Ciemat
Centro de
Investigaciones Energéticas,
Medioambientales
y Tecnológicas

Miner

Evaluación de productos informáticos para el Sistema de Adquisición de Datos del TJ-II

C. Crémy
E. Sánchez
A. Portas
J. Vega

Informes Técnicos Ciemat

809
Septiembre 1996

Informes Técnicos Ciemat

809
Septiembre 1996

Evaluación de productos informáticos para el Sistema de Adquisición de Datos del TJ-II

C. Crémy
E. Sánchez
A. Portas
J. Vega

Asociación EURATOM/CIEMAT para Fusión - 35

Instituto de Investigación Básica

Toda correspondencia en relación con este trabajo debe dirigirse al Servicio de Información y Documentación, Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas, Ciudad Universitaria, 28040-MADRID, ESPAÑA.

Las solicitudes de ejemplares deben dirigirse a este mismo Servicio.

Los descriptores se han seleccionado del Thesauro del DOE para describir las materias que contiene este informe con vistas a su recuperación. La catalogación se ha hecho utilizando el documento DOE/TIC-4602 (Rev. 1) Descriptive Cataloguing On-Line, y la clasificación de acuerdo con el documento DOE/TIC.4584-R7 Subject Categories and Scope publicados por el Office of Scientific and Technical Information del Departamento de Energía de los Estados Unidos.

Depósito Legal: M-14226-1995
NIPO: 238-96-001-0
ISSN: 1135-9420

Editorial CIEMAT

CLASIFICACIÓN DOE Y DESCRIPTORES

700320, 990200
DATA ACQUISITION SYSTEMS, TOKAMAK DEVICES, COMPUTER CODES, PROGRAMMING, COMPUTER PROGRAM DOCUMENTATION, THERMONUCLEAR REACTORS

"Evaluación de productos informáticos para el Sistema de Adquisición de Datos del TJ-II"

Crémy, C.; Sánchez, E.; Portas, A.; Vega, J.
34 pp. 8 figs. 21 refs.

Resumen

El Sistema de Adquisición de Datos del TJ-II debe proporcionar a sus usuarios un interfaz para realizar la programación de los canales de medida, visualizar señales de las descargas y realizar cierto procesado, todo ello durante la operación de la máquina. Por otra parte, se necesitan herramientas informáticas de alto nivel destinadas al análisis, al procesamiento y a la representación de los datos sea en tiempo de operación como en diferido. El presente informe resume la evaluación de un conjunto de paquetes informáticos llevada a cabo por el Grupo de Adquisición de Datos de la División de Fusión. Se han considerado los siguientes paquetes: **BuilderXcessory**, **X-Designer**, **Ilog Builder**, **Toolmaster**, **AVS 5**, **AVS/Express**, **PV-WAVE** e **Iris Explorer**. La evaluación de estos productos, resumida en el presente informe, ha permitido encontrar una solución global que responde a todos los requerimientos del Sistema de Adquisición de Datos del TJ-II.

"Software package evaluation for the TJ-II Data Acquisition System"

Crémy, C.; Sánchez, E.; Portas, A.; Vega, J.
34 pp. 8 figs. 21 refs.

Abstract

The TJ-II Data Acquisition System (DAS) has to provide a user interface which will allow setup for sampling channels, discharge signal visualization and reduce data processing, all in run time. On the other hand, the DAS will provide a high level software capability for signal analysis, processing and data visualization either in run time or off line. A set of software packages including **BuilderXcessory**, **X-Designer**, **Ilog Builder**, **Toolmaster**, **AVS 5**, **AVS/Express**, **PV-WAVE** and **Iris Explorer**, have been evaluated by the Data Acquisition Group of the Fusion Division. The software evaluation, resumed in this paper, has resulted in a global solution being found which meets all of the DAS requirements.

INDICE

I - Introducción.

El Sistema de Adquisición de Datos del dispositivo de fusión por confinamiento magnético TJII (SAD-TJII), debe proporcionar a sus usuarios dos niveles de servicio. Por un lado, una aplicación de tipo GUI (*Graphic User Interface*) que permita realizar la programación de los canales de medida del sistema, visualizar rápidamente el conjunto de las señales obtenidas durante la última descarga de la máquina y realizar cierto procesamiento sobre estas señales. Por otro lado, debe ofrecer herramientas de *software* de alto nivel destinadas al análisis, tratamiento y representación gráfica de datos experimentales, tanto durante la operación de la máquina como en diferido.

El presente informe describe la evaluación de productos informáticos llevado a cabo por el **Grupo de Adquisición de Datos** de la División de Fusión con el fin de encontrar una solución global a los requerimientos del SAD-TJII.

Dado que se hace uso de muchos términos ingleses (letra itálica), cuya traducción directa resulta difícil, se ha añadido un breve glosario que define tales términos en el contexto de esta presentación.

II - Requerimientos.

Las especificaciones funcionales del SAD-TJII y la solución informática (*hardware* y *software*) elegida para el servidor central del sistema (DEC ALPHA 8400) imponen ciertos requerimientos para los productos que se van a utilizar. Tienen que :

- estar disponibles, en versión depurada, para el sistema operativo DEC OSF/1 V2.0.
- funcionar en el entorno *X Window* [1][2] con interfaz gráfico de usuario *Motif* [3].
- permitir la representación gráfica de curvas x-y (1D), de curvas de nivel (2D), de volúmenes (3D) e incluso de visualizaciones 4D.
- no necesitar la presencia de aceladores gráficos en los *displays*.
- permitir la salida por impresora de gráficos.
- ser capaz de comunicar con los lenguajes de programación C [18] y Fortran [21].
- tener cierta facilidad de uso (en el caso de los productos destinados al análisis y a la visualización de datos).

Junto con las necesidades previamente enumeradas, sería muy deseable el disponer de capacidad operativa para realizar análisis estadístico, análisis espectral y procesado numérico de señales digitalizadas, todo ello de forma totalmente integrada.

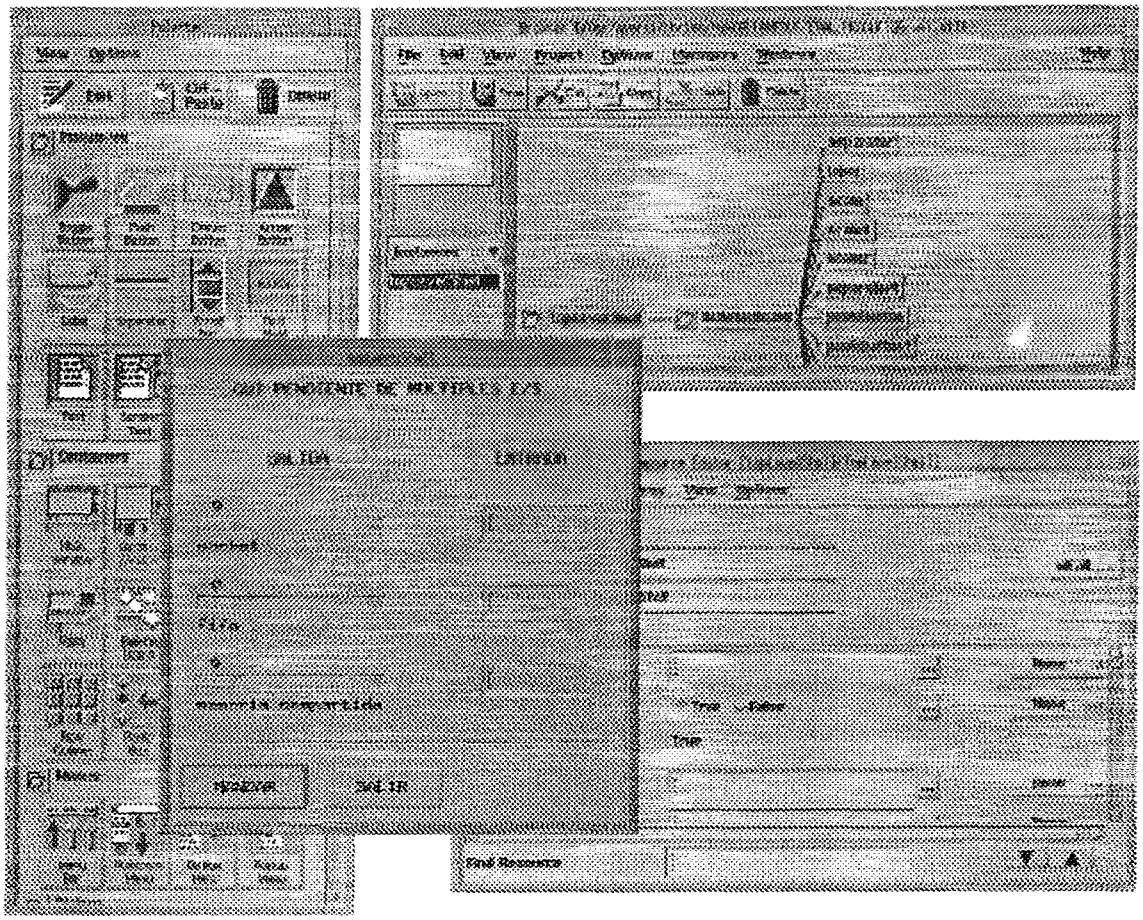

Figura nº 1 : Entorno de ventanas de builderXcessory

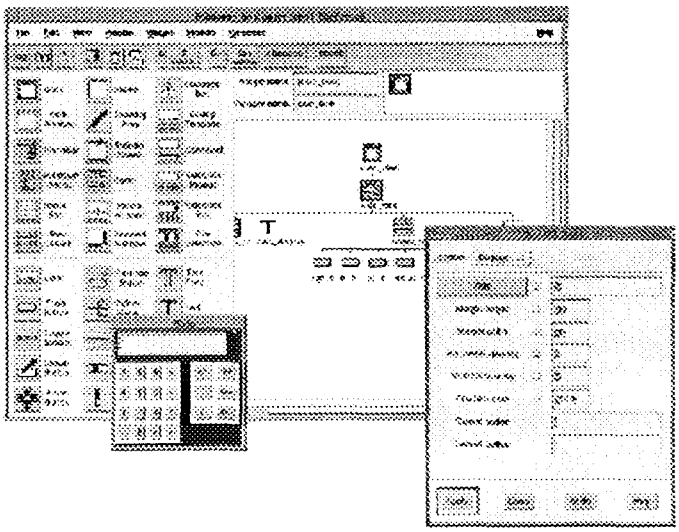

Figura nº 2 : Entorno de ventanas de X-designer.

III - Presentación de los productos evaluados.

Los productos evaluados se pueden agrupar en dos categorías. Los exclusivamente dedicados al desarrollo de GUIs, y los de análisis, tratamiento y representación gráfica; estos últimos pueden tener también capacidad para el desarrollo de GUIs.

En la primera categoría se encuentran :

- BuilderXcessory de **Integrated Computer Solutions Inc.** [4][5].
- X-designer de **Imperial Software Technology** [6].
- Ilog Builder de **Ilog S.A.**
- Toolmaster UIM/X de **Advanced Visual Systems Inc.** [7][8].

En la segunda categoría se encuentran:

- AVS 5 [9] [10] de **Advanced Visual Systems Inc.**
- AVS/Express de **Advanced Visual Systems Inc.**
- Toolmaster de de **Advanced Visual Systems Inc.**
- PV-WAVE de **Visual Numerics** [11][12].
- IDL de **Research Systems Inc.** [13].
- Iris Explorer de **Nag** [14] [15].

III.1 - Productos de desarrollo de interfaz de usuario.

Los cuatro productos presentados a continuación tienen una estructura y unas funcionalidades muy parecidas, de tal forma que la presentación del primero es más exhaustiva que la de los siguientes, para los cuales se citarán únicamente sus particularidades.

III.1.a - BuilderXcessory.

BuilderXcessory es una herramienta de desarrollo de GUIs que utiliza ***OSF/Motif*** y los ***Xt Intrinsics***. Este producto es un GUI por sí mismo que permite el desarrollo interactivo de una aplicación, su depuración antes de la compilación y la generación del código fuente.

El desarrollo de una aplicación se hace a partir de un conjunto de ventanas con las cuales, el desarrollador va a interactuar **(c.f. figura nº 1).** Este último dispone de una ventana principal que consta de tres partes: un menú que da acceso a todas las funcionalidades del producto, una zona donde se va a representar gráficamente la jerarquía de la aplicación y una zona donde aparecerán informaciones generales y mensajes de error. Otra ventana, llamada ***Widget Palette*** , contiene el conjunto de los *widgets* disponibles en el producto, ordenados en categorías (*primitives, containers, dialogs,* etc.). Por último, un editor permite visualizar y modificar los recursos (características geométricas, color, comportamiento, *callbacks, handlers,* etc.) del *widget* seleccionado.

El desarrollador, entrando en el ***Play Mode***, puede comprobar la apariencia (*look and feel*) de su aplicación antes de la fase de compilación. Sin embargo, este modo no permite verificar el funcionamiento de la aplicación ya que sólo se puede pueden llamar a *callbacks* predefinidos. A este efecto, builderXcessory tiene un modo de depuración, el ***Debug Mode***, donde se pueden usar herramientas como el **CodeCenter** y el **ObjectCenter** para completar la verificación antes de la compilación. También se puede integrar en entornos de desarrollo como **SunSoft WorkShop**, **DEC FUSE** o **CASEVision/WorkShop** de Silicon Graphics, dando acceso al sistema de control de código fuente **(SCCS)**.

BuilderXcessory permite una programación orientada a objetos en el sentido de que se pueden crear "trozos" de código modulares y reutilizables. El desarrollador puede crear colecciones, estilos y clases para agrupar *widgets* ademas de definir constantes, identificadores, tipos y procedimientos. Además la tarea del desarollador puede facilitarse gracias a un conjunto de rutinas de utilidad.

Una vez desarrollada la aplicación, se puede generar el código fuente en C, C++, y ***UIL*** (***OSF/Motif User Interface Language***). BuilderXcessory puede leer ficheros previamente creados en el formato *UIL* y también importar ficheros en formato *GIL* (***Guide Interchange Language***). El código generado se encuentra repartido en varios ficheros fuentes, separando la rutina principal, de las rutinas de creación de los *widgets*, de las rutinas de los *callbacks*, etc. También se genera un fichero de tipo *makefile* que permite la compilación y la edición de enlaces del conjunto de los ficheros de manera amigable.

III.1.b - X-designer.

X-designer es una herramienta interactiva para construir interfaces de usuario utilizando los *widgets* de ***OSF/Motif*** o los de otros *toolkits* de X; puede incluso generar aplicaciones para ***Microsoft Windows***.

Este producto tiene una estructura muy parecida a la de builderXcessory, con una paleta de *widgets*, una zona donde el desarrollador va a construir la jerarquía de *widgets* correspondiente a su aplicación y unos editores de recursos **(c.f. figura nº2)**.

X-designer se diferencia de BuilderXcessory en que no tiene capacidad para probar la aplicación antes de la compilación.

III.1.c - Ilog Builder.

Ilog es un paquete informático destinado al desarrollo de aplicaciones en lenguaje C++. El presente estudio se ha hecho sobre el componente Ilog Builder, que permite el desarrollo de GUIs en el entorno ***OSF/Motif*** en C, C++ y UIL.

Al igual que builderXcessory y X-designer, Ilog Builder es una herramienta de desarrollo interactivo de GUIs y de generación de código fuente.

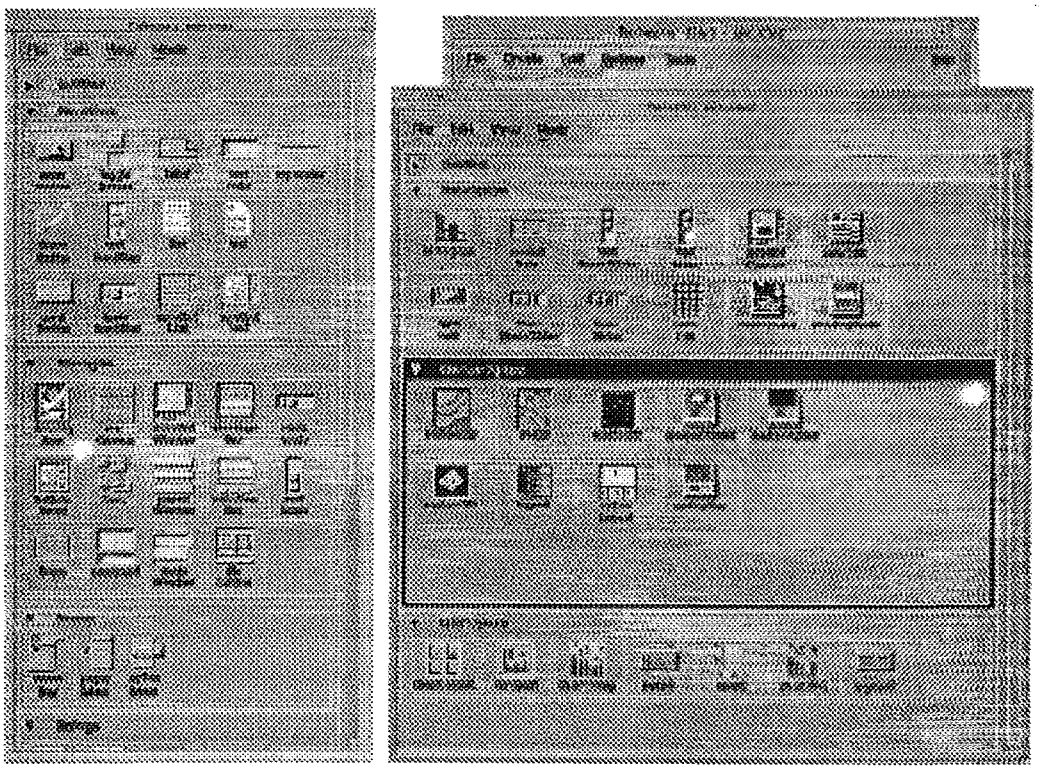

Figura nº 3 : Entorno de desarrollo de Toolmaster UIM/X.

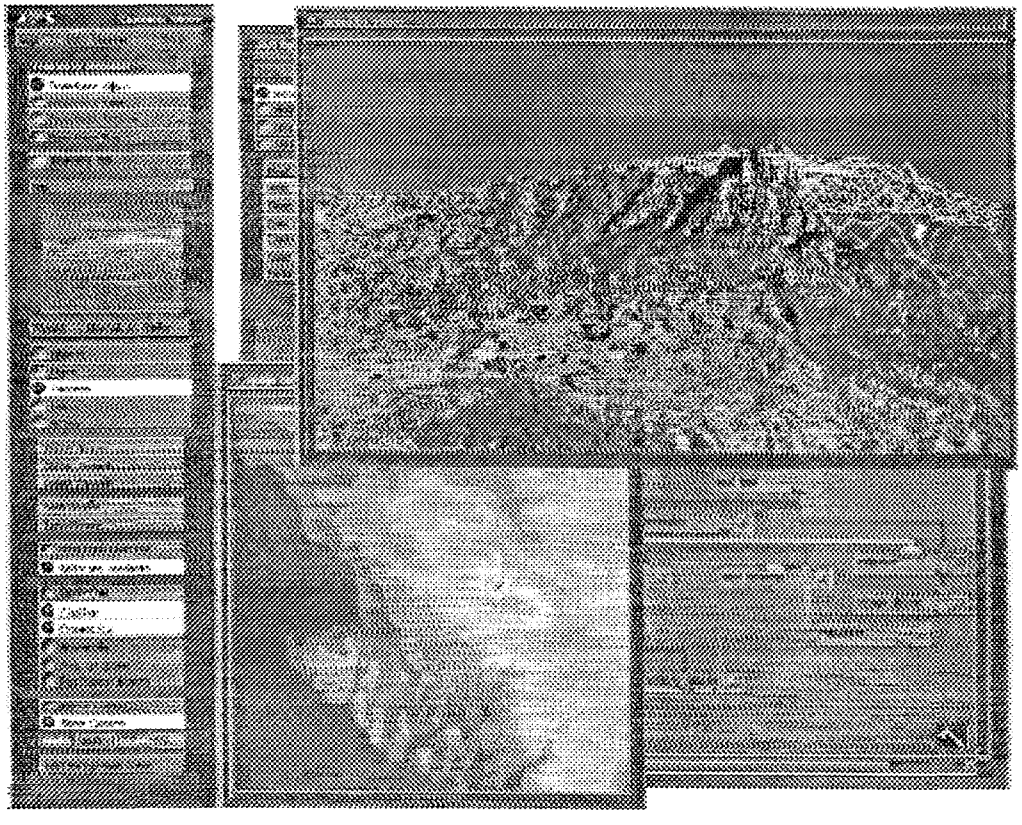

Figura nº 4 : Entorno de Visualización de AVS 5.

A diferencia de builderXcessory, Ilog Builder dispone de un modo de prueba que permite compilar y depurar la aplicación durante la fase de desarrollo.

III.1.d - Toolmaster UIM/X.

Toolmaster UIM/X es un componente del paquete Toolmaster destinado a la construcción de GUIs. Es muy similar a los productos descritos anteriormente.

Se diferencia por una parte, en el hecho de disponer de dos paletas de *widgets*: la paleta de los *widgets* de ***OSF/Motif***, común a todos los generadores de GUIs y la paleta de *widgets* de los otros componentes del paquete Toolmaster, es decir BaseXplore, ChartXplore y ContourXplore **(c.f. figura nº3)**.

Por otra parte, dispone de editores que permiten al desarrollador escribir y editar el código fuente (en C) de los *callbacks*, *handlers* u otras rutinas.

También dispone de un juego de rutinas de utilidad para gestionar, por ejemplo, los identificadores de *widgets*.

III.2 - Productos de análisis, tratamiento y representación gráfica de datos.

A continuación se introducen los productos de la segunda categoría. Se les hace una presentación general seguida de una descripción de los modelos de datos que manejan y de un resumen de sus funcionalidades, con particular énfasis en lo que concierne a las comunicaciones con lenguajes de programación.

III.2.a - AVS 5.

Presentación

AVS 5 es un producto interactivo de programación visual, orientado al manejo de módulos, cada uno de ellos realizando una tarea particular, con la posibilidad de interconectarlos. El usuario tiene a su disposición una ventana, llamada ***Network Editor***, que consta de tres partes **(c.f. figura nº 4)** :

- Una zona de menú principal para seleccionar el modo de funcionamiento.

- Una ventana llamada *Module Library*; es una paleta de botones colocados en columnas y que corresponden a los módulos disponibles con el producto.

- Una zona de edición, cuya funcionalidad consiste, por una parte, en posicionar los módulos previamente seleccionados en la paleta de botones, y por otra, en crear conexiones, formando así una estructura jerárquica o en otras palabras, un mapa de módulos.

Típicamente, un mapa contendrá un módulo de entrada para la lectura de datos brutos, unos cuantos módulos de tratamiento y uno o varios módulos de salida de representación gráfica. Una vez hecha la edición del mapa, el usuario pasa a modo de ejecución donde, por ejemplo, va a elegir un fichero de entrada mediante un *File Browser;* la selección de dicho archivo va a arrancar un "flujo" de datos que se propaga de módulo en módulo, hasta llegar a los últimos nodos del mapa. Visualmente, este flujo se materializa con un cambio de color de los módulos activos y de los enlaces.

Formatos y Entrada/salida de Datos

AVS 5 permite el manejo de varios tipos de datos los cuales se describen a continuación :

• El tipo *scalar* que corresponde a los enteros, reales, booleanos y las cadenas de caracteres.

• El tipo *field* que permite importar arreglos multidimensionales (1D, 2D, 3D, etc.) con tres tipos de mapeo de las coordenadas :

- *uniform* : hace un mapeo directo.
- *rectilinear* : asocia a cada dimensión un vector de coordenadas.
- *irregular* : es un mapeo explícito para cada elemento del arreglo.

• El tipo *Geometric Data* que permite el manejo de objetos en 2D y 3D.

• El tipo *Molecular Data* que da soporte para aplicación en química.

• El tipo *Colour Map* que corresponde a un arreglo 1D de vectores 4D cuyos componentes representan el tono, la saturación, la luminosidad y la opacidad de un color.

• El tipo *User defined* que permite la declaración de nuevos tipos de datos en el lenguaje C.

AVS 5 dispone de un conjunto de módulos de entrada, o *readers*, que permiten importar datos desde varios formatos estándares de representación (Postscript, TIFF, GIF, etc.) y transformarlos en los tipos internos descritos previamente. También dispone de módulos de salida, o *writers*, que realizan la tarea inversa. Además, el ADIA (*AVS Data Interchange Application*) permite realizar la importación de formatos no definidos.

Para la impresión de documentos gráficos e imágenes, AVS dispone de dos *drivers* Postscript : uno en escala de gris de 8 *bits* para las impresoras blanco y negro, y el otro en *True color* de 24 *bits* para las de color.

Visualización

En cuanto a la visualización de datos, AVS 5 incluye las funcionalidades siguientes:

• La extracción de superficie dentro de un volumen 3D (***slicing*** ortogonal y con cualquier orientación).

• La generación de isosuperficies desde un arreglo 3D de elementos escalares.

• El *rendering* de volúmenes.

• La construcción de contornos.

• La visualización de imágenes (conversión de valores en colores).

• La visualización 2D de datos 1D (*plots* X-Y, *scatters*, barras, gráfico en porciones, etc.) mediante un módulo llamado **AVS/Graph**.

También da la posibilidad de realizar ciclos de animación utilizando un módulo que genera un flujo de datos u otros módulos que pueden representar secuencialmente un conjunto de imágenes o de gráficas.

Comunicación con otros lenguajes

AVS 5 da soporte para la integración de código escrito en C (K & R , ANSI), C++ y Fortran 77. Esta integración se hace mediante la creación de nuevos módulos que pueden ser de dos tipos :

•El tipo ***subroutine*** que corresponde a módulos que serán controlados por el *kernel* de la aplicación y estarán implicados en el flujo de datos.

•El tipo *coroutine* que corresponde a módulos que se ejecutarán independientemente del *kernel* y que darán lugar a Entradas/Salidas asíncronas.

Un módulo generador de interfaz facilita la creación de nuevos módulos. A partir de una lista de especificaciones va a crear un archivo que contendrá el esqueleto del módulo que se quiere desarrollar.

En cuanto al intercambio de datos entre módulos, se pueden realizar comunicaciones, en el mismo *host*, vía *sockets* o a través de memoria compartida.

III.2.b - AVS/Express.

Presentación

AVS/Express es un entorno de desarrollo de aplicaciones que utiliza una programación visual orientada a objetos. Este producto proporciona una tecnología para manejar y visualizar gráficas avanzadas e imágenes y para representar datos. AVS/Express es un entorno abierto y extensible, que es transportable a muchas plataformas, tanto estaciones de trabajo como PCs.

Como en el caso de AVS 5, la programación visual consiste en la creación de un mapa de módulos cuya estructura aparece representada gráficamente.

Funcionalidades

AVS/Express proporciona una serie de juegos de herramientas que incluyen estructuras de datos y métodos para la visualización, el análisis de datos y el procesado de imágenes.

El componente *Graphic Display Kit* posee componentes para el *rendering* y la manipulación de textos, imágenes y objetos geométricos 2D y 3D. Estos componentes son: elementos de visualización, interfaces con estos elementos, primitivas de *rendering* y operadores de *picking*. Además, estos componentes se pueden reconfigurar, personalizar y extender. Los resultados obtenidos con este componente son óptimos si se utilizan aceleradores gráficos.

El componente *Data Visualization Kit* contiene objetos y estructuras de datos, y librerías diseñadas para visualizar y analizar datos científicos. Estos incluyen: técnicas escalares (isosuperficies, *slice*, y curvas de nivel), técnicas vectoriales (*stream lines* y *particle advection*), análisis de datos (muestreo y estadísticas) y proceso de datos (recogida de datos y análisis multivariante). Los objetos de la interfaz de visualización son polimórficos, de forma que pueden manejar tanto datos simples (octetos, enteros, caracteres, reales, etc.), como datos compuestos (estructurados, no estructurados y rectilíneos).

El componente *Image Kit* es una librería de funciones para analizar y manipular imágenes. Trabaja sobre imágenes simple o multibandas, sobre bandas individuales de imágenes múltiples y sobre subimágenes o regiones de interés. Las funciones de esta interfaz son de análisis (muestreo, cálculo de histograma, etc.), aritméticas (operadores lógicos de punto flotante), edición, filtrado, operaciones geométricas y transformadas rápidas de Fourier.

Comunicación con otros lenguajes

La integración del código fuente en C, C^{++} y Fortran se hace mediante la creación de nuevos objetos que se pueden integrar en el producto.

Creación de interfaz de usuario

AVS/Express dispone de un constructor de GUIs, llamado ***User Interface Kit,*** que proporciona a los desarrolladores los componentes necesarios para llevar a cabo la construcción y la organización de una aplicación de tipo GUI en los entornos ***OSF/Motif*** y ***Windows***. Este componente incluye, además de los *widgets* de ***OSF/Motif*** y ***Windows,*** un conjunto de *widgets* de mas alto nivel que integran capacidades numéricas y permite la integración de otras librerías de *widgets* simplemente incorporando otros objetos.

III.2.c - ToolMaster.

Presentación

Toolmaster es un paquete informático de desarrollo de aplicaciones gráficas que contiene los elementos siguientes :

• Toolmaster agX que es una librería gráfica en el entorno ***X Window.***

•Toolmaster BaseXplore que es una librería de *widgets* (basados en ***OSF/Motif***) y que completa las funcionalidades de Toolmaster agX.

• Toolmaster ChartXplore que es una librería de *widgets* y *gadgets* (también basados en ***OSF/Motif***) y diseñados para la visualización de *plots* x-y.

• Toolmaster CountourXplore que es una librería de *widgets* (nuevamente basados en ***OSF/Motif***) y diseñados para la representación de curvas de nivel en 2D, y 3D.

• Toolmaster UIM/X que es un generador de GUIs.

Toolmaster agX

Toolmaster agX es una librería de más de quinientas funciones que extiende el sistema ***X Window*** con rutinas gráficas de alto nivel y con capacidad de generación de archivos imprimibles. Esta librería soporta un amplio rango de formatos de datos y tipos de gráficas.

Toolmaster agX proporciona al usuario un control sobre la presentación y la apariencia, (ejes, leyendas, anotaciones, flechas, etc.) de las gráficas, que pueden ser lineas, barras, porciones, isolíneas, contornos 2D y 3D, etc.

La librería agX incluye las particularidades de ***X Window*** como el uso de un gestor de recursos, de contextos gráficos y de bucles de eventos ademas de permitir la salida en varias ventanas. Las llamadas a agX pueden combinarse con las capas subyacentes que forman ***Xlib, Xt Intrinsics*** y ***OSF/Motif.***

Toolmaster BaseXplore

Toolmaster BaseXplore es una librería de *widgets* de tipo *canvas* de editores de recursos y de *widgets* de control. Esta librería implementa el manejo de los eventos X y las operaciones interactivas dejando bajo responsabilidad del usuario la gestión de la visualización y de las otras funcionalidades de la aplicación.

Toolmaster ChartXplore

Toolmaster ChartXplore es un librería orientada a objetos de *widgets* y *gadgets* que extiende el *OSF/Motif* con funcionalidades para crear *charts* con varios juegos de datos, *view ports*, ejes, leyendas, anotaciones.

ChartXplore permite la implementación de operaciones de *picking* y de modificación sobre los datos y los componentes del *chart* y proporciona un juego importante de recursos para configurar su presentación. Por ejemplo, el AxisGadget tiene mas de cuarenta recursos para definir un eje.

Toolmaster ContourXplore

Toolmaster ContourXplore constituye una librería de *widgets* destinados a la exploración de datos de tipo *scatter* como contornos 2D, 3D y 4D. El diseño jerárquico de esta librería provee al usuario tanto de *widgets* compuestos con capacidad de edición mediante el ratón, como de los *widgets* gráficos con los cuales éstos están construidos.

Las técnicas avanzadas de manejo de contornos de esta librería dan al usuario la posibilidad de asociar un contorno a una región de los datos y de definir la resolución de *gridding* en función de la concentración de datos por ejemplo. La selección de datos permite operar sobre puntos, superficies y *slices* arbitrarios, a través de una malla (*grid*). ChartXplore provee también una amplia gama de recursos para el control de los ejes, de las mallas, de las leyendas, etc.

Comunicación con otros lenguajes

Las librerías de Toolmaster son librerías dinámicas y pueden enlazarse a programas en C y en Fortran. El inconveniente que presenta es la necesidad de licencia *run-time* para la ejecución de dichos programas.

Generación de GUIs

Toolmaster UIM/X, presentado en el párrafo III-1 es un componente del paquete Toolmaster.

III.2.d - PV-WAVE.

Presentación

PV-WAVE es un entorno visual e interactivo que permite la exploración, la manipulación y el análisis de datos numéricos. Este producto viene provisto de un conjunto de funciones gráficas, matemáticas y estadísticas, de funciones de tratamiento de imágenes, procesado de señal y herramientas de animación. Estas funciones están repartidas en varios módulos que forman el paquete informático PV-WAVE. Estos son:

- PV-WAVE: *Command Language*
- PV-WAVE: *Advantage*
- PV-WAVE: *Signal Processing Toolkit*
- PV-WAVE: *Point&Clik*
- PV-WAVE: *GT Grid*
- PV-WAVE: *Data Base Connection*
- PV-WAVE: *Maple*
- PV-WAVE: *UIM/X*

Intérprete de comandos y formatos de datos

En el corazón de PV-WAVE se encuentra el lenguaje de comandos (*Command Language* o CL), un lenguaje de cuarta generación (4GL) que representa el interfaz de usuario por defecto, para el acceso al conjunto de funcionalidades del paquete.

Este lenguaje permite, el manejo de tipos de datos estándares (escalares, ya sean enteros, reales, booleanos), arreglos multidimensionales y estructuras, así como el uso de modelos específicos como el *Asociative Array*, que corresponde a arreglos de variables de tipo diferente o de expresiones. La gestión de los datos durante una "sesión" con el CL es absolutamente dinámica en el sentido de que un identificador puede cambiar de tipo a medida que se llama a nuevos comandos.

Además de teclear los comandos uno por uno, el usuario dispone de otros mecanismos para llamar a las funciones disponibles con el CL. Por una parte, puede escribir el conjunto de sus comandos en un fichero de texto y posteriormente ejecutarlo como un *script*. Por otra parte puede escribir, siempre en un fichero de texto, un procedimiento o un programa completo con los comandos del CL, e invocar posteriormente al producto PV-WAVE para compilar y ejecutar el procedimiento desde el intérprete.

Funcionalidades

La funcionalidades de PV-WAVE se encuentran repartidas en los varios módulos previamente citados. A continuación se describen cada uno de estos elementos:

PV-WAVE *Command Language* da acceso al intérprete del CL, que viene provisto de las funciones siguientes:

- funciones matemáticas simples:
  - funciones trigonométricas elementales
  - logaritmos, funciones exponenciales
  - funciones de Bessel, etc.

- funciones estadísticas:
  media, desviación típica, *skewness*, *kurtosis*, etc.

- funciones de tratamiento de señal:
  FFT, transformada con *Wavelets*.

- instrumentos de trazado de curvas x-y, de contornos, de superficies y de visualización de volúmenes y de gráficas 4D.

El CL de PV-WAVE permite la importación y la exportación, tanto de datos formateados (ASCII, *bitmaps*, EPSI, XDR portable, HDF, etc.), como de datos sin formato. El CL también proporciona *drivers* para la impresión de documentos en los formatos Postscript, CGM, SIXEL, etc.

PV-WAVE: *Advantage* combina las funcionalidades de PV-WAVE *Command Language* y las librerías IMSL "c/Math/library" y c/Stat/library". Este elemento del paquete viene provisto de rutinas numéricas que realizan las funciones siguientes :

- Análisis de sistemas lineales y no lineales.
- Análisis de autovectores.
- Interpolación y la aproximación de datos.
- Integración y la diferenciación numérica de funciones.
- Resolución de ecuaciones diferenciales.
- Resolución de ecuaciones no lineales.
- Minimización de funciones.
- Cálculo de transformadas (FFT real o compleja, convolución discreta, correlación discreta).
- Uso de funciones especiales (Gamma, Beta, Bessel, etc.).
- Computo de estadísticos y la generación de números aleatorios.
- Uso de funciones de densidad de probabilidad estándares.
- Análisis de varianza.
- Cómputo de la correlación y de métodos de regresión.
- Uso de modelos ARMA.
- Análisis multivariante.

PV-WAVE: *Signal Processing Toolkit* es una colección de funciones de procesado digital de señales que completa PV-WAVE Advantage con las funciones siguientes :

- Modelado y análisis de señales y de sistemas.

- Aproximación y realización de filtros.
- Análisis espectral y computo de transformadas.
- Procesado estadístico de señales.
- Manejo de polinomios.
- El cálculo de filtros óptimos y uso de técnicas de minimización.
- Generación y representación gráfica de señales.

PV-WAVE: ***Point&Click*** es una interfaz gráfica preconstruida que permite un acceso "amigable" a casi la totalidad de las funciones de PV-WAVE. Este elemento ha desaparecido en la versión 6.0 del paquete y ha sido reemplazado por un conjunto de pequeñas aplicaciones cerradas, los ***VDA Tools,*** que se pueden llamar desde el CL. Esta versión también incluye una aplicación en forma de barra de ***Push Buttons,*** llamada ***Navigator,*** que permite un acceso amigable a los principales ***VDA Tools.***

PV-WAVE: ***Gt-Grid*** ofrece funciones de *gridding* y permite gestionar discontinuidades en los datos.

PV-WAVE: ***Database Connection*** permite a aquellos que utilizan Oracle y Sybase acceder directamente a sus bases de datos mediante comandos SQL.

PV-WAVE: ***Maple*** da acceso a funciones matemáticas simbólicas y numéricas. Es un sistema interactivo que permite, tanto el reconocimiento de ecuaciones y de notaciones matemáticas estándares, como la resolución de sistemas de ecuaciones diferenciales y el cálculo de transformadas. Este producto genera un código en C y en el CL de PV-WAVE que puede posteriormente utilizarse en otra aplicación.

Construcción de interfaz de usuario

PV-WAVE proporciona dos mecanismos para la creación de GUIs. Por una parte, a partir del CL, se dispone de un juego de *widgets* que corresponde a dos capas *software* por encima de la correspondiente al interfaz gráfico de usuario (***Motif, Openlook,*** etc.). La capa superior permite la creación de GUIs de manera relativamente simple, evitando la complejidad (y perdiendo parte de flexibilidad) de la programación del nivel de ***OSF/Motif*** o de los ***Xt Intrinsics,*** mientras que la capa inferior, de uso mas difícil, da acceso a gran parte de sus funcionalidades.

Por otra parte, existe un módulo llamado PV-WAVE UIM/X que es un generador de GUIs, como los productos presentados en el párrafo III-1. Sin embargo, este producto no está disponible para máquinas de arquitectura ALPHA AXP.

Comunicación con otros lenguajes

El CL de PV-WAVE proporciona varios mecanismos para la comunicación con códigos escritos en C y en Fortran.

Por una parte, es posible llamar a rutinas C y Fortran desde un programa escrito en el CL, mediante dos llamadas :

• El comando *SPAWN* crea un proceso para la ejecución de un comando y permite una comunicación a través de *pipes*.

• El comando *LINKNLOAD*, permite establecer un enlace dinámico de una rutina previamente compilada y presente en un fichero objeto. Este mecanismo sólo permite el paso de parámetros a la función y la devolución de un valor de retorno de parte de esta función.

Por otra parte, es posible llamar a comandos de CL de PV-WAVE desde un programa escrito en C o en Fortran mediante dos funciones :

• La llamada a *wavecmd* crea un proceso hijo que va a ejecutar el interprete de comandos de PV-WAVE. Sólo se crea un *pipe* unidireccional hacia el intérprete.

• La llamada a *cwavec* (resp. *cwavefor*) permite a un programa C (resp. Fortran) utilizar los archivos de PV-WAVE como librería estática y compartir un juego de variables durante su ejecución.

En cuanto a la compartición de volúmenes importantes de datos, la función *wavevars* permite a un programa C recuperar la totalidad de la variables definidas en una aplicación PV-WAVE. Para los programas Fortran, no existe una implementación directa sino un interfaz que realiza un enlace entre Fortran y la función *wavevars*.

Un último mecanismo de comunicación es la interoperabilidad de procesos en tiempo real que se puede hacer mediante RPCs, *named pipes* y DDE.

III.2.e - El IDL.

La presentación del producto IDL va a ser muy breve ya que este producto no es más que una implementación ligeramente diferente de PV-WAVE Advantage. Se corresponde con las primeras versiones del producto PV-WAVE, por lo que presenta unas funcionalidades inferiores a las de este producto.

IDL se diferencia sobre todo por el hecho de que no contiene las librerías IMSL, sino otras librerías llamadas *Mathematical Recipes*. Tampoco tiene librerías especialmente destinadas al procesado de señales.

La versión evaluada no dispone de un equivalente de los *VDA Tools* y resulta menos flexible en cuanto a la comunicación con otros lenguajes **(c.f. párrafo IV-4)**.

III.2.f - Iris Explorer.

Presentación

Iris Explorer es un sistema para la creación de mapas de visualización a partir de pequeñas herramientas de *software* llamadas módulos. Un mapa puede definirse como una colección de módulos que realizan una serie de operaciones sobre un juego de datos y produce una representación visual del resultado. Es un producto cuyo manejo es muy similar al de AVS 5.

Iris Explorer consta de cuatro componentes principales :

- El ***Map Editor***, que es una zona de trabajo donde se van a crear y desarrollar los mapas.

- El ***Module Librarian***, donde se van a seleccionar los módulos durante el desarrollo de un mapa.

- El ***DataScribe***, que es una utilidad de conversión para transformar datos entre Iris Explorer y otros formatos de datos.

- El ***Module Builder***, que permite la creación de nuevos módulos.

Formatos y Entrada/Salida de Datos

Iris Explorer opera sobre tipos de datos similares a los manejados por AVS 5:

- El tipo ***Parameter***, que corresponde a los enteros, reales y las cadenas de caracteres.

- El tipo ***Lattice***, que permite representar arreglos multidimensionales (1D, 2D, 3D, etc.) con tres tipos de coordenadas (uniformes, perimétricas, y curvilíneas):

- El tipo ***Pyramide***, que permite el modelado por elementos finitos y el modelado de moléculas.

- El tipo ***Pick Data***, que permite la selección de una parte de una imagen.

- El tipo ***User Defined Data***, que permite la declaración de nuevos tipos de datos en el lenguaje ETL (***Explorer Typing Language***), parecido al C.

Funcionalidades

Los módulos de Iris Explorer proporcionan funcionalidades para realizar :

- La representación 2D de datos 1D gracias a tres módulos :

• El *Gnuplot*, que es una implementación del programa interactivo de representación gráfica **gnuplot 3.0**.

• El *Graph*, que permite dibujar curvas x-y con varios canales de entrada.

• El *NagGraph*, que también es un módulo de representación de curvas x-y aunque más avanzado, por ejemplo en la configuración de los ejes.

• La visualización de superficies sombreadas y de imágenes 2D.

• El *rendering* de Volúmenes.

• El *slicing* de datos.

• El cálculo de contornos e isosuperficies.

• La salida de documentos en ***Encapsulated Postscript*** en archivo o directamente por impresora.

Comunicación con otros lenguajes

Como en el caso de AVS 5, la integración de código fuente del usuario puede hacerse mediante la creación de nuevos módulos. Iris Explorer da soporte para la integración de código escrito en C y en Fortan. Las rutinas del ***DataScribe*** permiten el mapeo de los parámetros de las funciones de usuario a los tipos de Iris Explorer. Luego, el ***Module Builder*** realiza la compilación del nuevo módulo y su integración en el ***Module Librarian***. A diferencia de AVS 5, todos los nuevos módulos están bajo el control del *kernel* y contribuyen al flujo de datos.

Iris Explorer también da soporte para la ejecución remota de programas y permite el intercambio de datos entre módulos a través de memoria compartida.

IV - Principales pruebas realizadas.

IV.1 - Introducción.

Los productos presentados en el párrafo anterior, han sido instalados en una estación de trabajo DEC 3000 ALPHA 600 AXP, que funciona bajo el sistema operativo DEC OSF/1 V2.0. Las pruebas se han desarrollado por un lado, directamente desde la consola de la estación es decir con un verdadero entorno ***X Window*** con ***OSF/Motif***, y por otro lado, desde un Macintosh PowerPC que dispone de un programa de emulación X (eXodus [17][18]).

Se ha dispuesto de licencias de demostración de alrededor de dos semanas para llevar a cabo la evaluación de los productos AVS 5, AVS/Express, IDL, Iris Explorer, X-designer, builderXcessory e Ilog. En el caso del paquete Toolmaster y de PV-WAVE , se ha podido alargar el periodo de prueba hasta más de un mes.

IV.2 - Ejecución de programas de demostración.

La primera fase de la evaluación, sobre todo de los productos de análisis y visualización, ha sido la ejecución de las aplicaciones de demostración incluidas en los paquetes. Este paso, aunque elemental, es relativamente importante ya que el fallo de una aplicación simple puede poner en evidencia una incompatibilidad entre el producto y el *hardware* o el *software*. Un caso concreto ha sido el de Iris Explorer cuyo módulo ***Render*** no ha podido ser utilizado en primera instancia ya que necesita la presencia de un acelerador gráfico y la instalación de librerías ***Open GL***. Luego se ha dispuesto de una versión no depurada del mismo módulo, que no necesita acelerador gráfico.

AVS 5, AVS / Express y Iris Explorer

Los productos de programación visual, AVS 5, AVS/Express y Iris Explorer, vienen con un conjunto de *mapas*, que ilustran el uso de sus principales módulos.

Toolmaster

El paquete Toolmaster viene con un conjunto exhaustivo de programas que ilustran las funcionalidades de las librerías agX, baseXplore, ChartXplore, y ContourXplore y que han sido compilados y ejecutados con éxito. Sin embargo, se ha podido comprobar que el manejo de estas librerías no es inmediato. En efecto, el código fuente de estos programas mezcla funciones de alto nivel con llamadas más elementales y se requiere un buen conocimiento del entorno *X Window* con *OSF/Motif* para enterderlos. También cabe precisar que sólo se ha dispuesto de la versión C de las librerías.

PV-WAVE e IDL

En el caso de PV-WAVE, se dispone de una aplicación cerrada llamada ***Wave_gallery*** que ilustra gran parte de las capacidades gráficas y de las funcionalidades de análisis de PV-WAVE Advantage. Además, en los documentos relativos a PV-WAVE , se encuentra una tutoría bastante representativa de la potencia del Command Language. Esta tutoría repasa todos los aspectos de la programación en el CL, desde el manejo de las variables, hasta la ejecución de animación, pasando por los diferentes tipos de visualización.

IDL dispone de una aplicación de demostración muy parecida a la anterior y de numerosos ejemplos en el *Programmer's Guide*.

IV.3 - Representación de señales reales.

Con todos los productos de análisis y visualización, se ha intentado leer ficheros (ASCII y binarios) de la base de datos de las máquinas TJI y TJIU, y representar señales en forma de curvas x-y. Se ha puesto un énfasis particular en lo que concierne a la presentación de los gráficos. En general se ha obtenido buenos resultados, excepto con Iris

Explorer cuyos módulos *Gnuplot*, y *Graph* han resultado poco estéticos y muy cerrados en cuanto a configuración, al menos en la versión probada.

En todos los casos, se ha podido grabar las gráficas en archivos con formatos de impresión estándar como *Postscript* o *cgm*. En este sentido Toolmaster es sin duda el producto que dispone de la más amplia gama de *drivers* de impresión.

IV.4 - Comunicación con otros lenguajes.

Se ha hecho un estudio comparativo de la integración de códigos escritos en C o en Fortran con el CL de PV-WAVE y el de IDL. Las diferencias entre estos productos están resumidas en la **tabla nº 1**, donde se pone en evidencia la gran flexibilidad de PV-WAVE.

IV.5 - Creación de GUIs.

Con el CL de PV-WAVE

Se ha escrito en el CL de PV-WAVE , una aplicación que podría ilustrar unas partes del interfaz de usuario del SAD-TJII **(c.f. figura nº5)**.

Esta aplicación consta de tres ventanas que simbolizan respectivamente, un sinóptico de la electrónica de adquisición de datos del SAD-TJII, un *Mainframe* VXI (un *chasis* donde se van a colocar los módulos digitalizadores de las señales) y una ventana donde se van a dibujar las señales. Desde el sinóptico, y mediante el ratón, el usuario puede seleccionar un *Mainframe* simbolizado por un rectángulo. La selección de uno de ellos provoca la desaparición de la primera ventana que es reemplazada por la del *Mainframe*. En ésta, siempre con el ratón, el usuario puede seleccionar un canal de medida, que corresponde a una señal. La elección de un canal va a eliminar la segunda ventana y crear la tercera donde se visualiza la señal. En este último nivel, el usuario dispone de dos barras de control para modificar el rango de visualización de la señal tanto en amplitud como en intervalo temporal. Las tres ventanas tienen un *Push Button* para salir de la aplicación. Este ejemplo ha permitido manejar muchas funcionalidades de X, como la creación y gestión de ventanas, el manejo del ratón, la utilización de *widgets* de tipo *Dialogs* y la importación de datos desde archivos. También ha puesto en evidencia la relativa facilidad de programación de GUIs "simples" que ofrece el CL de PV-WAVE.

Con los generadores de Interfaz de Usuario

En primer lugar, se han generado pequeñas interfaces (con un *widget container* y algunos *widget primitives*) para familiarizarse con el manejo de estos productos. Un ejemplo sencillo es el de una ventana (un *Bulletin Board* o un *Form*) que contiene un *Push Button* al cual se ha asociado un *callback* que finaliza el proceso.

PV-WAVE Advantage	IDL
SPAWN Comando de PV-WAVE que ejecuta un programa externo. Un *pipe* bidireccional permite transferir datos entre ambos procesos.	**SPAWN** Comando implementado en IDL.
wavecmd(), WAVECMD Funciones que permiten, desde un programa C o Fortran, de ejecutar comandos de PV-WAVE. Sólo se pueden mandar datos ASCII desde el programa externo hasta PV-WAVE.	Se espera una implementación en una futura versión de IDL.
LINKNLOAD Comando que proporciona el acceso a rutinas externas con un *enlace* dinámico.	**CALL_EXTERNAL** IDL proporciona el mismo mecanismo con el comando CALL_EXTERNAL.
cwavec(), CWAVEFOR Es posible enlazar estáticamente una aplicación C o Fortran con archivos de PV-WAVE y llamar directamente a funciones de PV-WAVE.	No tiene equivalente.
wavevars() Desde C, permite acceder a la totalidad de las variables PV-WAVE definidas al momento de la llamada. Desde Fortran el mecanismo es posible pero no directo. Hay que utilizar llamadas intermedias entre el Fortran y la función wavevars.	No tiene equivalente.
wave_execute() Proporciona un mecanismo para manejar variables WAVE, de manera más flexible que wavevars()	No tiene equivalente.
RPC CALL_UNIX permite usar una aplicación PV-WAVE como cliente y un proceso C como servidor. CALL_WAVE permite lo contrario.	**RPC** Solo se puede usar IDL como servidor RPC.

Tabla nº 1 : Estudio comparativo de la integración de código C o Fortran con el CL de PV-WAVE y de IDL

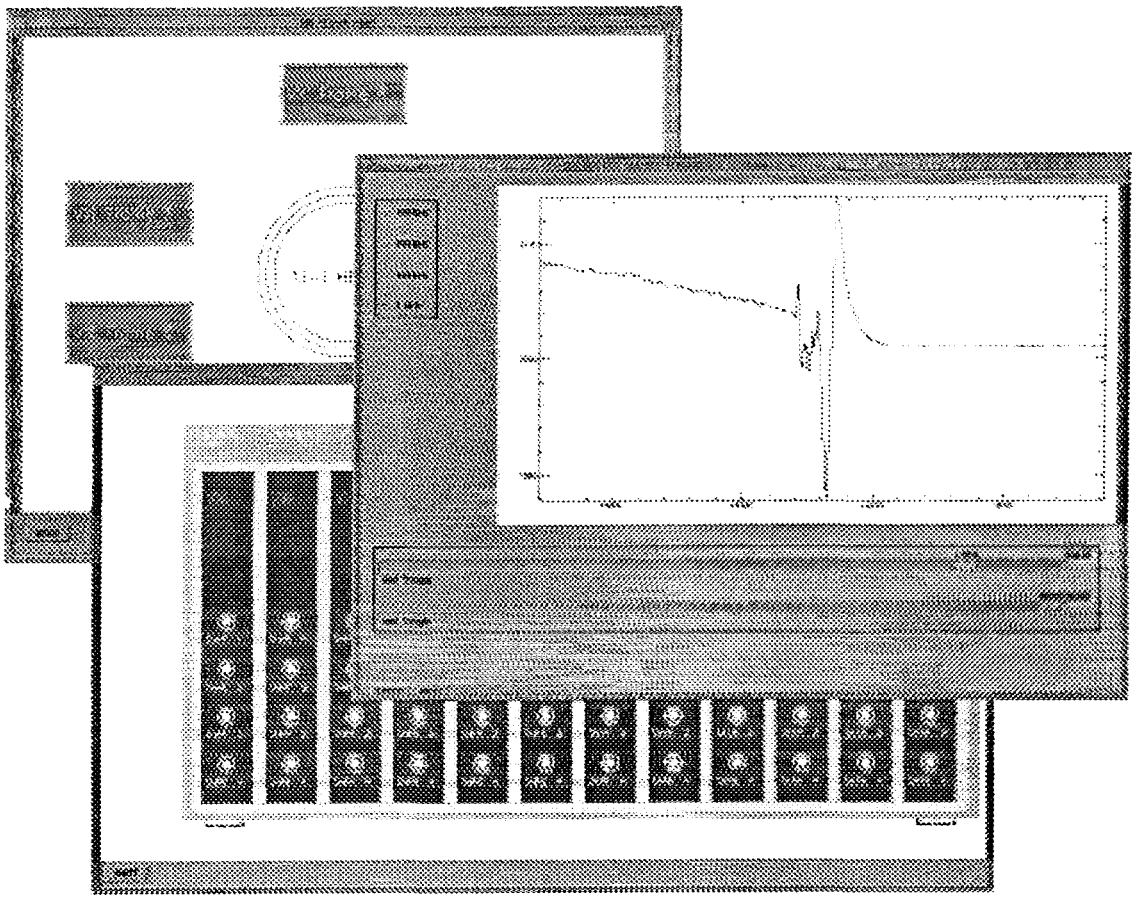

Figura nº 5 : Creación de un GUI con los widgets de PV-WAVE.

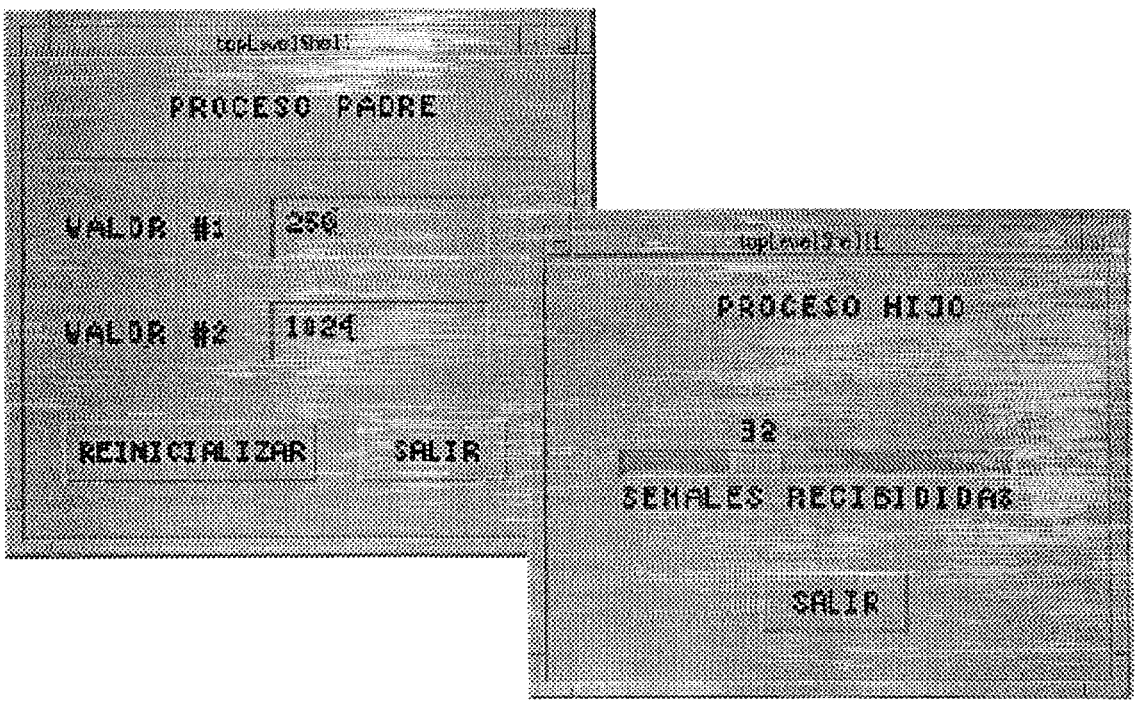

Figura nº 6 : Prueba de sincronización entre dos clientes X.

En segundo lugar se han desarrollado aplicaciones más sofisticadas. La **figura nº6** ilustra un ejemplo de programa que contempla la sincronización entre dos procesos mediante señales Unix. Un proceso, que consta de una ventana con dos campos de texto y dos botones, (uno para inicializar el contenido de los campos de texto y el otro para finalizar la aplicación), crea otro proceso donde aparece una ventana que contiene una barra de control y un botón. Después de haber apretado este botón, el proceso "hijo" manda periódicamente al proceso "padre" una señal. Al recibir la señal este último proceso modifica el contenido de los campos de texto. Con este ejemplo, se ha conseguido integrar un *handler* de señal en el código generado por el producto.

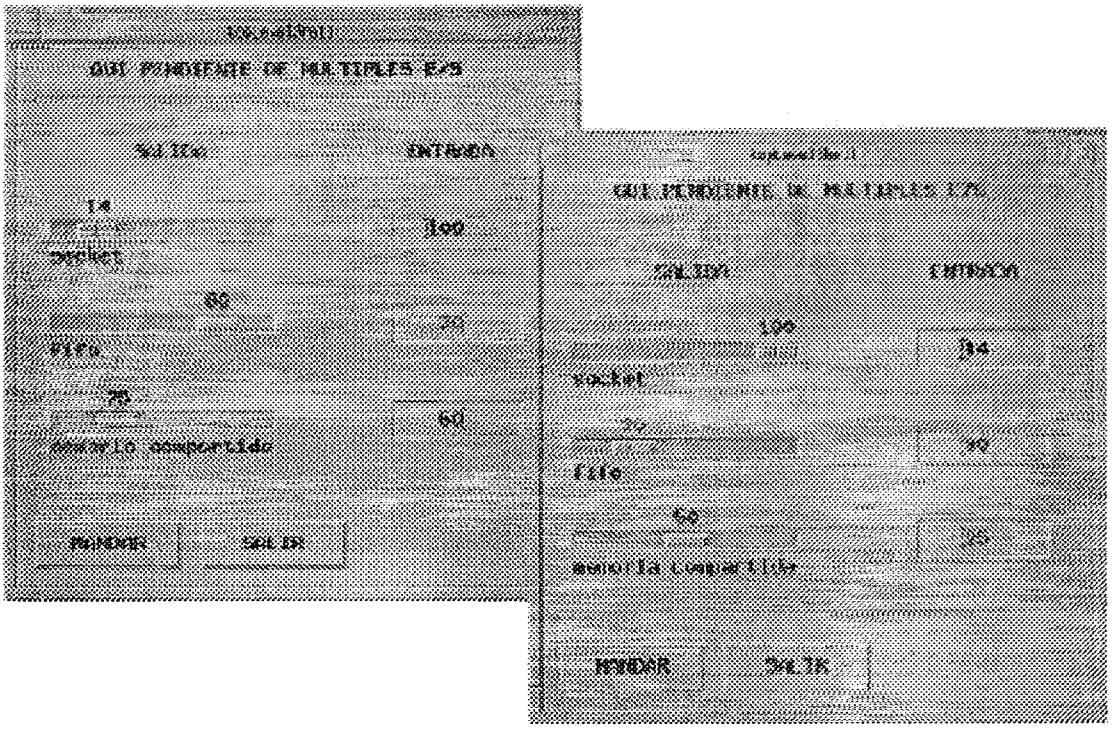

Figura nº 7 : Prueba de comunicación entre dos clientes X.

Otro ejemplo, ilustrado por la **figura nº7**, ha consistido en realizar la comunicación entre dos procesos clientes X, a través de *sockets*, de *fifo* y mediante memoria compartida. En este ejemplo, se ejecutan dos procesos idénticos que contienen cada uno *widgets* de configuración de parámetros enteros (barras de control) y *widgets* de visualización (campos de texto) de estos parámetros. El cambio de un parámetro en una ventana, moviendo con el ratón la barra de control correspondiente, se refleja inmediatamente en el campo de texto asociado de la otra ventana. Este programa ha permitido contemplar los mecanismos que permiten a un proceso cliente X, de estar pendiente de varias fuentes de Entrada/Salida simultáneamente además de los eventos X.

Por último, se han realizado programas relacionados con el diseño del SAD-TJII, como por ejemplo uno que simula la programación de los canales de medida u otro que crea ventanas en varios *displays*. La **figura nº8** ilustra un programa que emula un

osciloscopio donde se pueden visualizar cuatro señales, tanto en forma de puntos como en lineas. El usuario elige una señal de trabajo, mediante un *Option Menu*, o llevando el puntero del ratón sobre una curva y apretando el MB1 (*Mouse Button* nº 1). La aplicación permite varios controles sobre la señal de trabajo, como el cambio de los rangos en amplitud y en tiempo y el desplazamiento vertical de la curva, emulando un offset reglable. En este programa, se ha estudiado el manejo de llamadas de muy bajo nivel (***Xlib*** e ***Xt Intrinsics***) para pintar puntos y curvas, borrar zonas de un *widget* de tipo *Drawing Area* y gestionar las acciones realizadas con el ratón.

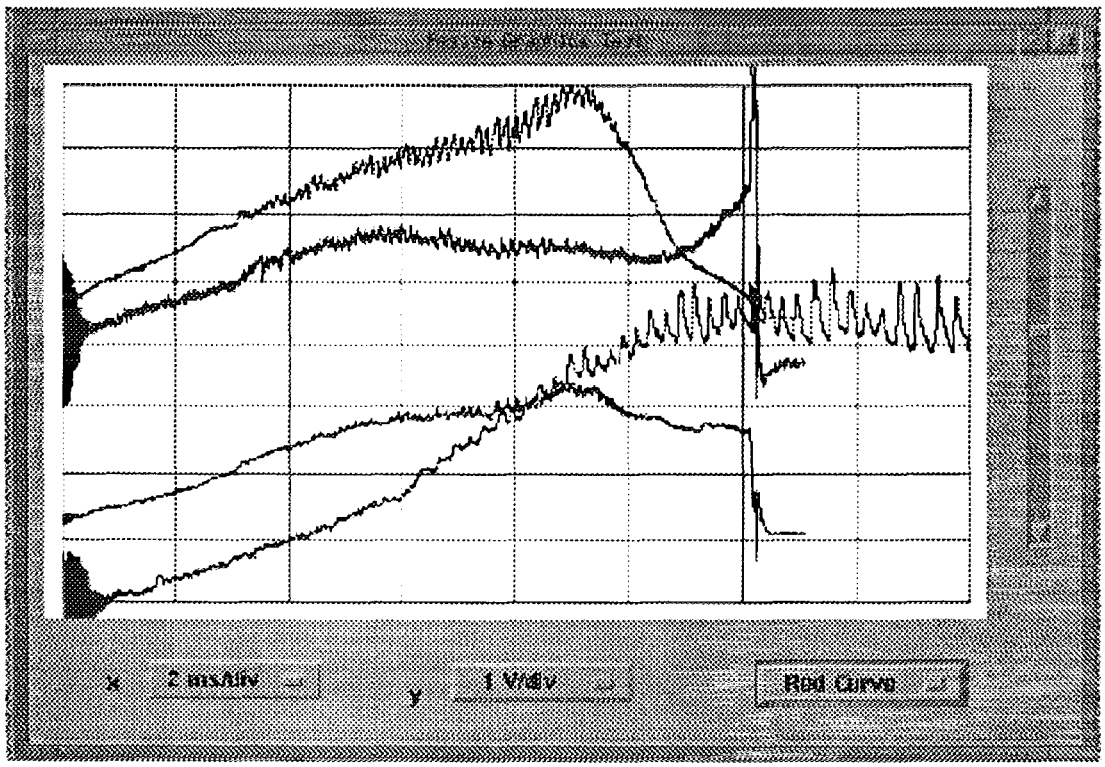

Figura nº 8 : Emulación de un osciloscopio.

IV.6 - Interconexión entre herramientas.

La programación de un GUI a partir del CL de PV-WAVE (o de IDL) resulta relativamente fácil. Sin embargo, como el resultado es un programa interpretado por el proceso *wave* o *idl*, los tiempos de ejecución no son óptimos. Por otra parte, el dibujo de gráficas en una *Drawing Area* mediante llamadas a *Xlib* no es inmediato, sobre todo cuando se quiere visualizar curvas en 2D y 3D. Un posible compromiso para guardar la rapidez de ejecución de un programa C sin tener que implementar rutinas gráficas complejas es la integración de las capacidades gráficas de PV-WAVE en una aplicación ejecutable, generada por un constructor de GUIs. Un prueba sencilla consiste en representar una curva gracias al comando *Plot* del PV-WAVE, dentro de una *Drawing area* creada a partir de builderXcessory.

La implementación de este programa es sencilla y consiste en integrar en el código fuente generado (en C) llamadas a comandos del CL de PV-WAVE , mediante las funciones

wavecmd o ***cwavec***. Para que desde los comandos de PV-WAVE se pueda reconocer la *Drawing Area* "ajena", el programa tiene que pasar como parámetro el identificador de la ventana (se puede obtener con una llamada a función ***XtWindow***).

V - Conclusiones.

La evaluación de los productos presentados en este informe ha permitido encontrar una solución informática satisfactoria para responder a las necesidades del SAD-TJII.

En lo que concierne a la creación de un GUI para la configuración de canales y la visualización de señales durante la operación de la máquina, se trata de una tarea de desarrollo que incumbe al grupo de adquisición de Datos, es decir a programadores profesionales. Este trabajo podría llevarse a cabo con el uso de uno los generadores de Interfaz de Usuario, a excepción de PV-WAVE : UIM/X que no está disponible para el sistema operativo DEC OSF/1. En el caso de los productos builderXcessory, Xdesigner e Ilog Builder, no se dispone de capacidad para el dibujo de curvas de tal forma que se tiene que implementar un juego de rutinas gráficas a partir de llamadas a ***Xlib*** o usar otras librerías de uno de los paquetes de visualización. La construcción del GUI también sería posible a partir de Toolmaster UIM/X, junto con las librerías del paquete Toolmaster, y a partir de AVS/Express. Estos dos productos proporcionan todas las funcionalidades gráficas necesarias. En cuanto al desarrollo del GUI, a partir de los *widgets* de PV-WAVE o de IDL, no constituye una solución satisfactoria debido a las limitaciones en tiempo de ejecución y en flexibilidad de programación que proporcionan estas herramientas. El interfaz de usuario del SAD-TJII es una aplicación crítica que no se puede ver limitada.

Por lo que concierne el análisis y a la visualización diferida de datos, se oponen tres filosofías : Una corresponde a la programación visual a partir de mapas (redes de módulos interconectados); es el caso de AVS 5, de AVS/Express y de Iris Explorer. Otra es la utilización de un lenguaje 4GL completada con un juego de aplicaciones cerradas como en el caso de PV-WAVE y de IDL. La última, es la de Toolmaster que corresponde al uso desde código fuente (C o Fortran) de un conjunto de librerías. El producto que cumple con todos los requisitos expresados para el tratamiento y la representación de datos (c.f. párrafo II) es PV-WAVE. Los productos IDL, AVS 5 y Iris Explorer han resultado menos flexibles, sobre todo en cuanto a la comunicación con otros lenguajes. Tampoco proporcionan el juego de rutinas de procesado de señal ofrecido con ***PV-WAVE : Signal Processing Toolkit***. AVS/Express y Toolmaster son ante todo herramientas de desarrollo al nivel del código fuente y en este sentido no parecen muy adecuadas para unos usuarios finales como son los físicos de la división de Fusión que utilizan la programación como un medio y no como un fin.

A la vista de estas observaciones, la solución que supone un mejor compromiso entre las dos necesidades del SAD-TJII, es decir la creación de un GUI para la operación de la máquina y el uso de herramientas de alto nivel para el tratamiento y la visualización de las señales, parece ser la combinación del paquete PV-WAVE con uno de los generadores de Interfaz de Usuario (builderXcessory, X-designer o Ilog builder).

GLOSARIO

Bitmap : Un *pixmap* de profundidad igual a 1.

Bulletin Board : Clase de *widget* de tipo *container*.

C : Lenguaje de programación de proposito general [18].

C++ : Lenguaje de programación orientado a objetos derivado del C [19].

Callback : Procedimiento que implementa una parte de las funcionalidades de una aplicación. Un *callback* se asocia con uno o varios *widgets* y se invoca cuando el usuario interactua con este o estos *widgets*.

Canvas : *Widget* compuesto que constituye una estructura a partir de la cual se puede construir un *widget* de más alto nivel o un elemento dentro de una aplicación.

Chart : Diagrama que representa una o varias curvas en forma de puntos, símbolos y/o lineas.

Colormap : Tabla de color que asocia a cada valor de un *pixel* un color es decir un nivel de rojo, verde y azul.

Container : Clase de *widget* que puede contener otros *widgets*; define el comportamiento de los *widgets* "hijos".

DDE : *Dynamic Data Exchange*; prótocolo de comunicación entre aplicaciones que se ejecutan en entornos con ventanas [20]

Dialogs : Clase de *widget* de tipo *container*.

Display : Dispositivo visualizador que consta de una o varias pantallas, un teclado y un ratón; se le asocia un *X server*.

Drawing Area : Clase de *widget* que permite visualizar imágenes o *pixmaps*.

Driver : Programa que permite el control de un dispositivo , interfaz de Entrada/salida para impresora, etc.

Fifo : *First In First Out*. Estructura de datos mantenida en el *kernel* de un sistema operativo e identificadas por un nombre de archivo. Permite comunicaciones bidireccionales entre procesos donde los datos se leen en el orden en que han sido escritos.

File Browser : *Widget* que permite la selección de un fichero dentro de un directorio. Permite también moverse dentro de la jerarquía de directorios.

FFT : *Fast Fourier Transform,* algoritmo rápido para calcular la transformada de Fourier discreta de una serie numérica.

Form : Clase de *widget* de tipo *container.*

Fortran : Lenguaje de programación de propósito científico diseñado en 1955 por IBM.

Fortran 77 : Una estandarización del lenguaje Fortran que data del 1977 [21].

4GL : Lenguaje de la cuarta generación. Lenguaje evolucionado que incluye llamadas de alto nivel que no se encuentran en los lenguajes de la generación anterior (Pascal, C, Fortran, etc.).

Gadget : Objeto similar a un *widget* de tipo *primitive* ;

GIL : *Graphic Interface Language;* formato utilizado para guardar un interfaz de usuario generado por el producto ***Devguide*** de Sun Microsystems Inc.

GUI : *Graphic User interface;* parte interactiva de una aplicación que tiene funcionalidades gráficas.

Gray Scale : Escala de gris; clase de *colormap* en cual los valores del rojo, del verde y del azul son iguales.

Grid : Malla 2D o 3D que permite referenciar un conjunto de puntos obtenidos con muestreo irregular.

Gridding : Métodos que permiten manejar conjuntos de datos muestreados irregularmente.

Handler : Procedimiento asociado a un evento; los eventos típicos son los movimientos del ratón en cualquier parte de un *widget,* las acciones del usuario con los botones del ratón, etc.

Hardware : Todo lo relacionado con el material electrónico de un sistema informático.

Host : Ordenador principal de un sistema.

Kernel : Conjunto de rutinas y estructuras de datos que constituyen el núcleo de un sistema operativo o de una aplicación.

Kurtosis : Momento de orden cuatro de una distribución de probabilidad.

Look and Feel : Aparencia y comportamiento de los elementos de un interfaz de usuario sin tener en cuenta las funcionalidades que les están asociadas.

Make : Utilidad que permite actualizar un programa en función de los últimos cambios efectuados en sus ficheros fuentes

Makefile : Fichero de tipo *script* utilizado por la utilidad *make.*

Mainframe : Unidad destinada a recibir módulos electrónicos y conectarlos a traves de un bus.

Option Menu : Menú formado por *widgets* de tipo *Push Button;* siempre muestra la opción elegida.

Picking : Elección de un subconjunto de puntos dentro de una estructura de datos.

Pipe: *First In First Out.* Estructura de datos mantenida en el *kernel* de un sistema operativo e identificadas por un descriptor de archivo. Permite comunicaciones bidireccionales sólo entre un proceso y sus procesos "hijos".

Named Pipe : Un *pipe* identificado por un nombre de tal forma que permite la comunicación entre dos procesos no relacionados.

Pixel Value : Valor de N bits, donde N es la profundidad de una ventana o de un *pixmap;* representa un índice en una tabla de color.

Pixmap : Tabla 2D de *pixels* de valor comprendido entre 0 y 2^N-1 donde N es la profundidad del *pixmap.*

Plot : Trazado de gráfico o dibujo.

Primitives : Clase de *widget* que no puede contener *widgets* "hijos".

Push Button : *Widget* que representa gráficamente una tecla.

Rendering : Técnicas que permiten extraer información desde una estructura de datos multidimensional.

RPC : *Remote Procedure Call;* protócolo de comunicación que permite la ejecución remota de programas, dentro de un modelo cliente/servidor.

Run-time : El tiempo de ejecución de un programa.

Scatter : Conjunto de puntos esparcidos.

Script : Fichero que contiene una lista de comandos que un intérprete va a ejecutar secuencialmente.

Skewness : Momento de orden tres de una distribución de probabilidad.

Slice : Corte dentro de un volumen.

Slicing : Obtención de un *slice.*

Sockets : Interfaz *software* con los prótocolos de TCP/IP.

Software : Todo lo relacionado con la lógica/programación de un sistema informático.

SQL : *Structured Query Language*; lenguaje estructurado destinado al manejo de una base de datos.

Stream Lines : Línea de corriente/flujo.

True Color : Clase de *colormap* donde el valor de un *pixel* está dividido en tres campos; cada uno de ellos corresponde a un índice para el rojo, el verde y el azul en una tabla de colores.

UIL : *User Interface Language* ; lenguaje destinado a guardar la jerarquía de *widgets* de una aplicación en un archivo separado del código fuente de la aplicación.

View Port : Zona de una interfaz de usuario donde se visualiza un diagrama, una imagen, etc.

Wavelets : Conjunto de transformaciones destinadas al análisis espectral de señales.

Widget : Componente de una interfaz de usuario; también llamado *widget instance* (instancia de un *widget*). Por ejemplo, un botón, un campo de texto, un menú, una zona gráfica, etc.

Widget Class : Clase o tipo de un *widget*; define el conjunto de atributos y operaciones soportados para una instancia particular de un *widget*.

Windows : Capa *software* que proporciona al sistema operativo MS-DOS capacidades par manejar ventanas e iconos.

X client : Proceso que crea ventanas en uno o varios *displays* X.

Xlib : Librería de rutinas del sistema *X Window*.

X server : Proceso servidor que trata las peticiones de los *X clients* y controla el *display* que le está asociado.

Xt Intrinsics : Módulo software que simplifica la escritura de aplicaciones para el sistema *X Window*.

X Window : Sistema de comunicación que permite el manejo de ventanas dentro de varios *displays* .

REFERENCIAS

[1] ROBERT W.SCHEIFLER & JAMES GETTYS, "X WINDOW SYSTEM"
Third EDITION
Digital Press.

[2] PAUL J.ASENTE & RALPH R. SWICK, "X WINDOW SYSTEM TOOLKIT The Complete Programmer's Guide and Specification"
Digital Press.

[3] "OSF/Motif™ Programmer's Guide release 1.2"
Prentice Hall.

[4] "Builder Xcessory 3.5 User's Guide"
UNIX Edition.

[5] "Builder Xcessory 3.5 reference Manual"
UNIX Edition.

[6] "X-Designer Release 4 User's Guide"
IMPERIAL SOFTWARE TECHNOLOGY

[7] "Toolmaster UIM/X User's Guide Release 6.4a"
ADVANCED VISUAL SYSTEM INC.

[8] "The UIM/X Developer's Guide Release 6.4a"
ADVANCED VISUAL SYSTEM INC.

[9] "AVS 5 User's Guide"
ADVANCED VISUAL SYSTEM INC.

[10] "AVS 5 Module Writer's Guide"
ADVANCED VISUAL SYSTEM INC.

[11] "PV-WAVE Advantage™, PV-WAVE Command Language™ User's Guide"
Visual Numerics

[12] "PV-WAVE Advantage™, PV-WAVE Command Language™ Programmer's Guide"
Visual Numerics

[13] "IDL User's Guide"
Research System's, Inc.

[14] "Iris Explorer programmer's Guide Release 3.0"
Nag.

[15] "Iris Explorer Module Writer's Guide Release 3.0"
Nag.

[16] "eXodus for Macintosh USER MANUAL Version 5.1"
White Pine.

[17] "Communications for eXodus USER MANUAL Version 2.2"
White Pine.

[18] Brian W. Kernighan & Dennis M. Ritchie "The C programming language"
Second Edition
Prentice Hall.

[19] Bjarne Stroustrup "The C++ programming language"
IEEE Software, 1986.

[20] "Microsoft Windows user's guide"
Microsoft.

[21] "Programming Language FORTRAN"
ANSI X3.9-1978.