Desarrollo de una aplicación CAE para el diseño de circuitos de ventilación

D. Ayala, N. Pla, A. Soto, M. Vigo y S. Vila *
Dept. Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Diagonal, 647, ETSEIB
08028 Barcelona
e-mail: (dolorsa, nuria, tonis, marc, sebas)@lsi.upc.es

Resumen

En este trabajo se presenta una aplicación CAE para el diseño de circuitos de ventilación. En ella se han desarrollado aspectos de diseño del modelo, de la interacción con el usuario y de simulación del comportamiento del sistema. Se utiliza un modelo no evaluado a dos niveles que permite interrogaciones geométricas y tecnológicas.

Uno de los principales objetivos del trabajo ha sido la portabilidad de la aplicación y, por ello, en su desarrollo se han usado herramientas independientes de la plataforma de implantación: se ha programado en ANSI C y se ha usado Tcl/Tk para la interacción y OpenGl como librería gráfica. La aplicación se ha desarrollado sobre Unix y se ha portado sin coste adicional sobre Linux, Windows 95 y Windows NT.

Palabras clave: CAE, modelos no evaluados, interacción, portabilidad.

1 Introducción

El trabajo presentado en este artículo se enmarca en un proyecto que se ha realizado mediante un convenio de colaboración con la empresa SOLER & PALAU, S.A. Este convenio se ha estructurado en dos fases. La primera ha consistido en la realización de un catálogo automatizado de ventiladores y la segunda en un sistema de ayuda al diseño de circuitos de ventilación.

*Los autores aparecen por orden alfabético.

El catálogo automatizado consta de dos aplicaciones. La primera de ellas, permite una consulta interactiva de ventiladores y accesorios. Aparte de las características textuales y técnicas, se ofrece información gráfica (fotografía y plano acotado del ventilador) e información tecnológica que muestra la curva característica del ventilador (caudal, presión). El usuario puede consultar interactivamente todos los puntos de dicha curva y además se ofrece la posibilidad de selección del ventilador adecuado indicando el punto de trabajo de la instalación que el sistema calcula a partir de datos técnicos que indica el usuario. La segunda aplicación del catálogo automatizado es la que permite su mantenimiento. El sistema permite generar distintos catálogos dependiendo de las zonas de venta a las que van dirigidos y mantiene 10 idiomas. La aplicación de consulta se distribuye entre los clientes de la empresa mientras que la de mantenimiento es accesible únicamente por el personal de la empresa.

La segunda fase, que es la que se presenta en este artículo, es una aplicación CAE para el diseño de circuitos de ventilación. En esta aplicación se ofrecen prestaciones para editar y visualizar un circuito de ventilación así como la posibilidad de calcular el punto de trabajo de la instalación diseñada. Esta fase conecta con la aplicación de consulta de la primera fase del proyecto ya que el citado punto de trabajo sirve para seleccionar el conjunto de ventiladores adecuados dentro del catálogo y se distribuye junto con ella.

En la segunda fase se han desarrollado aspectos de diseño del modelo, de la interacción con el usuario y de simulación del comportamiento del sistema. El modelo utilizado es un modelo no evaluado a dos niveles. Un primer nivel presenta una estructura arborescente que corresponde al eje del conjunto de tuberías que constituyen el sistema de ventilación. El segundo nivel del modelo consiste en la conversión de la estructura arborescente a un modelo de conductos. El modelo contiene elementos lineales, tipos distintos de entrada y salida de aire, codos y bifurcaciones, que pueden tener sección circular o rectangular. Finalmente, el modelo permite interrogaciones geométricas como distancias entre elementos, diámetro de secciones, etc. así como interrogaciones tecnológicas sobre las condiciones de funcionamiento de los distintos puntos del sistema: presión, caudal, pérdida de carga, etc.

Uno de los objetivos del trabajo ha sido la portabilidad de la aplicación y, por ello, en su desarrollo se han usado herramientas independientes de la plataforma de implantación: se ha programado en ANSI C y se ha usado Tcl/Tk [6] para la interacción y OpenGl [7] como librería gráfica. La aplicación se ha desarrollado sobre Unix y se ha portado sin coste adicional sobre Linux, Windows 95 y Windows NT.

El trabajo se estructura en los siguientes apartados. En primer lugar se

describe la arquitectura del sistema. El apartado tercero explica el modelo de conductos, el cuarto detalla los aspectos relacionados con la integración entre el diseño geométrico y los cálculos tecnológicos que se han llevado a cabo y el quinto describe el modelo de interacción que se han desarrollado. El sexto apartado comenta los aspectos referentes a la implementación y finalmente el último apartado extrae las conclusiones y el trabajo futuro a realizar.

2 Arquitectura de la aplicación

El sistema tiene cuatro componentes principales: la interfaz de usuario, los modelos, la escena y los cálculos tecnológicos (figura 1).

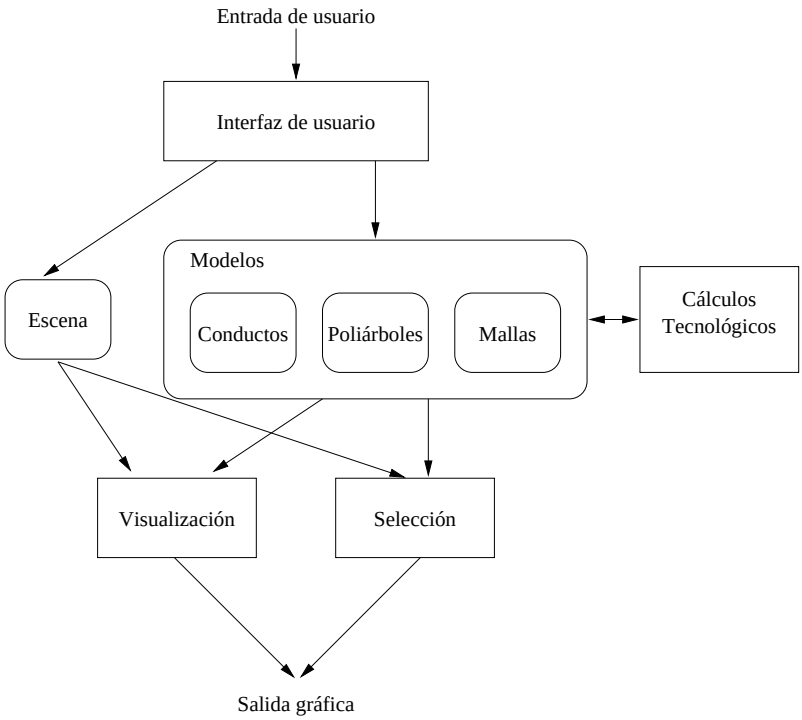

Figura 1: Arquitectura de la aplicación.

La *interfaz de usuario* tiene como misión gestionar la interacción con el usuario. El estilo de interacción es por manipulación directa [4]. Los criterios seguidos para su diseño se exponen en la Sección 5.

El sistema gestiona tres *modelos* principales que agrupan distintas en-

tidades interrelacionadas. Estos modelos son los conductos, los poliárboles y las mallas. Los modelos comparten como característica que todos ellos son modelos no evaluados. Los *conductos*, que modelan las tuberias de un sistema de ventilación, tienen topología de árbol y están compuestos por un conjunto de tramos conectados entre sí; véase la Sección 3.

Los *poliárboles* son una generalización de las polilíneas en el espacio que admiten una topología de árbol, es decir, admiten ramificaciones pero no ciclos. Se usan como ejes de los conductos de ventilación (figura 4). Pueden asociarse atributos tecnológicos tanto a los nodos como a las aristas de los poliárboles. Actualmente los atributos de los nodos y las aristas se usan durante el proceso de generación automática de un conducto a partir de su eje.

Las *mallas* son matrices de puntos en el espacio (figura 4). Una malla queda determinada por una posición en el espacio, su orientación, el volumen que ocupa en el espacio y la densidad de puntos en cada dirección. Las mallas son elementos auxiliares que se usan como soporte para posicionar otros elementos en el espacio.

Las instancias de las entidades descritas anteriormente se almacenan en una base de datos que llamamos *escena*. La escena és un conjunto de pares instancia-atributo. La escena se consulta durante el proceso de visualización y durante la selección de instancias. Para ello se ha establecido un convenio de atributos. Por ejemplo, se visualizan únicamente las instancias que tienen asociados los atributos `visible`, `seleccionada` o `seleccionable`, o bien, sólo pueden seleccionarse las instancias que tienen asociado el atributo `seleccionable`. Los atributos de las instancias cambían en función del estado de la interacción con el usuario.

Los *cálculos tecnológicos* sobre el sistema de ventilación se usan tanto para evaluar algunos parámetros que describen el comportamiento de los conductos como para generar nuevos conductos a partir de su eje y de los atributos tecnológicos con los que el diseñador ha etiquetado los nodos y aristas del poliárbol; véase la Sección 4.

3 El modelo de conductos

La característica más relevante del modelo utilizado para representar los conductos de aire es que se trata de un modelo en el cual la geometría no está evaluada [2]. Es decir, la geometría explícita (caras, aristas y vértices) sólo se utiliza en y para el proceso de visualización. Según este modelo, los conductos vienen representados por un conjunto de entidades paramétricas

sencillas, denominadas tramos, unidos entre sí por conexiones. A su vez, los tramos se representan a partir de un conjunto de parámetros de diversa tipología (posición, radio, longitudes, ángulos, materiales, etc.) y tienen asociada una lista de entidades geométricas que permiten su visualización.

La elección de un modelo no evaluado se fundamentó en las dos siguientes razones. En primer lugar, este modelo asegura más robustez, puesto que se evitan los errores numéricos asociados a todo proceso de cálculo. Dado que los cálculos se realizan directamente a partir de los parámetros y de la topología del modelo, no hay que basarse en valores numéricos que dependen de entidades geométricas de bajo nivel, como vértices y aristas de caras, en las que típicamente se suelen acumular los errores por propagación. Las ventajas del modelo propuesto se pueden comparar con las que proporcionan los manipuladores simbólicos frente al cálculo en coma flotante [8]. Y en segundo lugar, este modelo facilita la creación y posterior modificación de los conductos, puesto que ofrece una representación más cercana a las intenciones del usuario de la aplicación, resultando más eficiente. La mayoría de las modificaciones se reducirán al cambio de un simple parámetro, por ejemplo un ángulo o la posición de un tramo.

El nivel superior del modelo son los *conductos*, que representan un conjunto conexo de tuberías, incluyendo tubos rectos, codos, ramificaciones y accidentes (accesorios empotrados en el conducto, como ampliaciones, reducciones, válvulas y rejillas). La estructura topológica de un conducto puede identificarse con una estructura árbol (figura 2). En la raíz del árbol se situará un ventilador y en las hojas los elementos terminales de la tubería (rejillas de salida y campanas de extracción).

Como se ha dicho, cada conducto está compuesto por una serie de *tramos*, que corresponden a las piezas tecnológicas estándares (figura 3). Además de los ventiladores y elementos terminales, los tramos implementados pueden clasificarse como tramos rectilíneos, codos, bifurcaciones, cruces, reducciones, ampliaciones o válvulas. Se han implementado dos tipos geométricamente distintos de tuberías, los de sección circular y los de sección rectangular. Estas dos tipologías no son incompatibles, puesto que un conducto puede estar compuesto por tubos circulares y rectangulares, siempre que puedan conectarse adecuadamente mediante tramos de cambio de sección. El conjunto de tramos tipo no es fijo, sino ampliable según las necesidades del usuario. Una instancia de un tramo queda definida fijando su posición en el espacio, sus parámetros de dimensionado y las demás características como el material. La unión de dos tramos distintos se realiza a través de sus *conectores*, los cuales quedan definidos por su punto de conexión, el tipo (rectangular o circular), el plano de soporte y sus dimensiones.

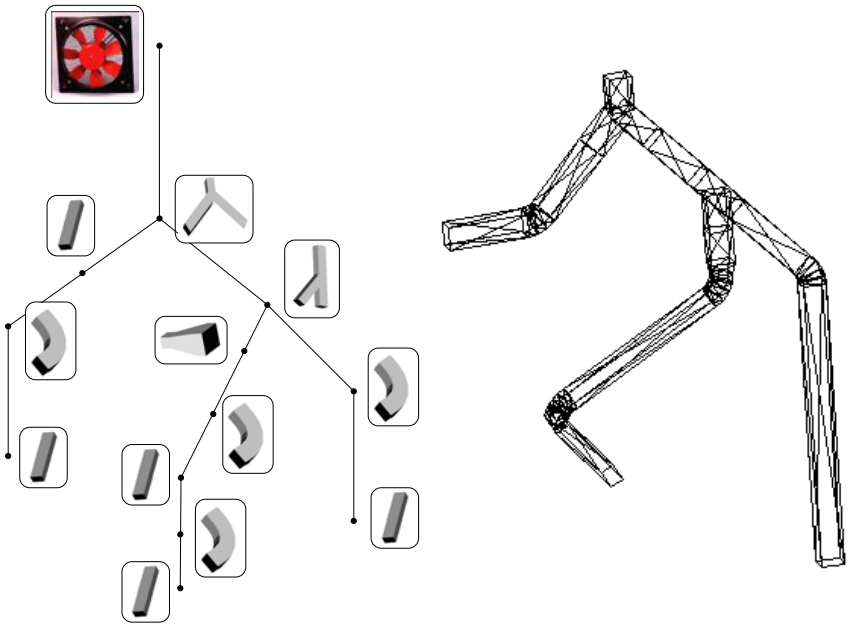

Figura 2: Los conductos de aire en el modelo quedan representados como un árbol de tramos

Dos tramos se pueden unir si los dos conectores son compatibles, es decir, si son del mismo tipo, tienen las mismas dimensiones y sus puntos de conexión y planos de soporte coinciden. En el caso de conectores rectangulares hay que verificar, además, que las direcciones verticales coinciden.

A la estructura árbol asociada al conducto se la denomina poliárbol porque además de los nodos de grado mayor que dos admite nodos de grado dos que corresponden a tramos lineales con dos conectores. De esta forma, la topología del polièarbol determina el tipo de tramo que debe colocarse en cada nodo, puesto que el grado del nodo debe coincidir con el número de conectores del tramo.

El usuario puede diseñar los conductos de aire siguiendo dos metodologías distintas: bien aditivamente, añadiendo nuevos tramos a un extremo de un conducto base, bien a partir de un esquema poliárbol con atributos asociados a los nodos y aristas. La aplicación ofrece herramientas para crear y editar los poliárboles eje y modificar los atributos.

La conversión de poliárboles a conductos se realiza automáticamente. Para ello no es necesario que se hayan definido todos los atributos del poliárbol debido a la interrelación que existe entre los atributos correspondien-

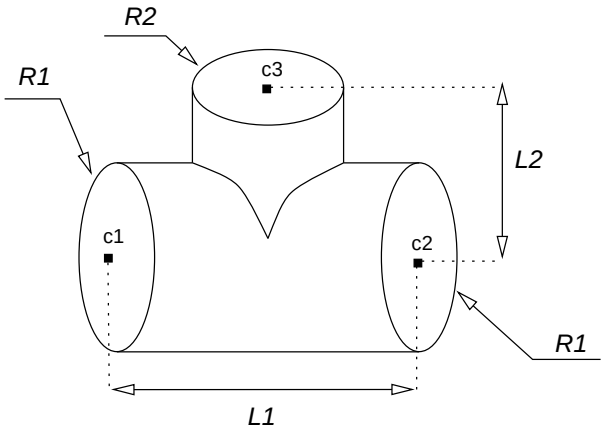

Figura 3: Ejemplo de tramo: bifuración en T con tres conectores circulares

tes a los parámetros tecnológicos (véase la Sección 4 donde se presentan los cálculos de los distintos parámetros tecnológicos). Existe la posibilidad de crear conductos coaxiales a los ejes del poliárbol o bien de que queden apoyados sobre ellos, para simular la construcción de tuberías adosadas a un soporte arquitectónico. Otros elementos auxiliares para la edición de los conductos incluyen las mallas de puntos y las polilíneas.

4 Cálculo del comportamiento y diseño

Uno de los principales objetivos de la aplicación presentada radica en el alto nivel de integración existente entre las herramientas destinadas a trazar el recorrido o forma de los conductos y las herramientas que se usan para determinar el comportamiento aerodinámico del sistema. Durante el proceso de diseño, esta integración permite al usuario un ciclo de simulación-corrección muy rápido resultando en un notable ahorro de tiempo de diseño.

El cálculo del comportamiento de un conducto se basa en diversos parámetros interrelacionados. Esencialmente, el caudal de fluido que circula por el conducto, la velocidad a la que circula y el diámetro (o diámetro equivalente en el caso de conductos rectangulares) se hallan relacionados entre sí. Así por ejemplo, para un mismo caudal, puede aumentarse o disminuirse la velocidad del fluido variando el diámetro del conducto. Por otro lado estas variables condicionan otros factores de comportamiento del conducto. En particular, el diámetro se ve limitado por el espacio en el que va a instalarse el conducto y además afecta al coste de la instalación: a mayor diámetro,

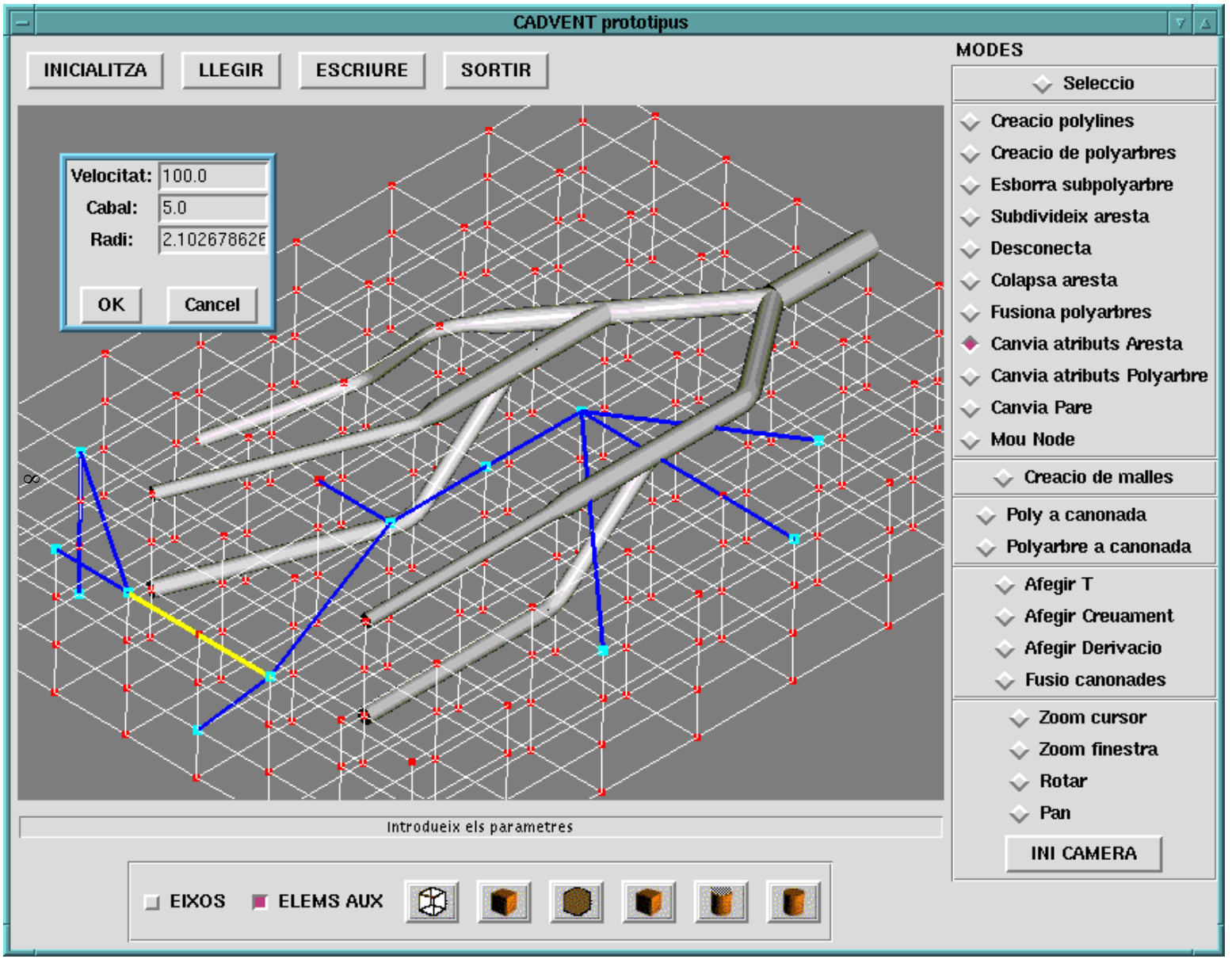

Figura 4: Modelo con un conducto, una malla y un poliárbol

mayor coste. La velocidad es un factor determinante del ruido que produce el conducto: a mayor velocidad, mayor ruido. Por último, el caudal está condicionado por las necesidades de ventilación para las que se diseña la instalación. Una introducción al proceso de cálculo en circuitos de ventilación puede hallarse en [5].

El diseñador, cuando concibe una instalación intenta conseguir un compromiso razonable entre las distintas variables. Para ello se requiere poder diseñar un conducto en cuanto a su geometría, accidentes y parámetros del fluido. Un conducto correctamente definido puede interrogarse para conocer sus características tecnológicas como el diámetro adecuado para una velocidad dada o la velocidad del fluido conocidos el diámetro y el caudal. La aplicación permite realimentar estos resultados y modificar automáticamente, por ejemplo, el diámetro de una parte del conducto.

Dos pautas de diseño habituales entre los diseñadores, soportadas por la aplicación, consisten en diseñar con diámetro constante o con velocidad constante. Ambas pautas pueden ser aplicadas globalmente a todo el conducto u, optativamente, a partes conexas del mismo. En el primer caso, se fija un diámetro constante para el conducto y se permite la variación del caudal (y por tanto de la velocidad). En el segundo caso, se fija una velocidad constante para el fluido y el diámetro varía según el caudal que conduce.

Por último, la aplicación permite calcular la pérdida de carga de un conducto. Este parámetro indica el trabajo que debe emplearse para mover el fluido dentro del conducto y es fundamental para determinar la máquina extractora o impulsora que debe instalarse en dicho conducto. La selección del ventilador se realiza automáticamente por la aplicación ofreciendo al diseñador un conjunto de alternativas entre las existentes en un catálogo de ventiladores.

5 Modelo de interacción

Los objetivos que han guiado el diseño de la interacción con el usuario son los siguientes:

1. Simplicidad de la interacción.
2. Claridad de la organización.
3. Accesibilidad de las operaciones principales.

4. Posibilidad de realizar cualquier operación en cualquier momento (*mode-free*).

5. Realimentación al usuario sobre el estado de la interacción.

6. Guiar al usuario sobre las operaciones posibles en cada momento.

Para alcanzar los objetivos 2 y 3, en el diseño del aspecto de la interfaz de usuario se ha evitado una organización excesivamente jerárquica de los menús (figura 4). Se han definido unos modos básicos de interacción (edición de poliárboles, edición de conductos de ventilación, edición de mallas, etc.) con el propósito de facilitar el acceso a las operaciones principales (objetivo 3) y a su vez simplificar la interacción (objetivo 1), concretamente reduciendo los desplazamientos del ratón entre el área gráfica y el área de menús (figura 4). Aparentemente, la inclusión de modos es contraria al objetivo 4. Sin embargo, se ha optado por mantener un compromiso entre los objetivos 4 y 1 garantizando que en cualquier momento pueda cambiarse de un modo a otro. Un criterio adicional que ha resultado especialmente satisfactorio para simplificar la interacción (objetivo 1) ha sido la minimización del número de *clicks* del ratón en cualquier momento de la interacción.

Los objetivos 5 y 6 requieren un control preciso sobre el estado de la interacción con el usuario. Para ello se ha especificado el comportamiento de la interfaz de usuario mediante una *red de transición de estados* (*state transition network*). Una red de transición de estados consiste en un conjunto de estados y un conjunto de arcos entre los estados cada uno de ellos etiquetado con el símbolo de entrada (*token*) que causa la transición del estado inicial del arco al estado final. Además de las etiquetas con los símbolos de entrada, pueden asociarse a los arcos procedimientos de la aplicación que son ejecutados al producirse la transición.

Myers, [4], clasifica las redes de transición de estados dentro de los estilos de especificación de interfaces de usuario basadas en el uso de un lenguaje. Entre los inconvenientes de la especificación mediante redes de transición de estados, Myers destaca que la especificación de interacción *mode-free* produce redes con gran cantidad de arcos entre estados y que es difícil gestionar redes de gran tamaño. Estos inconvenientes se subsanan parcialmente aumentando ligeramente la funcionalidad de las redes de transición de estados.

En primer lugar, estructuramos la especificación de la interfaz de usuario mediante subredes de pequeño tamaño de una forma análoga a los subprogramas de un cierto programa. Las subredes se alcanzan mediante transiciones especiales, análogas a las llamadas a subprogramas. En segundo lugar, dotamos a las subredes de memoria local en forma de registros y de

un mecanismo de intercambio de información entre subredes análogo a los parámetros de los subprogramas. Ambos mecanismos están inspirados en las transiciones de tipo *push* y *pop*, y en los registros de las ATN (*Augmented Transition Networks*), [1], respectivamente. A pesar del aumento de funcionalidad aportado a las redes de transición de estados su potencia descriptiva como lenguaje sigue siendo muy inferior a la de las ATN.

6 Aspectos de la implementación

Un objetivo importante planteado en el desarrollo de esta aplicación ha sido la portabilidad del producto resultante. Aunque se trata de un producto específico para el diseño de tuberías, se pretende no restringir el conjunto de plataformas sobre las que se pueda ejecutar. Esta portabilidad se ha conseguido gracias el uso de herramientas estándares, válidas para diversas plataformas.

Así, los lenguajes de programación utilizados en la implementación han sido ANSI-C, para las rutinas donde se precisa rapidez, y Tcl como lenguaje de alto nivel. Para la implementación de la interfaz de usuario se emplea la extensión Tk [6]. La utilización del lenguaje Tcl para la definición de los tramos facilita al usuario la ampliación del conjunto de tramos tipo.

Las rutinas de visualización consiguen la portabilidad mediante el uso de las rutinas gráficas de OpenGL [7] y la extensión de GLE que aporta facilidades de barrido generalizado. Estas librerías gráficas funcionan en la actualidad sobre la gran mayoría de plataformas, y la ejecución de la aplicación desarrollada sobre estaciones provistas de placas gráficas que soportan OpenGL ha probado ser muy eficiente. Para establecer el vínculo entre Tcl/Tk y OpenGL se ha utilizado la extensión ToGL.

Aunque el proyecto ha sido desarrollado mayoritariamente sobre Unix, a fin de aprovechar las comodidades que este sistema operativo ofrece, el porte a otras plataformas ha resultado ser muy rápido y sencillo. En la actualidad, existen versiones de la aplicación que funcionan sobre Linux, Windows 95 y Windows NT.

7 Conclusiones y trabajo futuro

Se ha desarrollado una aplicación para el diseño de circuitos de ventilación que ha comportado aspectos de modelado, interacción y simulación del comportamiento. Dicha aplicación ha sido encargada por una empresa fabricante de elementos para sistemas de ventilación y será distribuida por dicha

empresa entre sus clientes como ayuda al diseño de circuitos. El uso de un modelo no evaluado para los conductos implica que la aplicación goce de gran robustez y facilita los procesos de edición e integración entre el cálculo de la geometría y de la información tecnológica. La aplicación ofrece una interacción simple y amigable y goza de una alta portabilidad.

Actualmente se ha desarrollado una primera versión de la aplicación. Esta puede ser ampliada en varios aspectos:

- Incorporación de más facilidades de edición así como de la posibilidad de editar sobre el modelo superior de conductos.
- Posibilidad de obtener un despiece del circuito.
- Posibilidad de editar el entorno arquitectónico sobre el que se apoya el circuito de ventilación.

8 Agradecimientos

Este trabajo ha sido realizado mediante un convenio de colaboración con la empresa SOLER & PALAU, S.A. y ha sido parcialmente subvencionado mediante el proyecto CICYT TIC98-0586-C03-03.

Queremos agradecer a la empresa SOLER & PALAU, S.A. y al Sr. Joan Miró, en particular, el asesoramiento en los temas referentes al cálculo en circuitos de ventilación.

Así mismo queremos agradecer a los becarios programadores que han desarrollado este proyecto su entusiástica colaboración. Gracias a Xavi Gómez, Francesc Sala, Àlex Mañez, Laia Ferrer, Anna Manresa i Martí Sánchez.

Referencias

[1] E. Charniak and D. McDermott. *Introduction to Artificial Intelligence.* Addison-Wesley, 1985.

[2] M. Mäntylä. *An Introduction to Solid Modeling.* Computer Science Press, 1988.

[3] B. A. Myers. User Interface Software Tools. *ACM Transactions on Computer-Human Interaction*, 2(1):64 – 103, 1995.

[4] B.A. Myers. UIMSs, Toolkits, Interface Builders. Revised version of [3], http://www.cs.cmu.edu/∼bam, 1996.

[5] SOLER & PALAU S.A. Manual práctico de ventilación SP. Technical report, SOLER & PALAU, S.A.

[6] J. Welch. *Practical programming in Tcl and Tk.* Prentice Hall PTR, 1995.

[7] M. Woo, J.Ñeider, and T. Davis. *OpenGl Programming Guide. Second Edition.* Addison-Wesley Developers Press, 1997.

[8] C.H. Yap. *Handbook of Discrete and Computational Geometry*, chapter 35 Robust geometric computation. CRC Press LLC, 1997.