones es produciendo un mapa en dos dimensiones generalmente (como se mencionaba) que tracen (o mapeen como se dice comúnmente) las semejanzas de los datos agrupando objetos de características similares, y así conservando su topología. De esta manera los SOM's logran dos cosas, reducen dimensiones y visualizan semejanzas o clases.

V. 2. 2. El comportamiento de un SOM

En esta sección presentamos un ejemplo del comportamiento de un SOM para comprender de una forma práctica al algoritmo de Kohonen y sus mapas de auto-organización. En general, el ejemplo trata acerca de cómo los colores que nosotros vayamos metiendo en la red neuronal, se van acomodando en un lugar de dicha red formando así grupos uniformes de colores en ciertas zonas del mapa.

El problema que planteamos para nuestro ejemplo, es la clasificación de colores dentro de un espacio bidimensional. Nosotros contamos con las características que definen a cada fenómeno al que llamamos *color*. Estas características se encuentran en un espacio de datos, y son elementos de un vector. Cada elemento de un vector, representa un color. Así lo presentamos en la figura 5.2. El vector de pesos que en dicha figura mostramos, contiene varios elementos (colores), y están preparados para ser introducidos a la red neuronal que clasificará los colores en forma topológica.

Figura 5.2. Vector de pesos, un vector de la matriz **W**

Cada elemento (color) que se encuentra en el vector de pesos, es representado por un vector de tres dimensiones de datos (R*ed*: rojo, G*reen*: verde y B*lue*: azul), este ejemplo es para facilitar la analogía de los fenómenos que pueden presentarse en forma natural. Supongamos que cada dimensión (*red*, *blue* o *green*) debe tener una cantidad específica para que cuente como característica del color, y sea proyectando simplemente el color a la vista humana para la interpretación del fenómeno (que denominamos color). Cabe mencionar, que nuestra forma de representar el color no es una limitante, el color puede estar representado por más de 3 dimensiones, podemos agregar la intensidad y brillo a cada elemento por ejemplo.

El vector de pesos contiene dos componentes, donde el primero es precisamente la muestra de datos (el color), y el segundo componente es la posición en la que se encuentra el color en una matriz de datos. Esta matriz de datos la denominamos **W**, que es un conjunto de los vectores de pesos que mostramos en la figura 5.2. Esto significa entonces, que nuestra matriz de pesos es **W**, la cual es el conjunto de datos de entrada a la red neuronal. Nuestra matriz **W** se vería como un arreglo de dos dimensiones como el que presentamos en la Figura 5.3. Definimos que cada color es un peso y cada peso es un arreglo N-dimensional (en nuestro ejemplo con N=3) y el peso tiene una posición única en el conjunto de Colores que mostramos en la figura mencionada.

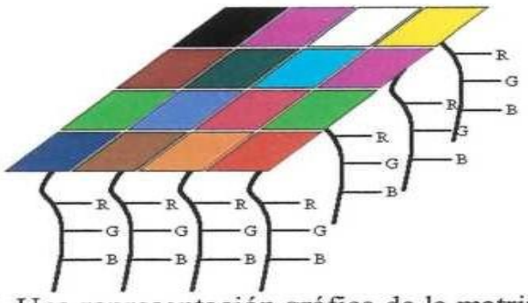


Figura 5.3. Una representación gráfica de la matriz de pesos **W**

Para darnos una idea de cuáles colores son los que podemos conocer a través de su vector de datos tridimensional, en la Tabla V.I, presentamos algunos colores, basándonos en el estándar RGB de los vectores para determinar colores, y el nombre del color que se representa.

Color	Vector		
	R = *red*	**G** = *green*	**B** = *blue*
Hueso	250	235	215
Café	165	042	042
Madera	222	184	135
Chocolate	210	105	030
Coral	255	127	080
Dorado	184	134	011
Caqui oscuro	189	183	107
Anaranjado	255	140	000
Salmón	233	150	122
...	...	...	...

Tabla V.I. Los vectores RGB de algunos colores

El algoritmo de Kohonen, lo que hace es, identificar como entrada de datos la matriz de pesos **W** e iniciar el proceso de la competencia por los datos (colores) entre las neuronas de la red. Esto con el fin de clasificarlas en el mapa de auto-organización. Ya que termina el proceso, se actualiza la **W**, y vuelve a generar nuevos colores para la entrada de datos a la red, y volver a competir y clasificar. En la siguiente sección, explicaremos a más detalle como funciona esta competencia entre las neuronas.

Poco a poco, los colores van clasificándose según sus características, hasta formar al SOM que presentamos en la figura 5.4 como el resultado de nuestro ejemplo planteado. En este SOM, mostramos que todos los colores semejantes se agrupan y forman una especie de color predominante.

Figura 5.4. Imagen que representa un SOM, para la clasificación de colores

V. 2. 3. Arquitectura del algoritmo

Después de presentar un ejemplo del comportamiento de un SOM, estructuremos su arquitectura, esto significa que detallaremos cómo es que la red neuronal trata a los datos que va adquiriendo como entrada y cómo es la forma en que se conserva la topología de los datos.

Para este algoritmo los datos que se introducen se representan en forma de nodos de entrada, como se muestra en la figura 5.5, un proceso donde se agrupan en una segunda capa de nodos según el promedio de las características semejantes de los nodos de entrada, donde el nodo "representante" mantiene las propiedades principales de todos los datos a quienes representa; el número de capas puede variar, y la última capa se considera la salida de la (ya formada) *Red* de información, de donde se obtienen los datos reducidos, listos para su visualización y también para su análisis.

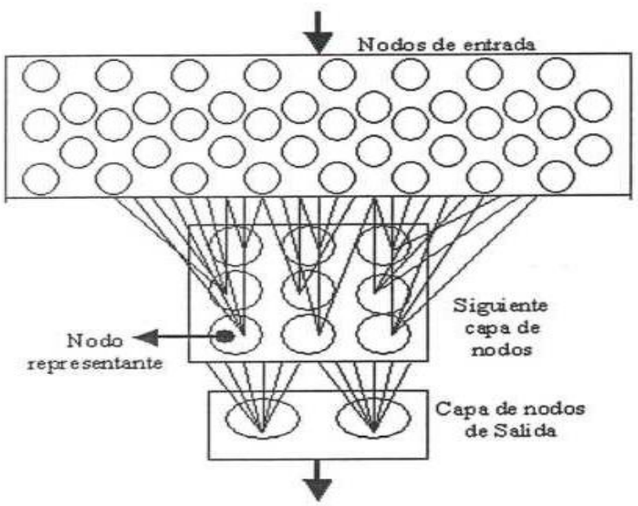


Figura 5.5. Red de información que reduce datos a través de nodos "representantes"

Auxiliándonos con el ejemplo presentado en la sección anterior, donde un vector se forma por tres dimensiones: R, G y B, estudiemos la topología de los datos que se van acomodando en el mapa. La figura 5.6 muestra la topología del algoritmo de Kohonen, donde se muestra un vector de pesos como entrada de datos, y cada elemento (nodo) del vector se conecta con todos los elementos (neuronas) de la red, con el fin de agruparse con el elemento de mayor similitud. Los datos de salida, que son las neuronas de la red, se van modificando al incremento del tiempo (o simplemente al incremento de iteraciones), con la estrategia competitiva para la actualización de sus pesos.

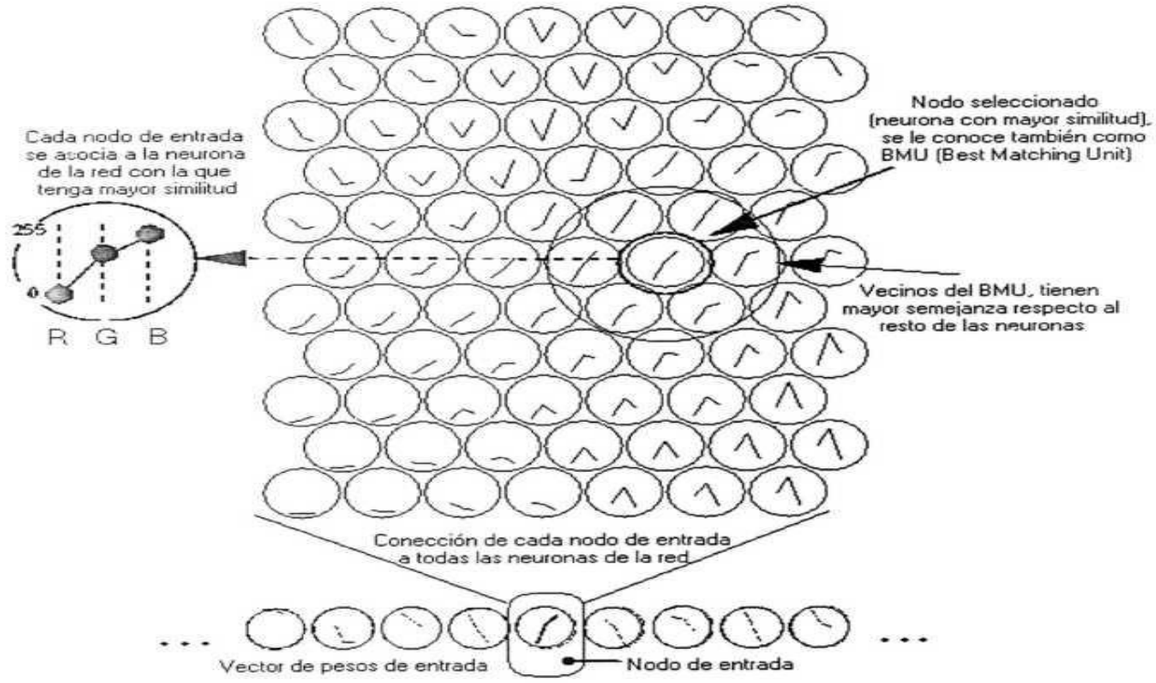


Figura 5.6. Topología del algoritmo de Kohonen

Cada vector del color, como los elementos de la Tabla V.I, se representa en forma de nodo, como se presenta en la figura 5.6, en el lado izquierdo de la figura, los valores de cada dimensión en los vectores van desde 0 hasta 255, en el nodo, cada dimensión se representa por una línea (imaginaria) vertical, donde los mismos valores se representan con un punto que determina la forma de la línea que se interpreta como el Color.

La Unidad con Mayor Similitud, es decir, el BMU (por sus siglas en inglés *Best Matching Unit*), es el nodo seleccionado como el ganador de la red neuronal para representar al nodo de entrada, y las neuronas que tienen mayor similitud al BMU, se encuentran como sus vecinas. Se presenta en la Figura 5.6 la ubicación del BMU y de sus vecinos en el SOM.

V. 2. 4. El algoritmo de Kohonen

Hemos explicado las bases para el entendimiento del algoritmote Kohonen, ahora estamos en posición de presentar los pasos de este algoritmo. Para el algoritmo desarrollado, se consideran lo siguientes términos, seguido de sus pasos:

Wij = Vector de pesos
i = índice de los nodos de entrada para **W** y **X**
j = índice de los nodos de salida para **W**
Xi = Vector de entrada (muestras)
rk = Distancia entre los vectores **X** y **W**
k = Cada uno de los elementos de **Wij**; total de neuronas de la red; índice para **r**.
BMU = Neurona (nodo de **W**) seleccionada como ganadora.
C(t) = Conjunto de nodos (vecinos) con mayor similitud con **BMU**.
a = Término de ganancia (*Gain term*), se considera el radio (o la razón) de aprendizaje
N = Número de dimensiones del vector **X**.
t = Tiempo o iteración.

1. Inicialización
   - **Wij**, se asignan valores pequeños de forma aleatoria.
   - Se determina el radio **a** inicial de aprendizaje, donde 0< **a** <1.
   - Se determina la topología del vecindario **C(t)**
2. Ciclo que repite los pasos del 3 al 8.
   3. Para cada vector de entrada **X** se repiten los pasos del 4 al 6.
      4. Se calculan las distancias **rj** de todos los nodos de entrada **X** con los de salida **Wij**, (suma del cuadrado de la distancia euclidiana de **X** y **Wij**):

         Para cada **k**, calcular: $r_j = \sum_{i=0}^{N} \left(X_i(t) - W_{ij}(t) \right)^2$

      5. Se selecciona el nodo que se encuentra en **Wij** en el momento en que adquiere **rj** el valor mínimo y se le asigna el nombre de **BMU**.
      6. Se actualizan los pesos del vector **W** que pertenecen a **C(t)** como sigue:

         Wij(t+1) = Wij(t) + a(t)[Xi(t) - Wij(t)]

   7. Se decrementa **a** en una pequeña cantidad, y el vecindario se reduce en el tiempo específico. El tiempo **t** aumenta en uno.
   8. Condición de paro: Hasta que el paso 2 se repita por lo menos 1000 veces o **W(t-1)** y **W** difieran por un épsilon determinado.
9. Fin del algoritmo

V. 3. Algoritmo EM

Presentamos el segundo algoritmo en esta sección, que proponemos como complemento para el problema de la visualización de datos, a través de la técnica de reducción de dimensión de datos. Basado en el modelo de variables latentes, es el algoritmo EM. Éste también es una red neuronal, basada en principios estadísticos.

V. 3. 1. Introducción

Para el problema de reducción de dimensiones de datos, nos encontramos con un modelo apropiado para su solución: El modelo de las variables latentes u ocultas. El modelo de variables latentes correlaciona puntos entre dos espacios, uno de ellos es el de las variables latentes **L**. **L** traza o mapea a través de una transformación de tipo no lineal[1], para nuestro interés, al conjunto de datos del segundo espacio al que nos referimos, llamémosle **D**. Dicha transformación obedece a un grupo de parámetros **W**, y al espacio de las variables latentes. Normalmente el espacio **L**<**D**, donde **D** es el espacio de datos, generalmente multidimensional. En la figura 5.7 presentamos al modelo de las variables latentes, el conjunto de puntos (izquierda) que tienen como dimensión (X_1,X_2) corresponde a las variables latentes, y a éstos se les aplica la transformada y(x,W) hacia el espacio de datos (derecha), sus dimensiones: (t_1,t_2,t_3).

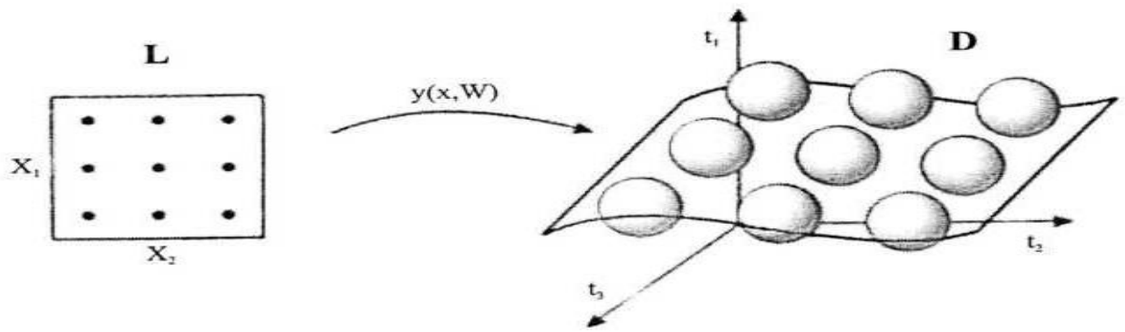


Figura 5.7. Modelo de las Variables Latentes

El objetivo del modelo de variables latentes es encontrar la representación de la densidad de probabilidad del conjunto de datos que se encuentran en el espacio **D** en términos de pequeños números de variables latentes o escondidas. Un modelo específico de variables latentes con transformación no lineal, es el mapeo topográfico generativo (llamado GTM por sus siglas en inglés: "*Generative Topographic Mapping*"), en el cual, los parámetros del modelo se determinan utilizando el algoritmo EM, "de la Maximización del valor Esperado" (*Expectation-Maximization*). **[Markus, 1998]**

1 El modelo se basa en una transformación, ya sea lineal o no lineal entre el espacio latente y el espacio de datos. Un modelo que se basa en una transformación lineal es el factor de análisis.

Para el desarrollo y explicación del algoritmo EM, nos basaremos entonces en el modelo GTM. El algoritmo EM produce probabilidades para máximas verosimilitudes ML por sus siglas en inglés "*Maximum-Likelihood*", para estimar parámetros con los que se realizan trazos o mapeos de muchos datos a un dato, estos parámetros pueden ser **W** y β que presentaremos adelante.

El algoritmo EM consta de dos pasos después de la inicialización de los datos, éstos son:

1. el paso del valor esperado, *Expectation* (**E**): El valor esperado es con respecto a un conjunto de variables ocultas utilizando la estimación de parámetros, y condiciones específicas (por ejemplo la distribución de los centros que también se presentarán adelante); Y,
2. el paso de la Maximización, *Maximization* (**M**): Este paso provee nuevas estimaciones de los parámetros;

Los pasos **E** y **M** se repiten hasta que se alcanza un criterio conveniente de la convergencia. En la práctica, el algoritmo converge después de un número relativamente pequeño de iteraciones.

Una aplicación importante de nuestro interés es la *visualización de los datos científicos*. Ya que los datos sean reducidos por este modelo, entonces los datos resultantes se visualizan con la ventaja de contar con la propiedad de la topología en los datos.

V. 3. 2. El modelo GTM

Como mencionábamos, GTM define un mapeo paramétrico NO lineal y(x, W) de un espacio de variables latentes L-dimensional a un espacio de datos D-dimensional. y(x, W) es continuo y diferenciable, traza (mapea) cada punto en el espacio latente a un punto en el espacio de datos.

Si definimos la distribución de probabilidad correspondiente al espacio de datos. Considerando una distribución gaussiana, la probabilidad quedaría como sigue:

$$p(t \mid x, W, \beta) = \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\sum_{d}^{D}(t_d - y_d(x,W))^2\right)} \qquad [1]$$

donde t es un punto en el espacio de datos y β^{-1} corresponde a la varianza, para una mejor definición consulte en el apéndice C, la sección 2.1. El modelo GTM se basa en estimar β, y W que haga [1], la función más probable con el mínimo de error.

Para integrar las variables latentes, obtenemos la distribución de probabilidad en el espacio de datos expresada como una función con los parámetros β y W, esta función está basada en la "Fórmula General del Modelo de Variables Latentes":

$$p(t \mid W, \beta) = \int p(t \mid x, W, \beta) p(x) dx \qquad [2]$$

La integral que mostramos no es analíticamente viable. Aún así, escogeremos p(x) para obtener una forma en particular, un conjunto de K funciones delta de pesos equivalentes en una matriz regular:

$$p(x) = \frac{1}{K} \sum_{k}^{K} \delta(X - X_k) \qquad [3]$$

Así a la integral [2] la convertimos en la siguiente sumatoria:

$$p(t \mid W, \beta) = \frac{1}{K} \sum_{k}^{K} p(t \mid X_k, W, \beta) \qquad [4]$$

Ahora tenemos al modelo donde cada centro de cada función delta (así nos referiremos para los puntos latentes) es un trazado (o mapeo) al centro de gaussianas. Es decir, cada punto latente será reflejado en el centro de gaussianas en el espacio de datos.

Ahora bien, dando un conjunto de datos finito $\{t_1,..,t_N\}$ podemos escribir la función de máxima verosimilitud (función *likelihood*) para el modelo GTM como sigue:

$$£ = \prod_{n}^{N} p(t \mid W, \beta) = \prod_{n}^{N} \left(\frac{1}{K} \sum_{k}^{K} p(t_n \mid X_k, W, \beta) \right) \qquad [5]$$

y se maximiza con respecto a W y β. También podemos escribir la función de máxima verosimilitud de la siguiente forma, a través de maximizar con la función logarítmica de [5]:

$$\ell = \sum_{n}^{N} \ln \left(\frac{1}{K} \sum_{k}^{K} p(t_n \mid X_k, W, \beta) \right) \qquad [6]$$

En el apéndice C, en la sección 2.2. presentamos una explicación más amplia de la función de máxima verosimilitud.

V. 3. 3. Formando al algoritmo EM para el modelo GTM

Para empezar a darle forma al algoritmo EM basándonos en el modelo GTM, damos como datos iniciales: W y β. Entonces en el paso **E** se calculan las responsabilidades R, que representan la probabilidad posteriori que el dato N-ésimo fue generado por el K-ésimo componente de la variable latente, para este concepto nos basamos en el teorema de Bayes, y su definición es:

$$R_{kn} = p(x_k \mid t_n, W, \beta) = \frac{p(t_n \mid x_k, W, \beta)p(x_k)}{\sum_{k'} p(t_n \mid x_{k'}, W, \beta)p(x_k')} \quad [7]$$

Esto es equivalente a tener la siguiente función, donde se deduce en el apéndice C, sección 2.1.:

$$R_{kn} = \frac{e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)}}{\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)}} \quad [8]$$

$$\text{donde } \Delta = \sum_{d}^{D} (t_d - y_d(x, W))^2 , \quad d \equiv n \text{ y } D \equiv N \quad [9]$$

En el paso **M**, estas responsabilidades actuarán como los pesos en la ecuación para actualizar W y para actualizar β. Prácticamente lo que se hace es mover cada componente del conjunto de datos hacia el más responsable de ellos.

Para actualizar β, se calcula la derivada de nuestra función de máxima verosimilitud con respecto a β, y se iguala a cero para encontrar su despeje. De ahí entonces llegamos a la siguiente ecuación con la que se actualiza β (que se deduce en el apéndice C, sección 2.3). ND es el total de los datos de entrada:

$$\frac{1}{\beta} = \frac{1}{ND} \sum_{n}^{N} \sum_{k}^{K} R_{kn} \Delta_{kn} \quad [10]$$

Para el modelo GTM la transformación de las variables latentes al espacio de datos, se trata de una combinación de parámetros no lineales donde se consideran, además de los pesos W, las funciones bases no lineales que constan de centros de distribuciones gaussianas. La transformación se denota con la siguiente función:

$$y_d(x, W) = \sum_{m}^{M} \Phi_m W_{md} \quad [11]$$

Las funciones bases ϕ se determinan con la referencia del conjunto de las variables latentes y se representan como lo mostramos enseguida:

$$\Phi_m(x) = e^{\left(-\frac{\|x - \mu(m)\|^2}{2\sigma^2}\right)} \quad [12]$$

donde μ es el conjunto de centros de las gaussianas, y σ es el ancho de éstas.

Para indicar cuales serán los centros, también tomamos como referencia a las variables latentes. Para explicar la ubicación de los centros, presentamos la figura 5.8 donde muestra el conjunto de las variables latentes (a la izquierda) y los centros marcados con un círculo (lado derecho) sobre las mismas variables latentes.

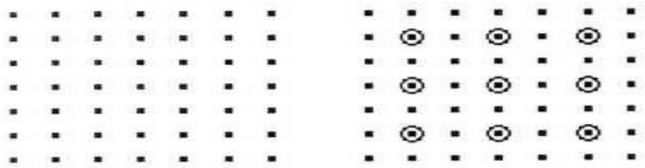


Figura 5.8. Conjunto de variables latentes (izquierda) y la ubicación de los centros (derecha)

En el paso M actualizamos los pesos W. Para deducir como se actualiza W se calcula la derivada de nuestra función de máxima verosimilitud con respecto a W, y se iguala a cero para encontrar su despeje. De ahí entonces llegamos a la siguiente ecuación con la que se actualiza W (que es deducida en el apéndice C, sección 2.4):

$$\left(\Phi^T G\Phi + I\lambda\right)W = \Phi^T RT \qquad [13]$$

donde,

1. Φ^T es la transpuesta de Φ
2. G es una matriz diagonal de K X K con los siguientes componentes:

$$G_{KK} = \sum_{n}^{N} R_{kn} \qquad [14]$$

3. I es una matriz identidad con la misma dimensión que $\Phi^T G\Phi$
4. λ es un parámetro de regularización:

$$\lambda = \frac{\alpha}{\beta} \qquad [15]$$

donde $\alpha = 0.01$

V. 3. 4. Inicialización de los datos

Los parámetros W y β requieren de una inicialización de sus valores. La inicialización de los valores para W se calculan de forma aleatoria con un rango de 0 hasta ς, donde ς es la varianza del conjunto de datos (de la matriz T), y se calcula como sigue, donde ND son el número de datos de T:

$$\varsigma = \frac{\sum_{n}^{ND}\left(y_n - \bar{y}\right)^2}{ND} \qquad [16]$$

donde: $$\bar{y} = \frac{\sum_{n}^{ND} y_n}{ND} \qquad [17]$$

Existe otra forma de inicializar W llamada PCA, que no será explicada. Pero debido a la elección que tomamos para inicializar W, el valor inicial para β se calcula de la siguiente forma:

$$\frac{1}{\beta} = \frac{1}{ND}\sum_{n}^{N}\sum_{k}^{K}\Delta_{kn} \qquad [18]$$

V. 3. 5. El algoritmo EM

Ya que describimos el modelo GTM, fácilmente puede formarse el algoritmo EM para modelos de variables latentes paso a paso, las definiciones de las siguientes variables y la explicación del paso correspondiente, lo presentamos seguido del algoritmo. El algoritmo EM tiene los siguientes pasos:

1. Capturar T, N y D
2. Determinar h y K
3. Generar X con dimensiones h X h
4. Determinar M
5. Generar el vector μ con dimensión M
6. Calcular ϕ
7. Generar W
8. Calcular Δ con dimensiones K X N
9. Inicializar β
10. Iniciar ciclo
    a. Calcular R
    b. Calcular G
    c. Actualizar W
    d. Actualizar Δ
    e. Actualizar β
11. Finalizar el ciclo

Este algoritmo se presentó en una forma generalizada, enseguida ampliaremos la mayoría de los pasos, en algunos otros no será necesario ahondar. En ocasiones presentamos referencia a la función que se utiliza en el paso correspondiente.

1. Los datos de entrada son: la matriz de datos T y sus dimensiones: N X D.
   Inicializaremos los datos necesarios para los pasos E y M.
2. Se determina el número de variables latentes K, esto es: K = (h)(h). Se requiere que exista por lo menos D variables latentes, entonces, para asegurarnos de ésto, calcularemos su dimensión h así: h = (D/2)+1. Para fines prácticos (como se explicará más adelante), igualmente se requiere que h sea impar, por lo tanto, si h es par entonces: h = h+1.
3. Se genera la matriz de las variables latentes X con dimensiones h X h.
4. Se determina el número de centros, por ende el número de funciones bases M. Se calcula de la siguiente forma: tomamos h y le restamos 1 para obtener par y para que al dividirlo entre dos nos arroje un entero, dividimos entre dos porque se desea menos de la mitad de variables latentes, y se eleva al cuadrado para obtener el total de centros que se requiere por toda la matriz X. Con esta explicación se forma la siguiente sencilla función para M: M = ((h-1)/2)^2. [11]
5. Se genera el vector μ que contendrá los centros de las gaussianas.
6. Se calculan las funciones bases ϕ con dimensiones K X M, y con $\sigma = 0.01$.
7. Se generan los pesos W de forma aleatoria.
8. Se calcula la matriz delta Δ con dimensiones K X N. [9]
9. Se calcula el valor inicial para β. [18]
   Ahora empiezan los pasos **E** y **M**, éstos se introducen en un ciclo:
10. Repetir las siguientes acciones:
    PASO E
    a. Se calculan las responsabilidades R. [8]
    b. Se calcula la matriz G. [14]
    PASO M
    c. Se actualizan los pesos W. [13]
    d. Se actualiza la matriz Δ con la nueva W. [9]
    e. Se actualiza β con la R y Δ. [10]
11. Fin del ciclo. El fin del ciclo se determina cuando β o W convergen, ésto significa: Cuando β anterior y β actual difieran por un valor épsilon, o bien, cuando W anterior o W actual sean quienes difieran por otro valor épsilon en cada uno de sus elementos.

V. 4. Transformada de Fourier

Presentamos la primera técnica para el análisis de los datos, que proponemos para nuestros datos reducidos y visualizados. Esta primera técnica es la Transformada de Fourier, basada en el análisis espectral de señales. En estas dos últimas secciones del capítulo, detallamos los conceptos básicos de las transformadas, debido a que el desarrollo e integración de estas transformadas, las propondremos como trabajo para siguientes versiones de la aplicación ViAn. Sin embargo, para el propósito de completar la aplicación presentamos los siguientes conceptos: Transformada de Fourier y Transformada de Wavelets.

La Transformada de Fourier (FT, por sus siglas en inglés: *Fourier Transformed*) se considera una herramienta clásica dentro de la ciencia e ingeniería para el análisis de datos. La FT es extensamente utilizada para análisis de una señal periódica, análisis de sistemas lineales; además es empleada en estudios de antena, óptica, modelado de procesos aleatorios, teoría probabilística, física cuántica y problemas de valores en la frontera; incluso ha tenido muchas aplicaciones exitosas en la resonancia de datos astronómicos.

La FT es una transformada lineal al igual que la transformada de Laplace. En esencia, la FT descompone o separa a algún fenómeno o función en forma de onda (sinoidal) o de frecuencias distintas. Esto es, se utilizan funciones suaves y bien localizadas en un intervalo determinado para la representación de tiempo-frecuencias. Esto nos lleva a pensar en la FT desde otra perspectiva, tal como una técnica matemática para transformar nuestras observaciones en señales de tiempo por frecuencia.

Existe un algoritmo computacional para la FT, conocido como la Rápida Transformada de Fourier (FFT, por sus siglas en inglés: *Fast Fourier Transformed*). Este algoritmo calcula la FT Discreta (DFT, por sus siglas en inglés: *Discrete Fourier Transformed*), debido a que en la computación se manejan datos discretos. La FFT no ha sido desarrollada para el sistema ViAn, sin embargo, si se utiliza y la presentamos en la aplicación, debido a que: en nuestro ambiente de implementación, el lenguaje IDL, contiene un módulo que procesa al algoritmo de la FFT, y es el módulo que utilizamos en nuestra aplicación ViAn.

El análisis con Fourier, tiene una desventaja. En el proceso de transformación para el dominio de frecuencia, se pierde información. Cuando observamos la FT de una señal, es imposible decir cuándo se presenta un evento en particular. Si las propiedades de la señal no cambian mucho al paso del tiempo, esto es, si las señales son *estacionarias*, esta desventaja no es muy importante.

Si embargo, debido a que las señales más interesantes contienen numerosas características no estacionarias o transitorias (por ejemplo derivas, tendencias, cambios abruptos, inicios y finales de eventos) y debido también a que, como mencionábamos, el análisis con Fourier no es apto para detectar estas señales, en muchas ocasiones se requiere entonces de otras técnicas de análisis de datos, una de ellas: Wavelets.

V.5. Transformada Wavelets

Nuestra segunda técnica para el análisis de los datos, que proponemos para nuestros datos reducidos y visualizados, la presentamos en esta sección. Esta segunda técnica es la Transformada de Wavelets, basada en el análisis de multirresolución. Como mencionábamos en la sección anterior, en estas dos últimas secciones del capítulo, explicamos lo básico de los conceptos de las transformadas, debido a que el desarrollo y la integración de éstas, las propondremos como trabajo para siguientes versiones de nuestro sistema ViAn.

La transformada de Wavelets (WT por sus siglas en inglés: *Wavelets Transformed*) es una herramienta muy popular en la actualidad para el análisis de imágenes y datos, además ha resultado muy efectiva en los casos de manejar grandes volúmenes de datos. Un Wavelet (una ondeleta) es una función en forma de onda, esto es, con duración limitada eficazmente, la cual tiene un valor promedio de cero.

Comparando a los wavelets con ondas sinoidales (las cuales son base en el análisis de Fourier), las sinoidales no tienen un límite de duración, esto significa que, ellas se extienden desde el infinito negativo hasta el infinito positivo. Además las sinoidales son lisas y predecibles, y los wavelets tienden a ser irregulares y asimétricos. El análisis con Fourier consiste en el análisis de ondas sinoidales de varias frecuencias. Los wavelets las analizan similarmente, con la diferencia de que lo realiza trasladando y escalando versiones de la ondeleta original denominada *Wavelet madre*.

Existe un método de sus creadores Burt y Adelson **[Burt, 1983]**, que fue el primero en introducirse como un método para el análisis de imágenes con escalas múltiples de sensitividad espacial. Este método presenta algunas características de la WT en un sentido relativamente intuitivo. Esta WT por el método de Burt-Adelson tiene la característica de ser discreta. A esta WT por Burt-Adelson, también la encontramos en la literatura como los algoritmos piramidales de Burt-Adelson para el análisis de datos o imágenes.

La transformada Wavelets es una técnica de multirresolución de patrones basada en la pirámide Laplaciana. Detallando lo que es la multirresolución, es un modelo que captura una amplia gama de niveles del detalle de un objeto y del cual se puede utilizar para reconstruir niveles de demanda. Dicho de diferente manera, esta técnica permite representar los objetos en distintos niveles de resolución, con los que es posible realizar manipulaciones sobre los mismos de forma más eficiente.

Ahora detallemos lo que es la construcción de un nivel de la pirámide Laplaciana. Ésta consiste en 3 pasos de progresión, observemos lo que presentamos en la figura 5.9.

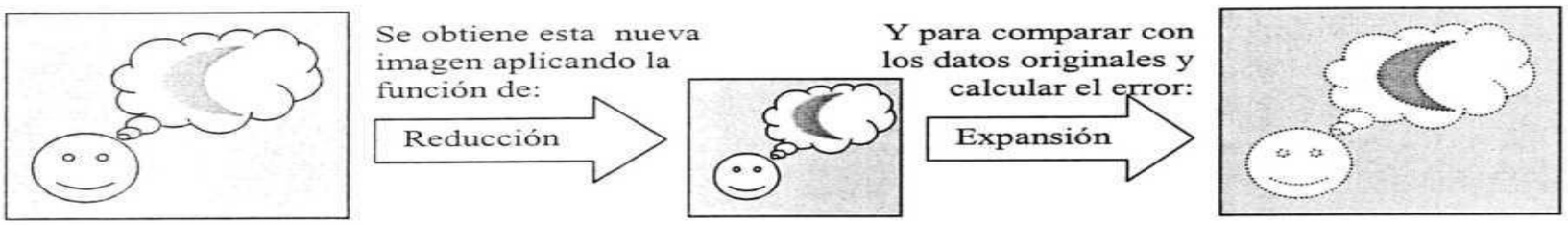


Figura 5.9. Pasos de progresión de un nivel para formar la pirámide Laplaciana

Primero un filtro pasa-bajo (*low-pass*) gaussiano llamado *reducción* se aplica a los datos reducidos M-dimensionales, así se consigue una versión más baja de la resolución de los datos de entrada de información. Se aplica la función inversa de *reducción* que se llama *expansión* se consigue una aproximación del nivel anterior. Si se calcula las diferencias entre los datos de entrada de información y los datos ampliados, se consiguen el nivel respectivo de la pirámide Laplaciana a los que se les llama componentes de alta frecuencia. Si se relanza ésto varias veces, pero no al nivel superior posible, se consigue la pirámide Laplaciana. Para el cálculo de error, es decir, la pérdida de información en los datos depende del número y de la distribución de los niveles de la cuantización. Se obtiene solamente el nivel más alto de la pirámide gaussiana.

CAPITULO VI.

Implementación del sistema ViAn

La mayor recompensa de nuestro trabajo no es lo que nos pagan por él, sino aquello en lo que nos convierte.

John Ruskin

CAPITULO VI.
Implementación del sistema ViAn

VI. 1. Introducción

La implementación del sistema ViAn, comprende los alcances y las limitaciones que tiene nuestro sistema ViAn en su primera versión para el desarrollo y presentación de nuestro proyecto de tesis. Para la implementación del sistema, es decir, para desarrollar el código que forma nuestra aplicación, es base principal el diseño del sistema. Aunque para su implementación (o programación), puede recurrirse también al análisis del sistema y a los antecedentes presentados.

Mostraremos una tabla con el total de módulos en el sistema ViAn, y expondremos una breve descripción de lo que realiza cada uno de estos módulos. Además, conoceremos las variables que forman nuestra estructura de datos E, y clasificaremos sus componentes, según las partes de la E detalladas en el diseño del sistema. También mostraremos al manual de usuario, utilizando el diseño de interfaz gráfica.

VI. 2. Especificaciones del desarrollo de ViAn

Para la primera versión de ViAn, implementamos (programamos) lo que se planteó, especificó y propuso en los capítulos anteriores, delimitando procesos, según nuestro alcance en tiempo y según nuestras prioridades. La aplicación cumple con los objetivos especificados y con los requerimientos.

Un punto a especificar, es la delimitación de la implementación a la lectura de los datos originales, denominados N. Esto es, el concepto de los proyectos no lo encontraremos en esta versión 01 del sistema. La extensión que determinamos para los proyectos ('.via'), la mantuvimos en el sistema, para almacenar los datos N. Y para esta versión, la extensión original de los datos N ('.dtN'), no la contemplamos. Contamos con otros tipos de archivos, los que contienen a los datos resultantes de la reducción de datos ('.dtR'), y los datos numéricos resultantes del análisis de datos ('.dtA'), que son desplegados y pueden ser almacenados en el SA, pero como mencionaba, no pueden ser leídos.

Una parte importante que proponemos para una siguiente versión de ViAn y que es digna de discusión, es el desarrollo de la transformada de Wavelets. Esta técnica de análisis de datos, es muy extensa y requiere de un estudio y desarrollo amplio. Si bien, ha sido estudiada, pero su detalle requiere de tiempo, el cual, como en todo proyecto, es limitado para su presentación.

Especifiquemos que, aunque solo se analizó y diseñó en un nivel superficial, se agregó el proceso de la reducción de dimensión y la visualización de datos con ambos algoritmos de forma simultánea. Este proceso se detalla en los requerimientos de interfaz y en el diseño de interfaz. Es considerado un requerimiento esperado, debido a que en el requerimiento **1e.** no indica que el despliegue de los resultados sean simultáneos para ser comparados.

Especificamos los requerimientos innovadores que se desarrollaron para ViAn. Uno de ellos es el Diseño de Interfaz Gráfica (que se presenta en el manual de usuario), que es parte de la atracción visual que proyectan las pantallas hacia el usuario. Esto consta de imágenes que identifican distintas actividades dentro del sistema, y consta de un logotipo propio de ViAn. Apreciamos este logotipo en la introducción a la aplicación y en un área de la interfaz gráfica.

Desde los requerimientos de interfaz, detectamos un botón en la parte superior que indicaba los datos originales N. En esta fase, utilizamos ese botón, para generar el evento que nos permite procesar actividades que son requerimientos innovadores para la aplicación. Estas actividades son: 1. La consulta de los datos N, que el sistema tiene cargado en memoria para procesarlos. 2. Modificar cualquier dato de la matriz, para posteriormente almacenarlos, ya sea: a) sobre-escribiendo los datos de la matriz en el archivo original, o b) escribiendo los datos actualizados en un archivo diferente. Pueden modificarse todos los datos, sin embargo, no pueden actualizarse las dimensiones.

En el momento de generar una nueva matriz de datos N, desde la aplicación, se especifica previamente las dimensiones de esta matriz. Una restricción para el usuario es que la matriz debe tener en cada dimensión un valor mínimo de dos (2). De lo contrario, no se procederá la creación de la matriz.

VI. 2. 1. Manual de usuario, utilizando al Diseño de Interfaz Gráfica

En el manual de usuario mostraremos a detalle los alcances, especificaciones y delimitantes que presentamos anteriormente. Por ejemplo, la información que localizamos en las pantallas del Diseño de Interfaz, en la parte inferior, pertenecen a datos correspondientes a los proyectos. Y explicamos que los proyectos han sido diseñados para una siguiente versión de la aplicación.

El manual de usuario lo localizamos en el apéndice D, y en el área de *ayuda* que contiene la opción de "Manual de usuario". Esta área se aprecia en la figura D2b) (apéndice D) donde nos muestra que se encuentra en el menú superior de la aplicación. Para realizar el manual de usuario, nos basamos en esta fase de implementación, enfocándonos en la funcionalidad del sistema desde la perspectiva del usuario. Aquí mismo utilizamos las ventanas que conforman al Diseño de Interfaz Gráfica. En este capítulo presentamos la pantalla principal de ViAn en la figura 6.1, la cual la identificaremos en el apéndice D como la figura D1.

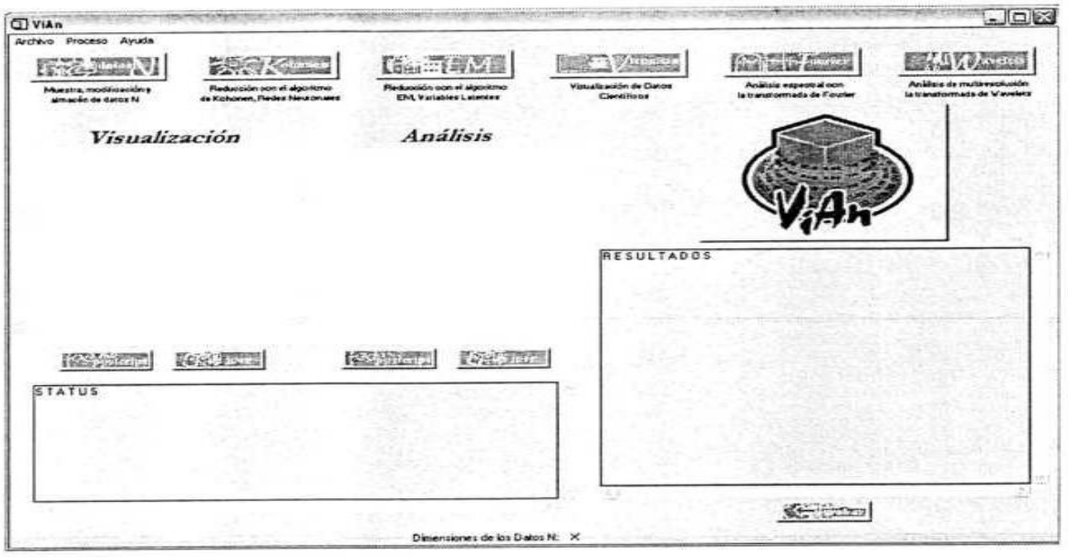


Figura 6.1. Pantalla principal de la aplicación

El diseño para la interfaz lo desarrollamos en tres etapas. La primera (presentada en el capítulo III) la nombramos "Requerimientos de interfaz", la segunda (capítulo IV) lleva el nombre de "Diseño de interfaz". Y por último, la tercera etapa es la que presentamos en esta fase, "Diseño de interfaz gráfica". En el diseño de interfaz gráfica mostramos las interfaces de la primera versión de la aplicación ViAn, a través de las cuales el usuario interactuará con el sistema.

VI. 2. 2. Módulos del sistema ViAn

Enseguida presentamos la tabla que contiene todos los procedimientos y funciones que conforman al sistema ViAn. Esta tabla indica si el módulo es una función o no lo es (si es procedimiento), contiene el nombre del módulo, una breve descripción del módulo, y el archivo al que pertenece (extensión '.pro'). Todos los archivos se encuentran agrupados en un proyecto llamado: **ViAn.prj**. El orden de los módulos en la tabla VI.I, es por aparición en el archivo indicado:

VI. 2. 3. Clasificación de nuestra Estructura de Datos (E)

Ahora, presentaremos la estructura de datos E, que se encuentra implementada. Relacionaremos las partes definidas en el diseño, con las variables que contiene. El nombre de la estructura E en la implementación: *EstructuraGeneral*. Accedemos a ella a través de un apuntador, que se llama: *EstGral*. Y nos referiremos a ella a lo largo del sistema como: E.

1. **Control del flujo de eventos:** Estos son los identificadores de las unidades de interfaz gráfica (Widgets), que permiten activar o desactivar al botón u opción del menú. Estos son: *bttn1_1, bttn1_2, bttn1_3:bttn1_3, bttn2_1, bttn2_2, bttn2_1_1, bttn2_1_2, bttn2_1_3, bttn2_3, bttn2_3_1, bttn2_3_2, bttn2_3_3, bttn3_1, bttn3_2, bttn3_3, bttn5, bttn6, bttn7, bttn8, bttn9, bttn10, bttn11, bttn12, bttn13, bttn14, bttn15.*

2. **Control del flujo de datos:** Estas variables son las que indican cuales fueron los últimos procesos en generarse, y otros indican si ya se realizó algún proceso con los datos indicados. Estas variables son las siguientes: *uVis, uAna, uAlm, ko_em, fo_wa, firstK, firstEM, firstV_K, firstV_EM, firstF_K, firstF_EM, firstW_K, firstW_EM.*

3. **Matrices y dimensiones de los datos:** Son matrices que contienen a los datos N y a los datos procesados, además, están las dimensiones de algunas de las matrices.
N, Ntmp, D, Dtmp, datosN, datostmp, datosM_K, datosM_EM, datosG_K, datosG_EM, datosA_F_K, datosA_W_K, matrizAuxF, datosA_F_EM, datosA_W_EM, matrizAuxW, dimKo, dimEM, EjeX, EjeY, varX, varY, Eje1, Eje2

4. **Despliegue de información:** Estos son identificadores de Widgets, para desplegar información, resultante de los procesos, ya sean matrices de datos o imágenes. También se encuentra un widget que despliega las dimensiones de los datos N con los que se está trabajando. *draw1, draw2, etiqueta4, txt1, txt2.*

5. **Información acerca del experimento:** Esta variable contiene al nombre del archivo en el que se encuentran los datos N, que se encuentran en la memoria del sistema. En versiones futuras de ViAn, se extenderá el número de variables para esta parte, ya que los proyectos sean desarrollados. El nombre de la variable es: *archN.*

Así, con este capítulo, concluimos con los pasos de la ingeniería del software, para contar con un soporte para la aplicación. Además de ser el cuerpo principal de la documentación de la tesis, una herramienta fundamental para obtener el título de licenciatura en la UABC.

CAPITULO VII.

Resultados

Todo lo que somos es el resultado de lo que hemos pensado; está fundado en nuestros pensamientos y está hecho de nuestros pensamientos.

Buda

CAPITULO VII.
Resultados

VII. 1. Introducción

Hemos presentado las diferentes etapas en el desarrollo de una aplicación de software orientado a un dominio de aplicación científico y tecnológico, con el fin de realizar la tarea de la visualización de datos científicos (VDC) a través de algoritmos para reducción de dimensión. Algoritmos, como mencionamos, con los que se mantiene la topología de los datos basados en redes neuronales. Podemos establecer que el proceso de desarrollo de software que utilizamos en la herramienta para la visualización es consistente y completo en términos de los requerimientos, que fueron detallados desde las tareas de lectura de datos N-dimensionales, los procesamientos para la reducción de dimensión de datos, la visualización de los datos en dimensión reducida (2D) desplegándolos en imagen, hasta el análisis de estas imágenes.

El sistema ViAn desarrollado en el lenguaje IDL, ofrece la funcionalidad de procesar datos a través de los algoritmos descritos, almacenar los resultados en forma numérica, almacenar imágenes resultantes de procesos con dos opciones de formato, además permite la entrada de datos de altas dimensiones directamente desde la aplicación para almacenarlos, o los lee desde archivos, y es capaz de modificarlos sobre-escribiéndolos o escribirlos como un archivo nuevo de datos originales.

En este trabajo de tesis, se desarrolló la aplicación específicamente con dos algoritmos para la reducción de dimensión de datos, y una técnica para el análisis de datos. En esta fase mostramos la funcionalidad con ejemplos de los resultados que ViAn obtiene en cada tarea para la reducción, visualización, análisis de datos, almacenamiento de datos y lectura de datos, con el fin de detallar los resultados obtenidos en el desarrollo de la aplicación. Nos basaremos en los procesos especificados en el análisis y el diseño del sistema.

En forma generalizada, obtuvimos como resultado de nuestro trabajo de tesis, el desarrollo de una **aplicación** de software de propósito específico. La aplicación se presenta como un **módulo** en el lenguaje de desarrollo IDL, el cual puede ser reusable. Desde del punto de vista del usuario, ViAn es una **herramienta** de apoyo para que el investigador la utilice en el manejo de datos de sus experimentos. Por último, ViAn es un **sistema** con módulos desarrollados con la ingeniería del software.

VII. 1. 1. Funcionalidad del sistema

Las pruebas que realizaremos en esta etapa (fase), serán solo en términos de la funcionalidad del sistema. Es decir, para verificar la funcionalidad del sistema se hicieron varias pruebas, donde se generan eventos (de forma interna o a través de un usuario) y se esperan las respuestas correctas del sistema. Para establecer qué respuestas son las correctas, consultamos el análisis y diseño del sistema de las fases desarrolladas para la aplicación. Específicamente, los modelados que consultamos son: los diagramas de secuencia, diagramas para el flujo de datos, modelado de flujo de procesos y el diseño de interfaz; presentadas en capítulos anteriores.

Enseguida presentamos algunos escenarios como ejemplos del uso del sistema, éstos se realizaron como parte de las pruebas, y así comprobamos la funcionalidad de la aplicación. Seleccionamos la presentación de los escenarios basándonos en el modelado de flujo de procesos.

a. Pre-procesamiento de datos Originales

El pre-procesamiento de datos, es la introducción y/o lectura de los datos originales de nuestro experimento en forma de matriz, se indica también las dimensiones de dicha matriz. Los siguientes procesos están representados en sub-procesos de diagramas presentados en los capítulos anteriores, como por ejemplo: en el caso de uso "Reducción de dimensión de datos N"; el modelado de flujo de procesos "Pre-procesamiento de datos N"; diagrama de estereotipo "Pre-procesamiento"; diagrama para el flujo de datos "Pre-procesamiento de datos N".

Para el flujo del pre-procesamiento de datos Originales (datos N), contamos con varios procesos en esta fase de implementación del sistema ViAn. Estos procesos inician a través de diferentes eventos generados por el usuario. Presentamos los procesos para la "Captura de datos N y su almacenamiento", para la "Lectura de un archivo de datos N" y para la "Actualización de datos N y su almacenamiento".

- Captura de datos N y su almacenamiento

a) b)

Figura 7.1. Opción de crear un archivo nuevo:
(a) Elección desde el menú principal. (b) Captura de dimensiones del archivo nuevo.

El procedimiento que seguimos para capturar los datos N y su almacenamiento, inicia en la elección de la opción "Archivo nuevo" del menú principal, como lo presentamos en la figura 7.1(a), e indicamos las dimensiones de nuestros datos a través de la ventana que muestra la figura 7.1(b).

Entonces procedemos a introducir los datos en la tabla que presenta la figura 7.2. Una vez introducidos los datos, se presiona el botón "OK" para seguir con en proceso de almacenar los datos en un archivo en el Sistema de Archivos (SA).

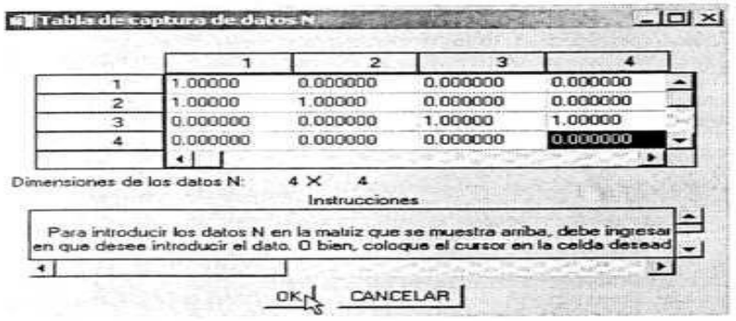


Figura 7.2. Tabla para capturar los datos N

Para almacenar los datos capturados, se indica el nombre del archivo con el cual se desea guardar a través de la ventana que muestra la figura 7.3. En la figura 7.4 presentamos el contenido del archivo almacenado, éste contiene los datos y sus dimensiones. Dicho archivo es almacenado con la extensión '.via'.

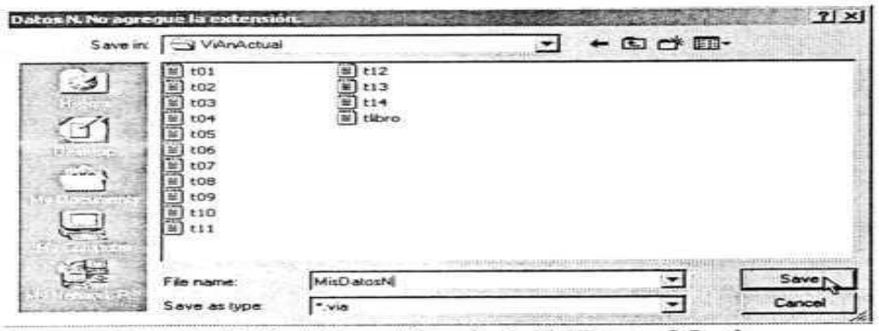


Figura 7.3. Almacén de MisDatosN.via

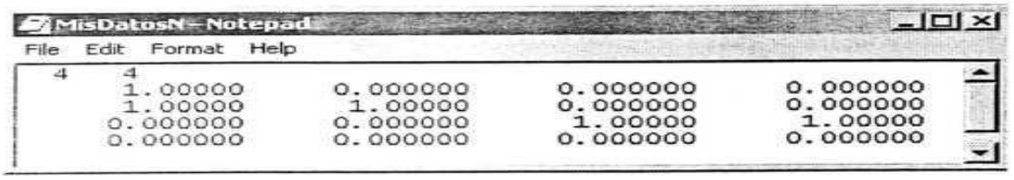
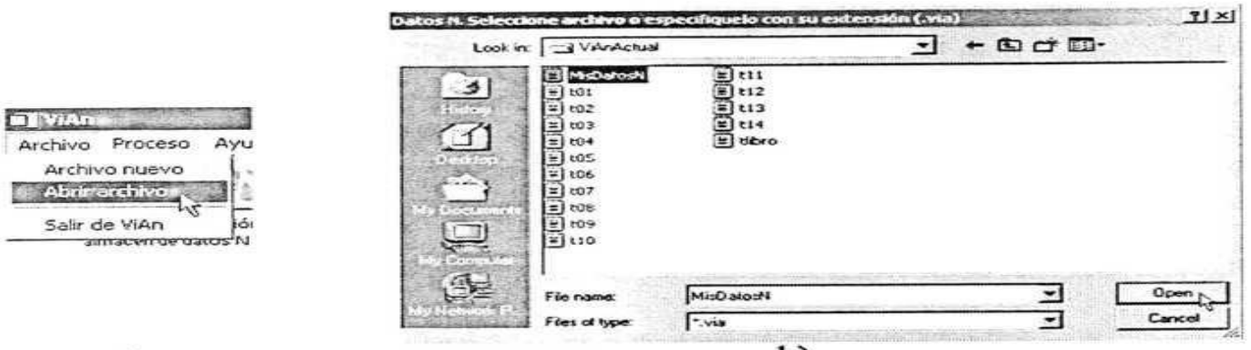


Figura 7.4. Archivo MisDatosN.via

- Lectura de un archivo de datos N

Para leer un archivo del SA, elegimos la opción "Abrir archivo" del menú principal como lo presentamos en la figura 7.5(a), enseguida se abre la ventana que presenta la figura 7.5(b) que muestra los archivos del SA, los cuales pueden leerse. Se elige uno, y al abrirlo, los datos que contiene este archivo se carga en la estructura de datos E del sistema para su procesamiento.



a) b)

Figura 7.5. Opción de abrir un archivo:

(a) Elección desde el menú principal. (b) Lectura de MisDatosN.via.

- Actualización de datos N y su almacenamiento

Al presionar el botón "datos N", como lo presenta la figura 7.6(a), se abre la ventana que se muestra en la figura 7.6(b) para el procedimiento de actualizar los datos N y almacenarlos nuevamente. En la tabla se despliegan los datos del archivo cargado en el sistema y es donde se pueden modificar. Se presiona el botón "sobre-escribir" para actualizar el archivo. Si desea que los datos modificados se almacenen en un nuevo archivo, se presiona el botón "escribir como...". En la figura 7.7 presentamos el archivo una vez actualizado.

a) b)

Figura 7.6. Opción para actualizar los datos N:

(a) Presionar botón "datos N". (b) Modificar datos N.

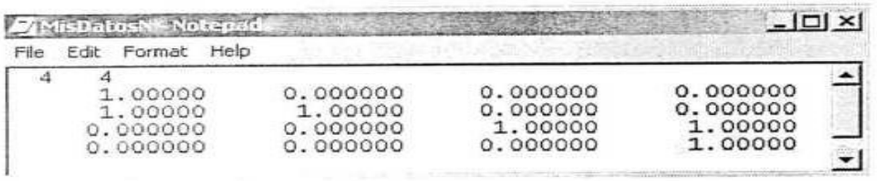


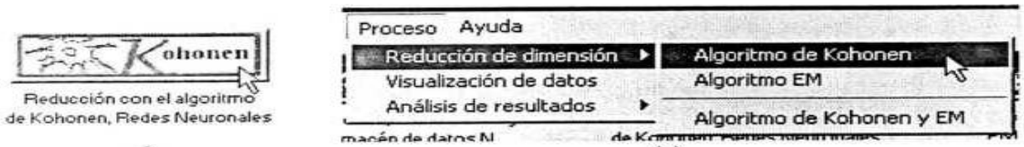
Figura 7.7. Archivo MisDatosN.via actualizado

b. **Reducción de dimensión de datos Originales**

Para poder visualizar los datos del experimento que se introdujeron al sistema, primero hay que reducir sus dimensiones. La Reducción de dimensión de datos Originales se logra a partir de eventos diferentes que indican el tipo de algoritmo con el que se realizará este procedimiento: con Kohonen o EM. Los diagramas relacionados con la reducción de dimensión de datos a lo largo del desarrollo del sistema se presentaron en los capítulos anteriores, algunos de estos diagramas son: en el caso de uso "Reducción de dimensión de datos N"; el modelado de flujo de procesos "Reducción de dimensión de datos N"; diagrama de estereotipo "Reducción"; diagrama para el flujo de datos "Reducción de dimensión de datos N".

Presentamos enseguida los procesos: "Reducción de datos con el algoritmo de Kohonen", "Reducción de datos con el algoritmo EM" y "Almacenamiento de datos resultantes de la reducción (Kohonen en el ejemplo)".

- Reducción de datos con el algoritmo de Kohonen



a) b)

Figura 7.8. Opción para reducir los datos N con el algoritmo de Kohonen: (a) Presionar botón "Kohonen". (b) Elección desde el menú principal.

Para obtener la reducción de dimensión de datos a través del algoritmo de Kohonen, se presiona el botón "Kohonen" o se elige la opción "Algoritmo de Kohonen" desde el menú principal, como lo presentan las figuras 7.8(a) y 7.8(b) respectivamente. Los datos N cargados se procesan a través de dicho algoritmo y se obtiene una matriz resultante de datos, listos para visualizarlos y/o analizarlos.

La figura 7.9 muestra los datos resultantes del proceso. Debajo del área para los resultados, se encuentra un botón ("grabar") para almacenar la matriz presentada en el SA.

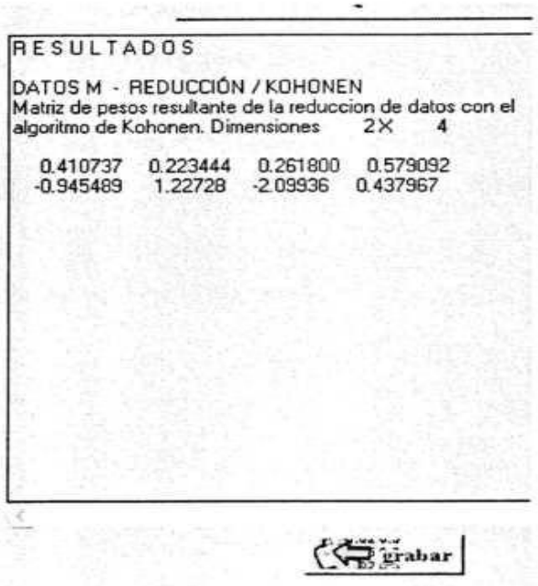
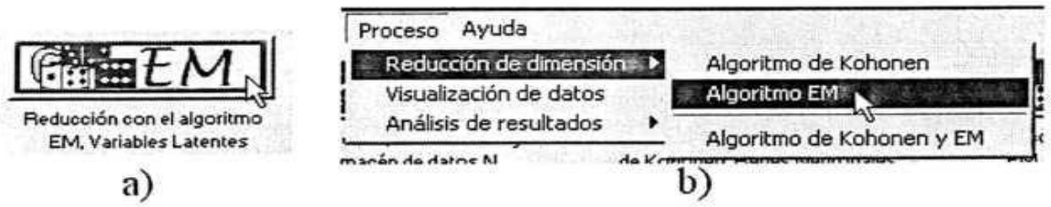


Figura 7.9. Datos resultantes de la reducción de dimensión con Kohonen

- Reducción de datos con el algoritmo EM



a) b)

Figura 7.10. Opción para reducir los datos N con el algoritmo EM:
(a) Presionar botón "EM". (b) Elección desde el menú principal.

Para obtener la reducción de dimensión de datos a través del algoritmo EM, se presiona el botón "EM" o se elige la opción "Algoritmo EM" desde el menú principal, como lo presentan las figuras 7.10(a) y 7.10(b) respectivamente. Los datos N cargados se procesan a través de dicho algoritmo y se obtiene una matriz resultante de datos, listos para visualizarlos y/o analizarlos. La figura 7.11 muestra los datos resultantes del proceso. Debajo del área para los resultados, se encuentra un botón ("grabar") para almacenar la matriz presentada en el SA.

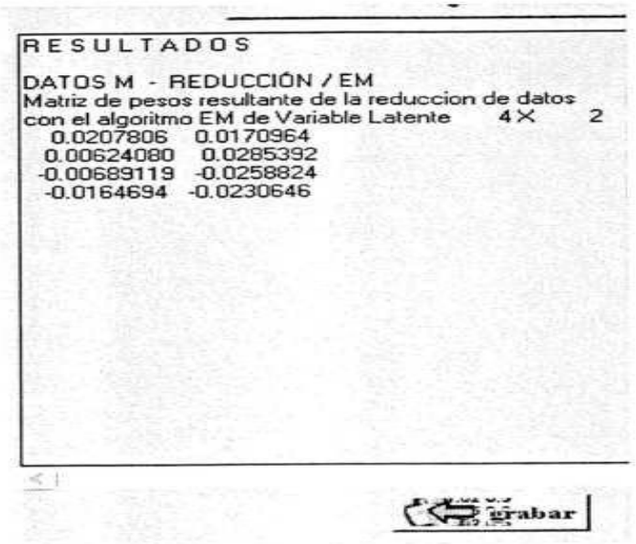


Figura 7.11. Datos resultantes de la reducción de dimensión con EM

- Almacenamiento de datos resultantes de la reducción (Kohonen en el ejemplo)

Los datos resultantes de la figura 7.9 pueden ser almacenados en el SA presionando el botón "grabar". Al presionar dicho botón, se abre la ventana que presentamos en la figura 7.12 con el fin de que se indique el nombre del archivo con el que se grabará la matriz en el SA. En la figura 7.13 mostramos al archivo MiReduccion.dtR, con la matriz y dimensiones de los datos resultantes de la reducción con el método de Kohonen.

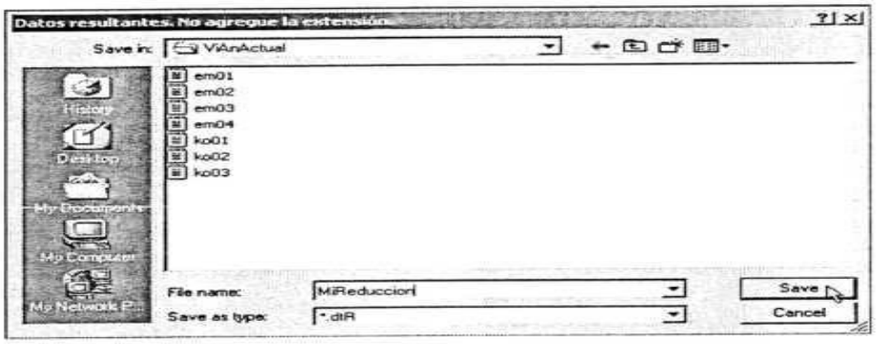
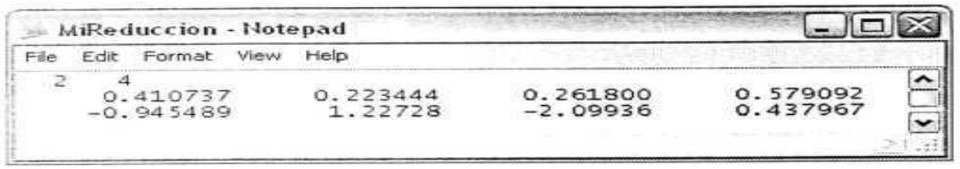


Figura 7.12. Almacén de MiReduccion.dtR



7.13. Archivo MiReduccion.dtR

c. Visualización de datos Reducidos

La Visualización de datos Reducidos se procesa después de reducir los datos con cualquiera de los dos algoritmos opcionales. Algunos de los procesos que presentamos en capítulos anteriores relacionados con la visualización de datos reducidos son: en el caso de uso "Visualización de datos M"; el modelado de flujo de procesos "Visualización de datos M"; diagrama de estereotipo "Visualización"; diagrama para el flujo de datos "Visualización de datos M".

Enseguida presentamos los procesos que pueden realizarse al generar eventos diversos relacionados con la visualización de datos Reducidos: la "Visualización de datos reducidos a través de Kohonen", "Visualización de datos reducidos a través de EM" y "Almacenamiento de imágenes visualizadas".

- Visualización de datos reducidos a través de Kohonen

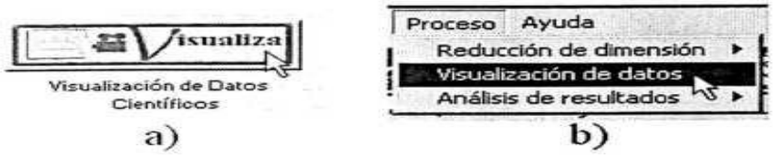


Figura 7.14. Opción para visualizar los datos reducidos:
(a) Presionar botón "Visualiza". (b) Elección desde el menú principal.

Para el procedimiento de visualizar los datos reducidos, se presiona el botón "Visualiza", como se muestra en la figura 7.14(a) o se selecciona la opción "Visualización de datos" desde el menú principal como se muestra en la figura 7.14(b).

Para el ejemplo de la figura 7.15 suponemos que se redujeron los datos a través del algoritmo de Kohonen, en la figura presentamos la imagen que se genera al visualizar los datos reducidos.

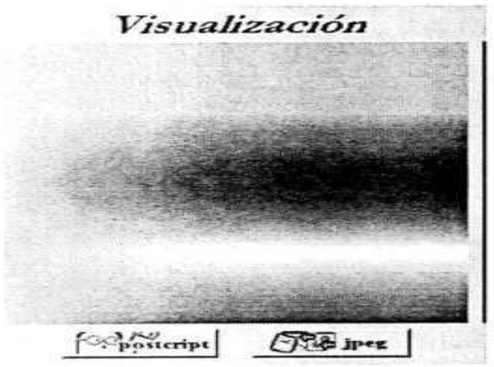


Figura 7.15. Imagen generada a través de la visualización de datos reducidos por Kohonen

- Visualización de datos reducidos a través de EM

En el caso de haber reducido a través del algoritmo EM, el proceso de visualización se selecciona tal y como se explicó en la figura 7.14. En la figura 7.16 se presenta la imagen generada a través de la visualización de datos reducidos por el algoritmo EM.

Figura 7.16. Imagen generada a través de la visualización de datos reducidos por EM

- Almacenamiento de imágenes visualizadas

El almacenamiento de imágenes se realiza a partir de presionar los botones que se encuentran debajo de estas imágenes precisamente. Se presentan dos botones, que determinan el tipo de dato gráfico, uno es jpeg y otro postcript. Enseguida presentamos dos ejemplos, el primero se presenta en la figura 7.17 y es el almacenamiento de la imagen generada a través de la visualización de los datos reducidos por Kohonen con el formato jpeg.

a) b)

Figura 7.17. Almacén de imagen generada por la visualización con el formato jpeg para los datos reducidos a través de Kohonen:
(a) Presionar el botón "jpeg". (b) Indicar el nombre del archivo.

El segundo ejemplo lo presenta la figura 7.18, es para el almacenamiento de la imagen generada a través de la visualización de los datos reducidos por EM con el formato postcript.

a) b)

Figura 7.18. Almacén de imagen generada por la visualización con el formato postcript para los datos reducidos a través de EM:
(a) Presionar el botón "postcript". (b) Indicar el nombre del archivo.

d. Análisis de datos Visualizados

Representamos al análisis de datos Visualizados con los diagramas: en el caso de uso "Análisis de datos G"; el modelado de flujo de procesos "Análisis de datos G"; diagrama de estereotipo "Análisis"; diagrama para el flujo de datos "Análisis de datos G". Estos se presentan en los capítulos anteriores, y son parte del análisis de datos.

El Análisis de datos Visualizados, se procesa a través de la transformada de Fourier, enseguida presentamos los procesos que se realizan a través de los diferentes eventos generados por el usuario para lograr este procedimiento. Tomamos como ejemplo el antecedente de reducir los datos con el algoritmo de Kohonen, y los procesos son: "Análisis con Fourier para los datos reducidos (Kohonen en el ejemplo)", "Almacenamiento de datos resultantes del análisis con Fourier" y "Almacenamiento de imágenes analizadas con Fourier".

- Análisis con Fourier para los datos reducidos (Kohonen en el ejemplo)

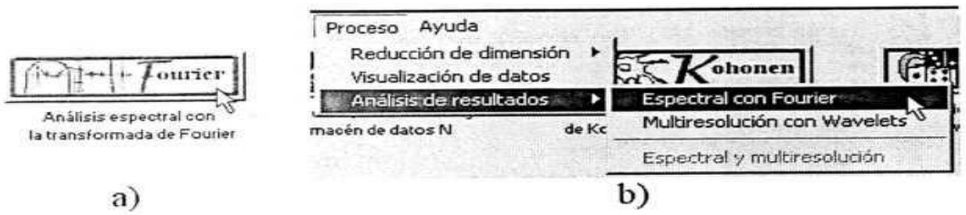


Figura 7.19. Opción para analizar los datos reducidos a través de Fourier:
(a) Presionar botón "Fourier". (b) Elección desde el menú principal.

Una vez reducidos los datos, pueden ser analizados a través de la trasformada de Fourier. Para iniciar el procedimiento del análisis de datos, se presiona el botón "Fourier" como lo muestra la figura 7.19(a), o bien, selecciona la opción "Espectral con Fourier" desde el menú principal como lo muestra la figura 7.19(b). Se genera una matriz de datos que se muestran en el área de resultados, como lo muestra la figura 7.20. También se genera la imagen que presentamos en la figura 7.21. Para dichos datos e imagen generados que presentamos en las figuras mencionadas, utilizamos como ejemplo la reducción de los datos N a través del algoritmo de Kohonen.

Figura 7.20. Datos resultantes del análisis con Fourier de los datos reducidos por Kohonen

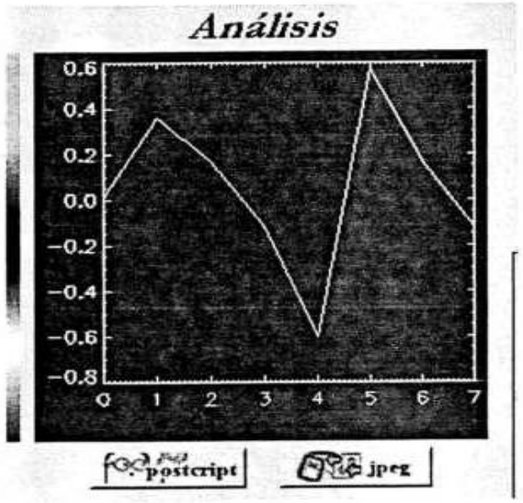


Figura 7.21. Imagen generada a través del análisis con Fourier de datos reducidos por Kohonen

- Almacenamiento de datos resultantes del análisis con Fourier

Los datos resultantes de la figura 7.20 pueden ser almacenados en el SA presionando el botón "grabar". Al presionar dicho botón, se abre la ventana que presentamos en la figura 7.22 con el fin de indicar el nombre con el que se grabará la matriz en el SA. En la figura 7.23 mostramos al archivo MiAnalisis.dtA, muestra la matriz y dimensiones de los datos resultantes del análisis con Fourier.

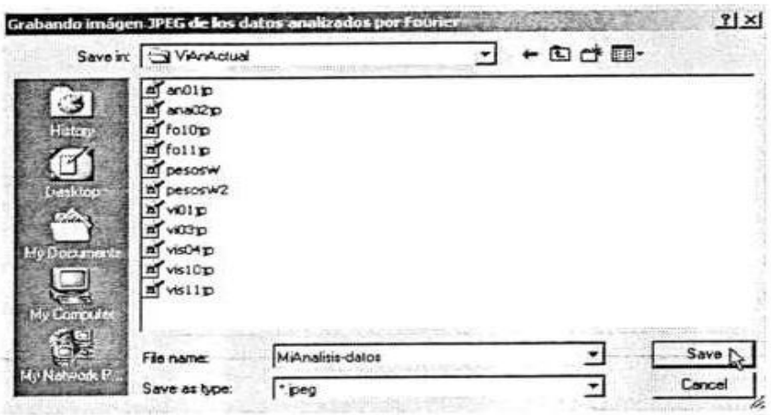


Figura 7.22. Almacén de MiAnalisis-datos.dtA

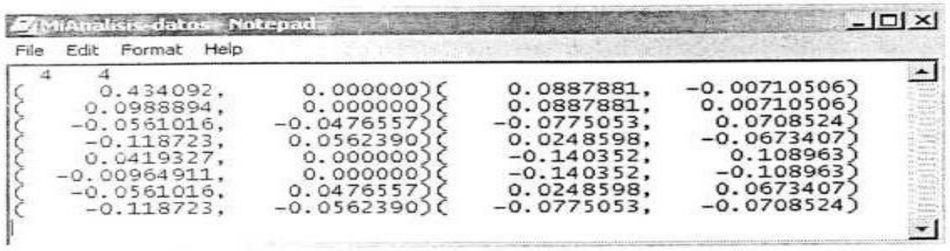


Figura 7.23. Archivo MiAnalisis-datos.dtA

- Almacenamiento de imágenes analizadas con Fourier

El almacenamiento de imágenes se realiza a partir de presionar los botones que se encuentran debajo de estas imágenes precisamente. Se presentan dos botones, que determinan el tipo de dato de salida de los resultados, uno es jpeg y otro postcript. En la figura 7.24 se presenta la ventana que se abre para solicitar el nombre del archivo con el cual se almacenará la imagen, esta ventana se abre al momento de presionar los botones mencionados que se muestran en la figura 7.21. Para la figura 7.24, se almacena como ejemplo, con el formato 'jpeg'.

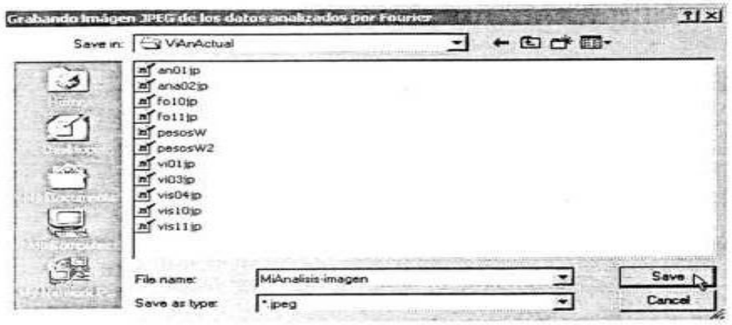


Figura 7.24. Almacén de MiAnalisis-imagen.jpeg

VII. 1. 2. Funcionalidad de los algoritmos de reducción

La funcionalidad de los algoritmos de reducción, se verificaron a través de pruebas con algunos datos específicos, generación de gráficas, seguimientos de los procesos y de los valores de nuestros datos. Enseguida presentaremos solamente una parte del proceso total de estas pruebas realizadas, con el fin de comprobar que la funcionalidad de los algoritmos de reducción es satisfactoria. Para consultar el significado de ciertos parámetros o características que especificamos en esta faceta (como por ejemplo la estructura de los pesos W), la información la presentamos en el capítulo V.

Los ejemplos en los que nos basamos, son datos que se generan a partir de una función sinoidal con 50 datos, esto es, manejamos una matriz de datos originales de 50 x 2 (cincuenta por dos). Las pruebas para estos ejemplos se desarrollaron en el espacio bidimensional (2D), por practicidad y con la finalidad de verificar de forma fácil y en parte gráfica que los resultados de los entrenamientos de los algoritmos de reducción de dimensión sean satisfactorios. Esta característica es exclusiva para el experimento que presentamos como ejemplo de la funcionalidad de los algoritmos. Las siguientes descripciones de las características de los algoritmos son las que se implementaron para el sistema ViAn, por lo tanto para cualquier matriz de datos originales y no solo para los ejemplos específicos de la función sinoidal descrita.

- **Entrenamiento con el algoritmo de Kohonen**

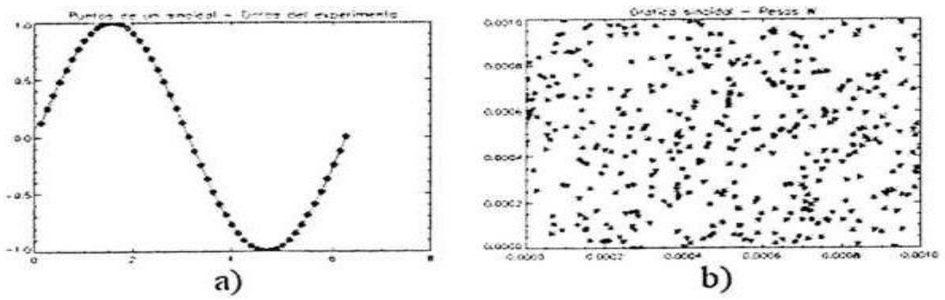
Las pruebas para en entrenamiento de datos a través del algoritmo de Kohonen, se realizaron con datos que pertenecen a una curva sinoidal en el espacio bidimensional, como se describió hace un momento. El algoritmo de Kohonen genera una "Matriz de Pesos" (de 20 x 20), el cual tiene el objetivo de mantener sus datos con las características topológicas de los datos originales. Estos pesos se encuentran en un espacio bidimensional, y cada peso se conforma de un conjunto de datos que depende directamente de la dimensión de los datos originales (recordemos que en nuestro caso son dos, por lo tanto, serían dos elementos por cada peso).

Se especificó un parámetro en nuestro algoritmo de Kohonen (NoVecin) que determina el número de neuronas vecinas (vecindarios) a lo largo del entrenamiento, este mismo auxilia a determinar los diferentes tamaños de los vecindarios: NoVecin + (NoVecin – 1). En un inicio, el primer tamaño del vecindario es de 5 elementos, en el siguiente cambio será de 3 elementos, en estos incluyendo al BMU (Unidad con Mayor Similitud) y por último de 1 elemento.

Para determinar cada cuando disminuye el vecindario se utiliza este mismo parámetro (NoVecin) y uno más itprom equivalente a 100, cada vez que se den itprom/NoVecin número de iteraciones, el vecindario disminuye. Esto significa que al llegar a las 100 iteraciones del entrenamiento, el vecindario desminuye al máximo, es decir a 1 elemento, el equivalente al BMU.

El parámetro itprom a su vez auxilia a determinar la condición de la disminución del parámetro de aprendizaje α, el cual en el inicio del entrenamiento es de valor 1.0. La condición de decremento del aprendizaje es: (0.001/a)^(1.0/itprom), lo que significa que al cabo de 100 iteraciones del entrenamiento, el aprendizaje es equivalente a 0.001.

Los datos originales que utilizamos para mostrar un ejemplo de la funcionalidad del algoritmo de Kohonen se presentan en la figura 7.25(a) que presenta la unión de 50 (puntos en la gráfica) datos originales del experimento. En la figura 7.25(b) presentamos los pesos en la primera iteración del entrenamiento. Lo que se espera al cabo de cierto número de iteraciones del algoritmo es que los pesos que se presentan en esta figura de forma dispersa, vayan auto-organizándose para caracterizar la forma de los datos originales.



a) b)

Figura 7.25. Espacio de los datos originales:

(a) Datos originales formando una gráfica sinoidal, es la unión de los puntos del espacio de datos.

(b) Pesos W dispersos en el espacio de datos, primera iteración del entrenamiento.

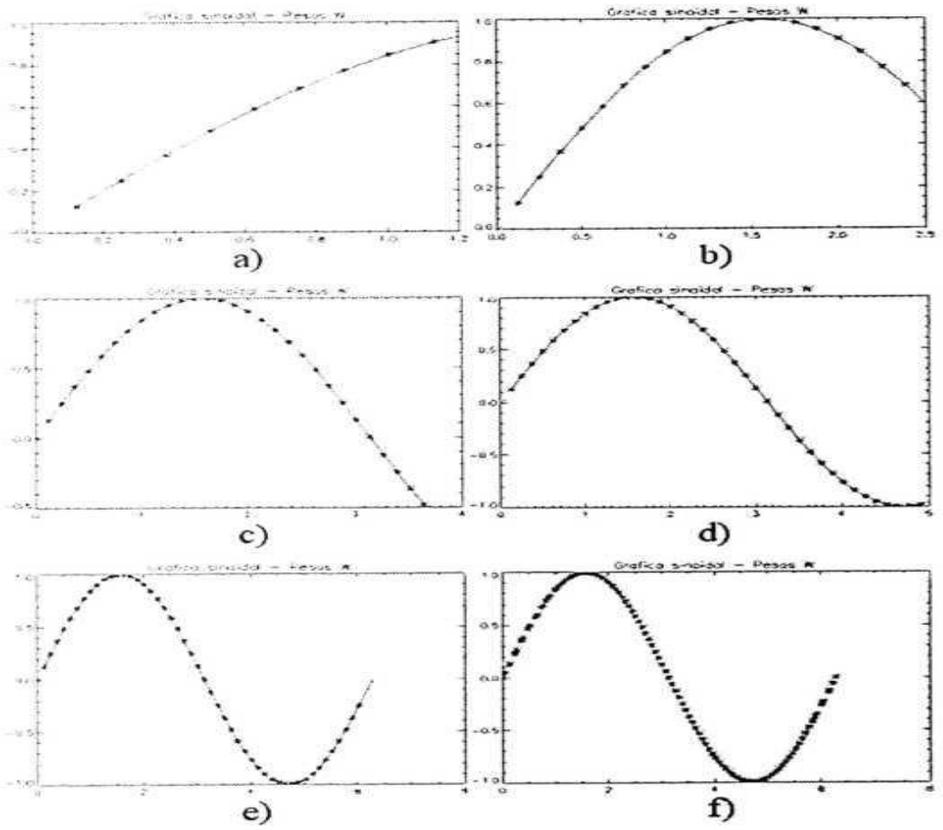


Figura 7.26. Gráfica de los pesos W en el espacio de datos:
(a) Pesos W en la iteración 10. (b) Pesos W en la iteración 20.
(c) Pesos W en la iteración 30. (d) Pesos W en la iteración 40.
(e) Pesos W en la iteración 50. (f) Pesos W en la iteración 60.

Las gráficas que presentamos anteriormente, son los pesos W en el espacio de datos. Las gráficas de estos puntos se lograron por el hecho de que nuestro espacio de datos es bidimensional. La figura 7.26 con sus incisos: (a), (b), (c), (d), (e) y (f) muestran los pesos W en las iteraciones del ciclo del algoritmo 10, 20, 30, 40, 50 y 60, respectivamente. Se muestra cómo los pesos W van asemejándose a los datos originales conforme avanzan las iteraciones.

En la figura 7.27 presentamos la gráfica de los pesos W resultantes al término del entrenamiento. En dicha figura presentamos los pesos W formando un tipo de Mapa de Kohonen. Es el contorno de nuestra matriz resultante del entrenamiento (de los pesos W). Los datos se agrupan en diferentes secciones respetando la topología que mantienen en el espacio de datos.

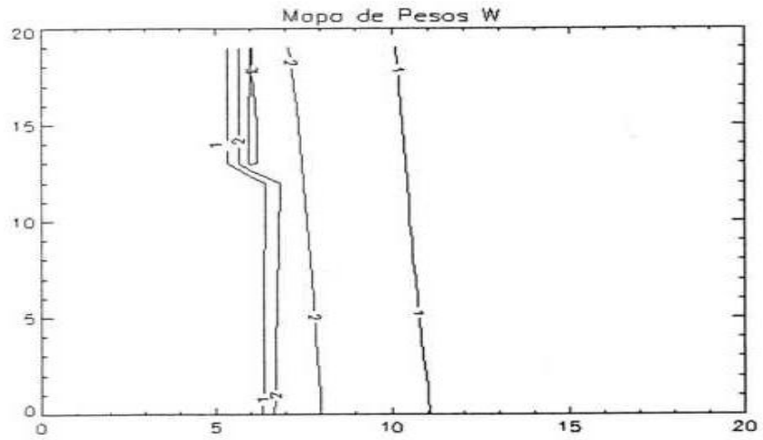


Figura 7.27. Imagen resultante de la matriz de pesos W.

- **Entrenamiento con el algoritmo EM**

El algoritmo para reducción de dimensión de datos EM asemeja de forma topológica a los datos originales de algún experimento en datos de un espacio bidimensional llamado "Variables Latentes" (o "Variables Ocultas"). El espacio de las Variables Latentes (VL) está representado por una matriz determinante (simétrica) de 5X5, donde los valores para cada dimensión son de -1 a 1. Existen unos parámetros del algoritmo EM (de nombre "centros") que auxilian en la formación de funciones bases. Estas funciones bases permiten visualizar claramente la semejanza entre estos parámetros y los datos originales.

Los centros se representan con una matriz determinante de 3X3, donde los valores para cada dimensión son de -1 a 1. Además denotan los centros de las gaussianas de las funciones bases no lineales. Las funciones bases se conforman de las funciones no lineales (son 9), funciones lineales (son 25, el número de VL) y una función constante equivalente a 1. El total de las funciones bases (35) se grafican en el espacio de datos para visualizar la semejanza entre ellos. Lo anterior se puede ejemplificar a través de una gráfica de los datos originales (seno en 2D), con el objetivo de que los centros se mantengan en el espacio bidimensional y se puedan visualizar.

Al momento de completar nuestra matriz de las funciones bases, antes de iniciar el ciclo de los pasos E y M, se inicializa W con la característica de que "la varianza de los datos originales" sea similar a "la varianza de las funciones bases multiplicadas por la matriz de pesos W".

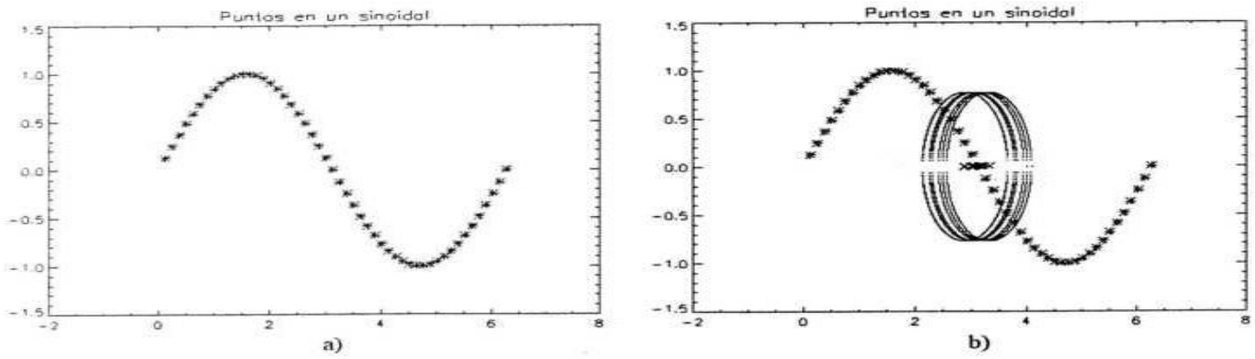


Figura 7.28. Gráfica sinoidal como datos originales:
(a) Iteración 0 del entrenamiento. (b) Iteración 1 del entrenamiento.

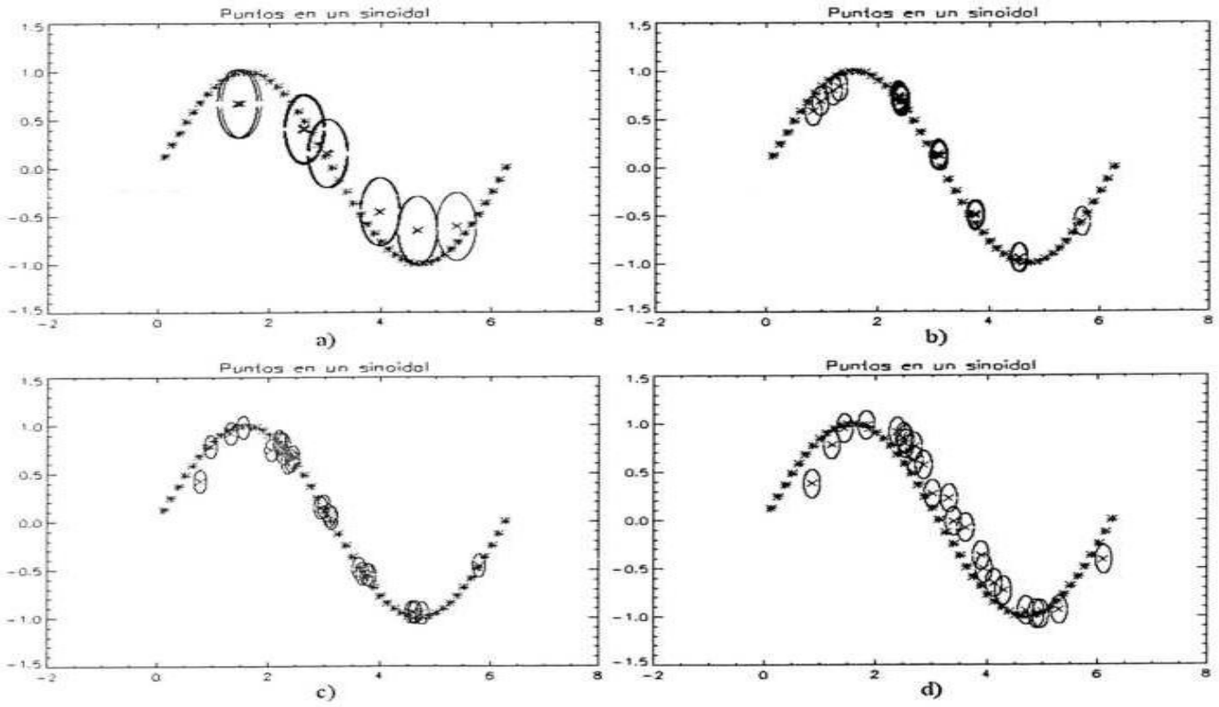


Figura 7.29. Gráfica seno en diferentes iteraciones:
(a) Iteración 2 del entrenamiento. (b) Iteración 3 del entrenamiento.
(c) Iteración 4 del entrenamiento. (d) Iteración 5 del entrenamiento.

La gráfica de la figura 7.28(a) muestra los datos del experimento. Estos datos se introducen al ciclo que da inicio al entrenamiento, en su primera iteración, las funciones bases se sitúan en un conjunto pequeño de la gráfica, esto lo presentamos en la figura 7.28(b). Conforme los ciclos se cumplen en el entrenamiento, las funciones bases van generándose en una forma semejando al sinoidal. Los centros de los círculos que se presentan en las gráficas se van acercando al conjunto de los datos que forman el seno. Los radios de dichos círculos son la varianza del parámetro Beta. Los resultados de las iteraciones durante el entrenamiento los presentamos en la figuras 7.29 las iteraciones desde dos a cinco.

Cuando el entrenamiento ha dejado de iterar, podemos estudiar nuestro espacio de variables latentes. Para visualizar nuestros datos, se requiere que las Variables Latentes (VL) sean de dos o tres dimensiones. Se podría graficar las responsabilidades en relación con las VL, pero serían gráficas ambiguas porque representarían solo una a una la relación de éstos. Lo que se requiere es conocer el comportamiento de todas las responsabilidades en relación con todas las VL. Para esto, el conjunto completo de las graficas mencionadas (es decir, de las relaciones entre las responsabilidades y las VL) se calcula a través del promedio de la multiplicación de ambas. A este dicho cálculo le llamamos Xmean o Xpromedio.

La gráfica de las Xpromedio en el espacio de las VL, que se presenta en la figura 7.30 se muestra en que forma las Xpromedio han sido afectadas por los las VL. Esta imagen es la que se presenta como el resultado de la reducción de dimensión de datos en nuestra aplicación ViAn.

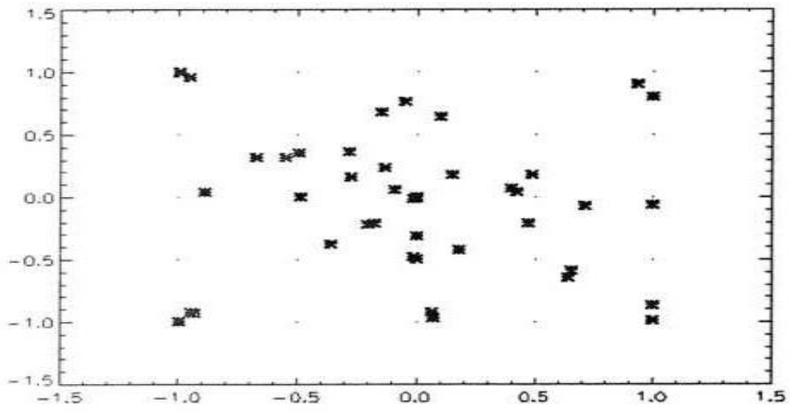


Figura 7.30. Gráfica de los pesos promedio en el espacio de las variables latentes

CAPITULO VIII.

Discusión

En una discusión, lo difícil no es defender
nuestra opinión, sino conocerla.
André Maurois

CAPITULO VIII.
Discusión

Presentamos una experiencia de desarrollo de un sistema, el cual se forma a través de un proceso de integración de técnicas de programación y modelados matemáticos para producir un software de calidad. Los componentes a integrar específicamente son: a) las cuatro diferentes estructuras con las que se conforman los algoritmos para lograr las tareas o procesos requeridas del sistema, dos para el proceso de reducción de dimensión de datos y dos más para el análisis de los datos visualizados; y b) el desarrollo del sistema en sí. Teniendo presente de qué trata este proceso de integración, podemos empezar a discutir varios aspectos de este proyecto de tesis en cuanto a la experiencia del desarrollo, los retos técnicos, los alcances y limitaciones del trabajo.

El primer punto que considero a discusión es respecto a los algoritmos que presentamos para el proceso de reducción de dimensión de datos, ambos algoritmos (Kohonen y EM) se encuentran ya estructurados dentro de diferentes campos de estudio. Kohonen en el área de Redes Neuronales Artificiales y EM con modelo de Variables Latentes en estadística y probabilidad. Lo que nosotros hicimos fue estudiar dichos algoritmos y, adaptarlos e integrarlos a nuestro problema de visualización. Mediante estos algoritmos se resuelve en gran medida el problema para el proceso de la reducción de dimensión de datos y de esta forma visualizarlos.

La utilidad y la comparación de estos dos algoritmos, es algo que puede explorarse para este proyecto. En esta primera versión de la aplicación, se desarrollaron los algoritmos sin este objetivo. Sin embargo, a través de nuestro trabajo logramos establecer los requerimientos, y se desarrollaron fases sólidas para que en un futuro ViAn estudie la utilidad y comparación de los algoritmos (Kohonen y EM).

Con respecto al análisis de datos, de igual forma son técnicas que no se desarrollaron para la elaboración de la tesis, sin embargo, se presentan en la misma aplicación para auxiliar al investigador en sus estudios, ofreciéndole a su vez la alternativa de mejorar su experimento al ver que no solo se puede limitar a visualizar sus datos, sino analizarlos con dos técnicas opcionales.

Sin embargo, a diferencia de los algoritmos para el proceso de reducción, estas técnicas se encuentran como módulos independientes en el IDL, y solo Fourier fue adoptado en el sistema. La

transformada de Wavelets se presenta en el IDL como todo un paquete de herramientas, el cual no se agrega como un módulo al sistema sino como una referencia. En trabajo futuro Wavelets puede modularse y adaptarse a las necesidades del sistema ViAn, por lo pronto, la encontraremos como referencia a un paquete muy amplio donde puede realizarse la tarea requerida.

Pasando a otro aspecto, existen muchos lenguajes visuales en los que se pueden desarrollar sistemas orientados a eventos. Uno de estos lenguajes es el que nos ofrece IDL, base de nuestra herramienta. IDL es una aplicación completa para realizar gráficas, objetos tridimensionales, animaciones, todo con un ambiente para el área de investigación, científico y tecnológico, que facilita al usuario su estudio. Como mencionaba, a través del lenguaje de programación propio del IDL, nos permite crear múltiples tareas, desde pequeños módulos hasta grandes sistemas (como al sistema ViAn), con las ventajas además de tener módulos desarrollados que pueden ser utilizados a nivel de programación. Es por estas ventajas que el sistema ViAn tiene como requerimiento: su desarrollo en este lenguaje, y no en un lenguaje convencional y poco familiarizado para los científicos.

Siguiendo con el lenguaje utilizado, durante la investigación y estudio del IDL, se observó que en el paso de información del módulo que declara la interfaz e inicializa los datos, el módulo que ejecuta las acciones dependiendo de los eventos generados, se podía enviar solamente un dato además de la estructura del evento EV que se envía por defecto (default). La estructura EV, no es creada por el programador, y se define en los "Antecedentes". Debido a la característica de solo permitir el envío de un dato, nuestro sistema se limitó en formar a una estructura de datos muy completa, en donde se contienen todos los datos necesarios para lograr el éxito en la función y funcionalidad de sistema durante la vida de éste. Entonces, al único dato que se puede enviar entre los módulos mencionados, lo convertimos en una estructura de datos, y la nombramos E.

Esta estructura de datos E fue detallada en el "Diseño del sistema". El hecho de poder manipular solamente una estructura de datos lo detectamos como una desventaja en el desarrollo del sistema debido a que se actualiza toda la estructura de datos al cumplirse las tareas del evento generado, además de que es una estructura muy compleja la cual requiere de muchos componentes para su explicación. Aún así, la parte operacional y manejo de datos son sencillos, al acceder los datos para actualizarlos o simplemente utilizarlos, únicamente se hace referencia a parte de la estructura de datos y el resto queda intacto. Un buen diseño de la estructura E y una correcta definición de las partes que la conforman fueron muy importantes. Así, se facilita la identificación de los datos para su modificación y actualización en los diferentes módulos del sistema.

Las partes más completas del sistema ViAn son su análisis y su diseño, además los requerimientos del sistema son prioridad en el desarrollo de este proyecto de tesis. Es por esto último que se han desarrollado características de la aplicación hasta cierto nivel, por ejemplo, el concepto de los "proyectos", y esto es lo que ahora presento como discusión. Los proyectos se diseñaron, pero una justificación de su ausencia en la implementación es que es un requerimiento innovador, lo cual significa que no afecta en ningún momento el cumplimiento de los objetivos del sistema. Esta parte si bien se diseñó, se omitió en el desarrollo para la elaboración de la tesis, para darle prioridad a la escritura de ésta y al buen funcionamiento y funcionalidad de la aplicación. En otros aspectos, el trabajo de esta tesis se delimitó en la fase de la implementación.

Queda claro entonces que se propone el desarrollo de estos aspectos para futuras versiones de la aplicación. Estas limitaciones para nuestra primera versión de la aplicación nos ayuda a obtener una buena calidad del trabajo durante un periodo de tiempo determinado. También es importante mencionar que los "proyectos" se agregaron al diseño, habiendo tenido la posibilidad de omitirlos en la implementación. Esto nos da una visión para ampliar el trabajo, y esto lo especificamos en el siguiente capítulo.

Enfoquémonos esta vez en el análisis y diseño de los algoritmos. Presentamos un capítulo para el "Análisis del sistema" que es el capítulo III, y tenemos otro para el "Diseño del sistema", correspondiente al capítulo IV, y en los dos se menciona que una parte de ellos lo conforma el capítulo V, "Análisis y diseño de los algoritmos". Realizamos el capítulo V, debido a que los algoritmos se desarrollan como una parte a integrar en el sistema. Estos algoritmos tienen una estructura, características y complejidad propias, por ende, necesitan de un análisis con una visión matemática (y algorítmica) diferente entre cada una de las técnicas. En resumen: en cada uno de los algoritmos presentamos su desarrollo desde un punto de vista específico al área de investigación en la que pertenecen. Y dentro del sistema ViAn, se visualizan como componentes que deben integrarse al análisis y al diseño del sistema. Las partes a integrar tienen una misma jerarquía, es decir, ninguna es más importante que otra.

Refiriéndonos a la ingeniería del software: Los modelados que conforman el análisis y el diseño del sistema son basadas en modelados pre-establecidos, algunos se cumplieron fielmente a la metodología estructurada que mencionamos, en otros únicamente fueron bases para notaciones adecuadas al sistema, y en otros modelados adoptamos notaciones en nuestro contexto. Este fenómeno ocurrió ya que las técnicas para desarrollar el análisis y el diseño del sistema no son rígidas, además fueron adoptadas en nuestro caso, esto debido a que el entendimiento en los modelados debe adaptarse al manejo de procesos y de eventos, presentándose en una forma clara y precisa de acuerdo a las características de ViAn.

Hemos tenido muy claro desde un principio las metodologías utilizadas para realizar el sistema, con esto me refiero a las diferentes perspectivas del análisis y diseño junto con los diagramas que los conforman. Además, hemos tenido siempre presente, tanto el cliente como la desarrolladora, al método que utilizaríamos, un método pre-establecido y muy óptimo llamado "El método espiral". Se tuvo varias versiones del sistema y de la documentación, donde se cumplían con los requerimientos especificados en cada versión, o bien, con la mayoría de éstos. Se realizaba una evaluación cada determinado tiempo cumpliendo un ciclo, donde se observaban las mejoras que se debían hacer, además de corregir malos entendidos entre el cliente y la desarrolladora, y confusiones en cualquiera de las dos partes.

Un ciclo del espiral se podía cumplir exitosamente con la intención de solo hacer una parte funcional. Ejemplifico un ciclo del espiral, para su mejor explicación, en la aplicación definimos perfectamente la interfaz gráfica para que funcionara la entrada de los datos junto con su captura y almacén, en un ciclo seguido, se modificaría la interfaz para que también funcione correctamente y despliegue los datos e imágenes resultantes de la reducción de dimensiones y el análisis de datos.

Otro ejemplo del proceso de desarrollo es la documentación de la tesis. En el diseño del sistema se realizaron diagramas para el flujo de datos, y en un segundo ciclo se definieron los elementos E (Estructura de datos) y SA (Sistema de Archivos) para formar la estructura del sistema y una segunda versión del flujo de datos. No se llevo a cabo un registro de cómo se aplicaba el método de espiral, ya que además de una definición meticulosa, no sería trascendente para el desarrollo del sistema. Eso implica gastos innecesarios de recursos como el tiempo, o diagramas sin utilidad posterior.

CAPITULO IX.

Conclusiones y trabajo futuro

He llegado a la conclusión, después de
muchos años de tristes experiencias, de que
uno no puede llegar a ninguna conclusión.
Vita Sackville-West

La recompensa del trabajo bien hecho es la
oportunidad de hacer más trabajo bien hecho.
Jonas Edward Salk

CAPITULO IX.
Conclusiones y trabajo futuro

Conclusiones

ViAn es una aplicación científica computacional con la cual el investigador introduce datos N-dimensionales, correspondientes a su experimento, reduce las dimensiones de sus datos, obtiene imágenes resultantes de la visualización de los datos reducidos, y analiza los datos visualizados. Todos estos procesos integrados en una sola herramienta. Así, ViAn ayuda al investigador a realizar operaciones con datos N-dimensionales teniendo mayor control y comprensión sobre los datos y los fenómenos producidos en su experimento.

En esta aplicación se presentan dos técnicas para la reducción de dimensión de datos, las cuales pueden ser comparativas, al igual que las técnicas que se presentan para el análisis de datos. Así, obtienen resultados comparativos y conclusiones más detalladas de la estructura subyacente de los datos. Los algoritmos de Kohonen y EM resultaron óptimos para cubrir nuestras necesidades acerca de reducir la dimensión de datos y así lograr la visualización de éstos. Con esto, consideramos que ViAn cumple con las expectativas originales, las cuales se especificaron en nuestros requerimientos.

Se cumplieron con los objetivos del trabajo en esta primera etapa del ciclo de vida del aplicación y se cubrieron los requerimientos del usuario. Las funciones de la aplicación están acorde al análisis del sistema y al diseño del sistema desarrollado, además de que la funcionalidad del sistema es clara y precisa. Desarrollamos todos los modelados para tener una base sólida y una documentación amplia y entendible de lo que hace la aplicación y de cómo se hace, para esto, nuestra base principal fue la ingeniería del software. Es debido a esta base que existe una buena conexión desde las fases de los requerimientos y los casos de uso, hasta las fases del diseño de interfaz y la implementación del sistema. Así, el sistema puede ser comprendido desde cualquier perspectiva presentada en el análisis, diseño e implementación.

Con respecto al análisis y diseño de los algoritmos, se obtuvo una explicación del desarrollo de las técnicas para la reducción de dimensión, sus algoritmos y la implementación de éstos. Para las técnicas de análisis de datos se obtuvieron módulos que cumplieron con procesos determinados, y presentamos una definición de las transformadas con las cuales se logra el análisis de los datos.

El lenguaje de programación IDL, es muy óptimo para desarrollar aplicaciones científicas con un enfoque orientado a eventos. Además, fue muy efectivo el método espiral que aplicamos durante el desarrollo de nuestro trabajo.

Como sucede en cualquier proyecto, existen partes que representan propuestas del sistema ViAn para su desarrollo en siguientes versiones de la aplicación. Sin embargo, estas partes se encuentran bien representadas y detalladas en los modelados de una forma muy clara para que se concluya de una forma muy practica en dichas futuras versiones.

Además que la funcionalidad efectiva en el análisis del sistema y diseño del sistema permiten la visión de mejorar el sistema y/o de ampliar sus procesos. Enseguida presentamos algunos aspectos como trabajo a futuro para esta aplicación.

Trabajo futuro

El sistema ViAn ofrece espacio para su ampliación y mejora como cualquier producto de software. Se estructuraron un análisis del sistema y un diseño del sistema muy completos, cumpliendo con los objetivos descritos para la aplicación. Pero además, surgieron requerimientos esperados y requerimientos innovadores, de los cuales se cubrieron todos los esperados y algunos innovadores, y quedo sin implementarse específicamente a la aplicación, un requerimiento innovador, se trata de los llamados "proyectos". Es entonces que los proyectos representan trabajo para continuar desarrollando el sistema ViAn.

Contamos con los algoritmos para la reducción y para el análisis. Dichos algoritmos son una parte del sistema en la cual puede profundizarse mucho, por supuesto, tomando como base lo ya presentado en la tesis. Pueden realizarse varias y diferentes pruebas a los algoritmos. Pueden optimizarse y formular un enfoque comparativo entre los dos algoritmos que realizan la misma tarea. Pueden adecuarse los parámetros de afinación para una mejor funcionalidad. Además se puede estudiar qué otra información necesaria puede darnos los algoritmos y de qué forma el sistema podrá ofrecerla.

Otra forma de ampliar aún más la utilidad de la aplicación, extendiendo y modificando el análisis y diseño del sistema, es a través de lo siguiente: El sistema ViAn reduce las dimensiones de una matriz que consta un conjunto de datos de algún experimento, experimento el cual tiene definidos sus elementos. También analiza un solo conjunto de datos.

Lo novedoso estaría si introducimos el factor tiempo a nuestro sistema, y así podremos entonces ver cambios entre un conjunto de datos y otro conjunto; ya sean analizados o reducidos solamente, es entonces que el sistema ViAn como un trabajo futuro, pueda reducir las dimensiones a un grupo de matrices, y analizar un conjunto de imágenes y dar como resultado los cambios sufridos a través del tiempo.

Aunque el lenguaje IDL se adapta a cualquier plataforma, existen detalles que hacen que las aplicaciones tengan características propias por el sistema operativo en el que se ejecuta. El sistema ViAn se adaptó para la plataforma Windows, y un trabajo futuro sería adaptar al sistema ViAn para las demás plataformas en las que IDL es soportado. Para lograr estos cambios, se requiere de la herramienta necesaria para el soporte de todas las plataformas y de un considerable tiempo de pruebas.

APENDICE A.

Análisis

La ciencia es el conocimiento organizado.

Herber Spencer

APENDICE A.
Análisis

Este apéndice es un complemento del capítulo III, se encuentran diagramas y elementos que forman parte del análisis del sistema. Por lo tanto, las referencias a las siguientes secciones las encuentra en dicho capítulo.

1. Requerimientos

El siguiente es un cuestionario que se aplicó para determinar los requerimientos (de usuario y de sistema) para el desarrollo de la aplicación, contestado por el Dr. Fernando Rojas Íñiguez (cliente de la aplicación y director de la tesis) y por la tesista Beatriz Benítez Mayo (desarrolladora de la aplicación):

General

1. ¿Quiénes serán los usuarios de la aplicación?

2. ¿Para qué tarea específica se necesita la aplicación?

3. ¿Qué documentación necesita como usuario de la aplicación?
   - Manual de usuario: _____ [SI/NO]
   - Manual de instalación: _____ [SI/NO]
   - Ejemplos de experimentos: _____ [SI/NO]
   - Otro(s): _____.

4. ¿Necesita ejemplos de experimentos en el menú de la aplicación?, _____
   ¿Cuántos y para qué área(s)?

5. ¿Desea conservar el experimento completo (datos de entrada, reducidos y analizados)?

6. ¿En qué lenguaje desea la implementación de la aplicación?

7. ¿Para alguna(s) plataforma(s) en particular, cuál(es)?

8. ¿Es necesario un sistema de seguridad?

9. ¿Desea comparar técnicas durante el desarrollo de la aplicación?

Entrada

10. ¿En qué forma necesita la captura de datos (matrices, imágenes, etc.)?

11. ¿Es necesario capturar serie de datos (cada serie para un tiempo t del fenómeno)?

12. ¿Cuál es el mínimo de serie de datos que se desea aceptar en el sistema?

13. ¿Cuál es el máximo de serie de datos que se desea aceptar en el sistema?

Proceso

14. ¿Qué técnica(s) específica(s) requiere para la reducción de datos?

15. ¿Qué técnica(s) específica(s) requiere para el análisis de los datos?

16. La reducción de los datos, ¿Para cuántas dimensiones desea que se Visualice?

17. ¿La Visualización de los datos la desea observar de forma dinámica, estática o tener la opción de ambas?

18. ¿Desea que las opciones para la reducción de los datos y para el análisis, [1] se le muestre en un momento determinado (como parte del diseño de pantallas en la aplicación) o [2] elegirlo cuando usted lo solicite (tener lo en el menú de opciones con las técnicas por defecto)?

19. ¿Qué información desea ver en el proceso de Visualización?

Salida

20. ¿Qué información desea ver como resultado de la reducción de los datos?

21. ¿En qué forma (matrices de datos, imágenes)?

22. ¿Qué información desea ver como resultado del análisis de los datos?

23. ¿En qué forma (matrices de datos, imágenes)?

24. ¿Qué información en general requiere cómo resultado de su experimento para su interpretación?

2. *Casos de uso*

Los casos de uso para la Reducción de dimensión de datos N, Visualización de datos M y Análisis de datos G, son parte de los casos de uso del nivel más bajo de nuestro Análisis del Sistema. Cada uno de estos tres diagramas representa las tareas principales de nuestro sistema ViAn.

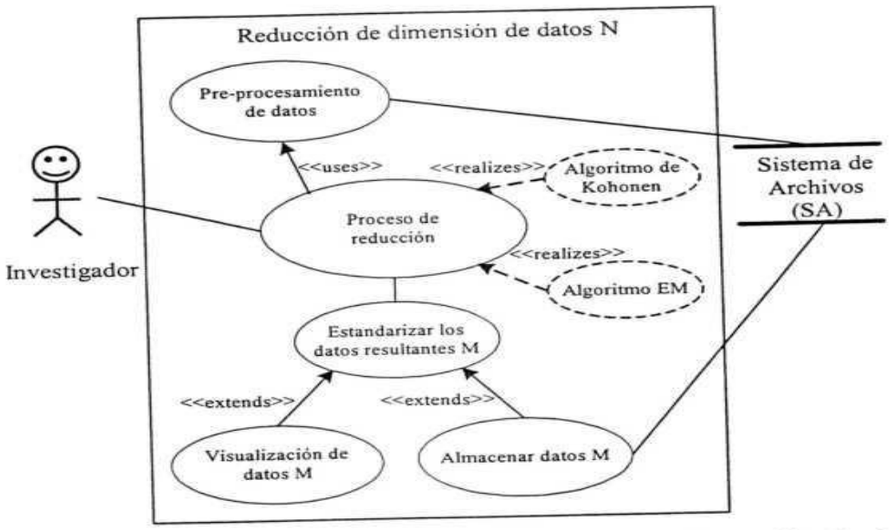


Figura A1. Diagrama de caso de uso de nivel 2: "***Reducción de dimensión de datos N***"

Nombre: Reducción de dimensión de datos N.

Actores: Investigador (usuario), SA (Sistema de Archivos).

Propósito: Reducir los datos N-dimensionales del experimento con el algoritmo de Kohonen y/o con el algoritmo EM.

Descripción: El investigador indica al sistema la opción de reducir los datos de su experimento, el usuario introduce los datos que se encuentran en el espacio N-dimensional a un formato de entrada que el sistema le presenta, se estandarizan y se almacenan en el Sistema de Archivos (SA), así es como se pre-procesan los datos. Otra opción para el pre-procesamiento de datos es leer directamente del SA al archivo que contiene los datos originales del experimento N previamente almacenados. El usuario indica que desea continuar con el proceso de reducción, se le presentan dos técnicas para la reducción de dimensión de los datos: 1. algoritmo de Kohonen y 2. el algoritmo EM, de estas dos técnicas, el usuario debe elegir una de las dos o si así lo desea, puede elegir a ambos algoritmos. Los datos resultantes del proceso de reducción se estandarizan para obtenerlos en un formato M. El sistema le presenta al usuario la opción de almacenar al

SA los datos reducidos M y la opción de visualizar los datos reducidos M. El usuario tiene la opción de almacenar los datos M al SA y después visualizarlos, o puede visualizarlos y después almacenarlos.

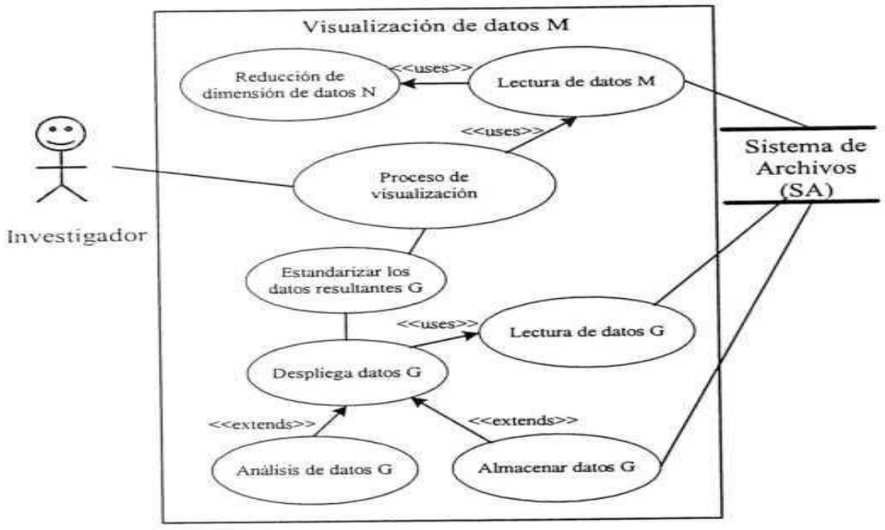


Figura A2. Diagrama de caso de uso de nivel 2: "***Visualización de datos M***"

Nombre: Visualización de datos M.

Actores: Investigador (usuario), SA (Sistema de Archivos).

Propósito: Visualizar los datos reducidos M.

Descripción: El investigador indica al sistema la opción de visualizar los datos reducidos M. El sistema procede a la lectura de los datos M, los cuales se obtienen del resultado del proceso de reducción de datos N, si el usuario desea leer los datos M de un archivo, el sistema le permite la búsqueda del archivo dentro del SA donde se encuentran los datos reducidos M almacenados. Con los datos M listos, se envían al proceso de visualización, los datos resultantes se estandarizan en un formato G y se envían al despliegue en pantalla estos datos G que representan a los datos del experimento ya visualizados. Otra opción para el despliegue de datos, es leer directamente del SA al archivo que contiene los datos visualizados G previamente almacenados. El sistema le presenta al usuario la opción de almacenar al SA los datos visualizados G y la opción de analizar los datos G. El usuario tiene la opción de almacenar los datos G al SA y después analizarlos, o puede primero analizar los datos y después almacenar al SA los datos G visualizados.

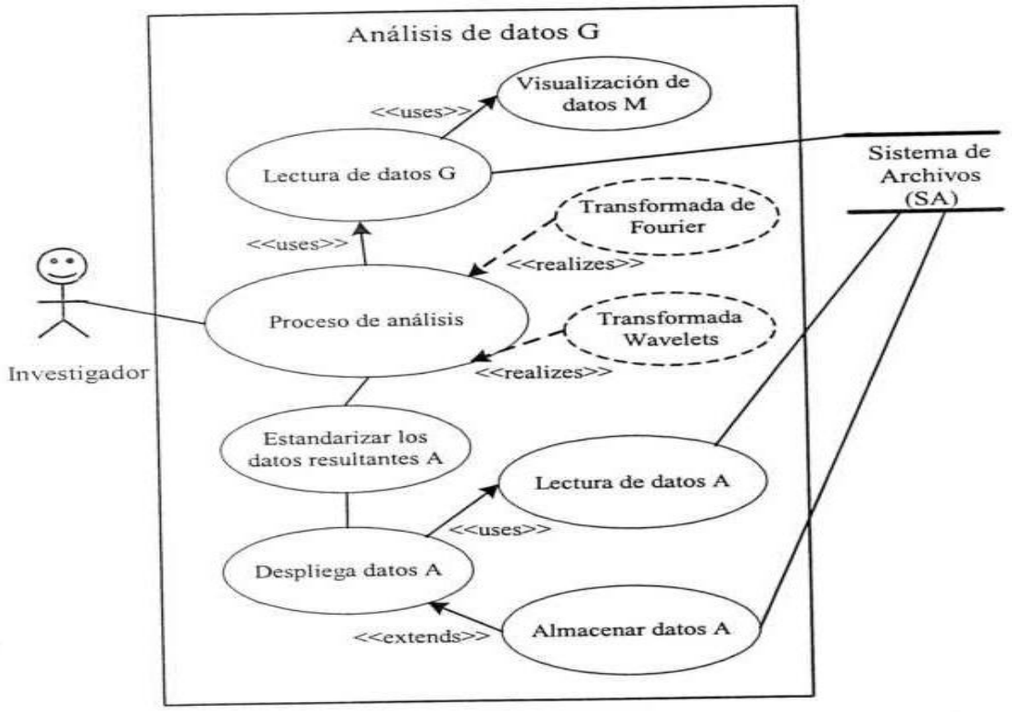


Figura A3. Diagrama de caso de uso de nivel 2: "***Análisis de datos G***"

Nombre: Análisis de datos G.

Actores: Investigador (usuario), SA (Sistema de Archivos)

Propósito: Analizar los datos G con multirresolución y/o Fourier.

Descripción: El investigador indica al sistema la opción de analizar los datos visualizados G. El sistema procede a la lectura de los datos G, los cuales se obtienen del resultado del proceso de visualización de datos M, si el usuario desea leer los datos G visualizados de un archivo, el sistema le permite la búsqueda del archivo dentro del SA donde se encuentran los datos visualizados G almacenados. Con los datos G listos, se envían al proceso de análisis, donde se aplica uno de los dos métodos de análisis (o ambos, si es que así lo desea el usuario): 1. es con la transformada de Fourier y 2. es con la transformada Wavelets (ondeletas). Los datos resultantes se estandarizan en un formato A y se envían al despliegue en pantalla, los datos A representan al fenómeno y sus cambios que conforman el experimento del investigador. Otra opción para el despliegue de datos, es leer directamente del SA al archivo que contiene los datos analizados A previamente almacenados. El sistema le presenta al usuario la opción de almacenar al SA los datos analizados A.

3. *Modelado de Procesos*

Se puede observar una clara relación entre los casos de uso anteriores y los modelados siguientes. La diferencia está en que los casos de uso son el análisis del sistema desde el punto de vista del usuario, y el modelado de procesos es el análisis del mismo sistema desde el punto de vista del sistema mismo, a través de procesos. Algún caso de uso se conformará de uno o más procesos, otros casos serán solamente salidas o entradas de procesos.

En el caso de la figura A4, el subproceso "pre-procesamiento" se forma por otros subprocesos más específicos, éstos se presentan en la figura A5. El modelado de flujo de procesos para la Reducción de dimensión de datos N que presentamos en la figura A4 cuenta con dos entradas al proceso, una es para el pre-procesamiento de datos N, y la otra es para la inicialización de los datos para la reducción, el pre-procesamiento permite la salida del proceso o continuar con la inicialización de los datos. Una vez inicializados, tenemos dos opciones para la reducción, uno con el algoritmo del Kohonen y el otro con el algoritmo EM, al término de cualquiera de ellos se procede a estandarizar los datos resultantes y salimos de proceso, o bien, almacenamos los datos M y posteriormente salimos del proceso.

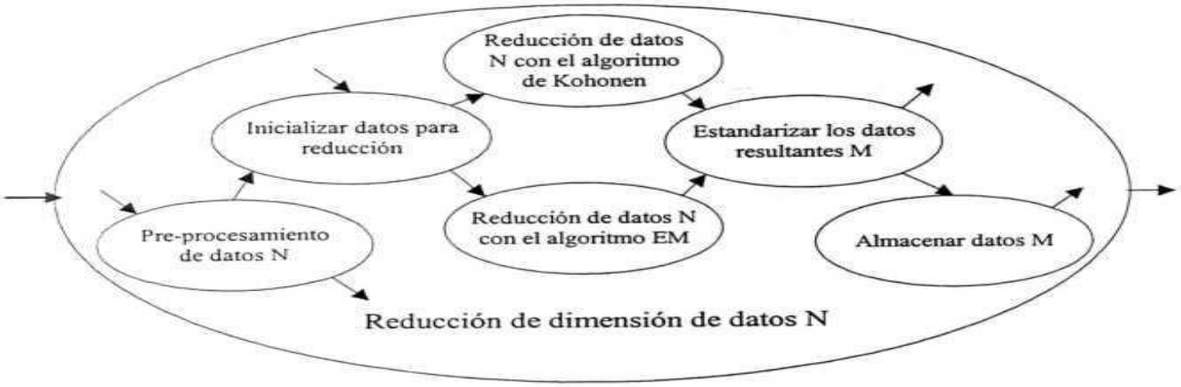


Figura A4. Modelado de flujo de procesos: "**Reducción de dimensión de datos N**"

Ahora presentamos el subproceso de la figura A4, el modelado de flujo de procesos para el Pre-procesamiento de datos N en la figura A5, como un proceso donde tiene dos opciones para la entrada. Si se desea la lectura de los datos N, procede y se sale del proceso, en la otra opción, tenemos que mostrar un formato de entrada para después introducir los datos del experimento, procedemos a estandarizar los datos N para su reducción, se almacenan los datos N para seguir con la salida del proceso.

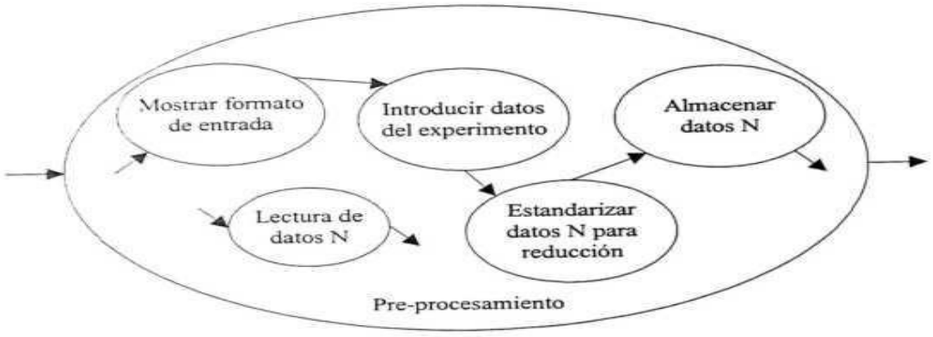


Figura A5. Sub-proceso de la figura A4, Modelado de flujo de procesos: **"Pre-procesamiento de datos N"**

En la figura A6 presentamos el modelado de flujo de procesos para el Análisis de datos G con tres opciones para la entrada del proceso, en una mostramos la lectura de los datos M, que procede a la inicialización de los datos para el análisis, siendo este mismo entrada para el proceso, después de inicializados los datos, contamos con dos formas de análisis de datos, uno es con la transformada de Fourier y otro es con la trasformada Wavelets (ondeletas), al término de cualquiera de ellos, se estandarizan los datos resultantes A, se procede al despliegue de los datos y de ahí contamos con dos opciones, una es la salida del proceso y otra es almacenar los datos A para entonces seguir con la salida del proceso. En la tercera entrada al proceso, realizamos la lectura de los datos A, procedemos con el despliegue de los datos que tiene dos opciones a su término como lo mencionamos. Haré una pausa para hacer notar que en este modelado de procesos pueden leerse datos M (datos reducidos) y no datos G (datos visualizados), ya que los datos M pueden ser procesados nuevamente y los datos G por ser imagen no pueden procesarse, pero todo esto lo estará mejor explicado en el capítulo del diseño del sistema.

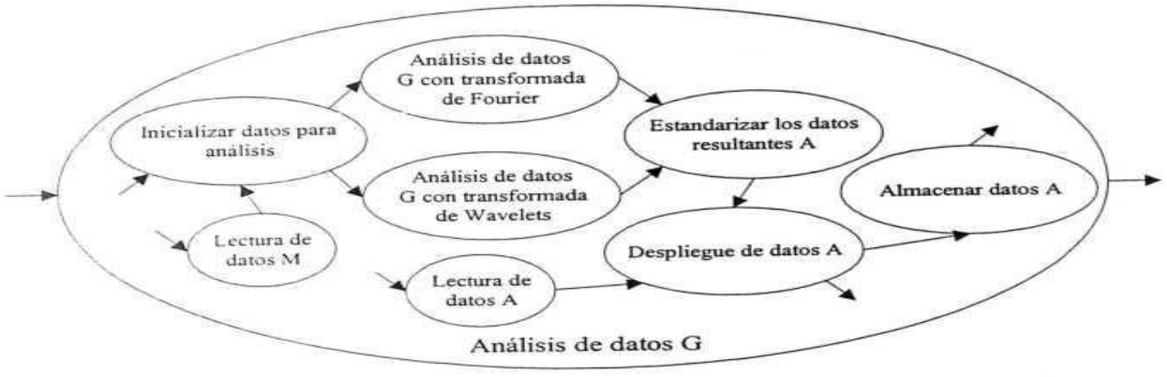


Figura A6. Modelado de flujo de procesos: **"Análisis de datos G"**

4. *Estereotipos*

Para los estereotipos contamos con los diagramas de secuencia. Los siguientes tres diagramas son parte del diagrama de estereotipos que puede observarse en la figura 3.8 en el capítulo III (Análisis del sistema). Los diagramas corresponden al pre-procesamiento de datos, a la Reducción de datos y al Análisis de éstos. Cada uno viene con su explicación, pero en general sucede que:

En la figura A7 del diagrama de secuencias para el pre-procesamiento, el usuario solicita el formato para la captura de los datos de su experimento, introduce los datos N y se estandarizan, después tenemos dos opciones, una es para enviarlos al proceso de reducción de los datos y otra es para almacenarlos en el Sistema de Archivos (SA), y así se finaliza el proceso. Si bien, después de almacenar los datos el usuario desea enviar estos mismos al proceso de reducción, también cuenta con esa vía para hacerlo.

En la figura A8, tenemos al diagrama de secuencia para la Reducción de datos, donde el usuario busca un archivo de datos N en el SA, el sistema prepara los datos para inicializarlos y los envía al proceso de reducción, el resultado se estandariza en datos M, y se envían estos nuevos datos al proceso de Visualización de los datos. El usuario cuenta con la opción de enviarlos al SA para almacenarlos, antes de enviarlos directamente al proceso de Visualización. También puede almacenar los datos y finalizar al proceso.

Por último presentamos en la figura A9 al diagrama de secuencia para el análisis de los datos visualizados (Recordemos que el diagrama de secuencia para la Visualización de los datos se encuentra en la figura 3.9 ubicada en el capítulo III – Análisis del sistema). En este diagrama observamos que el usuario busca un archivo de datos G en el SA, se preparan los datos para su análisis, se envían los datos al proceso de análisis, y se envían los resultados del análisis al proceso de desplegar los datos, de ahí se envían al SA como datos A para su almacén y se termina el proceso. Ahora, si en lugar de realizar la búsqueda de datos G, se buscan datos A, se envían al proceso de despliegue de datos y se finaliza el proceso.

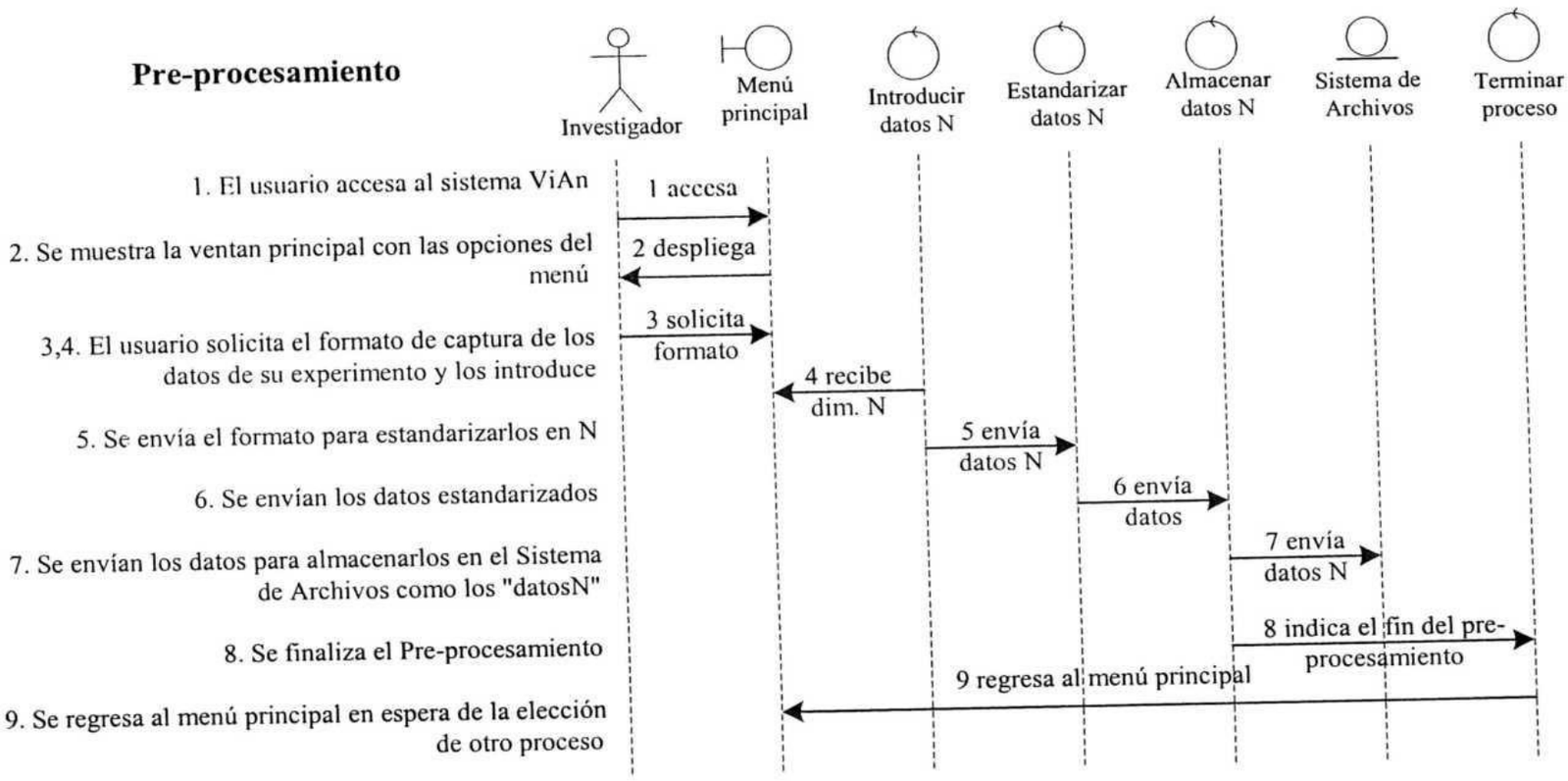


Figura A7. El investigador elige los procesos para *pre-procesar* los datos N (originales)

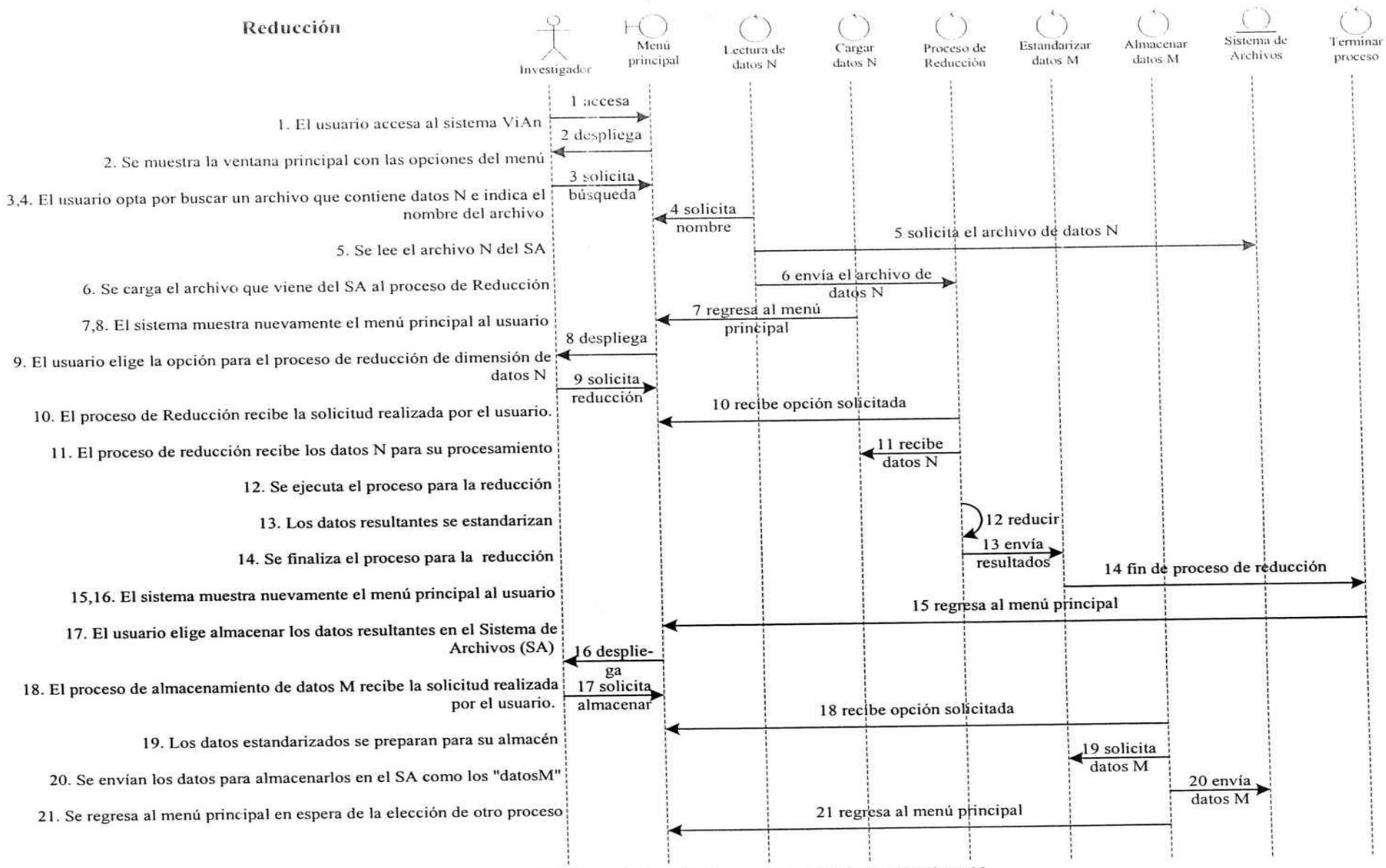


Figura A8. El investigador elige los procesos para*Reducir* los datos N

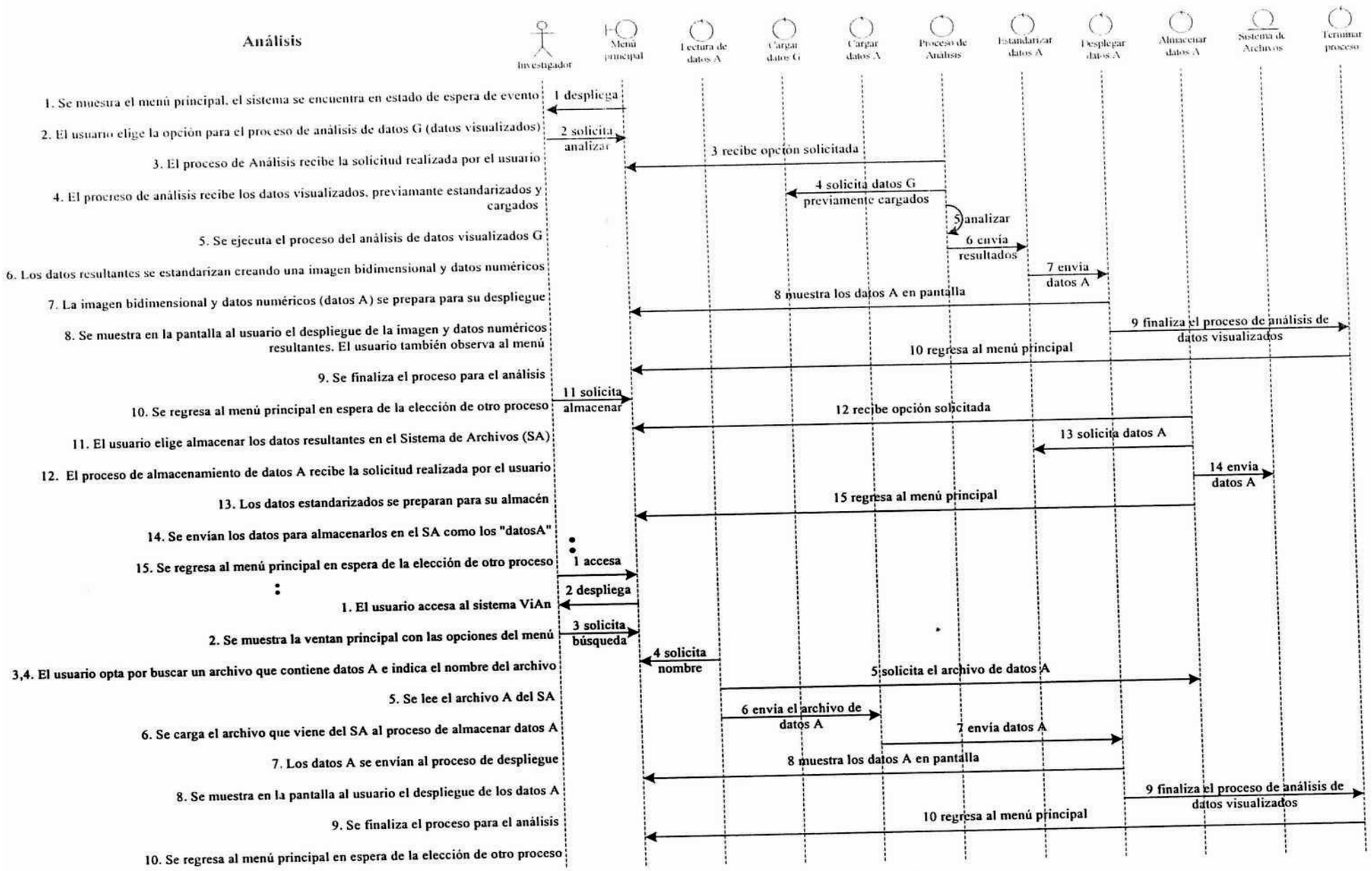


Figura A9. El investigador elige losprocesos para *Analizar* los datos visualizados

5. Requerimientos de interfaz

Las siguientes pantallas complementan el requerimiento de interfaz, presentado en el análisis del sistema. Con esto nos basaremos para la primera versión del diseño del sistema.

Figura A10. Pantalla para la reducción de datos

En la figura A10 se presenta la pantalla para la reducción de datos que muestra la matriz resultante de la reducción de los datos originales en el área correspondiente, y en el área del estatus indica el estado "reduciendo".

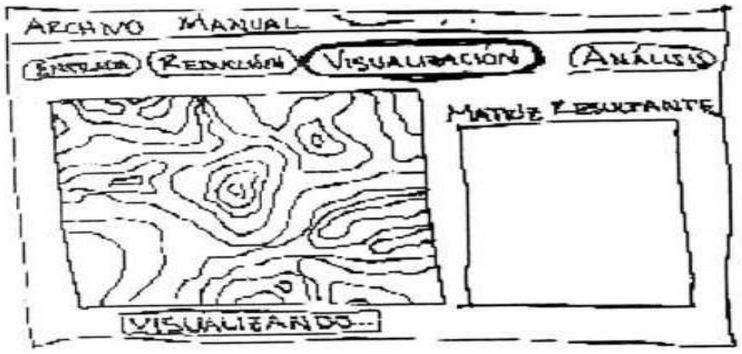


Figura A11. Pantalla para la visualización de datos reducidos

En la figura A11 se muestra la pantalla para la visualización de datos reducidos que presenta la imagen generada por los datos reducidos en el área correspondiente, y en el área del estatus indica el estado "visualizando".

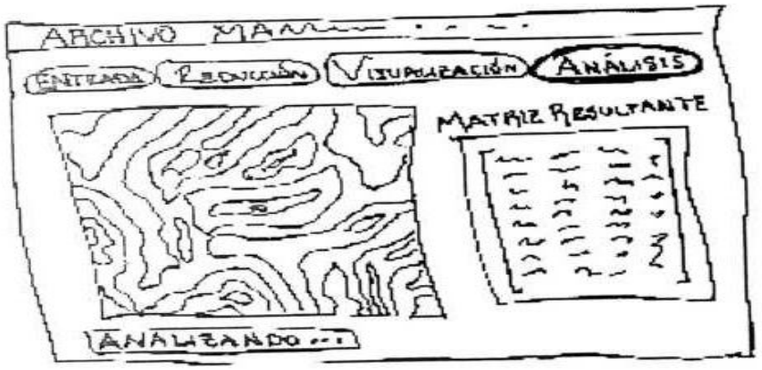


Figura A12. Pantalla para el análisis de datos visualizados

En la figura A12 se presenta la pantalla para el análisis de datos visualizados que muestra la imagen generada por los datos reducidos y la matriz resultante, ambos en su área correspondiente, y en el área del estatus indica el estado "analizando".

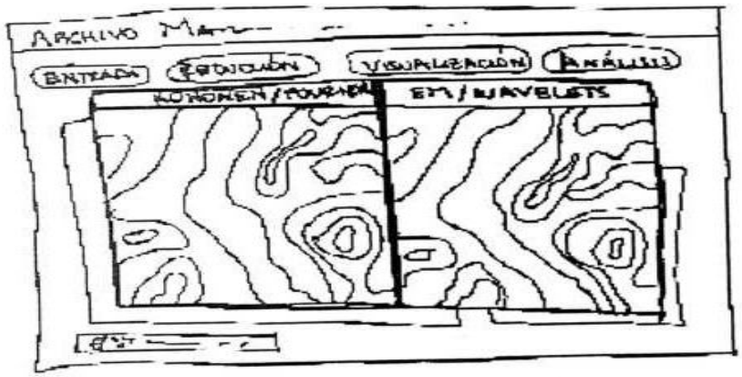


Figura A13. Pantalla para la opción ambos, indica el proceso con dos técnicas (para la reducción o para el análisis)

En la figura A13 se muestra el caso de que el usuario decida procesar la reducción o el análisis con ambas técnicas según el proceso, ya sea el algoritmo de Kohonen y el algoritmo EM para la reducción o la transformada de Fourier y la transformada Wavelets (ondeletas) para el análisis, se abrirá una ventana donde habrá dos áreas de dibujo, en las que se desplegarán las imágenes resultantes del proceso elegido, en la primera (izquierda) se observará la imagen resultante de Kohonen o de Fourier y en la segunda (derecha) la imagen generada por EM o por Wavelets, según el proceso solicitado por el usuario. En esta ventana se refleja nuestro requerimiento **1e**.

APENDICE B.

Diseño

La verdadera grandeza de la ciencia
acaba valorándose por su utilidad.
Gregorio Marañon

APENDICE B.
Diseño

Este apéndice es un complemento para detallar lo presentado en el capítulo IV: diagramas y elementos que forman parte del diseño del sistema. Por lo tanto, las referencias a las siguientes secciones las encuentra en dicho capítulo.

1. Estructura del sistema

Presentamos en la tabla BI con los nombres de los módulos en la columna 1, la explicación de cada módulo (segunda columna), así entenderemos su significado y funcionalidad, en la tercera columna se indica si el módulo es un procedimiento o una función. Y en la cuarta columna mostramos si es que se trata de un módulo que genera una ventana de widgets, es decir, si crea una interfaz gráfica. Si en este caso se trata de un módulo que no sea el principal, significa que es una ventana visualmente independiente pero que es parte de la principal.

Nombre del módulo	Significado y Funcionalidad	Proc/ Func	Crea inter.
ViAn	Interfaz gráfica principal del sistema, inicializa todos los datos que están en E y las características de la pantalla.	Proc	Si
ViAn_event	Es el principal del sistema y es generado por *ViAn*. Se encuentra en la espera de un evento para ejecutar un proceso. Cuando se cumple un proceso, aquí es la parte donde se regresa el sistema para la espera de otro evento.	Proc	
nuevo_accion	Permite la creación de nuevos datos originales de algún experimento, se define sus dimensiones. Se auxilia de los siguientes cuatro módulos, generando a su vez al siguiente.	Proc	Si
nuevo_event	Solicita las dimensiones de los datos N y espera la generación de un evento que permita avanzar con el proceso.	Proc	
capturar_datosN	Genera la pantalla donde se muestra una matriz para la introducción de los datos N por parte del investigador.	Proc	Si
capturar_datosN_event	Solicita la introducción de datos N y espera la generación de un evento para seguir con el proceso.	Proc	
archivar_datosN	Se encarga de almacenar los datos N y sus dimensiones en el SA y los guarda también en la E.	Proc	
abrir_accion	Permite abrir un archivo que se encuentre en el SA, detecta qué tipo de datos se abrió si texto o imagen, además identifica por quién ha sido procesado y entonces llama a los módulos indicados según el proceso correspondiente.	Proc	
kohonen_accion	Prepara los datos necesarios para el desarrollo del algoritmo de Kohonen y para que realice bien su función, manda llamar al módulo siguiente y recibe directamente los datos resultantes para enviarlos a la E.	Proc	
kohonen	Realiza el algoritmo de Kohonen para la reducción de dimensión de datos basado en las redes neuronales.	Func	

EM_accion	Prepara los datos necesarios para el desarrollo del algoritmo EM y para que realice bien su función, manda llamar al módulo siguiente y recibe directamente los datos resultantes para enviarlos a la E.	Proc
EM	Realiza el algoritmo EM para la reducción de dimensión de datos basado en el modelo de variables latentes.	Func
visualiza_accion	Aquí se convierten los datos que fueron reducidos por cualquiera de los algoritmos (Kohonen o EM) en imagen, se guardan en E.	Proc
fourier_accion	Prepara los datos necesarios para el desarrollo de la transformada de Fourier y para que realice bien su función, manda llamar al módulo siguiente y recibe directamente los datos resultantes para enviarlos a la E.	Proc
Fourier	Realiza el algoritmo para desarrollar la transformada de Fourier con el fin de lograr un análisis espectral.	Func
wavelets_accion	Prepara los datos necesarios para el desarrollo de la transformada Wavelets y para que realice bien su función, manda llamar al módulo siguiente y recibe directamente los datos resultantes para enviarlos a la E.	Proc
wavelets	Realiza el algoritmo para desarrollar la transformada de Fourier con el fin de lograr un análisis de multirresolución.	Func
despliegue_datosG	Muestra la imagen reducida y creada con anterioridad en la interfaz gráfica para que el usuario la observe y tenga la opción de almacenarla en el SA.	Proc
despliegue_datosM	Muestra la matriz de los datos reducidos en la interfaz gráfica para que el usuario observe sus resultados y tenga la opción de almacenarlos en el SA.	Proc
despliegue_datosA	Muestra en la interfaz gráfica a los datos resultantes del análisis con Fourier o con Wavelets. Los datos numéricos y la imagen resultante de la transformada.	Proc
gRes_accion	Graba los resultados numéricos (matrices de datos) de cualquiera de los procesos con sus respectivas dimensiones. Se auxilia del siguiente módulo.	Proc
archivar_accion	Almacena en el SA los datos resultantes de cualquier proceso, ya sean matrices de datos con sus respectivas dimensiones o una imagen creada por un proceso.	Proc
gImg_accion	Graba las imágenes resultantes de cualquiera de los procesos. Se auxilia del módulo anterior.	Proc

Tabla BI. Los módulos de la estructura del sistema y sus significados

2. *Flujo de datos*

En esta sección del apéndice presentamos los diagramas para el flujo de datos correspondientes a la reducción y al análisis de los datos. Y también presentamos una tabla con los datos que se encuentran en los tres diagramas para el flujo de datos, y el significado de cada dato.

Primero mostramos la figura B1 que muestra al diagrama para el flujo de datos correspondiente a la "Reducción de dimensión de datos".

En ella observamos que el *Pre-procesamiento de datos N* es formado por dos módulos, uno de ellos es *abrir_accion* que lee del SA los *datosN* y sus dimensiones *N* y *D* para entonces enviárselos a la estructura E junto con el nombre del archivo *archN* que fue leído. De la misma forma el módulo *nuevo_accion* envia a la E los mismos datos mencionados, pero envía también al SA los *datosN* y sus dimensiones *N* y *D*.

Una vez que la E tiene esta información, envía los *datosN*, *N* y *D* a cualquiera de los dos módulos: *kohonen_accion*, que es para el proceso de *Reducción de datos N con el algoritmo de Kohonen,* que envía como datos de salida *datosM_K*, *dimW* y *D*; o *EM_accion* para el proceso *Reducción de datos N con el algoritmo EM,* que envía *datosM_EM, dimR* y *N*.

El módulo siguiente *despliegue_datosM* recibe datos de cualquiera de estos dos algoritmos durante el mismo evento, este módulo es para el proceso de *Estandarizar los datos resultantes M* y tiene como datos de salida *datosM*, *MN* y *MD*. E envía también a *gRes_accion* los mismos datos, módulo para el proceso de *Almacenar datos M*, que envía a su vez esos mismos datos al SA y envía *archM* a la E.

En la figura B2, presentamos el diagrama para el flujo de datos del "Análisis de datos G". En este caso, existen dos procesos para un mismo módulo que se repite tres veces *abrir_accion*. Para uno de los procesos *Lectura de datos M*, lee del SA a *datosM, MN* y *MD* y los envía a la E junto con *tipo* y *archM*.

Los mismos datos que recibió a excepción de *archM* los envía a cualquiera de los dos procesos: *fourier_accion* para el proceso de *Análisis de datos G con transformada de Fourier* que tiene de salida los datos *datosA_F, AN_F, AD_F,* y *tipo*; o *wavelets_accion* para el proceso de *Análisis de datos G con transformada Wavelets* que tiene de salida *datosA_W, AN_W, AD_W* y *tipo*.

Regresemos ahora al proceso *Lectura de datos A* donde *abrir_accion* lee del SA a *datosA1, AN* y *AD* o a *datosA2*, según los que haya leído los envía a E y a su vez junto con *tipo* al módulo *despliegue_datosA* del proceso *Despliegue de datos A*, que se repite en el mismo proceso, y en esta ocasión, el módulo *despliegue_datosA* tiene como salida los *datosA2* que los envía a la E, y la E los lleva al módulo *glmg_accion* que realiza el mismo flujo de datos ya explicado.

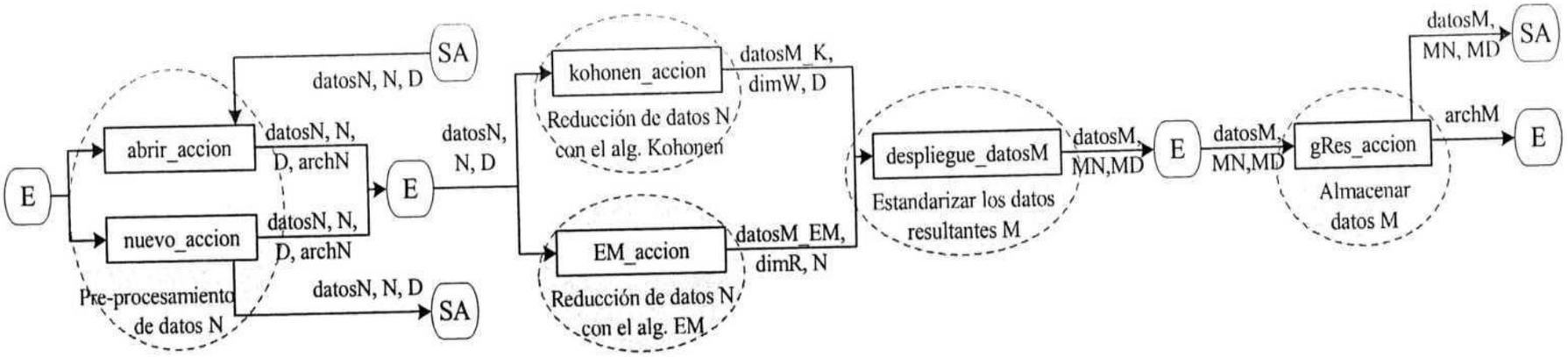


Figura B1. Diagrama para el flujo de datos originales: **"Reducción de dimensión de datos N"**

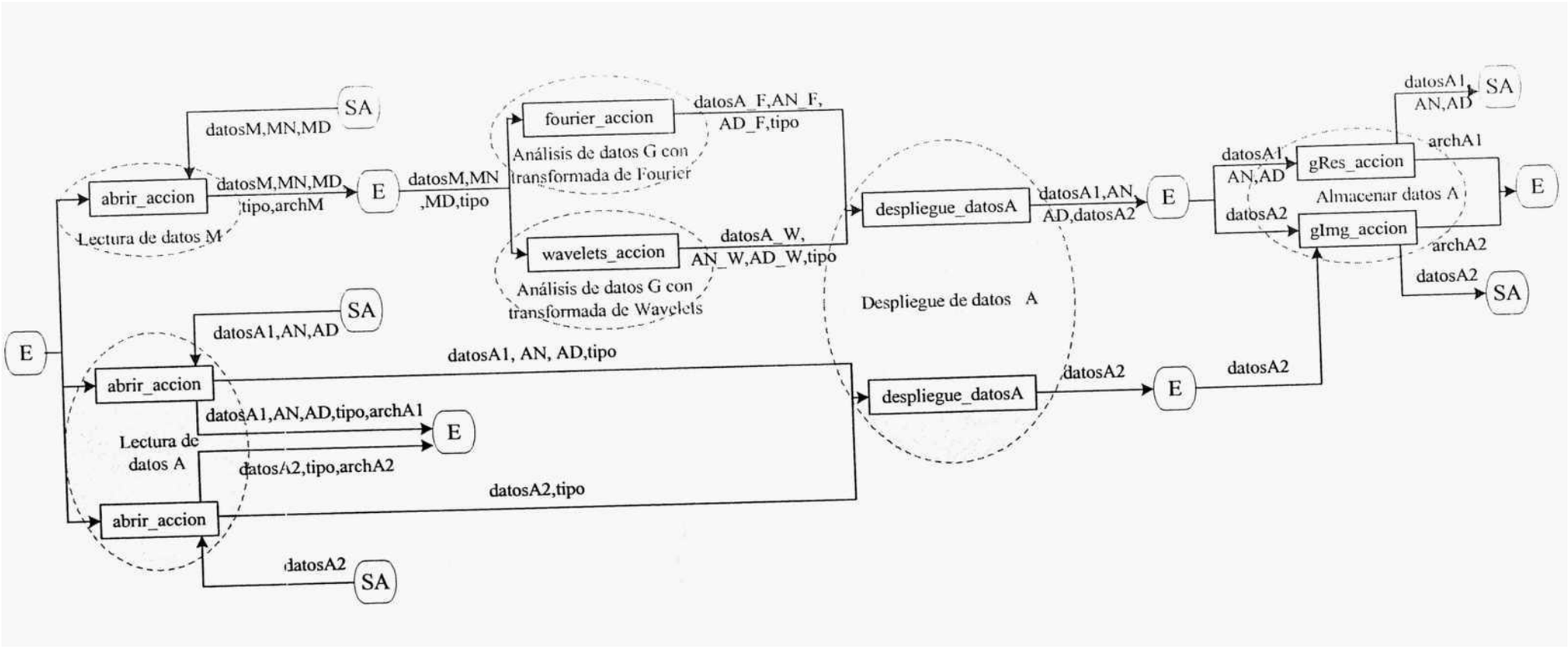


Figura B2. Diagrama para el flujo de datos visualizados: "**Análisis de datos G**"

En la tabla BII que presentamos enseguida, tenemos los datos que fluyen entre los módulos de los diagramas presentados para los flujos de datos. En la primera columna están los nombres de los datos que se presentan por orden de aparición en los diagramas (primero: capítulo IV; segundo: apéndice B), en la segunda columna mostramos sus significados y en la tercera columna mostramos a qué sección de la E pertenece este dato, las secciones de la E fueron definidas en el capítulo IV, en la Organización de información, se presenta con mayor claridad en la figura 4.4. Como se aprecia en la siguiente tabla, la mayoría de los datos pertenecen a la sección de *matrices y dimensiones de los datos* (3), ya que son los que principalmente se desean modelar.

Nombre del dato	**Significado**	**Sección de E**
datosM	Matriz de los datos reducidos ya estandarizados, es decir, que fueron el resultado de cualquiera de los dos procesos (Kohonen o EM), y está lista para su almacén o es así como se leyó del SA.	3
MN	Indica el número de datos (renglones) de la matriz *datosM*. Este dato es el que se almacena junto con la matriz correspondiente al SA o bien, fue leía de un archivo también del SA.	3
MD	Indica las dimensiones (columnas) de la matriz *datosM*. Este dato es el que se almacena junto con la matriz correspondiente al SA o bien, fue leía de un archivo también del SA.	3
archM	Es el nombre del archivo que se almacenó o se leyó del SA que contenía los *datosM*.	4
tipo	Indica si los datos leídos del SA son texto o si es una imagen. Es un auxiliar para el flujo de datos.	2
datosG	Imagen de los datos visualizados, y está lista para su almacén o es así como se leyó del SA.	3
archG	Es el nombre del archivo que se almacenó o se leyó del SA que contenía los *datosG*.	4
datosN	Matriz de los datos originales del experimento, y está lista para su almacén o es así como se leyó del SA. Esta matriz puede formarse con la captura de datos nuevos en el sistema.	3
N	Indica el número de datos (renglones) de la matriz *datosN*. Este dato es el que se almacena junto con la matriz correspondiente al SA o bien, fue leía de un archivo también del SA.	3
D	Indica las dimensiones (columnas) de la matriz *datosN*. Este dato es el que se almacena junto con la matriz correspondiente al SA o bien, fue leía de un archivo también del SA.	3
archN	Es el nombre del archivo que se almacenó o se leyó del SA que contenía los *datosN*.	4
datosM_K	Matriz resultante de los datos reducidos por el algoritmo de Kohonen.	3
dimW	Indica las dimensiones (columnas) de la matriz *datosM_K*.	3
datosM_EM	Matriz resultante de los datos reducidos por el algoritmo de EM.	3
dimR	Indica las dimensiones (columnas) de la matriz *datosM_EM*.	3
datosA1	Matriz de los datos analizados ya estandarizados, es decir, que fueron el resultado de cualquiera de los dos procesos (Fourier o Wavelets), y está lista para su almacén o es así como se leyó del SA.	3
AN	Indica el número de datos (renglones) de la matriz *datosA1*. Este dato es el que se almacena junto con la matriz correspondiente al SA o bien, fue leía de un archivo también del SA.	3
AD	Indica las dimensiones (columnas) de la matriz *datosA1*. Este dato es el que se almacena junto con la matriz correspondiente al SA o bien, fue leía de un archivo también del SA.	3
archA1	Es el nombre del archivo que se almacenó o se leyó del SA que contenía los *datosA1*.	4
datosA2	Imagen de los datos analizados, y está lista para su almacén o es así como se leyó del SA.	3
archA2	Es el nombre del archivo que se almacenó o se leyó del SA que contenía los *datosA2*.	4
datosA_F	Matriz resultante de los datos analizados por la transformada de Fourier.	3
AN_F	Indica el número de datos (renglones) de la matriz *datosA_F*.	3
AD_F	Indica las dimensiones (columnas) de la matriz *datosA_F*.	3
datosA_W	Matriz resultante de los datos analizados por la transformada de Wavelets.	3
AN_W	Indica el número de datos (renglones) de la matriz *datosA_W*.	3
AD_W	Indica las dimensiones (columnas) de la matriz *datosA_W*.	3

Tabla BII. Los datos modelados pertenecientes a la E, y sus significados correspondientes

3. *Flujo de eventos*

Los diagramas para el flujo de eventos faltantes en el capítulo IV, los presentamos enseguida. Los eventos generados que representamos en los diagramas de flujo de eventos, en esta sección y en el capítulo IV, los explicamos brevemente en la tabla BIII, que detallamos más adelante.

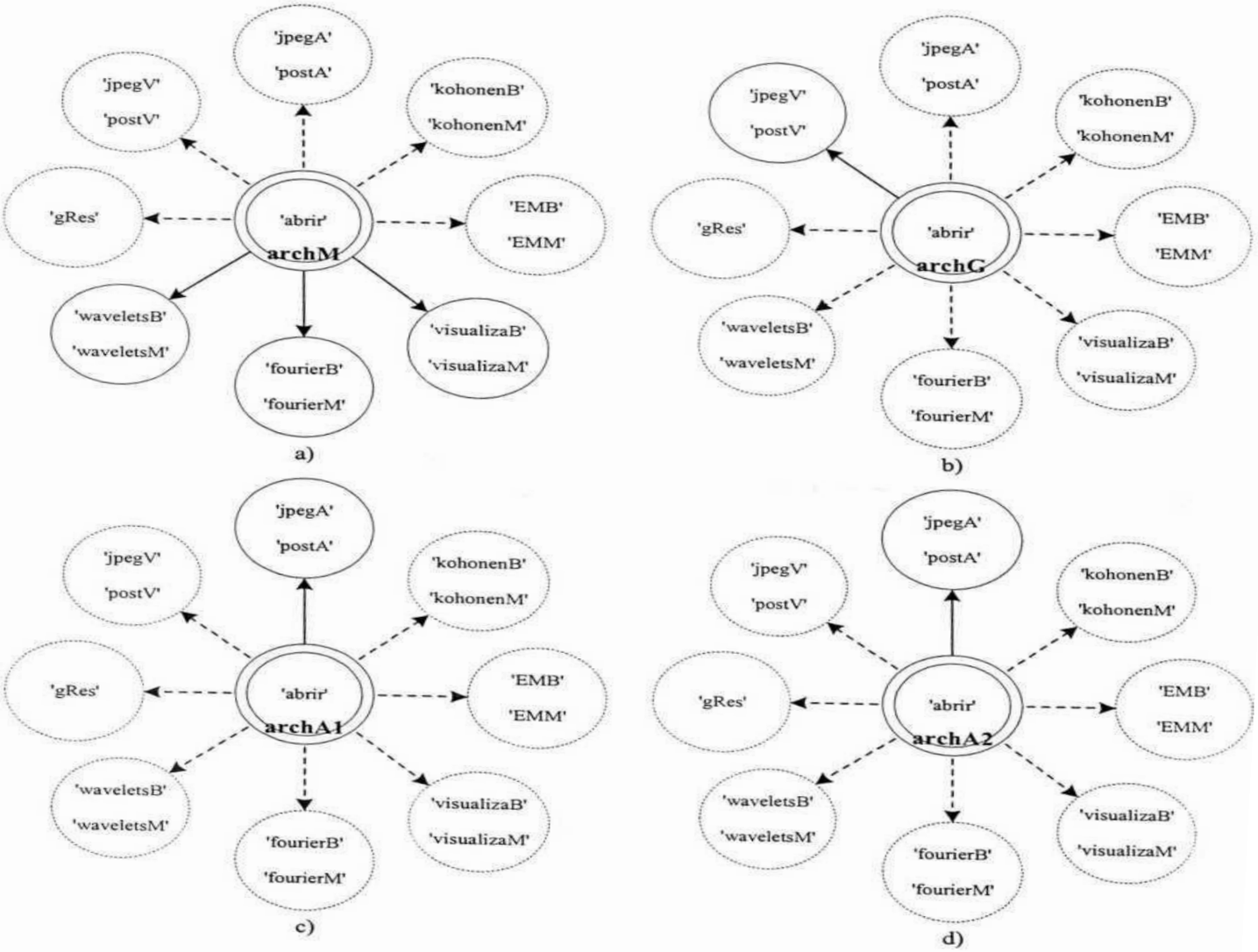


Figura B3. Diagramas de flujo de eventos para *abrir*:
(a) En caso de manipular al dato *archM*. (b) En caso de manipular al dato *archG*.
(c) En caso de manipular al dato *archA1*. (d) En caso de manipular al dato *archA2*.

En la figura B3 está el flujo de eventos para *abrir* para los diferentes casos de manipulación de datos, salvo el caso de *datosN* que fue mostrado en el capítulo IV. En (a), el evento *abrir* activa los eventos que permiten el proceso para el análisis de datos y para la visualización de datos desactivando al resto, los datos que manipulan son los *datosM*, los ya reducidos. Para (b), *abrir* activa los eventos que permiten grabar una imagen visualizada, maneja *datosG*. Los *datosA1* los presentamos en el (c), donde *abrir* activa los eventos para almacenar imágenes analizadas, al igual que en el (d) que manipula los *datosA2*.

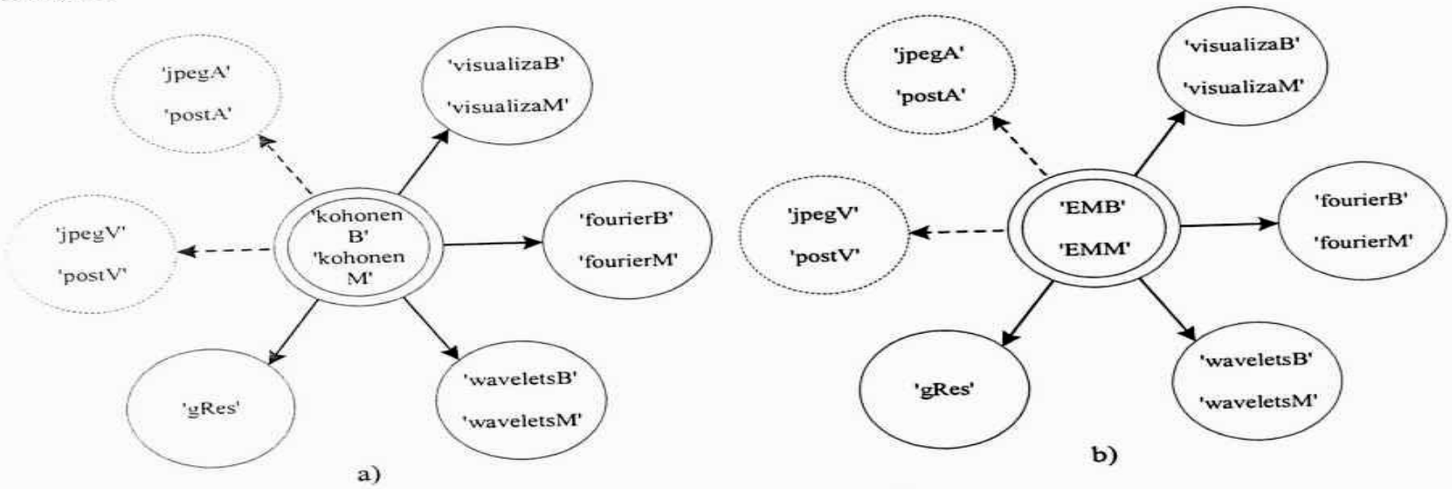


Figura B4. Diagrama para el flujo de eventos para la reducción de dimensión de datos:
(a) Para el algoritmo Kohonen. (b) Para el algoritmo EM.

Los eventos que se activan al procesar la reducción de dimensión con cualquiera de los algoritmos Kohonen o EM son *visualizaB, visualizaM, fourierB, fourierM, waveletsB, waveletsM* y *gRes*, y se desactivan *jpegA, postA, jpegV,* y *postV,* como lo presentamos en la figura B4 para los eventos de *kohonenB* y *kohonenM* en el (a) y en el (b) para los eventos *EMB* y *EMM*.

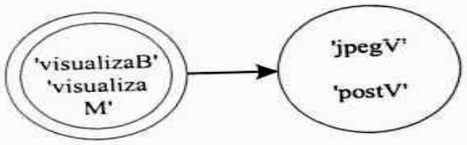


Figura B5. Diagrama de flujo de eventos para la Visualización de datos

En la figura B5 los eventos *visualizaB* y *visualizaM* llaman al mismo proceso, y al generarse éstos, solo activan los eventos *jpegV* y *postV* que también realizan el mismo proceso, solo con algunas características que los hacen identificarse.

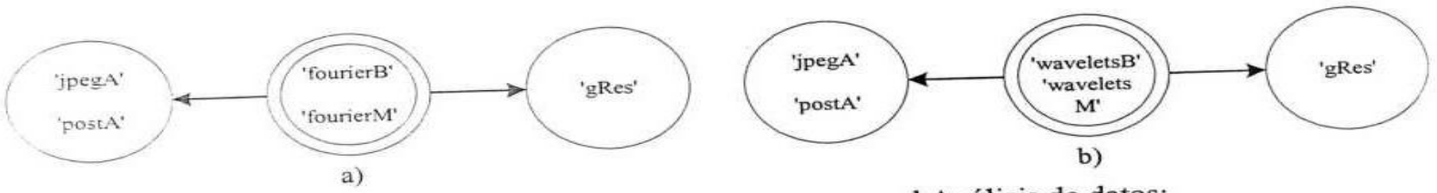


Figura B6. Diagrama de flujo de eventos para el Análisis de datos:
(a) Eventos para Fourier. (b) Eventos para Wavelets.

Para procesar el análisis de datos, pueden generarse cuatro eventos; dos para el análisis con Fourier, que son los eventos *fourierB* y *fourierM* que se modelan en la figura B6(a), los cuales activan *gRes* y *jpegA* junto con *postA*. A estos mismos eventos los activan los otros dos eventos de la figura B6(b), *waveletsB* y *waveletsM*, que procesan el análisis con Wavelets (ondeletas).

En la tabla BIII presentamos los significados de los eventos que mostramos en los diagramas de flujo de eventos de todo el diseño del sistema. En la primera columna se muestra el nombre del evento y en la segunda su significado con una breve explicación.

Evento	Significado
abrir	Evento que procesa la apertura de algún archivo que se encuentre en el SA, el flujo de eventos también depende de los datos que contenga el archivo que abra, no solo del evento por si mismo.
Nuevo	Permite la introducción y almacenamiento de los datos brutos (originales) del experimento a procesar.
kohonenB	Procesa la reducción de la dimensión de datos con el algoritmo de Kohonen, esta opción se presenta en un botón de la interfaz por ello la terminación con "B".
kohonenM	Procesa la reducción de la dimensión de datos con el algoritmo de Kohonen, esta opción se presenta en el menú principal de la interfaz por ello la terminación con "M".
EMB	Procesa la reducción de la dimensión de datos con el algoritmo EM, esta opción se presenta en un botón de la interfaz por ello la terminación con "B".
EMM	Procesa la reducción de la dimensión de datos con el algoritmo EM, esta opción se presenta en el menú principal de la interfaz por ello la terminación con "M".
visualizaB	Ejecuta el procesa de la visualización de los datos previamente reducidos, esta opción se encuentra como un botón de la interfaz, tiene la terminación "B".
visualizaM	Ejecuta el procesa de la visualización de los datos previamente reducidos, esta opción se encuentra en el menú de la interfaz, tiene la terminación "M".
fourierB	Procesa el análisis de datos con la transformada de Fourier, esta opción se presenta en un botón de la interfaz por ello la terminación con "B".
fourierM	Procesa el análisis de datos con la transformada de Fourier, esta opción se presenta en el menú principal de la interfaz por ello la terminación con "M".
waveletsB	Procesa el análisis de datos con la transformada Wavelets, esta opción se presenta en un botón de la interfaz por ello la terminación con "B".
waveletsM	Procesa el análisis de datos con la transformada Wavelets, esta opción se presenta en el menú principal de la interfaz por ello la terminación con "M".
gRes	Permite grabar la matriz resultante del último proceso realizado en el SA, como un archivo de modo texto.
jpegV	Ejecuta el proceso que almacena la imagen visualizada en el SA con el formato jpeg, extensión '.jpeg'.
postV	Ejecuta el proceso que almacena la imagen visualizada en el SA con el formato postcript, extensión '.ps'.
jpegA	Ejecuta el proceso que almacena la imagen analizada en el SA con el formato jpeg, extensión '.jpeg'.
postA	Ejecuta el proceso que almacena la imagen analizada en el SA con el formato postcript, extensión '.ps'.

Tabla BIII. Eventos y sus significados

4. *Diseño de interfaz*

Las ventanas que complementan al diseño de interfaz presentado en el capítulo IV, las exponemos en esta sección, estas ventanas se basan en las pantallas del pre-diseño de interfaz (requerimientos de interfaz) detalladas en el análisis del sistema. Las pantallas específicas en las que se basan las siguientes ventanas, las presentamos en el apéndice A.

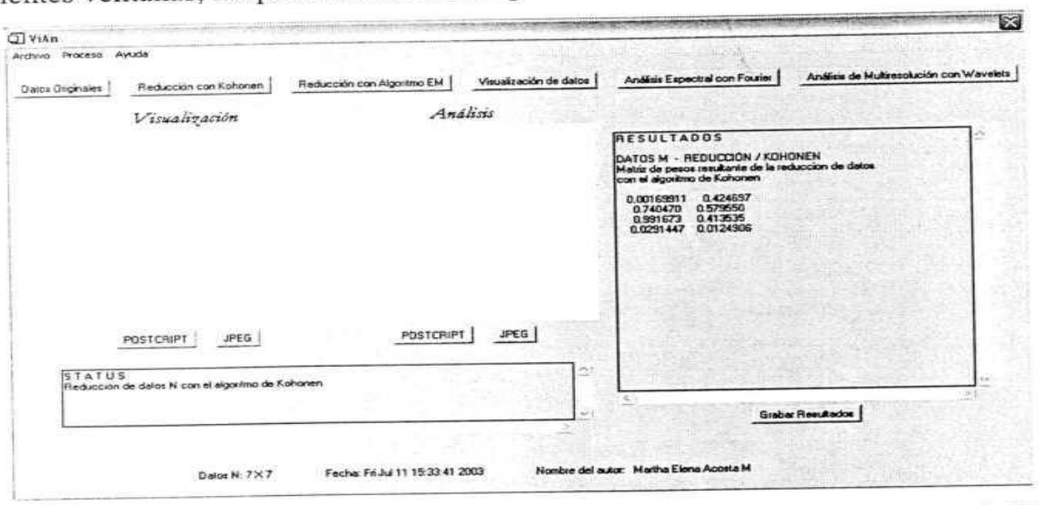


Figura B7. Ventana para la reducción de dimensión de los datos con el algoritmo de Kohonen

En la figura B7 mostramos la ventana para la reducción de dimensión de datos con el algoritmo de Kohonen. Esta ventana se crea después de que el usuario genera el evento para el proceso de la reducción de los datos N con el algoritmo de Kohonen, a través del botón *Reducción con Kohonen* que se modela en la figura B7. El resultado del proceso, es desplegado en forma matricial en la región de los *resultados*. Esta ventana tiene como base la pantalla A10 (apéndice A), y la ventana de la figura B7 diseña en forma detallada, especificando una técnica para el proceso de reducción.

El diseño en forma detallada, especificando otra técnica para el proceso de reducción, lo presentamos en la ventana de la figura B8 que también se basa en la pantalla A10. En la figura B8 presentamos la ventana para la reducción de dimensión de datos con el algoritmo EM. Esta ventana se crea después de que el usuario genera el evento para el proceso de la reducción de los datos N con el algoritmo EM, a través del botón *Reducción con EM* que se modela en la figura B8. Observamos que en el área de *resultados* los datos resultantes del proceso son desplegados en forma matricial. Estos datos pueden ser almacenados en SA (sistema de archivos), al generarse el evento correspondiente al botón que se modela como *Grabar Resultado*. Este suceso también puede ser generado para el caso de la figura B7.

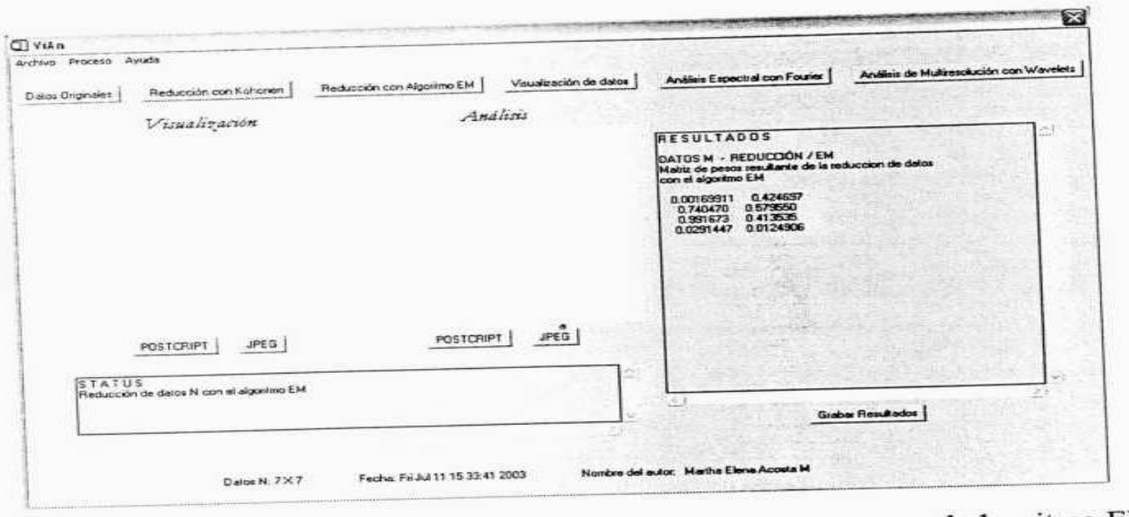


Figura B8. Ventana para la reducción de dimensión de los datos con el algoritmo EM

En la figura B9 presentamos el diseño de la ventana para el proceso de la visualización de datos reducidos por cualquiera de los dos algoritmos de reducción. Con el botón *Visualización de datos* generamos el evento que corresponde al proceso que despliega la imagen creada a través de los datos reducidos. La imagen se despliega en el área correspondiente a la visualización, la cual puede ser almacenada en el SA, en el momento que el usuario genere el evento que solicite su almacén. Se diseñaron dos botones bajo la región donde se despliega la imagen, para almacenar la imagen en el SA, uno de ellos le corresponde el formato postcript y el otro botón el formato jpeg. Esta ventana se basa en la figura A11.

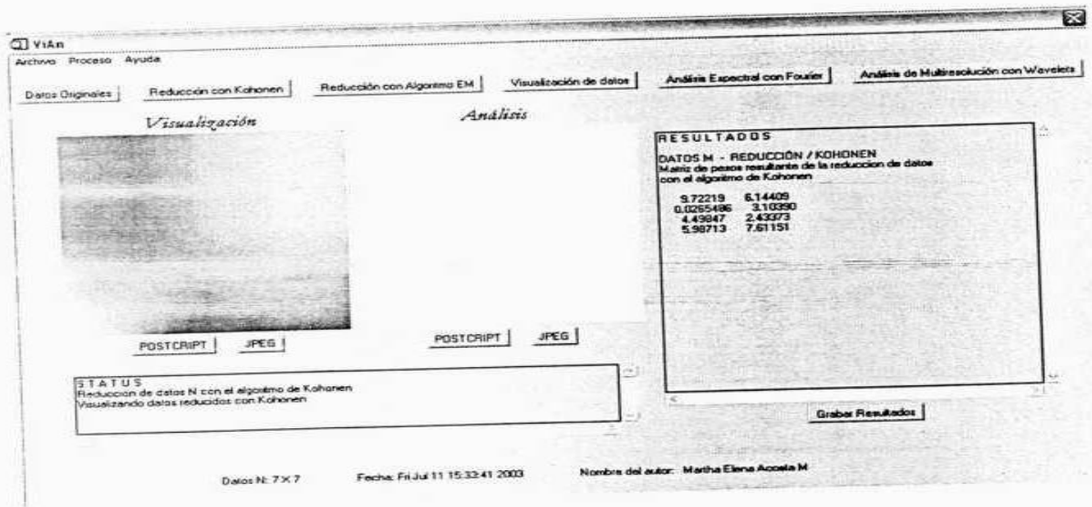


Figura B9. Ventana para la visualización de datos reducidos

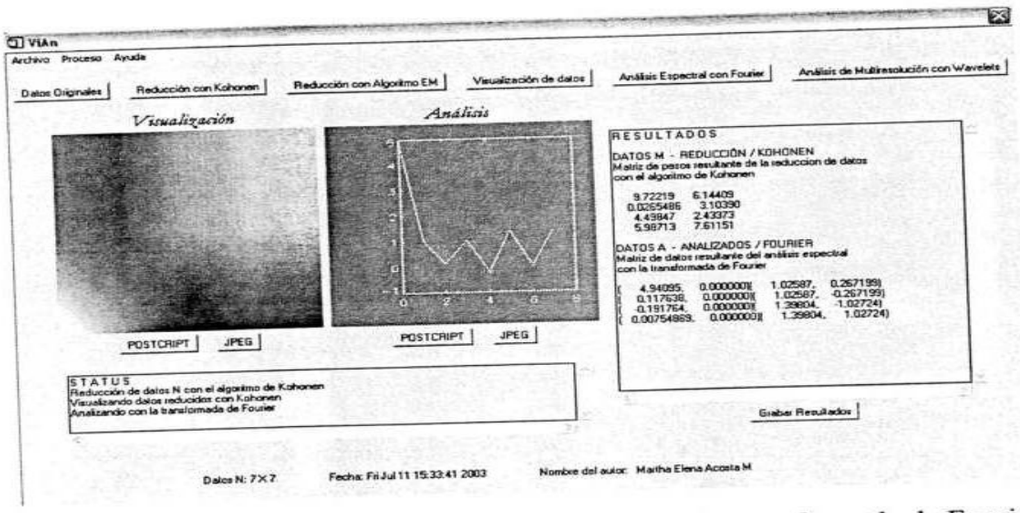


Figura B10. Ventana para el análisis de los datos con la transformada de Fourier

En las figuras B10 y B11 presentamos las ventanas que diseñamos para el proceso del análisis de datos, con la transformada de Fourier y la transformada de Wavelets (ondeletas), respectivamente. Para la primera técnica (o transformada), le corresponde el botón *Análisis espectral con Fourier* y para la segunda, le corresponde el botón *Análisis de multirresolución con Wavelets*. Ambas ventanas muestran el despliegue de los resultados del análisis, en forma matricial (área de *Resultados*) y en forma de gráfica (región para la imagen del análisis). Ambas ventanas se basan en la figura A12. Los datos resultantes de estos procesos, pueden ser almacenados en el SA, a través de los botones diseñados bajo cada resultado.

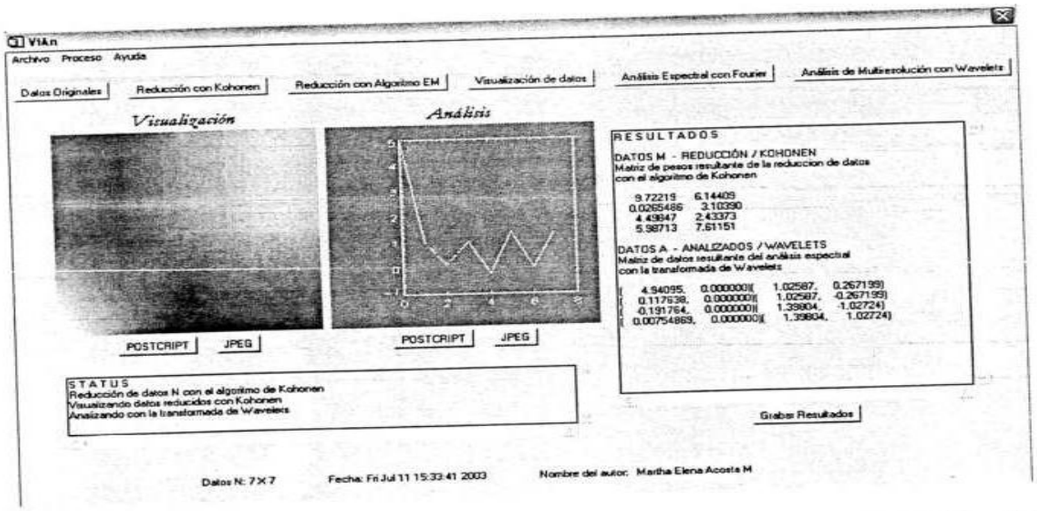


Figura B11. Ventana para el análisis de los datos con la transformada de Wavelets

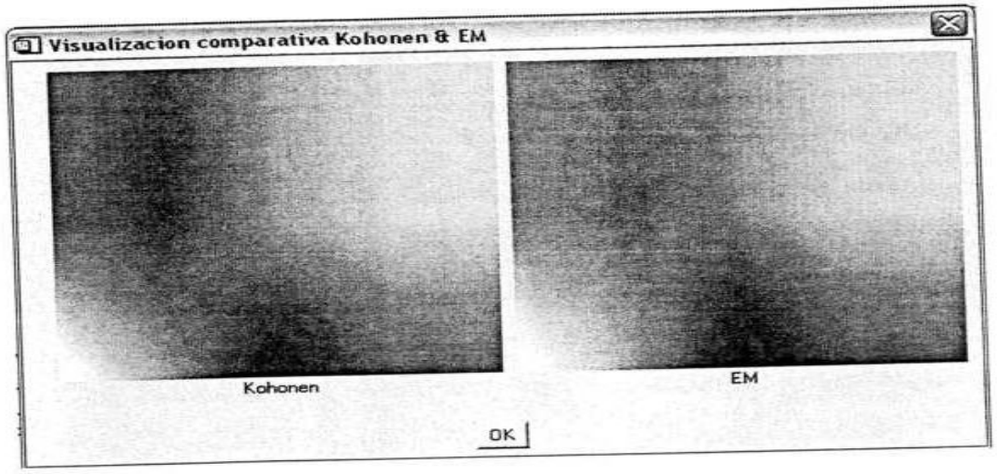


Figura B12. Ventana para la visualización de datos con ambos algoritmos de reducción

Las figuras B12 y B13 diseñan las ventanas basadas en la figura A13 (dependientes de la ventana principal), en las que el usuario genera el evento correspondiente a los procesos de la reducción o el análisis con ambas técnicas según opción. Reducción con el algoritmo de Kohonen y el algoritmo EM para figura B12 y el análisis con la transformada de Fourier y Wavelets para la figura B13. Estas ventanas contienen dos áreas en las que se despliegan las imágenes resultantes del proceso elegido, en la primera (izquierda) se presentará la imagen resultante de Kohonen o de Fourier y en la segunda (derecha) la imagen generada por EM o por Wavelets, según el proceso solicitado por el usuario.

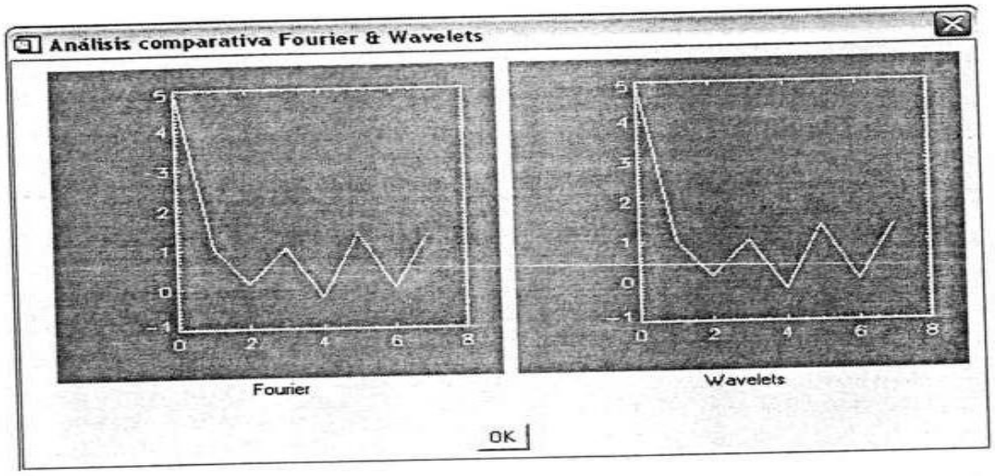


Figura B13. Ventana para el análisis de datos con ambas transformadas

APENDICE C.

Algoritmos

Los científicos no persiguen la verdad, es
ésta quien los persigue a ellos.
Karl Schlechta

APENDICE C.
Análisis de los algoritmos

Este apéndice es un complemento del Análisis de los algoritmos, aquí se encontrarán conceptos y deducciones que ayudarán a la comprensión de los problemas, funciones y/o ecuaciones que se tratan a lo largo del desarrollo de los algoritmos (desde su análisis hasta su implementación y pruebas). Por lo tanto, las referencias a las siguientes secciones las encuentra en la sección del Análisis de los algoritmos.

1. Algoritmo EM para el modelo de las Variables Latentes

En el algoritmo EM para el modelo de las variables latentes, necesitamos deducir varias funciones con respecto a los elementos del algoritmo, así como comprender conceptos del modelo. Esta sección nos ayudará a explicar partes del algoritmo y en qué conceptos se basa su desarrollo, aunque no se profundice en algunas definiciones, esta sección será lo suficientemente explícita para comprender el desarrollo e implementación del algoritmo EM.

1.1. Responsabilidades

En la determinación de los pesos, es necesario calcular las responsabilidades, que no es otra cosa que la probabilidad de que el dato n-ésimo haya producido un efecto en la variable latente k, para una W y una β dadas. Las responsabilidades se calculan de la siguiente forma, basándose en el teorema de Bayes, de esta misma forma las presentamos también en la función [5.8]:

$$R_{kn} = p(x_k \mid t_n, W, \beta) = \frac{p(t_n \mid x_k, W, \beta)\, p(x_k)}{\sum_{k'} p(t_n \mid x_{k'}, W, \beta)\, p(x_k')} \qquad \text{[C1]}$$

donde tenemos que la distribución de probabilidad se define de una forma particular, es decir es una distribución gaussiana isotrópica del ruido, es definida de la siguiente forma y es equivalente a la función [5.7]:

$$p(t \mid x, W, \beta) = \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left\{-\frac{\beta}{2}\sum_{d}^{D}(t_d - y_d(x,W))^2\right\}} \qquad \text{[C2]}$$

Donde, β^{-1} denota la variación del ruido. Ahora retomamos [5.9], y con ello obtenemos que:

$$p(t \mid x, W, \beta) = \left(\frac{\beta}{2\pi}\right)^{\frac{N}{2}} e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)} \qquad [C3]$$

Ahora, adaptando la función [C1], se presenta de la siguiente forma:

$$R_{kn} = \frac{\left(\frac{\beta}{2\pi}\right)^{\frac{N}{2}} e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)}}{\left(\frac{\beta}{2\pi}\right)^{\frac{N}{2}} \sum_{k}^{K} e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)}} \qquad [C4]$$

Y ahora, las responsabilidades R quedan como se presenta en la siguiente función:

$$R_{kn} = \frac{e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)}}{\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\Delta_{kn}\right)}} \qquad [C5]$$

A partir de la función [C5], las responsabilidades se encuentran aptas para implementarlas, esta misma función la encontramos en el documento como la función [5.8].

1.2. Función de probabilidad Likelihood

Basándonos en el teorema de Bayes podemos obtener la función de probabilidad conocida como la función likelihood. Se elige una hipótesis de que ocurra un evento T_n que tenga la máxima probabilidad de ocurrir sobre el resto de los eventos (grupo de eventos: T). Entendamos sobre este contexto que le llamamos "evento" a un dato del conjunto de datos T. El evento del que depende que ocurra T_n es *w*; esto significa que, para que se elija al dato T_n como representante depende del conjunto de pesos *w*. W es el conjunto de vectores *w* que representan a los pesos.

Ahora, se requiere de la maximización de la probabilidad condicional de los datos W, dados los T datos, esto es: p(T|*w*). En otras palabras p(T|*w*) es la probabilidad de *w* que define la distribución de probabilidad sobre el espacio de datos del conjunto T condicionado por *w*. Tenemos que p(w) es la distribución de probabilidad apriori sobre los pesos *w*. Así tenemos la suma de todas las probabilidades de que T ocurra condicionado por los pesos *w* escrita de la siguiente forma:

$$p(T) = \int p(T \mid w) p(w) dw \qquad [C6]$$

Con estos términos, la función de distribución de probabilidad que utiliza el teorema de Bayes se expresa como sigue:

$$p(w \mid T) = \frac{p(T \mid w)\,p(w)}{\int p(T \mid w)\,p(w)\,dw} \qquad [C7]$$

De aquí nos interesa la probabilidad máxima p(T|w) para llegar al concepto de la función likelihood, ya que esta distribución de probabilidad es muy peculiar dado el concepto de las variables latentes[1]. En la sección 1.1. se definió la distribución gaussiana isotrópica del ruido [C2], ahora la utilizaremos para definir:

$$p(w \mid T) = \frac{\prod_{n}^{N}\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\, t_n - y(x_k, w\|^2\right)}}{\int \prod_{n}^{N}\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\, t_n - y(x_k, w\|^2\right)} dT} \qquad [C8]$$

el numerador es equivalente a:

$$\prod_{n}^{N}\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\|t_n - y(x_k, w\|^2\right)} = \exp\left\{-\left(-\sum_{n}^{N}\ln\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\|t_n - y(x_k, w\|^2\right)}\right)\right\} \qquad [C9]$$

y podemos deducir el denominador como sigue:

$$\int \prod_{n}^{N}\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\|t_n - y(x_k, w\|^2\right)} dT = \int \sum_{k_1}^{K}\cdots\sum_{k_N}^{K}\prod_{n}^{N} e^{\left(-\frac{\beta}{2}\|t_n - y(x_k, w\|^2\right)} dT$$

$$= \sum_{k_1}^{K}\cdots\sum_{k_N}^{K}\int e^{\left(-\sum_{n}^{N}\frac{\beta}{2}\|t_n - y(x_k, w\|^2\right)} dT$$

$$= K^N\left(\frac{2\pi}{\beta}\right)^{\frac{ND}{2}} \qquad [C10]$$

Retomando [C8] y utilizando [C9] y [C10] obtenemos lo siguiente:

$$p(T \mid w) = \frac{\exp\left\{-\left(-\sum_{n}^{N}\ln\sum_{k}^{K} e^{\left(-\frac{\beta}{2}\|t_n - y(x_k, w\|^2\right)}\right)\right\}}{K^N\left(\frac{2\pi}{\beta}\right)^{\frac{ND}{2}}} \qquad [C11]$$

[1] La función [C7] tiene el formato de la Fórmula general del modelo de las variables latentes.

donde detectamos la siguiente función, que le llamaremos "Función de error":

$$S_T(w,\beta) = -\sum_n^N \ln \sum_k^K e^{\left(-\frac{\beta}{2}\|t_n - y(x_k,w)\|^2\right)} \qquad [C12]$$

Si a esta función de error la regularizamos, obtenemos entonces nuestra función likelihood:

$$S_T = -\sum_n^N \ln\left(\frac{1}{K}\sum_k^K p(t_n \mid X_k, W, \beta)\right) = \ell \qquad [C13]$$

que es equivalente a la función ℓ ([5.6]) del documento. Donde, conocemos la distribución de probabilidad en [C2], y que se puede re-escribir de la siguiente forma:

$$p(t \mid x, W, \beta) = \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\sum_d^D (t_{nd} - \Phi_k W_d)^2\right)} = p_{kn} \qquad [C14]$$

De aquí podemos deducir que $Y = \Phi W$, donde Y es una matriz de K X D, ϕ es una matriz de K X M que contiene el centro de las gaussianas, y W es una matriz de M X D que contiene los pesos y los parámetros diagonales.

Recordemos que las responsabilidades R se basan de igual forma en el teorema de Bayes, y además en la distribución de probabilidad que obtuvimos en [C14]. Entonces re-escribamos las Responsabilidades R que tenemos en [C1], como sigue:

$$R_{kn} = \frac{p_{kn}}{\sum_{k'}^K p_{k'n}} \qquad [C15]$$

1.3. Deducción para W

Para la deducción de W, necesitamos la función likelihood que presentamos en [C13] y le aplicamos su derivada con respecto a W. Y encontramos su condición de maximización para encontrar el parámetro W. Utilizando [C13], [C14] y [C15], el desarrollo de la derivada es:

$$\frac{\partial \ell}{\partial W} = \frac{\partial S_T}{\partial W_{ij}} = -\sum_n^N \frac{1}{\sum_{k'}^K \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\sum_d^D (t_{nd} - \Phi_k W_d)^2\right)}} \cdot$$

$$\sum_k^K \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\sum_d^D (t_{nd} - \Phi_k W_d)^2\right)} \beta(t_{nj} - \Phi_k W_j)\Phi_{ki}$$

$$= -\sum_{n,k}^{N,K} \frac{p_{kn}}{\sum_{k'}^{K} p_{k'n}} \cdot \beta(t_{nj} - \Phi_k W_j)\Phi_{ki}$$

$$= -\sum_{n,k}^{N,K} R_{kn}\beta(t_{nj} - \Phi_k W_j)\Phi_{ki} \qquad [C16]$$

Ahora, [C16] lo igualamos a cero:

$$-\sum_{n,k}^{N,K} R_{kn}\beta(t_{nj} - \Phi_k W_j)\Phi_{ki} = 0$$

$$-\sum_{n,k}^{N,K} R_{kn}\beta\Phi_{ki}t_{nj} + \sum_{n,k}^{N,K} R_{kn}\beta\Phi_{ki}\Phi_k W_j) = 0$$

$$\sum_{n,k}^{N,K} R_{kn}\beta\Phi_{ki}\Phi_k W_j) = \sum_{n,k}^{N,K} R_{kn}\beta\Phi_{ki}t_{nj}$$

$$R\beta\Phi\Phi W = R\beta\Phi T$$

$$\beta\Phi R\Phi W = \beta\Phi RT$$

Aqui introducimos Φ^T que equivale a $\beta\Phi$, y G, su definición la presentamos en [14]. Además, introducimos un parámetro de regularización de los pesos, con el fin de que no difieran los elementos entre ellos por una cantidad notable. La siguiente ecuación es la que presentamos en [13]:

$$(\Phi^T G\Phi + I\lambda)W = \Phi^T RT$$

Por ultimo despejamos W y la fórmula para actualizar los pesos nos queda de la siguiente forma:

$$W = (\Phi^T G\Phi + I\lambda)^{-1}\Phi^T RT \qquad [C17]$$

1.4. Deducción para β

Para la deducción de β, necesitamos la función likelihood que presentamos en [C13] y le aplicamos su derivada con respecto a β. Y encontramos su condición de maximización para encontrar el parámetro β. Utilizando [C13], [C14] y [C15], el desarrollo de la derivada es:

$$\frac{\partial \ell}{\partial \beta} = \frac{\partial S_T}{\partial \beta} = -\sum_{n}^{N} \frac{1}{\sum_{k'}^{K}\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\sum_{d}^{D}(t_{nd} - \Phi_k W_d)^2\right)}} \cdot$$

$$\frac{1}{K}\sum_{k}^{K}\left[\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\sum_{d}^{D}(t_{nd}-\Phi_k W_d)^2\right)}\left(-\frac{1}{2}\sum_{d}^{D}(t_{nd}-\Phi_k W_d)^2\right)+\right.$$

$$\left. e^{\left(-\frac{\beta}{2}\sum_{d}^{D}(t_{nd}-\Phi_k W_d)^2\right)}\left(\frac{D}{2}\right)\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}-1}\left(\frac{1}{2\pi}\right)\right]$$

Basándonos en la función [9], tenemos que: $\Delta = \sum_{d}^{D}(t_{nd}-\Phi_k W_d)^2$. Entonces, continuando el desarrollo de la derivada, obtenemos lo siguiente:

$$= -\frac{1}{K}\sum_{n,k}^{N,K}\frac{\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} e^{\left(-\frac{\beta}{2}\Delta\right)}\left(\frac{D}{2\beta}-\frac{1}{2}\Delta\right)}{\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}}\sum_{k'}^{K} e^{\left(-\frac{\beta}{2}\Delta\right)}}$$

$$= -\frac{1}{K}\sum_{n,k}^{N,K}\frac{e^{\left(-\frac{\beta}{2}\Delta\right)}}{\sum_{k'}^{K} e^{\left(-\frac{\beta}{2}\Delta\right)}}\cdot\left(\frac{D}{2\beta}-\frac{1}{2}\Delta\right)$$

De lo anterior podemos identificar las responsabilidades R, y obtenemos lo siguiente:

$$= -\frac{ND}{\beta}+\sum_{n,k}^{N,K} R_{kn}\Delta_{kn} \qquad [C18]$$

Ahora, [C18] lo igualamos a cero:

$$-\frac{ND}{\beta}+\sum_{n,k}^{N,K} R_{kn}\Delta_{kn} = 0$$

$$\sum_{n,k}^{N,K} R_{kn}\Delta_{kn} = \frac{ND}{\beta}$$

Por último despejamos 1/β, para actualizar β. Observemos que la actualización está en función de las Responsabilidades R. La fórmula equivalente a [10], es la siguiente:

$$\frac{1}{\beta} = \frac{1}{ND}\sum_{n,k}^{N,K} R_{kn}\Delta_{kn} \qquad [C19]$$

APENDICE D.

Implementación

Si buscas resultados distintos,
no hagas siempre lo mismo.

Albert Einstein

APENDICE D.
Implementación

Este apéndice es un complemento de la fase que corresponde a la Implementación del sistema ViAn. Presentamos el manual de usuario, utilizando el Diseño de Interfaz Gráfica. Las referencias a la siguiente sección las encuentra en el capítulo VI.

1. El manual de usuario y el Diseño de Interfaz Gráfica

Esta información es de gran utilidad para el usuario, ya que va dirigido a él. Con este manual, el usuario aclarará sus dudas con respecto a la funcionalidad de la aplicación, incluso puede comprender que uso tienen ciertos procedimientos. Este manual de usuario, corresponde a los procesos, actividades y eventos que puede realizarse en la primera versión del sistema ViAn.

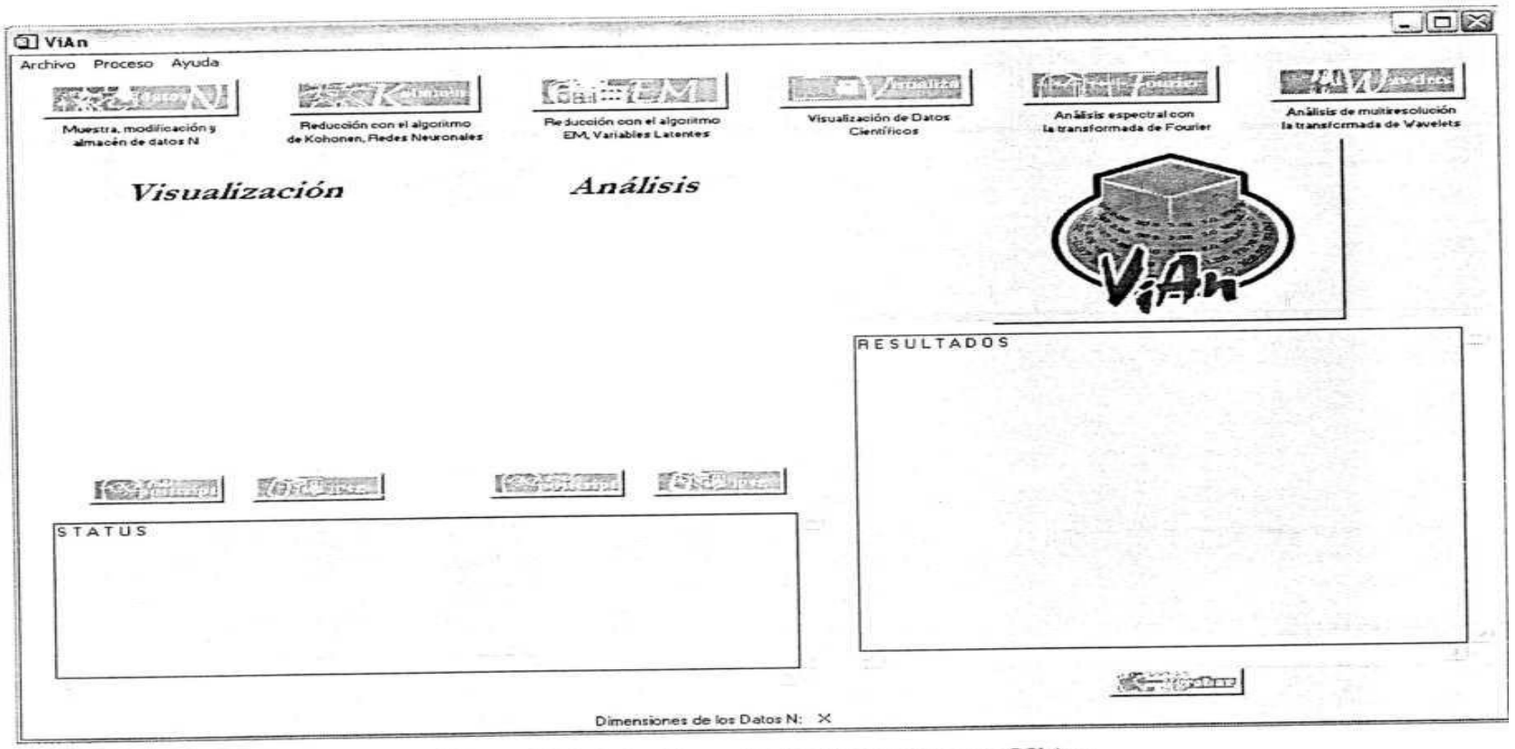


Figura D1. Interfaz principal del sistema ViAn

Con respecto a la función y funcionalidad de estas interfaces, son muy similares a las detalladas para el diseño de interfaz. Las diferencias que encontramos entonces entre el diseño de interfaz y el diseño de interfaz gráfica, son los aspectos visuales de los botones, en ocasiones los nombres. Y, en el diseño de interfaz gráfica, se muestra como se va desenvolviendo el flujo de los eventos que presentamos en el capítulo IV. Además, refleja limitaciones del sistema, como la implementación del análisis de datos con la transformada de Wavelets para la primera versión del sistema ViAn, entre otras más que explicamos en el capítulo VI.

La interfaz principal del sistema ViAn (figura D1) muestra la ventana a través de la cual el usuario iniciará a utilizar la aplicación. Todos los botones aparecen deshabilitados a excepción del botón que muestra el logotipo del sistema ViAn, al presionar este botón, se abrirá una ventana que muestra el nombre del sistema, versión y sus autores (figura D2(a)). Esta ventana también se puede abrir al presionar la opción de "Acerca de ViAn", localizado en el menú superior, en la sección de "Ayuda".

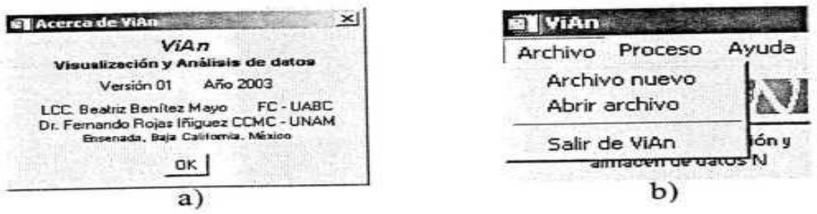


Figura D2. Eventos iniciales del sistema ViAn: (a) Información acerca de ViAn
(b) Opciones de la parte de Archivo del menú superior

Para iniciar con los procesos deseados, es necesario introducir los datos del experimento, ya sean datos nuevos o datos previamente almacenados en el sistema de archivos (SA). Estas opciones las encontramos en el menú superior, como se muestra en la figura D2(b), presionando los eventos de nombre: "Archivo nuevo" para introducir la matriz de datos nuevos, o "Abrir archivo" para seleccionar el archivo que contiene los datos originales. Una tercera opción que se le presenta al usuario en esta sección del menú superior, es el de la salida del sistema ViAn. Estas tres opciones, además, se encuentran activas en cualquier momento a lo largo de la vida de la aplicación.

Para abrir un archivo previamente almacenado en el SA, elegimos la opción que mencionábamos anteriormente, la cual se presenta en la figura D2(b) (Abrir archivo). Se abrirá una ventana donde el usuario puede elegir al archivo deseado, el archivo debe tener la extensión '.via'. Al abrirlo, el sistema ViAn carga en su memoria los datos correspondientes (la matriz de datos N y sus dimensiones). Se especifica en el área de Status (estado) el hecho de que se abrió en un archivo de datos N.

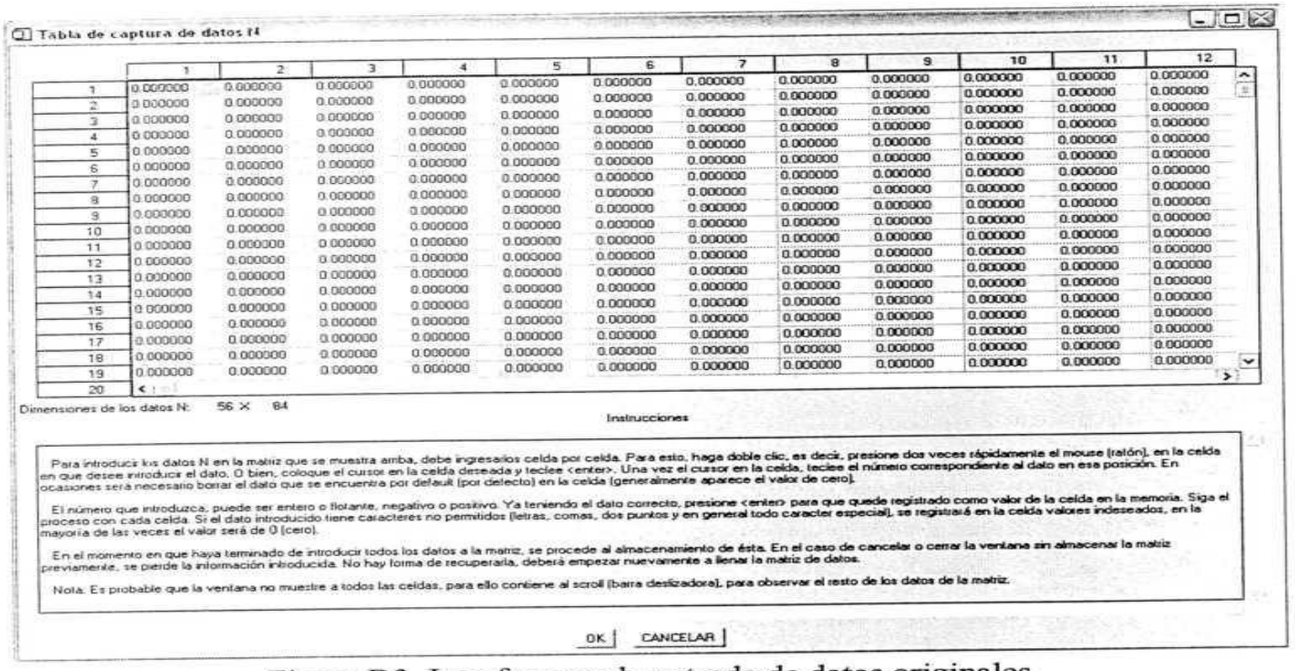


Figura D3. Interfaz para la entrada de datos originales

Si el usuario elige abrir un nuevo archivo, entonces se abrirá una ventana, la cual solicitará las dimensiones de la matriz de los datos N. Estas dimensiones deben ser igual o mayores a dos (2). Enseguida se abrirá una nueva ventana, donde se muestra una matriz con las dimensiones indicadas. En esta segunda ventana, que presentamos en la figura D3, el usuario puede introducir sus datos N, celda por celda. Y por último se abre una ventana más, solicitando el nombre del archivo que contendrá la matriz de los datos introducidos. En este momento, en el área de status se explica que se ha abierto un archivo de datos N. Si no se siguen estos pasos correctamente y de forma completa, entonces el sistema no tendrá en memoria ninguna matriz de datos N, y no se podrá proseguir con los procesos.

Una vez teniendo en memoria los datos N, preparados para sus procesos, pueden ser consultados y modificados. Esto nos lo muestra la figura D4 donde se despliega en la ventana la matriz de datos N. El usuario puede consultar los datos y cerrar la ventana. Una opción más, es modificar algunos, varios o todos los valores sin ser alteradas las dimensiones. Si el usuario ha actualizado los datos deseados, la matriz puede sobre-escribirse, es decir, puede almacenarse con el nombre original del archivo, perdiendo los datos anteriores. Sin embargo, el usuario cuenta con la flexibilidad de escribir la matriz actualizada en un nuevo archivo, y así no pierde los datos originales ni los modificados. Es entonces que aparece en el área del status, el nuevo archivo cargado en memoria del programa.

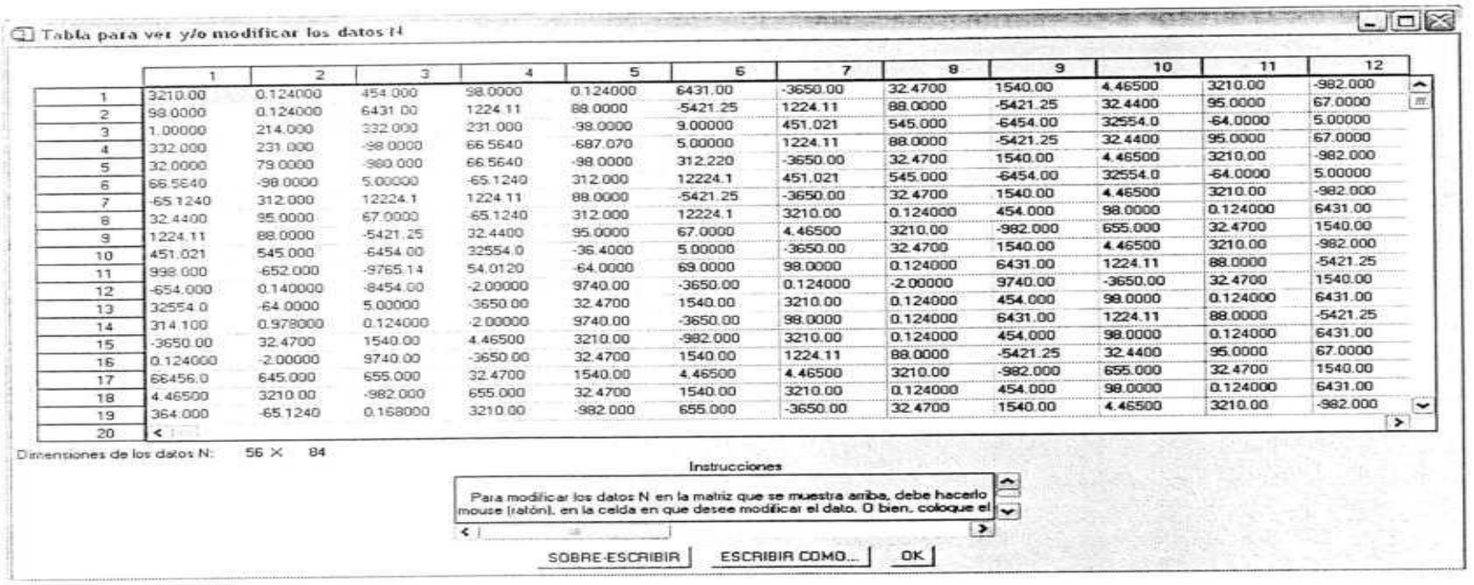


Figura D4. Interfaz para la consulta y/o modificación de los datos originales

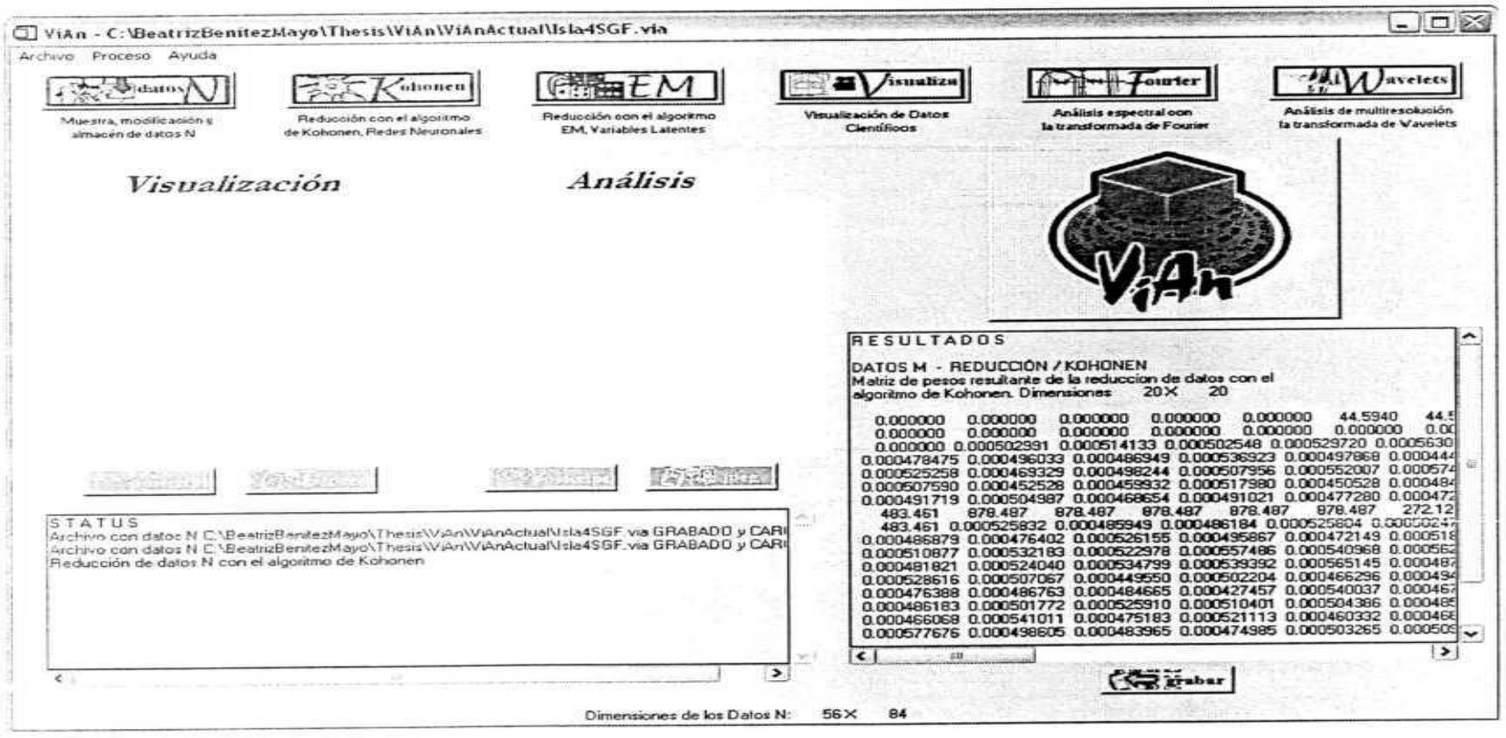


Figura D5. Interfaz para la reducción de dimensión de los datos con el algoritmo de Kohonen

Desde el momento en que se cargaron en la memoria del sistema ViAn los datos N, se encuentran preparados para la reducción de sus dimensiones a través de cualquiera de los dos algoritmos que la aplicación ofrece. Ya sea con el algoritmo de Kohonen, del cual la ventana D5 presenta sus datos numéricos resultantes en el área de resultados y se despliega en el status, el proceso que se cumple: reducción con Kohonen. O bien, si se reduce con el algoritmo EM, aparecería la ventana de forma similar, con la diferencia de que se indicaría que la reducción fue realizada con el algoritmo EM. La matriz que se muestra en el área de resultado (figura D5), puede ser almacenada a través de la opción que genera el botón "grabar" que se presenta debajo de dicha área. Se solicita el nombre del archivo en que se almacenará la matriz resultante de la reducción correspondiente. El archivo es de extensión '.dtR'.

En el menú superior podemos encontrar la sección de Proceso. En esta sección se localizan las opciones para reducir con Kohonen o EM, de analizar con Fourier o EM, y de visualizar datos. Estas mismas opciones las encontramos en forma de botones, como lo hemos mostrado en las figuras de las ventanas del diseño de interfaz. Además, en esta sección, se encuentran las opciones para reducir con ambos algoritmos simultáneamente, y así poder visualizar ambos resultados con el procedimiento adecuado. De igual forma encontramos la opción de analizar los datos con ambas transformadas simultáneamente. Las explicaciones de sus respectivas ventanas las mostraremos adelante.

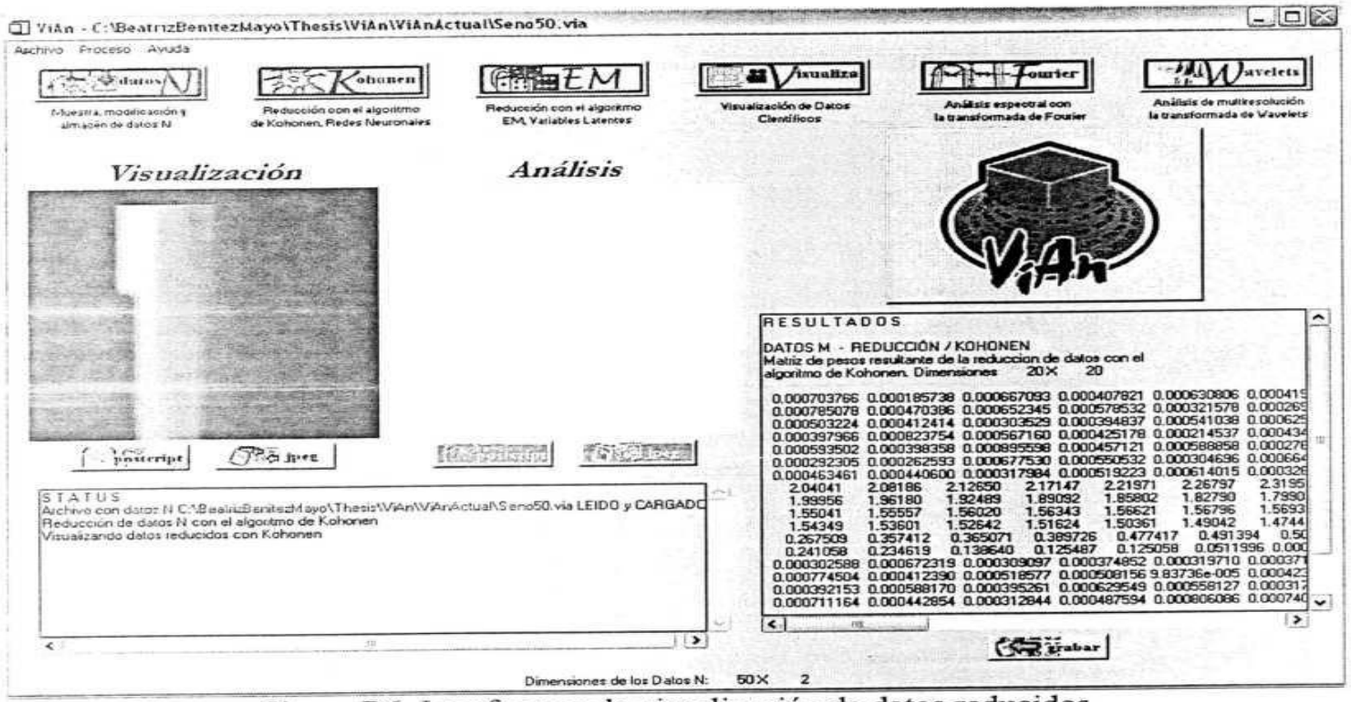


Figura D6. Interfaz para la visualización de datos reducidos

El usuario puede visualizar los datos, una vez reducidos los datos N. La imagen se despliega en el área de Visualización, es entonces que se activan los botones que permiten la opción de almacenar la imagen en cualquiera de los dos formatos que se aprecian en la figura D6. Uno de ellos es el formato postcript (botón izquierdo) y el segundo formato es el jpeg (botón derecho). Al presionar cualquiera de los dos, enseguida se solicita el nombre con el cual el usuario desea almacenar su imagen en el SA.

Para que el proceso del análisis de datos pueda ser utilizado por el usuario, debe haber datos reducidos en la memoria de ViAn. Esto es, después de aplicarse la reducción, se activan los procesos para la visualización, y para el análisis de los datos reducidos. El análisis de los datos no depende del proceso de visualización. Si el usuario elige la opción de analizar datos con Fourier, se despliega una imagen en el área de Análisis y se muestran los datos numéricos en el área de resultados.

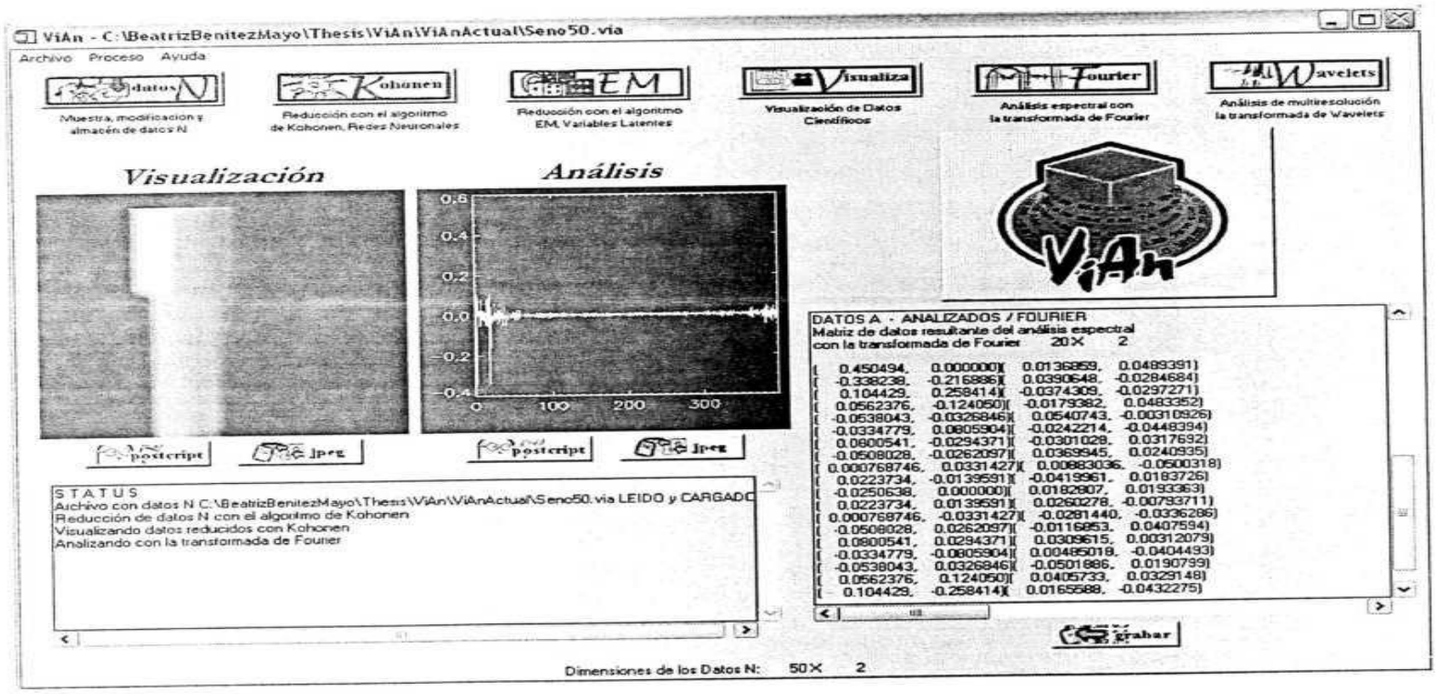


Figura D7. Interfaz para el análisis de los datos con la transformada de Fourier

Así se presenta en la figura D7. Pueden almacenarse en el SA los datos resultantes del análisis. En el caso de la matriz numérica, presionamos al botón "grabar" y enseguida se solicita el nombre del archivo. Para almacenar la imagen, existen dos opciones: el formato postcript (botón izquierdo) y el formato jpeg (botón derecho).

En el caso de optar por el análisis con la transformada Wavelets, aparecerá una referencia a lo que existe desarrollado en el ambiente IDL. En la figura D8 presentamos la ventana que el usuario observará al momento de elegir la transformada de Wavelets. Es la ventana de búsqueda en la documentación del IDL, donde ubicamos el paquete de wavelets en IDL "*IDL Wavelets Toolkit*". Por esta situación, el análisis de los datos reducidos con ambas transformadas simultáneamente lo proponemos para una futura versión del sistema ViAn.

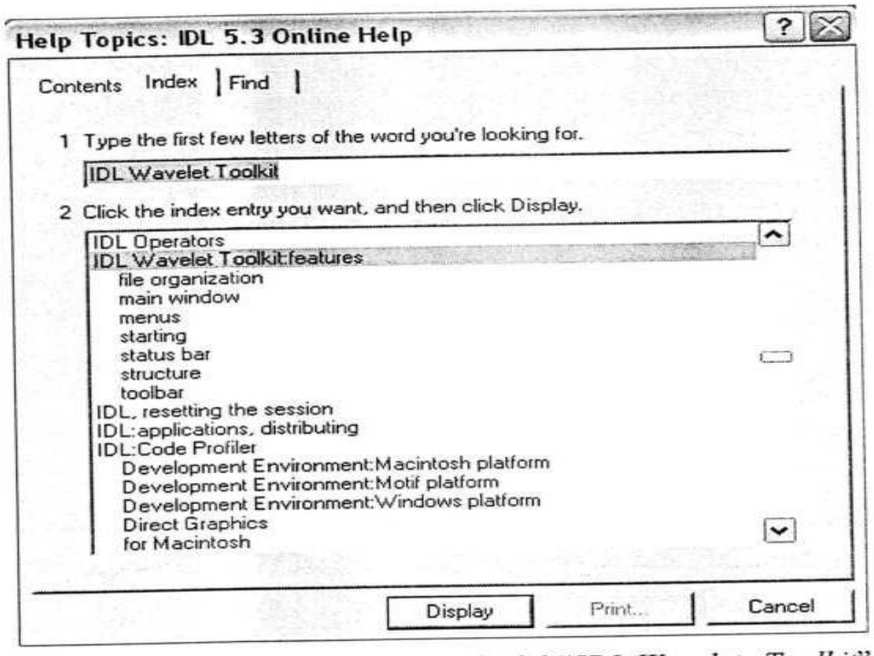


Figura D8. Interfaz para la búsqueda del "*IDL Wavelets Toolkit*"

En el proceso de reducción, mencionábamos que existe la opción de ejecutar ambos algoritmos (Kohonen y EM) y obtener simultáneamente dos matrices resultantes, las cuales se despliegan en el área correspondiente. Solo que, en la memoria del sistema, queda registrada ninguna matriz de los datos reducidos para su almacenamiento en el SA. Para almacenar los resultados, debe procesar con el algoritmo correspondiente de forma individual.

En la figura D9 presentamos la ventana que muestra el proceso de reducir con ambos algoritmos, y su visualización. Observamos las dos imágenes generadas con el proceso de visualización, siempre y cuando, el último proceso correspondiente a la reducción haya sido con ambos algoritmos simultáneamente. Desde dicha ventana, el usuario cuenta con la restricción de almacenar cualquiera de las imágenes en el SA. Solo podrá consultarlas y así compararlas.

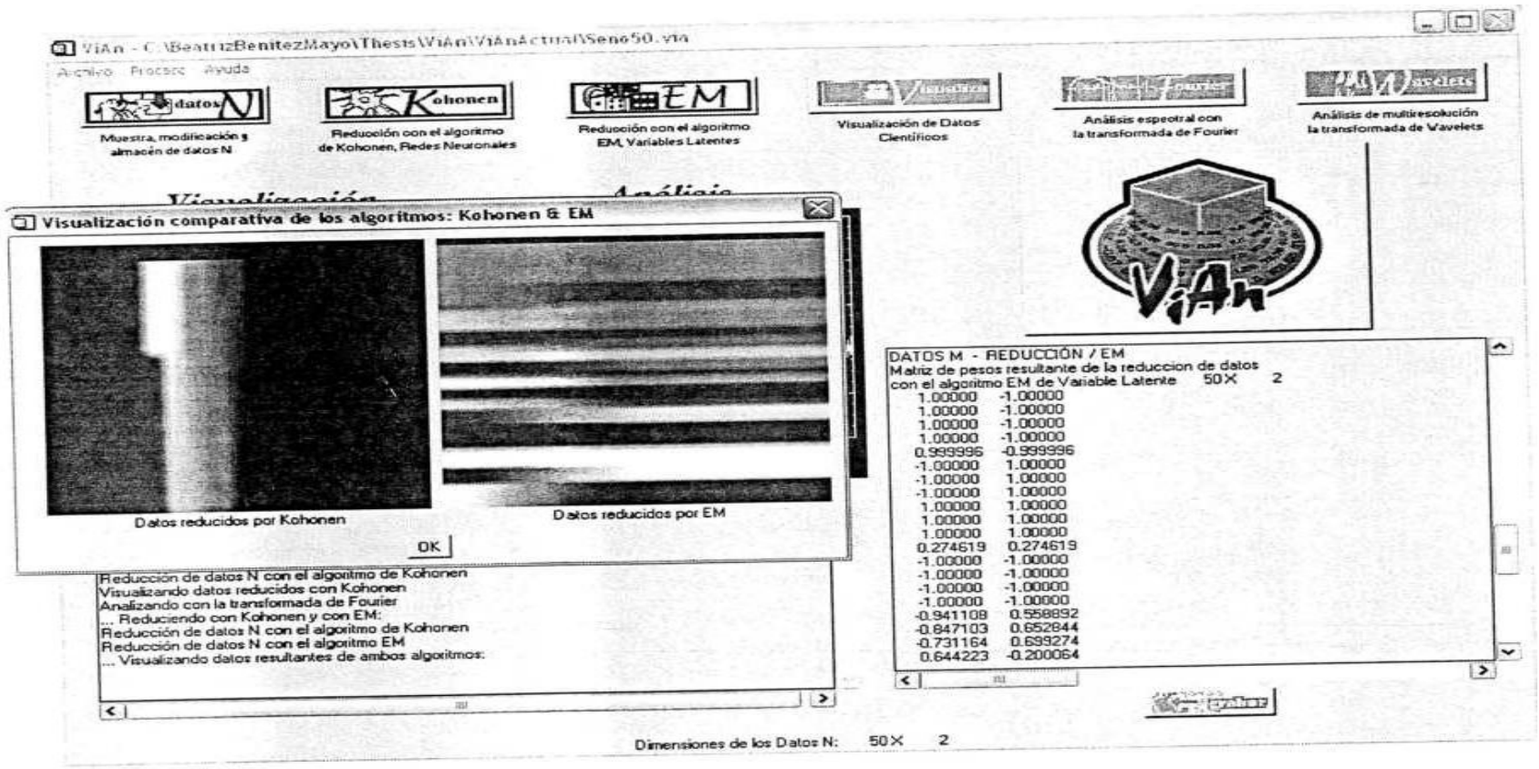


Figura D9. Interfaz para la visualización de datos
con la reducción previa de datos con ambos algoritmos (Kohonen y EM)

Literatura Citada

Literatura citada

Presento las referencias en las que nos apoyamos durante el desarrollo del proyecto de tesis. Lo que estudiamos para comprender y formar nuestros propios criterios, los conceptos e ideas que adoptamos, fueron tomados de publicaciones (artículos, tesis, cursos, manuales), sitios en internet y en libros. No toda la literatura que encontramos aquí la veremos como referencia en los textos de la tesis, ya que algunos de ellos nos sirvieron solo como lectura, como consulta o como estudio del contexto correspondiente.

Bibliografía

[Bishop, 1995] Christopher M. Bishop. ***"Neural Networks for Pattern Recognition"***. Clarendon Press - Oxford. New York, USA, 1995. 482 páginas.

[Booch] Grady Booch. "*Object-Oriented Analysis and Design*". Segunda edición. Prentice Hall. Addison-Wesley. 595 páginas.

[Burden, 1993] Richard L. Burden, J. Douglas Faires. ***"Numerical Analysis"***. Publishing Company. Boston, Massachussets. 1993. 768 páginas.

[Fanning, 2000] David W. Fanning. "***IDL Programming Techniques***", 2da. Edición. Fanning Software Consulting. Estados Unidos, 2000. 445 páginas.

[Fausett, 1994] Laurene Fausett. ***"Fundamentals of neural networks – architectures, algorithms, and applications"***. Prentice Hall International Editions. New Jersey, Estados Unidos. 1994. 461 páginas.

[Gumley, 2002] Liam E. Gumley. "*Practical IDL programming*". Morgan Kraufmann Publishers, Estados Unidos. 2002. 508 páginas.

[Jacobson, 2000] Ivar Jacobson, Grady Booch, James Rumbaugh. "***El proceso unificado de desarrollo de Software***". Addison Wesley. Madrid España, 2000. 438 páginas.

[Pressman, 1997] Roger S. Pressman. "***Ingeniería del Software, un enfoque práctico***", 4ta. Edición. Mc Graw Hill. España, 1997. 581 páginas.

[Rao, 1998] Raghuveer M. Rao, Ajit S. Bopardikar. "***Wavelet Transforms – Introduction to Theory and Applications***". Addison Wesley Longman. Massachusetts, Estados Unidos. 1998. 310 páginas.

[Rojas, 1996] Rojas, Raúl. *"Neural Networks. **A systematic Introduction**"*. Springer. Springer-Verlag Berlín Heiderberg. Alemania, 1996. 502 páginas.

Publicaciones

[Aldrich, 1998] Chris Aldrich. "***Visualization of transformed multivariate data sets with autoassociative neural network***". Pattern Recognition Letters, ELSEVIER, 1998.

[Bartfai, 1993] G. Bartfai, M. Elliffe, P. Wood. "***SOTA - a visualisation tool for ART neural networks***". Department of Computer Science, Victoria University, Wellington, New Zealand. 1993.

[Bishof] Horst Bishof, Axel Pinz, Walter G. Kropatsch. "***Visualization Methods for Neural Networks***". Department for Pattern Recognition and Image Processing, Technical University of Vienna.

[Bishop, Svensén, 1996] Christopher M. Bishop, Markus Svensén, Christopher K. I. Williams. "***EM Optimization of Latent-Variable Density Models***". Neural Computer. Research Group, Aston University, Birmingham, UK. 1996.

[Bishop, Tipping, 1998] Christopher M. Bishop, Michael E. Tipping. ***"A Hierarchical Latent Variable Model for Data Visualization"***. IEEE Transactions on pattern analysis and machine intelligence, Vol. 20, No.3. Marzo 1998.

[Burt, 1983] Peter J. Burt, Edward H. Adelson. "***The Laplacian Pyramid as a Compact Image Code***". IEEE. 1983.

[Carr, 2001] Matthew Carr, Richard Cooper, Maggie Smith, M. Turhan Taner, Joel Walls. "***The integration of surface seismic and borehole data using artificial neural network clustering methods***". ASEG 15th Geophysical Conference and Exhibition, Brisbane. 2001.

[Espadero, 1999] José Miguel Espadero, Luis Rincón. ***"Herramientas de modelado multirresolución de imágenes 3D utilizando wavelets"***. Departamento de Tecnología Fotónica y Departamento de Ciencias Experimentales, Universidad Politécnica de Madrid. España, 1999.

[Jain, 2000] Anil K. Jain, Robert P.W. Duin, Jianchang Mao. ***"Statistical Pattern Recognition: A Review"***. IEEE. Transactions on Pattern Analysis and Machine Intelligence. 2000.

[Kohonen, 1990] Teuvo Kohonen. "***The Self-Organizing Map***". Department of Computer Science, Helsinki University of Technology, Finland. 1990.

[Lippmann, 1987] Richard P. Lippmann. "***An Introduction to Computing with Neural Nets***". IEEE. 1987.

[Markus, 1998] TESIS: Johan Fredrik Markus Svensén. "*GTM: Generative Topographic Mapping*". Aston University. 1998.

[Manduca, 1994] Armando Manduca. "***Multi-Parameter Medical Image Visualization With Self-Organizing Maps***". Department of Physiology and Biophysics, Mayo Clinic and Foundation, Rochester, USA. Member IEEE. 1994.

[Meng, 2000] Zhuo Meng, Yoh-Han Pao. "***Visualization and self-organization of multidimensional data through equalized orthogonal mapping***". Computer. Associates Int. Inc., Independence, OH, USA. 2000.

[Munro, 1992] Paul W. Munro. "***Visualization of 2-D Hidden Unit Space***". Department of information science University of Pittsburgh, Pittsburgh PA. IEEE. 1992.

[Myklebust, 1995] Gaute Myklebust, John G. Solheim, Erik Steen. "***Speeding up small sized Self Organizing Maps for use in visualization of multiespectral medical images***". The Norwegian Institute of Technology, Trondheim, Norway. IEEE. 1995.

[Newman, 2000] Timothy S. Newman, Ning Tang. "***Approaches that exploit vector-parallelism for three rending and volume visualization techniques***". Department of Computer. Science, University of Alcfabama, 2000.

[Ornes, 1997] Chester Ornes, Jack Slansky. "***A neural network that visualizes what it classifies***". Department of Electronic & Computer. Engineering, University of California. 1997.

[Phillies, 1996] George D.J. Phillies. "***Wavelets: a new alternative Fourier Transform***". Department of Physis and Worcester Polytechnic Institute, USA. 1996.

[Serrano] Eduardo Pedro Serrano. "*Introducción a la transformada Wavelet y sus aplicaciones al procesamiento de señales de emisión acústica*". Escuela de Ciencia y Tecnología – Universidad Nacional de General San Martín.

[Soo-Chang, 2001] Pei Soo-Chang, Ding Jian-Jiun. "***Two Dimensional Affine Generalized Fractional Fourier Transform***". IEEE. Transactions on signal processing. 2001.

[1999] MANUAL: "***Getting Started with IDL***". Versión 5.3. Research Systems. Septiembre 1999. 206 páginas.

[2001] CURSO: Bioinformatics course supplement. ***"Latent Variable Models"***. SNU CSE Artificial Intelligence laboratory (SCAI). 2001.

Sitios en red

[Sitio 1]. An Introduction to Fourier Theory (Una introducción a la teoría de Fourier)
http://aurora.phys.utk.edu/~forrest/papers/fourier/

[Sitio 2]. Las Redes Neuronales Artificiales
http://ciberconta.unizar.es/LECCION/REDES/INICIO.HTML

[Sitio 3]. Introducción al método de Monte Carlo
http://csep1.phy.ornl.gov/mc/node1.html

[Sitio 4]. Self-Organizing Maps (Mapas de auto-organización). Tom Germano, 1999
http://davis.wpi.edu/~matt/courses/soms/

[Sitio 5]. Survey of Multiresolution Modeling (Examen del modelado de multiresolución)
http://graphics.cs.uiuc.edu/~garland/CMU/multires/survey.html

[Sitio 6]. Scientific Visualization with AVS (Visualización científica con AVS)
http://oscinfo.osc.edu/training/avs/AVS/AVS-schedule.html

[Sitio 7]. Scientific Visualization Studio, SVS (Estudio de la visualización científica)
http://svs.gsfc.nasa.gov/

[Sitio 8]. The Adaptive Resonance Theory (ART) clearinghouse (Cámara de compensación ART)
http://web.umr.edu/~tauritzd/art/index.html

[Sitio 9]. La transformada de Fourier
http://www.arrakis.es/~ppriego/fourier/transf_f.htm

[Sitio 10]. Information Visualization (Visualización de la información)
http://www.cc.gatech.edu/classes/AY2000/cs7450_fall/

[Sitio 11]. Laplacian Volume Pyramid (Pirámide del volumen Laplaciano)
http://www.cg.tuwien.ac.at/studentwork/VisFoSe98/bernd/laplac.htm

[Sitio 12]. Self-Organizing Maps In Natural Language Processing, Thesis. (Mapas de auto-organización en procesamiento del lenguaje natural, Tesis)
http://www.cis.hut.fi/~tho/thesis/index.html

[Sitio 13]. Kohonen Networks (Redes Kohonen)
http://www.cs.bham.ac.uk/~jlw/sem2a2/Web/Kohonen.htm

[Sitio 14]. Visualization (Visualización)
http://www.cs.uct.ac.za/courses/CS400W/Visualization/viscourse.html

[Sitio 15]. Conceptos básicos
http://www.gc.ssr.upm.es/inves/neural/ann2/concepts/concepts.htm

[Sitio 16]. Urban Image Analysis Using Adaptive Resonance Theory (Análisis de una imagen urbana utilizando la teoría de resonancia adaptativa.
http://www.gisdevelopment.net/aars/acrs/2000/ts9/imgp0002.shtml

[Sitio 17]. Redes Neuronales Artificiales
http://www.inf.udec.cl/~intartif/neuralnetworks.html

[Sitio 18]. Scientific Visualization at the Supercomputing Institute (Visualización científica en el instituto de supercómputo)
http://www.msi.umn.edu/user_support/scivis/scivis-list.html

[Sitio 19]. Scientific Visualization Laboratory (Laboratorio de visualización científica)
http://www.scivis.gatech.edu/

[Sitio 20]. Speech Technology Center (Centro tecnológico del lenguaje)
http://www.speechpro.com/eng/products/visvoice.html