Portage du logiciel Formes

Cyrille George, Hervé Lequay

736

Portage du Logiciel Formes

Rapport de Recherche
Juin 1990

Cyrille George • Hervé Lequay
Ecole d'Architecture de Lyon

formes

CONVENTION N° 88 01384 00 223 75 01
Ministère de l'Equipement et du Logement, Direction de l'Architecture et de l'Urbanisme,
Sous-Direction des Enseignements et des Professions, Bureau de la Recherche Architecturale

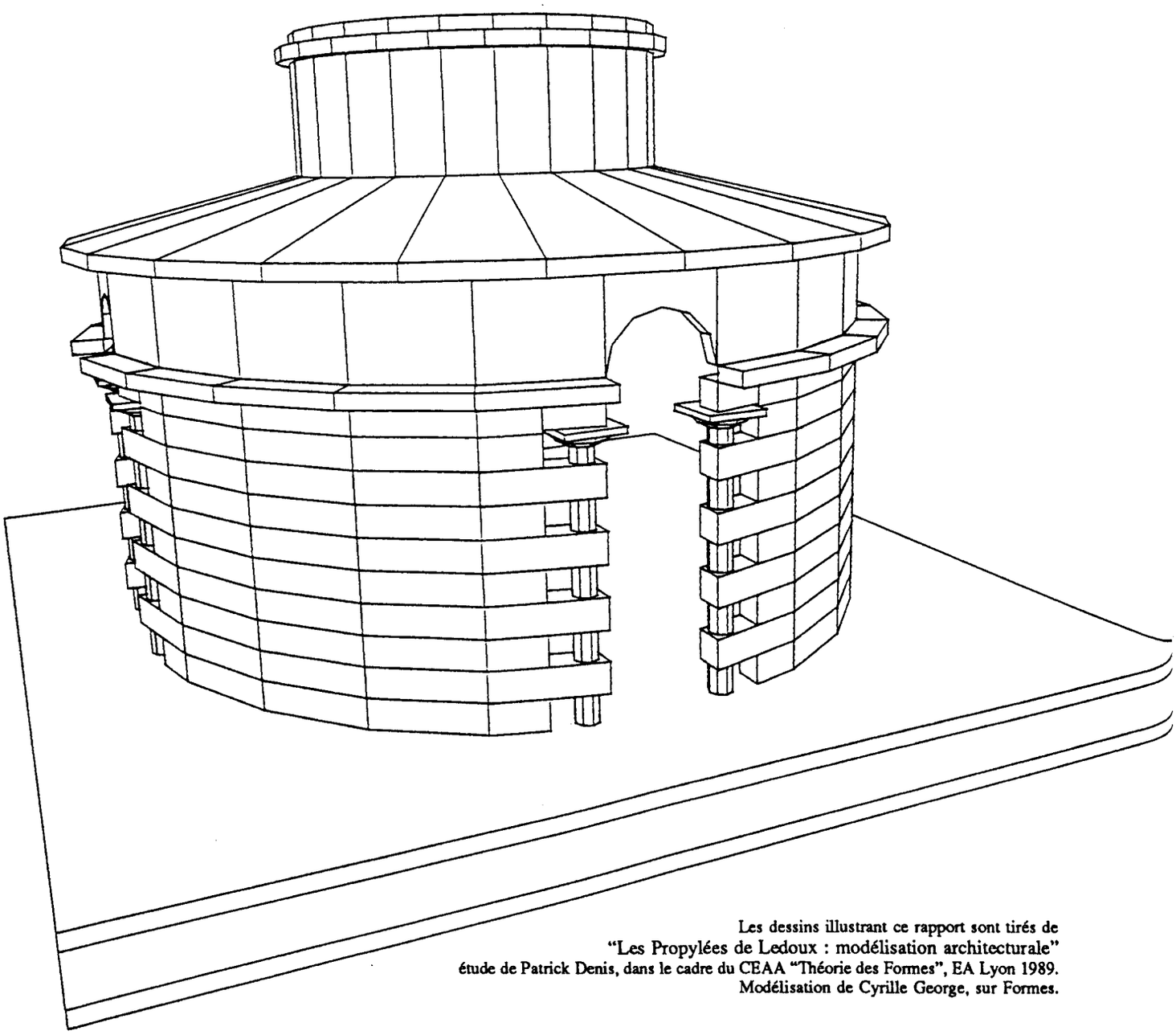
Portage du Logiciel Formes

Rapport de Recherche
Juin 1990

Cyrille George • Hervé Lequay
Ecole d'Architecture de Lyon

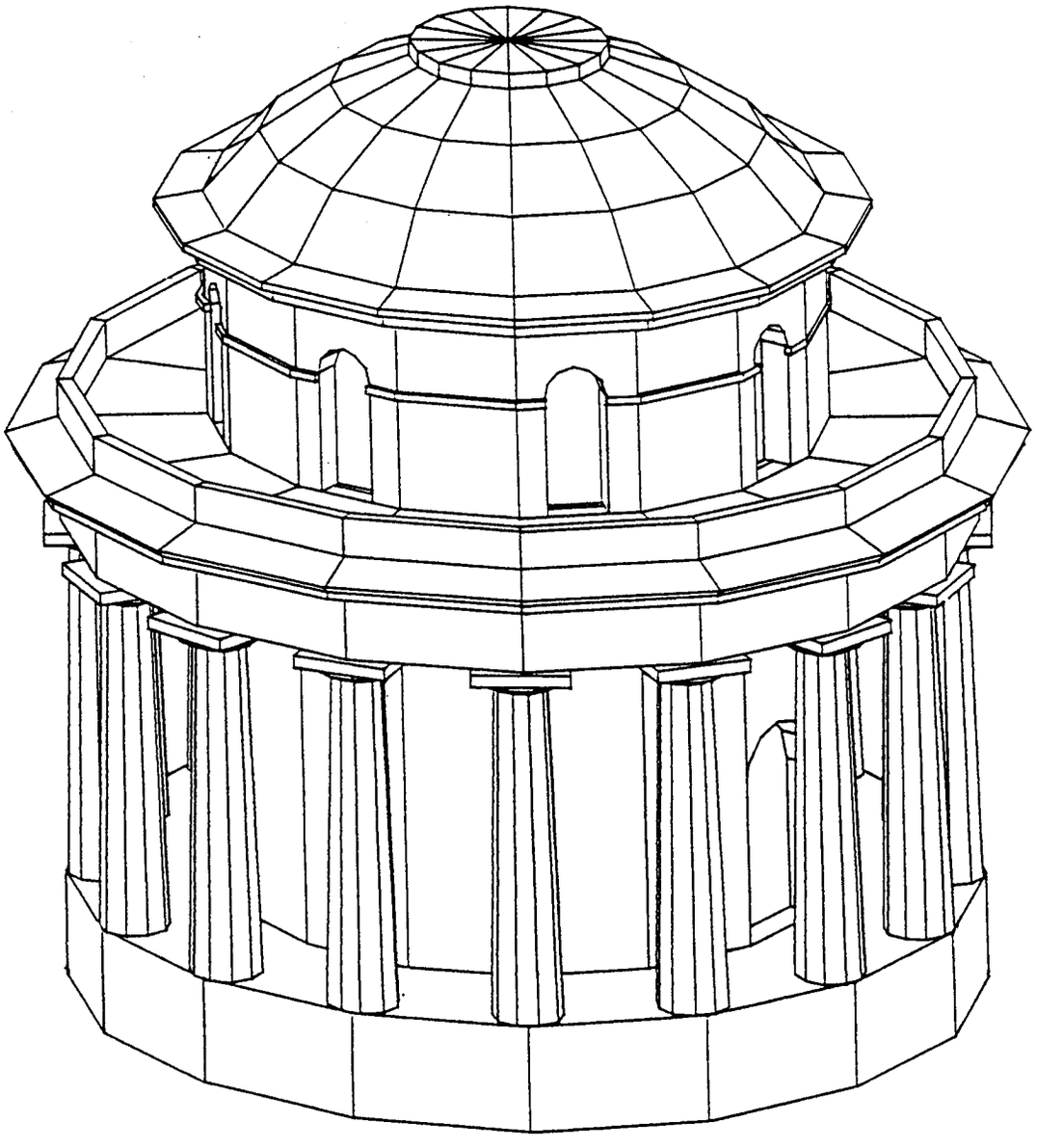
CONVENTION N° 88 01384 00 223 75 01
Ministère de l'Equipement et du Logement, Direction de l'Architecture et de l'Urbanisme,
Sous-Direction des Enseignements et des Professions, Bureau de la Recherche Architecturale

"Le présent document constitue le rapport final d'une recherche remise au Bureau de la Recherche Architecturale en exécution du programme général de recherche mené par le Ministère de l'Equipement et du Logement avec le Ministère de la Recherche.
Les jugements et opinions émis par les responsables de la recherche n'engagent que leurs auteurs".



Les dessins illustrant ce rapport sont tirés de
"Les Propylées de Ledoux : modélisation architecturale"
étude de Patrick Denis, dans le cadre du CEAA "Théorie des Formes", EA Lyon 1989.
Modélisation de Cyrille George, sur Formes.

Sommaire



Formes

Formes est un modeleur 3D de type volumétrique orienté design, incluant des fonctionnalités propres aux volumes architecturaux, aussi bien au niveau de la conception des volumes que de leur représentation.

Conçu dans un but pédagogique, il offre à l'utilisateur un ensemble de fonctions simples de génération de formes, sans limitation de complexité dans leur description. La phase de conception est réalisée interactivement en trois dimensions, à partir de projections orthographiques, avec contrôle immédiat sur une vue perspective. Tous les types de représentation couramment utilisés en architecture sont disponibles (perspectives, projections parallèles, coupes), et leurs paramètres aisément modifiables. Il inclue un algorithme de suppression des faces cachées orienté traceur, et une palette de mise en couleur des images produites.

Historique du logiciel

1984 Premier modeleur 3D opérationnel, 'Bole', dessiné sur Apple II, écran graphique Secapa, traceur Calcomp, table à digitaliser. Auteurs : Hervé Lequay, Denis Boussant, dans le cadre de l'enseignement informatique.
Le logiciel est utilisé pour la modélisation et la reconstitution d'un site archéologique, sur commande de la direction des Antiquités Historiques.

1986 Modeleur Formes version 1, dessiné sur BFM 186 couleur, traceurs Calcomp et Benson, table à digitaliser.
Auteurs : Hervé Lequay, Denis Boussant, dans le cadre du TPFE.

1987 Formes version 2, extensions fonctionnalités, manuel d'utilisation.
Auteurs : Jérome Demiaux, Jean-Marc Duport, sur financement BRA/DAU.
Le logiciel est diffusé dans les écoles d'architecture, présenté en France et à l'étranger. Il est utilisé en pédagogie et dans la modélisation de nombreux projets (commandes extérieures).

1988 Module supplémentaire d'opérations booléennes sur les volumes, implanté sur Formes pour test.
Auteur : Jérome Demiaux.

1988 Le portage sur environnement Macintosh d'Apple est envisagé, pour bénéficier de l'implantation massive de cette gamme de micro-ordinateurs, et de sa simplicité d'utilisation, qui en fait un outil privilégié pour la pédagogie.

La version BFM

L'attrait de la première version de Formes réside dans sa simplicité d'utilisation, bénéficiant d'une ergonomie poussée. Chaque utilisateur peut configurer à sa guise son environnement de travail, et aucun mode opératoire spécifique n'est obligatoire. La construction des volumes et

leur manipulation peut s'effectuer par des procédures variées, avec différents modes de saisie, permettant ainsi de choisir à tout moment le meilleur outil pour la tâche désirée.

Interface

L'utilisateur peut choisir entre une configuration composée de quatre cadres de tracé (trois projections orthographiques et une vue de contrôle), ou une configuration plein écran, agrandissement d'une des vues précédentes. Ces quatre cadres peuvent être agrandis ou disposés de manière différente.

Le texte nécessaire soit à l'information de l'utilisateur, soit à la saisie de données, est affiché en surimpression sur les tracés.

La modélisation et la manipulation des objets se fait indépendamment avec le clavier, avec des touches d'accès direct à certains éléments géométriques, numériquement, ou encore par l'intermédiaire d'une table à digitaliser. Ces différents modes de saisie sont accessibles à tout moment, y compris lorsqu'une commande est en cours de traitement.

Base de données

Le modèle de données utilisé est un modèle volumétrique, autorisant l'élimination des parties non visibles des objets sur les représentations. Les entités auxquelles a accès l'utilisateur sont les facettes et les volumes.

Représentations

La plupart des modes de représentation couramment utilisés en architecture et en design sont disponibles: projection perspective, projection parallèle axonométrique, isométrique ou orthographique, avec ou sans coupe, avec ou sans élimination des parties cachées.

L'algorithme de projection a été dessiné de façon à faciliter le paramètrage des représentations. Les paramètres accessibles sont position de l'oeil, position du point de visée, angle d'ouverture. La coupe, si elle existe, s'effectue par un plan perpendiculaire à l'axe du regard passant par le point de visée.

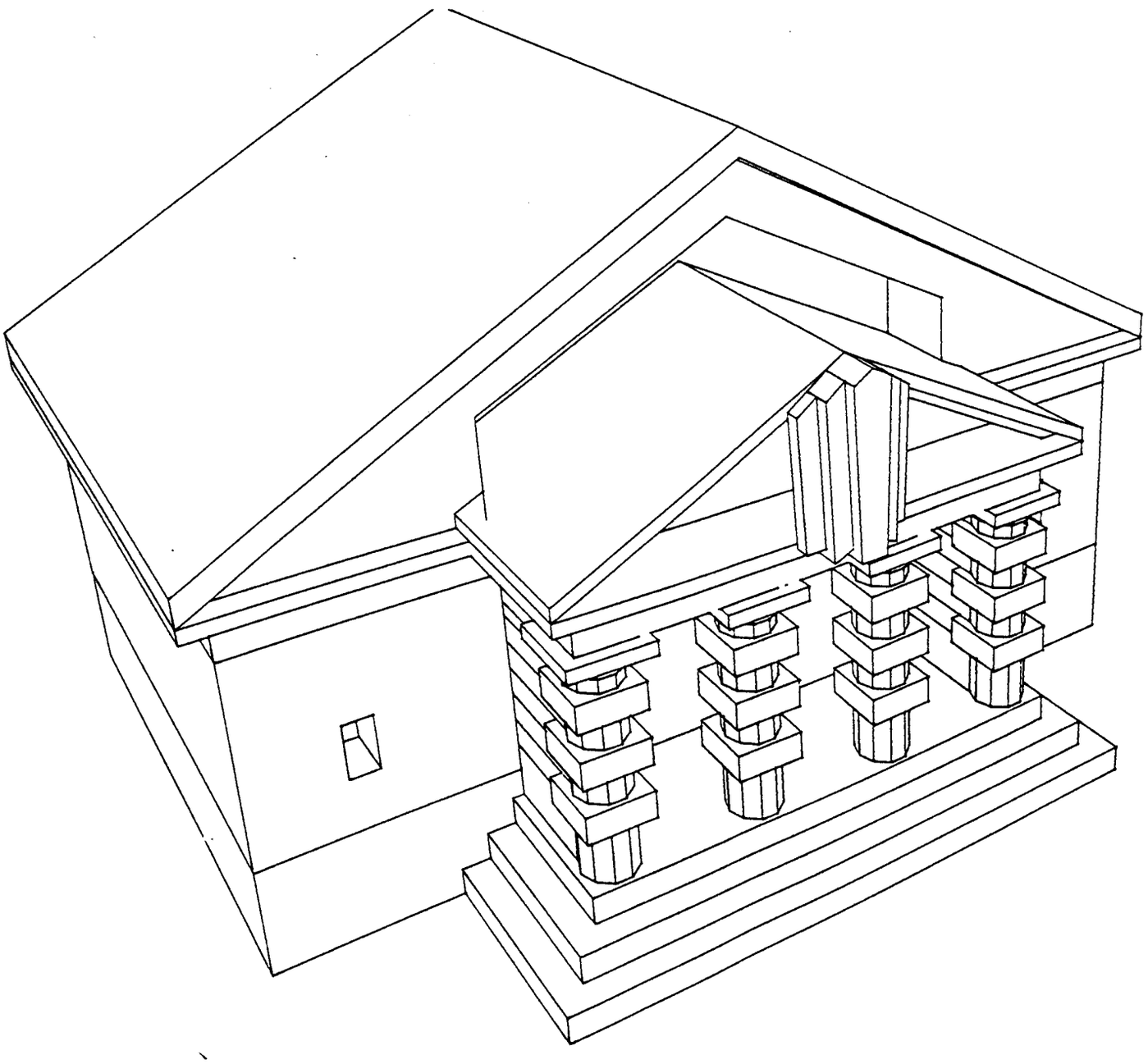
L'algorithme d'élimination des faces cachées est orienté ligne, afin de permettre la sortie sur traceur, et de conserver la précision de calcul quelle que soit l'échelle de tracé. Il a été écrit spécifiquement pour Formes, et autorise la modélisation de surfaces et volumes concaves.

Le portage sur environnement Macintosh répondait à plusieurs attentes :

- bénéficier d'une puissance de calcul accrue, pour envisager une extension de ses fonctionnalités (opérations booléennes, synthèse d'image, etc) ;
- par l'intermédiaire des développements récents sur l'interface logiciel/utilisateur, implantés sur les machines Apple, augmenter encore sa simplicité d'utilisation et sa convivialité ;
- en 1988, quatre ans après l'apparition de la gamme Macintosh, très peu de logiciels de CAO étaient disponibles, et surtout trop peu d'entre eux offraient une interface comparable. Il était nécessaire de s'insérer rapidement dans ce marché prometteur alors en pleine extension.

Autant le portage d'un logiciel d'une machine à une autre peut être rapide si ces matériels sont compatibles (utilisent le même système d'exploitation), autant le passage de l'univers IBM à l'univers Apple est long et délicat.

Dans le monde IBM, l'écriture d'un logiciel se fait en complète indépendance par rapport aux logiciels préexistants. Chaque développement nécessite une refonte complète de l'interface. Tous les logiciels sont différents, leurs modes de fonctionnement sont différents, leurs apparences sont différentes. IBM, jusqu'à récemment, ne se préoccupait guère du confort des utilisateurs de ses machines.
Apple, lors de la création du Macintosh, a voulu modifier radicalement les relations entre l'homme et la machine, et a édicté un ensemble de règles à l'usage des développeurs. Cette sorte de guide de programmation de l'interface permet d'offrir à l'utilisateur final une convivialité sans égale.



I. L'environnement Macintosh

"Le Macintosh a été dessiné pour s'ouvrir sur une audience de non-programmeurs. Pour atteindre ce but, les applications pour Macintosh doivent être faciles à apprendre et à utiliser (...) L'utilisateur doit se sentir maître de la machine, et non l'inverse."

'The Macintosh User Interface Guidelines' [1]

Une des plus importantes qualités que doit avoir une application pour Macintosh est la simplicité d'apprentissage et d'utilisation. Pour ceci, il faut que l'utilisateur retrouve dans toute nouvelle application les modes opératoires qu'il connaît.

A cette fin, Apple encourage l'utilisation intensive de la 'ToolBox' (boîte à outils) contenue dans le système de l'ordinateur, et qui constitue le cœur de l'interface. Cette boîte à outils est un ensemble de procédures complexes, gérant aussi bien la mémoire de l'ordinateur, la souris, le clavier, mais aussi tout ce qui est visible par l'utilisateur, écran, fenêtres, menus, etc. Cette boîte à outils est le lien entre toutes les applications écrites pour le Macintosh, qui leur donne leur air de famille, et qui permet à l'utilisateur de manipuler facilement, intuitivement, tout nouveau logiciel.

Si l'avantage d'un standard de développement d'interface est un gain énorme pour l'utilisateur, qui lui rend transparentes toutes les opérations qui ne sont pas directement liées à son travail, et le décharge de toute la gestion de la machine elle-même, il n'en va pas de même pour le programmeur.

Celui-ci, s'il n'a pas à chaque application à réinventer la roue, si l'interface logicielle de son application est déjà contenue en pièces détachées dans l'ordinateur, est contraint pour la mettre en œuvre d'ingurgiter le contenu complet de la boîte à outils, c'est à dire quelques 500 routines et 3000 pages de documentation. Une connaissance trop fragmentaire de cet environnement de programmation, ou la négligence des règles édictées par Apple dans 'The Macintosh User Interface Guidelines', entraînera soit des dysfonctionnements du logiciel (contexte non prévu, non-compatibilité ascendante sur les machines ultérieures), soit des manques au niveau de l'interface et de sa souplesse, entraînant rapidement un rejet de la part de l'utilisateur.

L'interface Macintosh

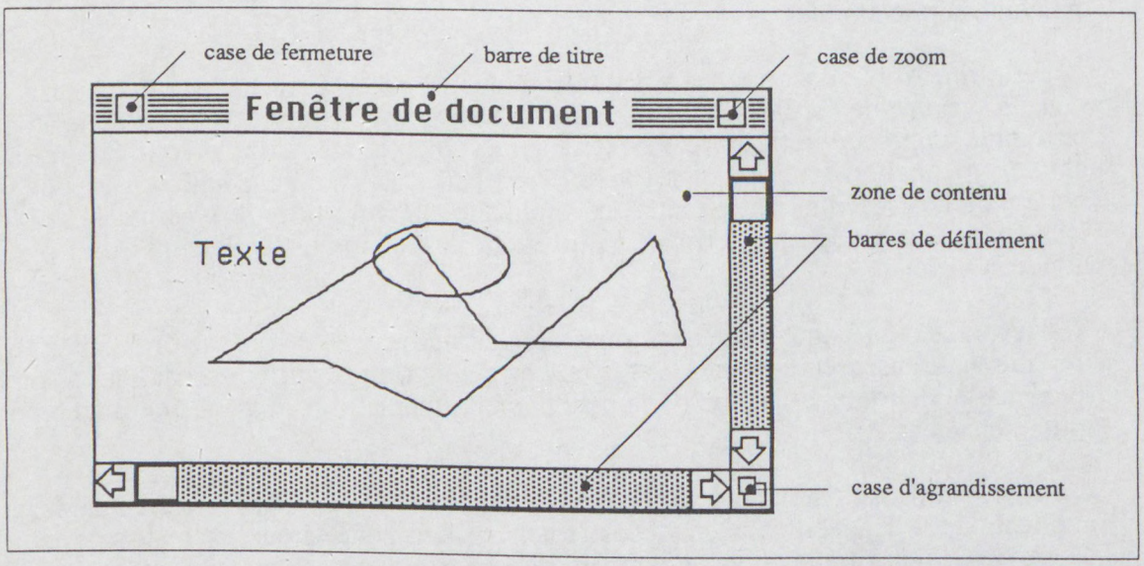
L'interface dessinée par Apple à destination des développeurs comprend un grand nombre d'entités, graphiques ou autres, permettant de contrôler l'apparence de l'application, et de présenter sous divers formats les informations à destination de l'utilisateur.

Les entités de plus haut niveau sont les fenêtres, les documents, les menus.

1. *'The Macintosh Interface GuideLines'*, recommandations Apple pour le design et le comportement des applications destinées à Macintosh.

Sur Macintosh, tout est affiché graphiquement ; il n'existe pas de mode texte. Toutes les informations affichées le sont dans des fenêtres, généralement rectangulaires. Dans le cas qui nous concerne, le dessin assisté, l'application pourra permettre à l'utilisateur d'afficher dans différentes fenêtres différentes vues d'un même objet, donc d'une même information.
Ces fenêtres peuvent être déplacées ou agrandies à volonté, passées en premier ou en arrière plan.

Le type de fenêtre le plus commun est le type 'document'. C'est dans ce type de fenêtre que l'information que traite l'utilisateur sera affichée (un texte dans un traitement de texte, un dessin dans un logiciel graphique). Ces fenêtres comprennent plusieurs zones, à usages différents, que l'on retrouvera dans tous les logiciels.

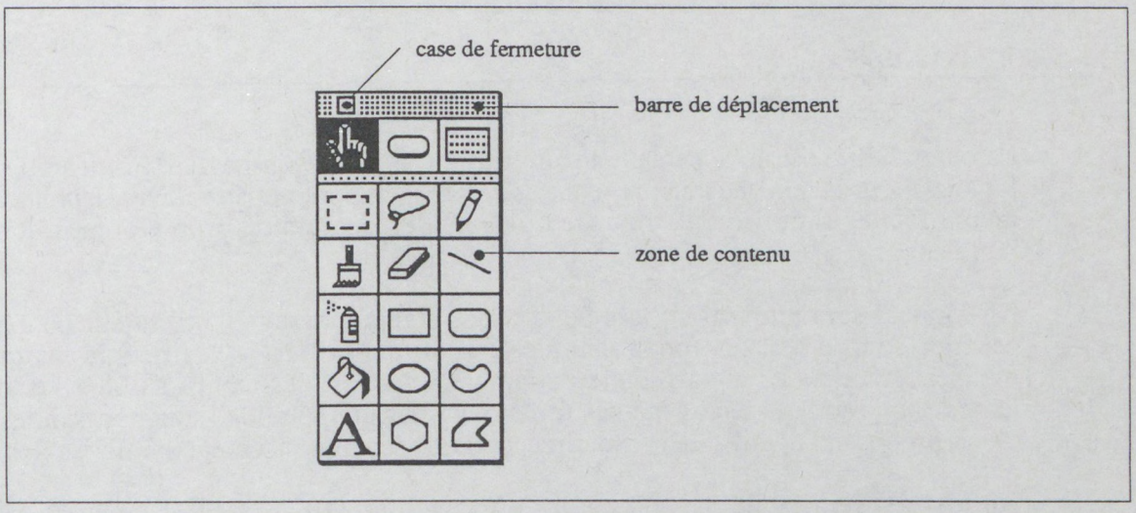


Fenêtre standard de document

La *zone de contenu* est le rectangle où sont affichées les informations. Sa taille dépend de la taille de la fenêtre.
La *barre de titre* est dans la plupart des cas obligatoire, puisqu'elle permet à l'utilisateur de reconnaître au premier coup d'oeil le contenu de la fenêtre. Ce titre peut être le nom du document affiché (le nom du fichier par exemple), ou encore le nom de la vue présentée (vue de face, perspective, etc). C'est par la barre de titre que l'utilisateur déplace la fenêtre.
La *case de fermeture* permet de fermer la fenêtre, ou de la masquer.
La *case de zoom* fait passer la taille de la fenêtre entre son état actuel et l'état standard, qui est généralement la dimension totale de l'écran.
La *case d'agrandissement* permet de redimensionner la fenêtre.
Les *barres de défilement* permettent de déplacer la fenêtre sur le document, c'est à dire de déplacer la zone visible du contenu.

Une application spécifique pourra changer l'apparence générale des fenêtres de document, tout en en conservant le comportement.

Un autre type de fenêtre communément utilisé dans les logiciels à vocation graphique est la fenêtre de palette. Comme son nom l'indique, elle présente à l'utilisateur, sous forme d'icônes, la palette des outils disponibles au niveau courant d'utilisation. Ces palettes complètent les menus, qui présentent une liste limitée de fonctions essentielles.



Exemple de fenêtre de palette (tiré d'HyperCard)

De même que les fenêtres de document, les fenêtres de palette comprendront une zone de contenu, une case de fermeture, et une zone qui permettra de les déplacer à l'écran. Par contre, leur taille reste fixe, et elles restent en permanence au premier plan, de façon à survoler les fenêtres de document.

Nous verrons ultérieurement comment ces types de fenêtres sont implantés dans Formes, de façon à suivre au mieux les besoins d'un logiciel de Dessin Assisté.

Les documents

Le concept de document est une partie importante de l'interface Macintosh. Par document, on entend communément l'ensemble des données sur lesquelles travaille l'utilisateur au cours d'une session. Celles-ci peuvent être textuelles, graphiques, ou mixtes. Les informations que contient un document peuvent être sauvegardées dans un fichier, en mémoire de masse.

Un logiciel de traitement de texte, ou un logiciel graphique 2D présentera dans une seule fenêtre l'ensemble du document, même si la présentation des informations se fait sous des formes différentes. Dans un logiciel graphique en trois dimensions, il est évident que l'utilisateur a besoin de lire les informations affichées sous des angles différents, et simultanément. Un logiciel qui ne présenterait à l'utilisateur qu'une vue des volumes qu'il gère ne saurait prétendre à un véritable travail en trois dimensions.

Les menus

Un menu présente à l'utilisateur une liste des actions qu'il peut mettre en œuvre à un moment donné de la session de travail. Le mode de fonctionnement d'un menu est simple : ouvrir le menu en le désignant à la souris, choisir un article du menu, lâcher la souris. Ce mode opératoire, s'il est facile à utiliser, puisqu'il n'est pas nécessaire de

mémoriser les commandes, comporte deux inconvénients : l'utilisateur n'a pas en permanence sous les yeux les commandes disponibles, et la liste des actions présentées est nécessairement limitée. Les fenêtres de palette permettent de contourner ces deux obstacles.

La convivialité

La convivialité est un aspect d'une application trop souvent négligé jusqu'à présent. Les logiciels lourds tels que ceux rencontrés en CAO ne sont ni particulièrement attractifs, ni même faciles à utiliser. Ils imposent des modes opératoires trop éloignés de ceux de l'utilisateur.

Un logiciel sera attractif si, lors de la phase d'apprentissage, l'utilisateur peut saisir au premier coup d'oeil les fonctionnalités qui lui sont offertes, les opérations qu'il peut réaliser, grâce à des menus clairs et compréhensibles, des icônes lisibles ; si, après un temps d'apprentissage, ces mêmes fonctionnalités et opérations sont accessibles rapidement, proposant à l'utilisateur expérimenté des raccourcis aisément mémorisables.

Un logiciel sera facile d'emploi si les informations gérées peuvent être affichées de la manière la plus naturelle possible, le plus rapidement possible ; si les actions de l'utilisateur sont immédiatement répercutées à l'écran, et si elles peuvent être invalidées.

Un logiciel de CAO gère souvent une grande masse d'informations. La modification de ces informations entraîne une réactualisation du dessin qui peut durer très longtemps. Même si le gain de temps par rapport à un dessin manuel est appréciable, l'utilisateur peut se trouver dans une situation où il passe plus de temps à attendre la régénération des tracés qu'à manipuler effectivement ses données. Peu de logiciels permettent à l'utilisateur de continuer à travailler pendant les réactualisations, ou même d'interrompre le tracé en cours.

Les logiciels de CAO existants

Dans la gamme des logiciels de CAO disponibles dans l'univers Macintosh, on trouve des logiciels de finalités et d'ambitions différentes : du simple modeleur peu onéreux, utile en phase de recherche volumétrique, au progiciel coûteux, assurant l'ensemble de la chaîne de production architecturale, de l'esquisse au descriptif en passant par les plans de détail.
Il est évident qu'aussi bien l'interface que la facilité d'apprentissage et d'usage de ces logiciels diffèrent.

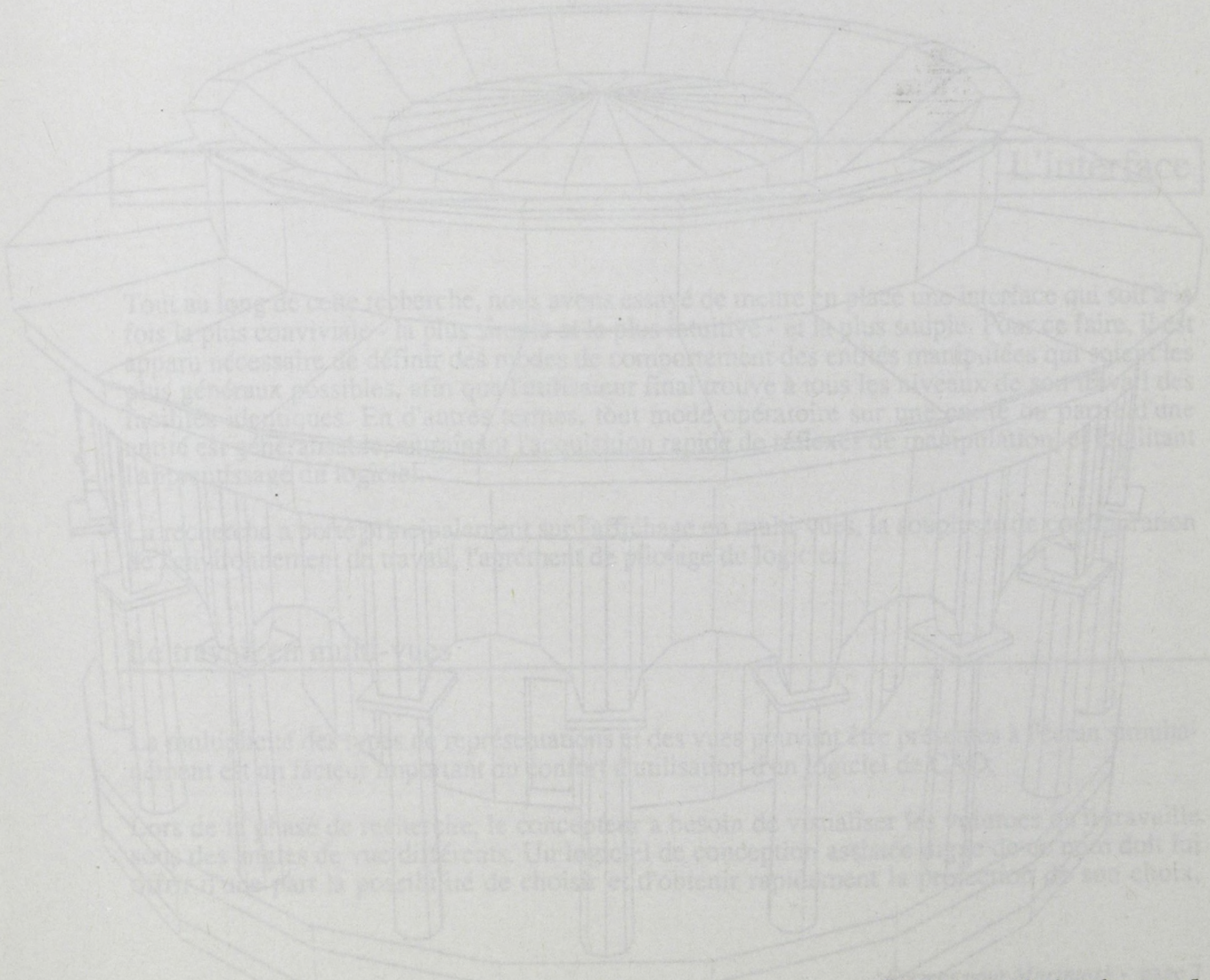
Dans un modeleur de faible coût, comme SpaceEdit™ de Abvent, nous trouverons une unique fenêtre de document, découpée en quatre zones, chacune correspondant à un type de projection : trois projections orthographiques, une vue perspective. Une palette attachée à la fenêtre présente les opérations les plus couramment utilisées.
SpaceEdit est d'apprentissage simple, mais les tracés obtenus ne peuvent prétendre à un usage professionnel (qualité du dessin, mauvais algorithme de faces cachées, peu d'outils de gestion de page).

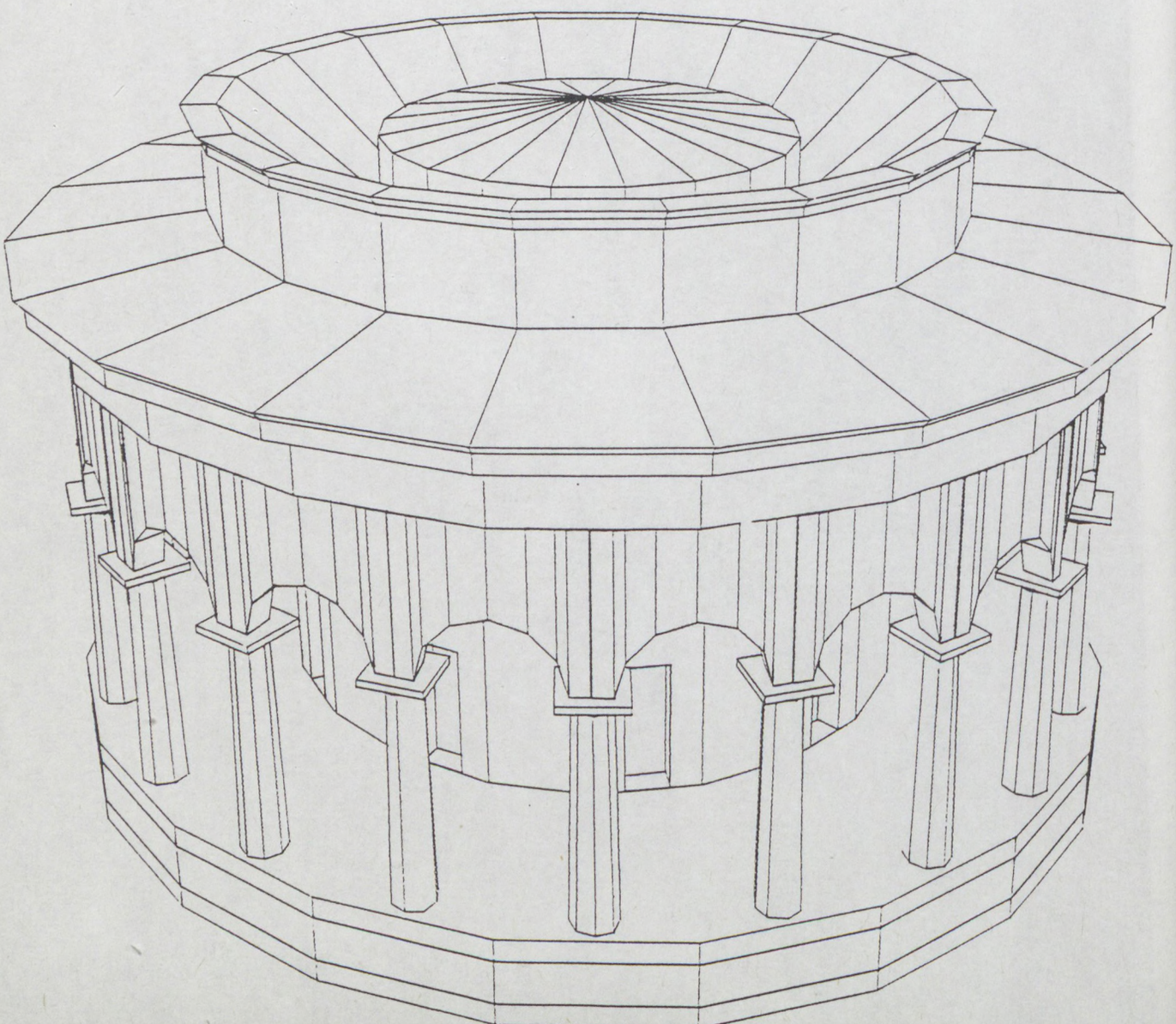
Un logiciel de CAO professionnel, tel que Architrion™ de Gimeor, proposera des outils beaucoup plus sophistiqués, permettant la prise en charge de la quasi-totalité de la chaîne de production, jusqu'à l'élaboration de métrés et de devis.

Mais apparemment, ce gain en puissance se fait au détriment de la qualité de l'interface. L'utilisateur n'a sous les yeux qu'une seule projection, lui interdisant la visualisation en trois dimensions ; la conception se fait en deux dimensions, la saisie de la troisième se faisant numériquement. Il est délicat dans ce cas de parler de vrai 3D.
De plus, les outils de modélisation et de modification des volumes ne sont pratiquement accessibles que par menu. L'utilisateur débutant n'a d'autres choix que de se reporter à la documentation, rendant l'apprentissage du logiciel particulièrement rebutant.

A l'opposé en matière d'interface utilisateur, Zoom, édité par Abvent. Ce logiciel, orienté design et image de synthèse, offre une convivialité exemplaire. L'affichage est multi-fenêtre, ce qui permet, contrairement à SpaceEdit, de ne faire apparaître que les projections indispensables. Les outils disponibles s'affichent dans plusieurs fenêtres de palette, chacune regroupant un ensemble de fonctionnalités de même type d'usage.

Beaucoup de logiciels professionnels gagnent en convivialité en accélérant la gestion du graphique. Mais ce gain de temps, et de productivité, se fait au détriment soit de la complexité des objets qu'ils gèrent (Architrion ne gère que des volumes parallèlépipèdiques), soit de la qualité des dessins (SpaceEdit fait facilement jusqu'à 30% d'erreurs dans l'élimination des faces cachées).





II. Formes sur Macintosh

L'ambition qui a présidé au portage de Formes sur l'environnement Macintosh était de réaliser un logiciel qui intègre à la fois tous les apports de cet environnement (convivialité, facilité d'apprentissage et d'usage), et la simplicité et la puissance de la première version de Formes.

La première phase du portage a porté sur la découverte de l'environnement de programmation du système Macintosh, la recherche de l'interface "idéale" d'un logiciel de CAO, l'expérimentation de l'interface et l'implantation d'une base de données géométrique 3D.
Mais l'ampleur du travail à effectuer pour obtenir une maquette rapidement utilisable a conduit à un portage direct de machine à machine, avec les seules modifications essentielles au changement de processeur, sans intégration des développements réalisés sur l'interface.

La suite de ce rapport sera donc divisée en deux parties principales :

- Présentation des recherches effectuées en première phase (interface, convivialité, base de données)
- Le portage proprement dit.

L'interface

Tout au long de cette recherche, nous avons essayé de mettre en place une interface qui soit à la fois la plus conviviale - la plus simple et la plus intuitive - et la plus souple. Pour ce faire, il est apparu nécessaire de définir des modes de comportement des entités manipulées qui soient les plus généraux possibles, afin que l'utilisateur final trouve à tous les niveaux de son travail des facilités identiques. En d'autres termes, tout mode opératoire sur une entité ou partie d'une entité est généralisable, entraînant l'acquisition rapide de réflexes de manipulation, et facilitant l'apprentissage du logiciel.

La recherche a porté principalement sur l'affichage en multi-vues, la souplesse de configuration de l'environnement de travail, l'agrément de pilotage du logiciel.

Le travail en multi-vues

La multiplicité des types de représentations et des vues pouvant être présentes à l'écran simultanément est un facteur important du confort d'utilisation d'un logiciel de CAO.

Lors de la phase de recherche, le concepteur a besoin de visualiser les volumes qu'il travaille sous des angles de vue différents. Un logiciel de conception assistée digne de ce nom doit lui offrir d'une part la possibilité de choisir et d'obtenir rapidement la projection de son choix,

mais, et c'est le plus important, lui permettre de créer et de modifier les volumes depuis n'importe laquelle de ces projections.

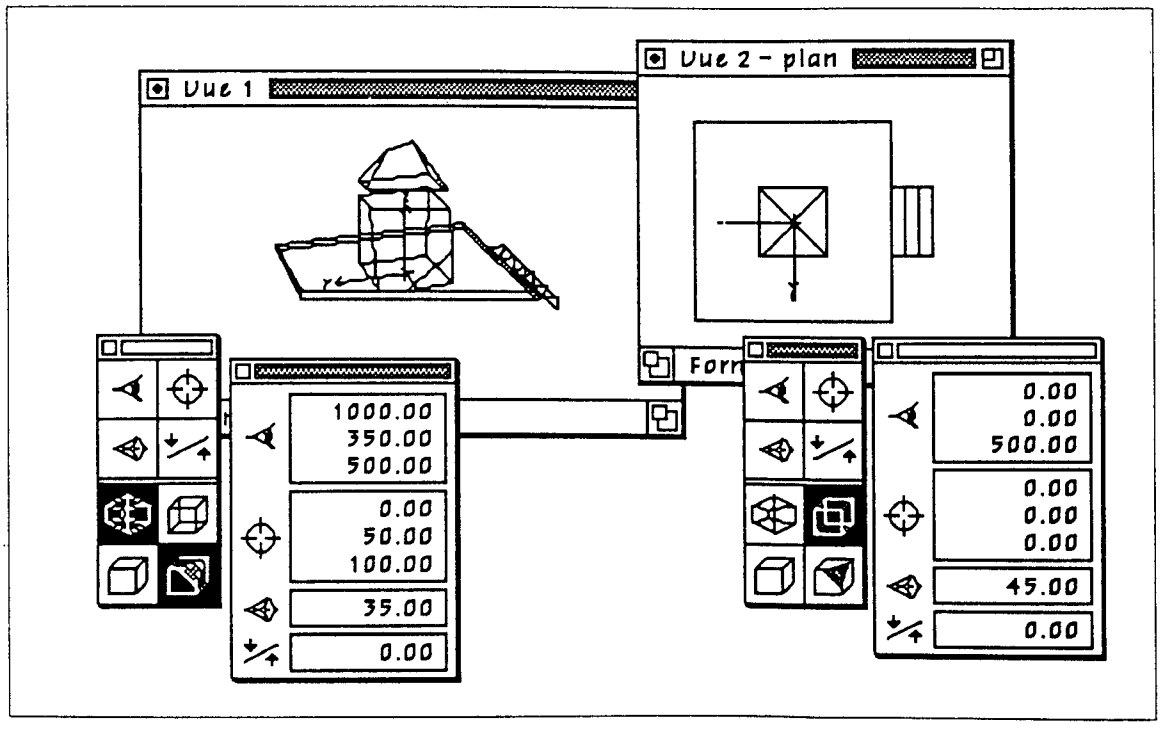
S'il est évidemment difficile, quelle que soit la maîtrise de l'utilisateur sur son outil, de manipuler des données en trois dimensions sur une projection non orthographique, rien ne peut et doit empêcher les générations ou transformations géométriques sur une coupe, une élévation, une vue de dessous, etc.

La solution qui fut mise en œuvre dans la version BFM de Formes est de présenter en standard un écran découpé en quatre projections, trois vues orthographiques et une perspective de contrôle. Cette configuration pouvait être modifiée, soit en déplaçant et agrandissant ces cadres de tracé, soit en commutant entre les quatre vues et une vue plein écran.

Bien entendu, le portage de Formes dans l'univers Macintosh profite pleinement de la puissance du gestionnaire de fenêtres.

Nous avons vu précédemment qu'une fenêtre au sens Macintosh présente l'ensemble des informations constituant un document. Dans Formes, chaque fenêtre n'est qu'une représentation particulière des objets manipulés. Ainsi, toutes les fenêtres dites de document ont un seul et même statut, celui de vue sur les objets, et donc un même comportement. Aucune différenciation n'est faite entre un plan, une coupe, une façade, une perspective.

L'utilisateur crée une fenêtre, et définit pour celle-ci les paramètres de projection : position de l'observateur, direction du regard, angle d'ouverture, et type de projection : perspective, parallèle axonométrique, parallèle oblique.



Chaque fenêtre a ses propres paramètres de projection

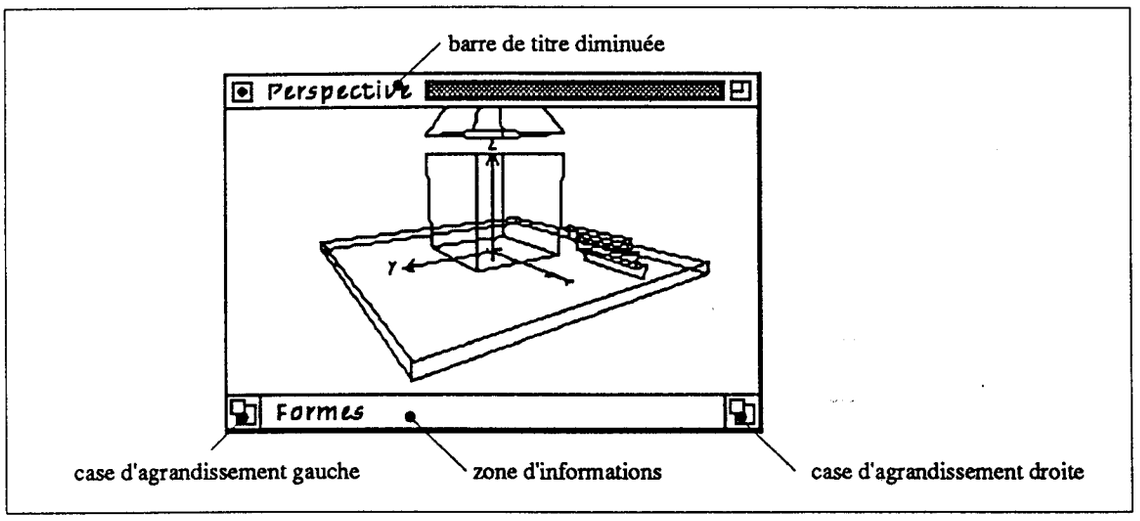
Toutes les opérations sont possibles dans toutes les vues : création et manipulation d'objets, désignation, élimination des parties cachées, etc. Le confort d'utilisation est donc considérable, le travail étant possible quelle que soit la fenêtre de premier plan.

Le choix de design et de comportement des fenêtres de document a été guidé par plusieurs priorités :

- permettre la définition et le paramètrage rapide de représentations multiples ;
- réserver la plus grande surface possible de l'écran pour la représentation graphique ;
- offrir un confort extrême à l'utilisateur dans la manipulation et l'accès aux représentations qu'il aura défini ;
- autoriser un travail et un maniement identiques quel que soit le type de représentation ;

Design

Pour ce faire, les fenêtres de document destinées à recevoir les représentations des objets manipulés ont été redessinées, tout en suivant les recommandations du guide Apple.



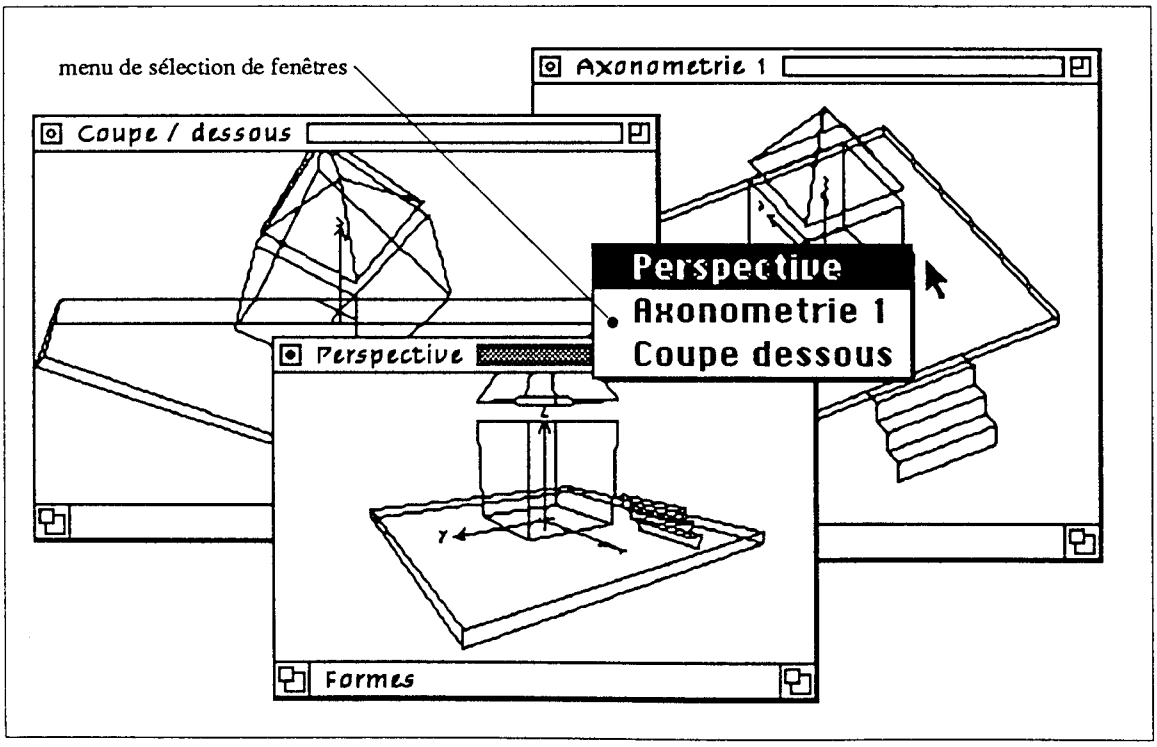
Fenêtre de document Formes

La barre de titre a vu sa hauteur diminuée, de façon à introduire en partie basse une zone de texte servant à l'affichage d'une ligne de statut et d'assistance, sans pour autant réduire de façon significative la taille de la zone de contenu.
Une case d'agrandissement a été rajoutée dans la partie gauche de la fenêtre, pour permettre un élargissement à partir de cette région, impossible à réaliser avec la fenêtre de document traditionnelle - pour laquelle il faut d'abord déplacer la fenêtre sur la gauche, puis l'agrandir à droite, puis déplacer son contenu.

Comportement

Le comportement de ces fenêtres diffère aussi du comportement des fenêtres prédéfinies.

Dans l'interface standard, le passage d'une fenêtre en avant-plan nécessite de la désigner à la souris, et donc qu'elle soit visible. Dans Formes, une touche permet d'obtenir sous la souris, à n'importe quel endroit de l'écran, un menu déroulant permettant de sélectionner la fenêtre à amener en premier plan ; de même, tout clic souris dans une zone n'appartenant pas à une fenêtre fait apparaître ce menu. L'utilisateur a ainsi accès instantanément à la liste des représentations qu'il aura défini.



Sélection des fenêtres en ligne

Toujours dans l'interface standard, le déplacement ou l'agrandissement d'une fenêtre d'arrière plan est impossible, imposant à l'utilisateur une gymnastique rapidement fastidieuse pour configurer son espace de travail. La gestion avant/arrière-plan a été repensée, permettant ainsi de manipuler les fenêtres quelles que soient leurs positions. De la même façon, le déplacement de la zone de visualisation (de la partie visible) de la projection d'une fenêtre d'arrière-plan est possible.

Ces facilités autorisent une grande souplesse de configuration de l'espace de travail, d'autant plus que celle-ci est mémorisable, et sauvegardée en fin de session.

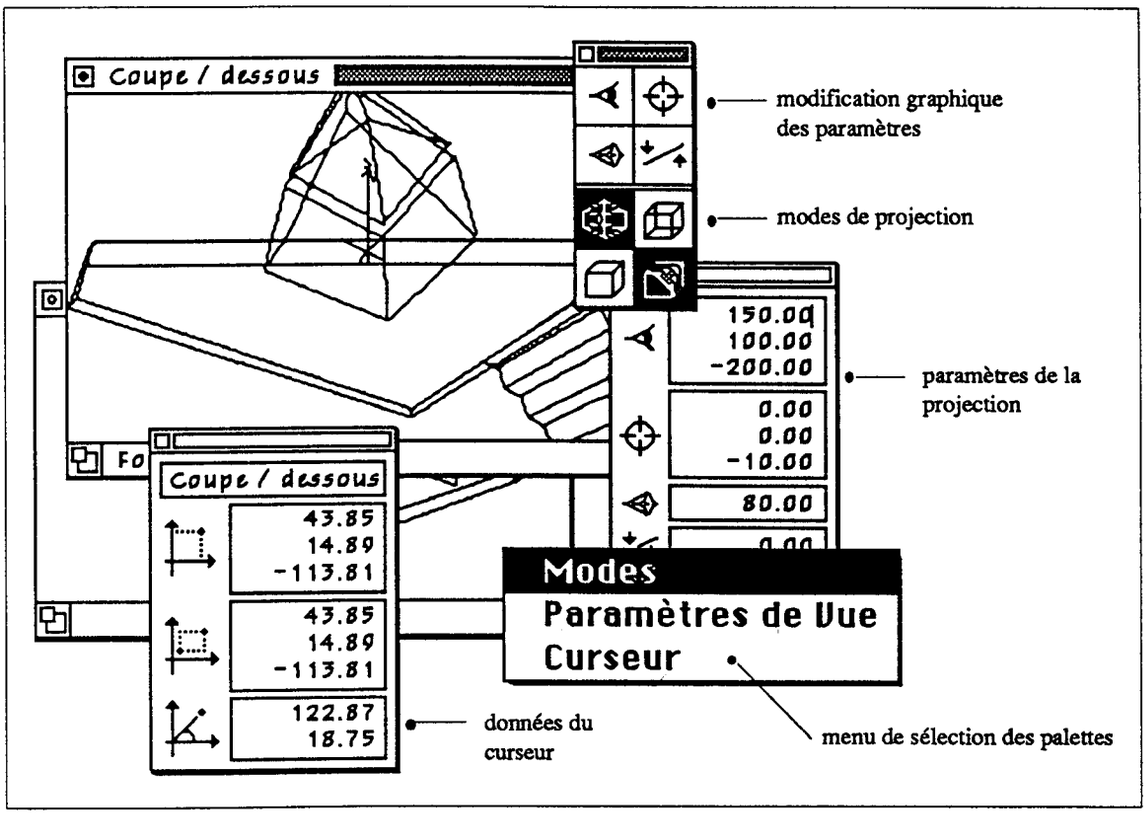
Les fenêtres de palette

Nous avons vu que le nombre important de fonctions que comporte un logiciel de Dessin Assisté nécessite l'usage de fenêtres de palettes, permettant à l'utilisateur d'avoir en permanence sous les yeux, à l'endroit où il veut, l'ensemble des commandes utilisables à un moment donné de la session de travail.

Toujours dans l'optique de faciliter la mise en place et le maniement des entités composant l'espace de travail de l'utilisateur, nous avons défini un type de fenêtres de palette dont le design est similaire et le comportement identique à celui des fenêtres de document :

- avec combinaison d'une touche du clavier, sélection de la fenêtre de palette à amener en premier plan, sans que celle-ci soit visible ;
- manipulation des fenêtres d'arrière-plan sans passage obligatoire en avant-plan.

Les fenêtres de palette essentielles sont celles qui permettent de modifier les paramètres de représentation des fenêtres de document.



Les fenêtres de palette et le menu de sélection en ligne

La convivialité

La convivialité d'un logiciel quelconque, a fortiori de CAO, repose comme nous l'avons vu sur la simplicité d'utilisation et la cohérence des réponses du logiciel aux actions de son utilisateur. Ce comportement général du logiciel peut être rapidement atteint si l'on suit à la lettre les recommandations Apple, aboutissement de longues recherches en ergonomie. Mais l'aspect le plus crucial et le plus difficile à gérer, dans des applications qui traitent une masse importante de données, est le temps de réponse du logiciel.

Deux points importants méritent d'être traités en profondeur :

- en aucun cas l'utilisateur ne doit être bloqué dans son travail parce que le logiciel est engagé dans une phase de calcul gourmande en temps machine (cas du dessin d'une projection, de l'élimination des parties cachées) ;
- le logiciel doit avoir un minimum d'intelligence pour arrêter une tâche en cours d'exécution, si une modification du contexte de celle-ci la rend obsolète : par exemple, il est inutile de poursuivre le calcul d'une projection si l'utilisateur désire modifier un des paramètres de cette projection ; ou encore, continuer d'éliminer les parties cachées d'une perspective n'a plus de sens si l'utilisateur veut supprimer l'un des volumes de la scène.

Le travail en multi-tâches

L'interactivité maximum est atteinte lorsque le logiciel est réellement multi-tâches. Ce qui signifie que le temps machine doit être partagé entre différentes tâches, avec différents niveaux de priorité.

La tâche de plus haute priorité est bien évidemment la réponse aux actions de l'utilisateur ; le deuxième niveau concerne les tâches d'actualisation des dessins sur lesquels celui-ci travaille ; le dernier niveau de priorité est celui de tâches pouvant se dérouler en arrière-plan, et qui n'ont pas d'interférences avec le travail en cours : gestion de la mémoire, transfert d'informations entre logiciels et/ou mémoires de masse, impression, élimination des parties cachées, coloration d'image (jusqu'à un certain stade).

L'"intelligence" du logiciel est de gérer les relations entre les tâches en cours d'exécution et leurs niveaux de priorité. Par exemple, la tâche de mise à jour du contenu d'une fenêtre devra prendre en compte les nouvelles mises à jour qui peuvent intervenir pendant son exécution. De même, la procédure d'élimination des parties cachées en cours dans l'une des fenêtres devra juger si la modification apportée aux données dans une autre fenêtre mérite sa propre interruption ou au contraire sa continuation [1].

Après test, on s'aperçoit que les techniques développées apportent un confort inégalé. Sur un logiciel courant, jusqu'à 20% du temps machine est utilisé pour des tâches de calcul ou de rafraîchissement de dessins, si le nombre d'objets présents est important. Sur Formes, la plupart des calculs s'effectuant en tâches de fond, étant interruptibles à volonté, l'utilisateur a toujours un contrôle complet sur le déroulement du logiciel, celui-ci étant constamment en attente de ses ordres.

La difficulté majeure rencontrée lors du développement fut, outre l'écriture d'un gestionnaire de tâches, de réécrire les procédures de gestion de fenêtres, qui n'étaient pas à l'origine prévues pour un fonctionnement en tâches de fond différées [2].

La base de données

Formes a été dessiné pour produire rapidement des représentations en trois dimensions d'objets complexes, sans prétendre offrir une assistance à la conception qui, par principe, devrait intégrer, en plus de la description géométrique des objets, des descriptions qualitatives qui permettrait la gestion de descriptifs, de métrés, de calculs de structure ou de thermique, etc. La base de données est évidemment extensible, n'interdisant en rien l'accroissement des fonctionnalités dans cette direction, mais seules les données géométriques sont actuellement implantées.

Plusieurs critères ont guidé le choix d'un modèle de données :

- une description surfacique de la géométrie des objets ;
- la souplesse qui permettra à l'utilisateur de créer des objets de la manière la plus intuitive possible, sans aucune limite de complexité ;
- l'optimisation de la gestion de la mémoire, afin de profiter au mieux de la puissance du gestionnaire du système Macintosh [3] ;
- l'optimisation du parcours de la base de données, afin d'accélérer les tâches de calcul et d'affichage.

---

1. voir annexe 'Gestion des tâches'.

2. Une tâche différée ne sera exécutée que lorsque l'opération en cours sera terminée. Seul l'installation de la tâche est effectuée en temps réel, le gestionnaire se chargeant de sa mise en route en temps voulu.

3. Le gestionnaire de mémoire (*Memory Manager*) du système Macintosh offre de nombreuses procédures permettant les allocations et désallocations de zones de la mémoire.

Description volumétrique

La description surfacique est la seule capable d'autoriser l'élimination des parties cachées des représentations des objets, pour l'obtention d'images de rendu lisibles et réalistes.

Les entités

- Le point est la description dans un repère orthonormé à trois dimensions d'un sommet d'un objet ;
- la facette est la description point par point d'une face de l'objet ; l'arête est donc un simple lien entre deux points consécutifs d'une facette, un segment limite des plans des faces de l'objet ;
- un objet est une liste de facettes.

Lorsque un nombre important d'objets sont présents, il devient indispensable de permettre leur regroupement ou association dans une même entité, de façon à faciliter leur repérage et leur gestion. A donc été introduit un niveau hiérarchique supérieur, le groupe, qui est une liste d'objets, soit élémentaires, soit hiérarchiques eux-mêmes.

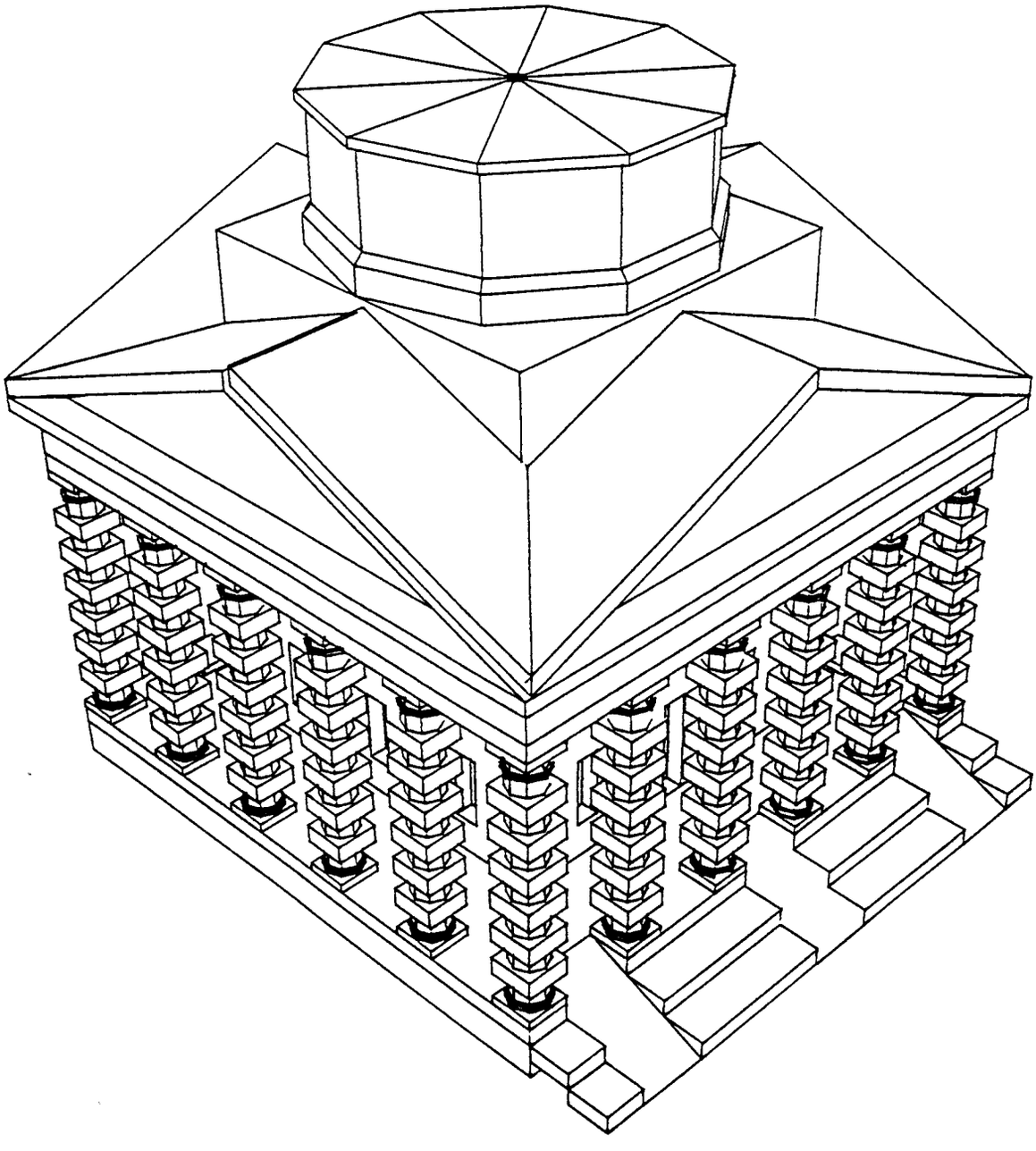
Toutes ces entités sont accessibles à l'utilisateur, pour lui laisser une totale liberté de création. En contrepartie, le logiciel, n'opérant aucun contrôle sur la cohérence des descriptions, ne pourra pas assister l'opérateur lors de manipulations ; par exemple, ne déterminera pas si des facettes des objets sont gauches, si des facettes ne sont pas entièrement clôturées, si des volumes s'interpénètrent.

Implantation

L'implantation de la base de données devait profiter de la puissance des outils du Macintosh, en particulier en ce qui concerne l'optimisation de la gestion de la mémoire. Ainsi, les allocations d'objets sont faites dynamiquement, à l'inverse de la version BFM. Le logiciel suit au mieux les besoins en mémoire des objets manipulés, permettant ainsi un meilleur fonctionnement général du système en environnement multi-applications[1].

Une des volontés qui ont présidées au choix du modèle d'implantation était de simplifier au maximum la gestion des entités géométriques, pour accélérer l'écriture des procédures. Un modèle de liste chaînée a été étudié. Sa structure facilite la gestion des insertions/suppressions d'entités, et le parcours de la structure complète (pour le repérage, les affichages, les calculs) [2].

1. Le *MultiFinder*, système multi-applications du Macintosh, autorise le fonctionnement simultané de plusieurs applications, qui peuvent ainsi coopérer et s'échanger leurs données, permettant à l'utilisateur de se créer un véritable environnement de travail.
2. voir annexe 'Base de données'.



III. Le portage de la version BFM

La réécriture complète de Formes à partir des recherches décrites ci-avant semblait être la meilleure façon de bénéficier de la qualité de l'interface du système Macintosh, tout en transcrivant au coup par coup les fonctionnalités qui faisait l'intérêt de la version BFM. Mais cette tâche paraissait hors de propos, au vu du temps de développement nécessité. Avant toute poursuite de ces recherches, nous avons voulu tester Formes dans une version la plus proche possible de l'original, avec les seules modifications rendues indispensables par le changement de machine et de système d'exploitation.

Le transfert

Il existe sur Macintosh des lecteurs 5 pouces 1/4, pour lire des disquettes formattées sur un PC IBM. Bien que les ordinateurs BFM soient décrits par leur constructeur comme des machines compatibles IBM, le formattage de disquettes n'est pas strictement identique sur les deux matériels. Il n'a donc jamais été possible de récupérer les fichiers sources du logiciel par simple transfert par disquettes.

Pour récupérer les fichiers sources du logiciel (12 fichiers textes; environ 12600 lignes, représentant 350 000 caractères), il a fallu utiliser une transmission par câble, sur interface série - après fabrication du câble -, et un logiciel de transfert de fichiers format ASCII.

La traduction

La traduction de la version BFM en version Mac n'a pas posé de problème au niveau strictement syntaxique, le langage utilisé, *Pascal*, étant relativement ancien et normalisé. Par contre, le vocabulaire lui-même, ainsi que l'enrobage, est différent. Cette partie du rapport, pour ne pas rentrer trop avant dans les détails techniques, sera une simple énumération des différences essentielles de programmation entre un système MS DOS et Mac OS.

La notion d'événement

Une application typique sur Macintosh est pilotée par des *événements*. A chaque fois que l'utilisateur presse le bouton de la souris, frappe une touche du clavier, insère une disquette, le système notifie à l'application un événement qui décrit l'action effectuée: son lieu, son type, son objet, sa date, etc. L'application décide, en fonction de cet événement, des opérations à réaliser : fermer une fenêtre si l'utilisateur a cliqué dans sa case de fermeture, effectuer l'action correspondant à la touche frappée, mettre à jour une fenêtre passant en premier plan, etc. La réponse

faite à l'action de l'utilisateur, l'application se remet en attente d'un nouvel événement. Le cœur du programme est donc une *boucle de traitement d'événements*.

Ce mode de fonctionnement est totalement différent de la méthode classique qui consiste à diriger constamment l'utilisateur au long d'une chaîne de d'actions enchaînées logiquement. Au contraire, celui-ci peut opérer dans l'ordre qui lui semble le plus approprié. Tout au moins, il peut connaître à tout moment l'ensemble des opérations qui lui sont accessibles à un instant donné, et choisir parmi celles-ci.

Dans la version BFM de Formes, cette notion de boucle de traitement des événements avait déjà été introduite. L'application était en permanence en attente d'un ordre de l'utilisateur : déplacement du curseur, touche clavier. Par contre, l'utilisateur n'était prévenu qu'a posteriori que sa commande n'était pas valide à cet instant, par un message laconique.

De plus, les types d'événements Macintosh sont beaucoup plus variés, du fait même que les objets qui composent son interface sont plus divers :

- événements souris : *mouseDown*, *mouseUp*, *mouseMoved* ;
- événements clavier : *keyDown*, *keyUp*, *autoKey* ;
- événements disquettes : insertion ;
- événements fenêtres : activation, déactivation (changement de plan) ;
- événements système : *suspend*, *resume* (commutation d'application) ;
- ...

Le portage de Formes sur Mac a donc nécessité une réécriture partielle de la boucle principale de traitement des événements, et son extension.

De même, dans la version BFM, le clavier et la tablette à digitaliser étaient les seuls modes de saisie. Dans la version Mac, il a fallu autoriser la saisie par souris. L'utilisateur agit, par son intermédiaire, sur les menus et les fenêtres, mais aussi sur la saisie des objets et sur leurs transformations.

La gestion des commandes

Dans la version BFM, l'utilisateur choisissait ses actions par l'intermédiaire du clavier, et accessoirement d'une tablette à digitaliser - permettant le déplacement du curseur, et possédant une partie du menu -. Chaque touche du clavier était affectée à l'une des quelques cinquante fonctions disponibles.

Sur Macintosh, il est hors de question de procéder de la même façon. L'utilisateur doit avoir accès aux commandes par l'intermédiaire soit de menus (avec des équivalents claviers pour certains), soit de fenêtres de palette, dont les symboles représentent les diverses actions qu'elles permettent.

Nous ne pouvions dans cet phase de transfert consacrer notre temps au dessin de fenêtres de palette, seules capables de contenir sous forme condensée une telle somme de fonctions.

La solution choisie est une solution hybride : offrir les principales fonctions dans des menus, l'appel à ces menus plaçant l'utilisateur dans un mode donné, où seules sont activées les commandes valides. Les fonctions secondaires sont accessibles au clavier, avec répercussion dans une fenêtre spécifique textuelle.

La gestion de l'écran

Dans la version BFM, l'application Formes se servait de la totalité de la surface de l'écran, et combinait mode texte et mode graphique. Sur Macintosh, il est indispensable de séparer les informations textuelles des dessins. Cette séparation entre informations, nous l'avons vu, s'effectue par l'intermédiaire des fenêtres. L'utilisateur agit sur la position et la taille de ces fenêtres, modifiant ainsi la répartition des surfaces affectées aux éléments d'information affichés.

La gestion des fenêtres est un processus complexe. Le système Mac OS met à disposition des développeurs des outils (routines) leur simplifiant la tâche.

Le portage a nécessité l'intégration dans cet environnement d'un logiciel qui supposait que tout l'écran lui appartenait. Pour accélérer le transfert, nous avons modifié le programme de façon à gérer une fenêtre de taille fixe, initialisée dans les règles (en prenant en compte les dimensions de l'écran lors du démarrage de l'application), mais que l'utilisateur ne pourrait ni déplacer ni agrandir.

Au-dessus de cette fenêtre principale, destinée à recevoir le graphique, plane une fenêtre plus petite, déplaçable, qui affiche les informations textuelles. Cette fenêtre peut être masquée.

Si le Macintosh est en mode multi-applications - sous MultiFinder -, la seule possibilité de commuter entre Formes et d'autres applications éventuelles est de choisir leur nom dans le menu standard du Finder. Cette solution permet donc de respecter un minimum les recommandations d'Apple, ou du moins de ne pas affoler un utilisateur non averti.

Une autre concession à l'interface Macintosh est l'intégration du menu standard d'édition, qui permet de transférer des informations d'une application à l'autre, par l'intermédiaire du *presse-papiers*. Dans Formes, ce menu permet de transférer les dessins en mode point, à destination d'une application quelconque qui pourra les imprimer. De la même façon, la sauvegarde des images peut se faire en mode vectoriel, en format PICT standard.

La gestion de la mémoire

L'environnement MS-DOS, sur IBM et compatibles, est mono-application, c'est à dire qu'une application tourne seule dans l'ordinateur. La gestion de la mémoire est alors simplifiée, puisqu'à chaque moment, l'application connaît l'espace mémoire qu'elle utilise, et donc l'espace restant.
Dans l'environnement Macintosh, depuis l'apparition du MultiFinder, système multi-application, un logiciel ne peut présumer de la place mémoire qui lui est attribuée, ni de celle qu'il pourra obtenir.

Dans cet environnement, il est nécessaire de prendre toutes les précautions lors de l'utilisation de la mémoire. Le gestionnaire de mémoire du Macintosh, intégré au système d'exploitation, peut prendre en charge le partage, l'allocation et la désallocation de la mémoire, ainsi que son compactage et la récupération de l'espace devenu vacant.

Pour cette raison, la version Mac a nécessité la refonte complète de la mémorisation des données, avec modification des bases de données, et un accroissement notable des procédures de vérification de validité des allocations.

Le résultat

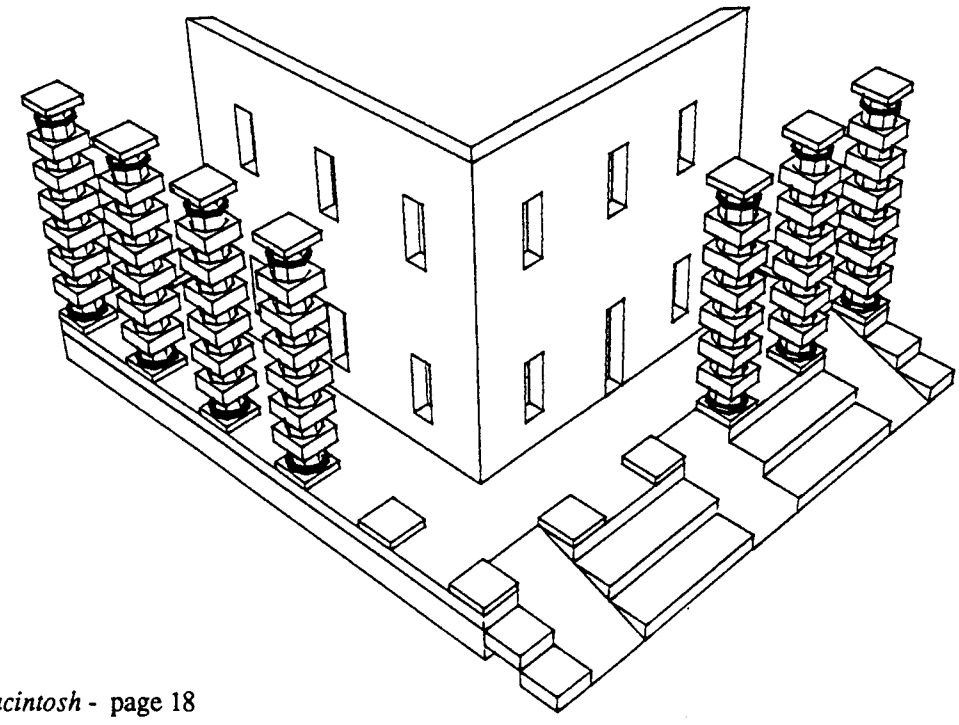
Le transfert direct des sources a été rapide - environ deux mois -. Le résultat en est une application hybride, ou plutôt une sous-application Macintosh. A part les éléments de l'interface qui permettent un fonctionnement en fenêtres et sous MultiFinder, aucune des recommandations de 'The Macintosh User Interface Guidelines' n'a été respectée. Le logiciel est opérationnel, mais non diffusable. Il ne peut servir ni à la pédagogie, ni à une utilisation en situation réelle. Il peut par contre être utilisé pour la modélisation d'objets à destination du modeleur du didacticiel *Perspectiva* sur l'Algorithme Perspectif.

Toutes les fonctions de la version BFM ont été réimplantées, sauf pour la partie élimination des parties cachées des perspectives produites. A ce niveau, c'est l'ensemble de l'algorithme, très demandeur en précision mathématique, qui est à remanier, les deux processeurs des machines traitant différemment les fonctions mathématiques sur les réels.

En conclusion, nous estimons que le portage pur et dur, s'il nous paraissait à priori inenvisageable, au vu des différences essentielles de philosophie entre le monde IBM et le monde Macintosh, est faisable à relativement peu de frais. Il n'en reste pas moins que le logiciel résultant n'est en aucun cas un logiciel pour Macintosh. Son comportement est par trop similaire à celui de la version BFM pour être mis entre les mains de qui n'en serait pas l'auteur.

Les tests qui ont été réalisés avec cette version sont néanmoins très prometteurs : peu de logiciels existants offrent de telles fonctionnalités, avec une telle souplesse et simplicité de mise en œuvre, avec des résultats de cette qualité. Une vraie version Macintosh, réécrite de bout en bout, mais basée sur les mêmes fonctionnalités, serait réellement un excellent logiciel d'apprentissage de la CAO.

Des déceptions persistent : nous nous attendions, bien que la version BFM soit déjà rapide - au vu des benchmarks réalisés -, à une amélioration des performances globales du logiciel. L'amélioration n'est que d'environ 20%. Il est évident que cette version n'utilise pas au mieux la boîte à outils du Macintosh, et qu'un gain supérieur nécessiterait une réécriture plus complète des algorithmes critiques.



IV. Bilan

Le travail sur le portage du logiciel Formes d'un environnement IBM à un environnement Macintosh a donc consisté en deux phases : développement d'une interface utilisateur spécifique et novatrice, préparant une réécriture complète du logiciel original, et transfert direct des sources de la version BFM, pour pré-test des fonctionnalités.

Faute de temps, l'intégration des fonctionnalités de Formes BFM dans le nouveau Formes Mac ne sera pas réalisée. Par contre, les recherches tant au niveau interface (fenêtrage, gestion des événements), qu'au niveau opérationnel (multi-tasking), et l'expérience acquise dans la programmation, sont réutilisées dans d'autres projets moins ambitieux mais plus novateurs, et dans l'enseignement de la méthodologie de l'informatique.

Il ressort néanmoins que, pour le peu de fonctionnalités intégrées à la version purement Macintosh, le confort d'utilisation apporté par les recherches menés en ergonomie du logiciel est incomparable à celui des logiciels concurrents.

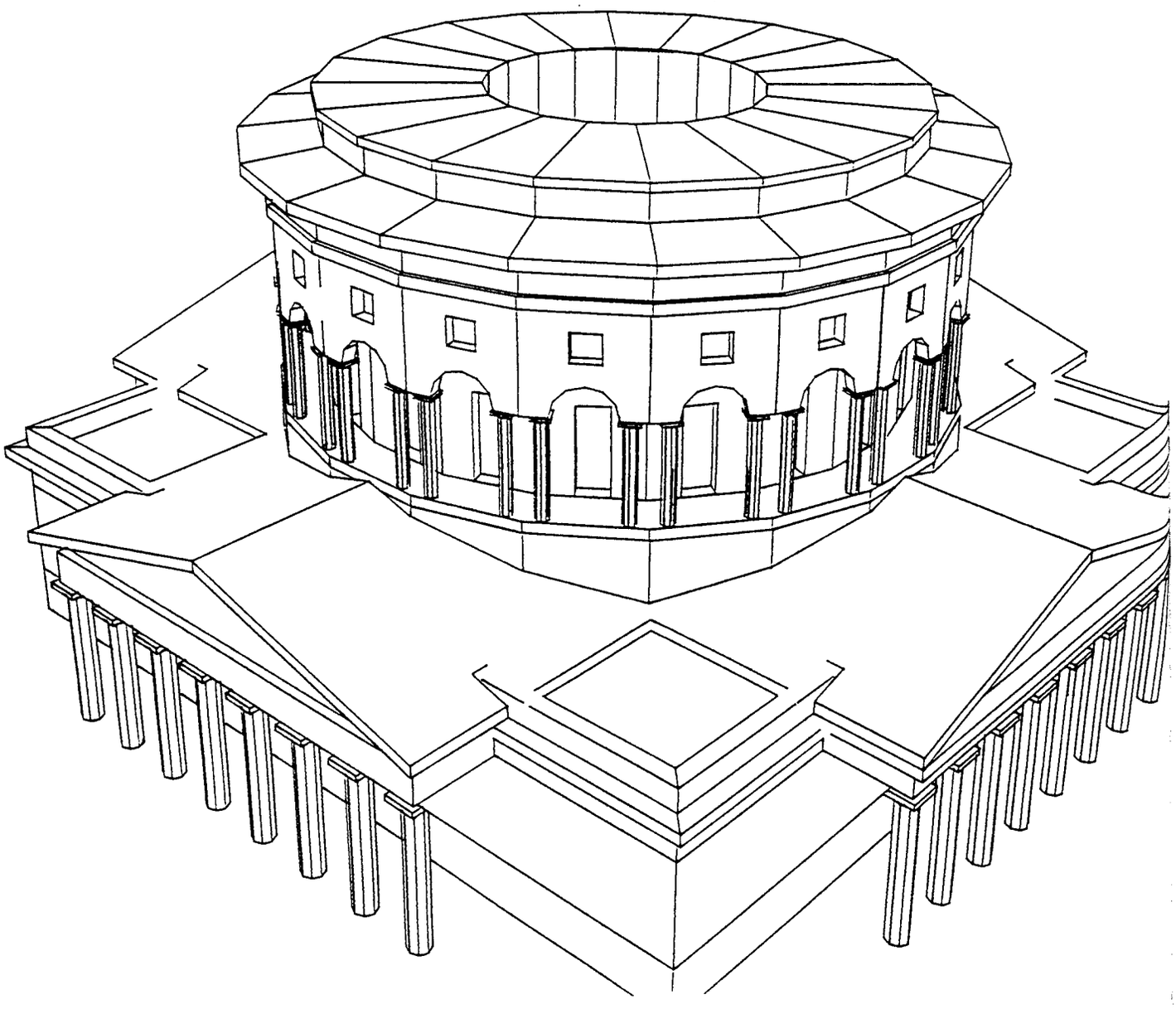
De la même façon, les études sur la gestion de tâches différées donnent à cette version de Formes un agrément d'usage à notre avis inégalé. Aucun logiciel, en effet, ne gère en tâche de fond les mises à jour de ses fenêtres, qui peuvent être de grosses consommatrices de temps machine, et sont souvent non interruptibles - si elles le sont, c'est définitivement. Formes, quant à lui, permet à l'utilisateur de continuer à travailler, et reprend dès que possible ses mises à jour.

Deux constats négatifs majeurs ressortent de ces deux ans d'études.

Le premier, que tout différenciant les deux environnements, de la philosophie aux outils, il est difficilement envisageable de réaliser un portage rapide qui puisse aboutir à un produit efficace et, surtout, répondant aux normes de qualité en usage. Lors de l'apparition du Macintosh, alors que l'offre en matière de logiciels de CAO était encore faible, le portage aurait pu aboutir à un produit diffusable, à faible coût. Aujourd'hui, les éditeurs de logiciels possèdent une grande expérience du développement sur Mac, et proposent à bas prix des logiciels de bon niveau. Il est trop tard pour que Formes, tout au moins dans ses fonctionnalités actuelles, puisse concurrencer qui que ce soit.

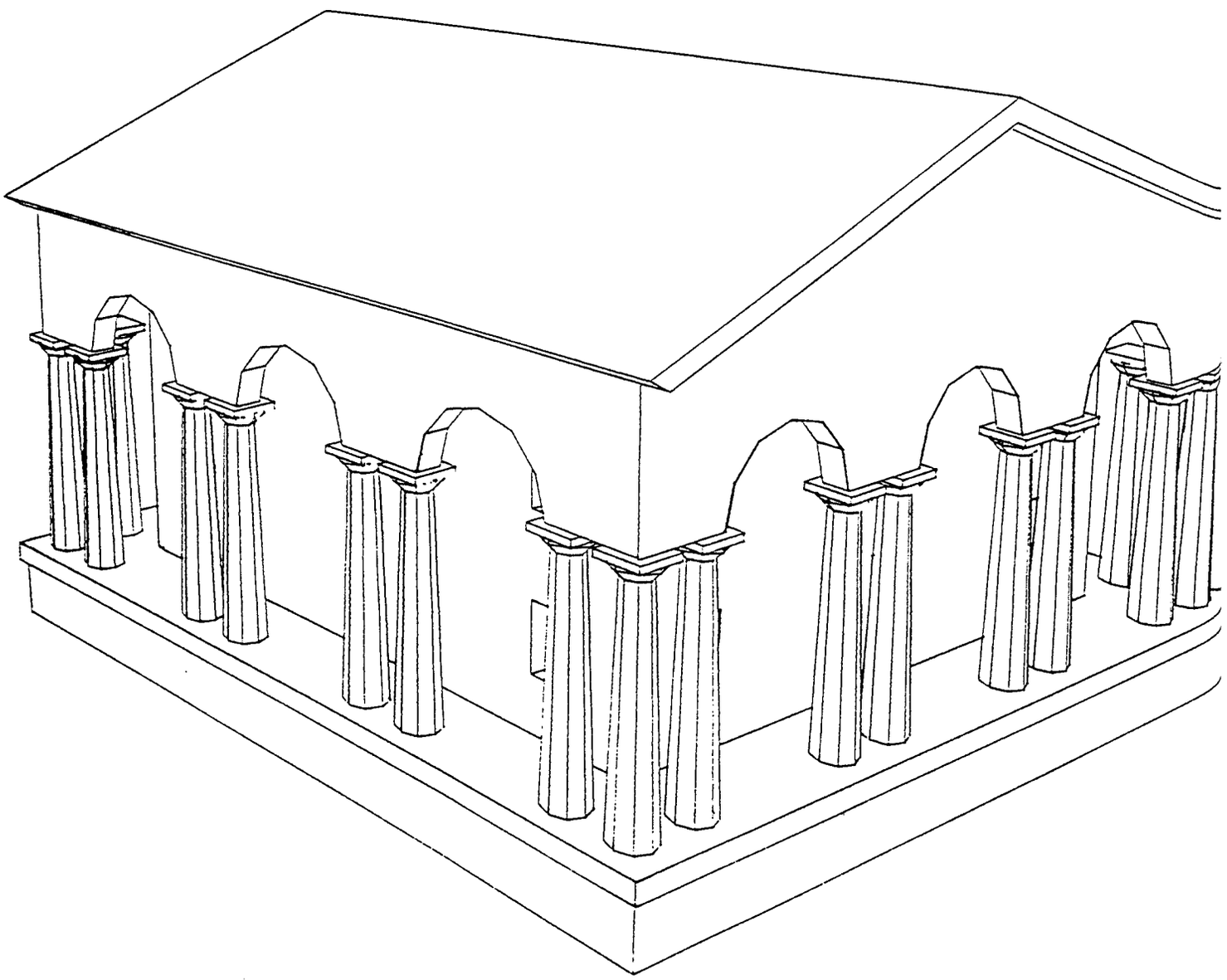
Le second, que la réalisation d'un logiciel de CAO, logiciel complexe, est une recherche hors d'échelle pour une petite équipe, découvrant un nouvel environnement. Ce constat est aggravé par le fait qu'il y a une distorsion évidente entre les moyens et l'expérience amassée soit par d'autres équipes de recherche, soit par les éditeurs, et ceux dont nous disposions.

L'expérience acquise lors de l'écriture des versions BFM et Mac sera bien évidemment utile lors des recherches ultérieures, aussi bien au niveau algorithmique pur (modélisation, algorithme perspectif, élimination des parties cachées), qu'au niveau de l'ergonomie et du design d'interfaces. Cette expérience a déjà été utile lors du développement du didacticiel "Perspectiva" écrit en collaboration avec l'Ecole d'Architecture de Grenoble, et dont le cœur est l'algorithme perspectif utilisé dans Formes.



Annexes

tomes



La base de données

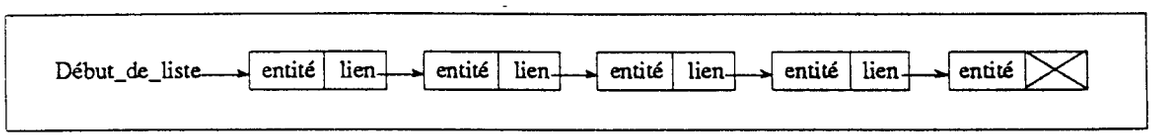
Plusieurs critères ont guidé le choix d'un modèle de données :

- une description surfacique de la géométrie des objets ;
- l'optimisation de la gestion de la mémoire, afin de profiter pleinement de la puissance du gestionnaire du système Mac OS (Memory Manager) ;
- la souplesse qui permettra à l'utilisateur d'intervenir à tous les niveaux de la hiérarchie de description : objets, facettes, arètes, points, attributs ;
- l'optimisation du parcours de la base de données, pour l'accélération des tâches d'affichage et de calcul.

Structuration des données

Nous avons choisi une structure de *listes chaînées*, identique pour tous les niveaux de hiérarchie. Cette structure permet d'une part de résoudre le problème de l'allocation mémoire, puisque le logiciel n'utilisera que la place demandée par les objets existants à un instant donné, et d'autre part de permettre l'écriture de procédures génériques de gestion de ces listes, applicable à n'importe quel type d'entité géométrique : le point, l'arète reliant deux sommets, la facette, l'objet, le groupe.

Une liste chaînée est une structure contenant un seul type d'entité, cette entité, outre ses données propres, contenant un *lien* avec l'entité suivante de la liste. Parcourir la liste revient à suivre ces liens, connaissant la première entité de la liste.



Liste chaînée

Un lien est un *pointeur* permettant de retrouver l'entité suivante (entité *pointée*).
Nous appellerons *début_de_liste* le pointeur pointant sur la première entité, *fin_de_liste* le pointeur vide, c'est à dire qui signale la fin de la liste.

L'algorithme de parcours de liste pourrait se décrire :

```
Parcours(liste):
      entité = début_de_liste
      Tant que Lien(entité) ≠ Fin_de_liste
            entité = Suivante(entité)
```

Une autre manière de décrire l'opération de parcours de la liste est d'utiliser une définition récursive :

Parcourir une liste depuis une entité, c'est :
Parcourir la liste des suivantes de cette entité

Les deux lignes sont équivalentes, et indépendantes de la liste sur laquelle on opère ; considérant une liste comme déterminée par sa première entité, nous décrirons l'opération 'Opérer sur une liste' comme 'Opérer sur la première entité de la liste'.

L'algorithme de parcours devient :

```
Parcours(entité):
      Parcours(Suivante(entité))
```

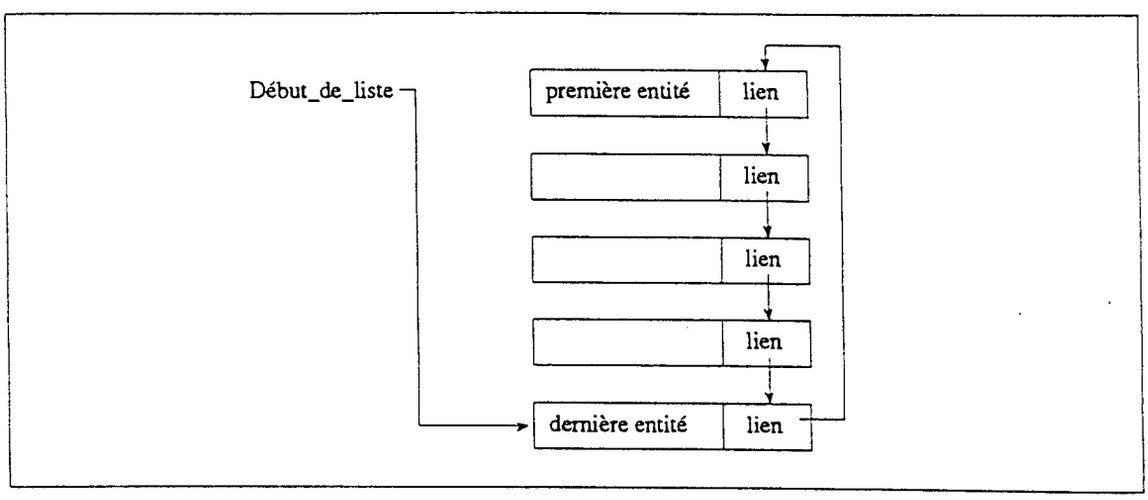
Le début du parcours est l'exécution de Parcours(début_de_liste). Le parcours devra s'arrêter lorsque nous atteindrons la fin de liste :

```
Parcours(entité):
      si Lien(entité) ≠ fin_de_liste
            Parcours(Suivante(entité))
```

Toute opération sur une liste chaînée prendra donc la forme:

```
Opération_sur(entité):
      si Lien(entité) ≠ fin_de_liste
            Opération_sur(Suivante(entité))
            Effectuer l'opération sur (entité)
```

Dans Formes, la structure de liste employée est une structure circulaire, le pointeur de tête de liste pointant sur la dernière entité, qui pointe à son tour sur la première entité. Cette structure présente le grand avantage de permettre une insertion immédiate d'un nouvel élément en fin de liste, sans parcours préalable. De plus, le parcours de la liste peut se faire à partir de n'importe quel endroit ; cet avantage autorise notamment, en cas de recherche d'éléments particuliers, de commencer le parcours par l'objet suivant celui précédemment repéré, permettant ainsi le repérage successif de toutes les entités correspondant aux critères de recherche (cas du repérage graphique avec superpositions, de l'identité de nom, etc).



Liste chaînée circulaire

Les entités de Formes

La structure de liste, malgré ses qualités, présente le défaut de nécessiter la mémorisation d'un lien entre entités ; la place requise par ce lien, en corrélation avec le nombre des enti-

tés, peut dans certains cas dégrader sensiblement le rapport entre souplesse de gestion mémoire et mémoire exigée ; ce sera le cas si la taille de l'information nécessaire à la description de l'entité elle-même est proche de la taille nécessaire au lien.
Par exemple, si une entité a une taille de 4 unités, et que le lien de chaînage demande aussi 4 unités, alors la place nécessaire au stockage de toute l'information est le double de la place nécessaire à la description des entités.

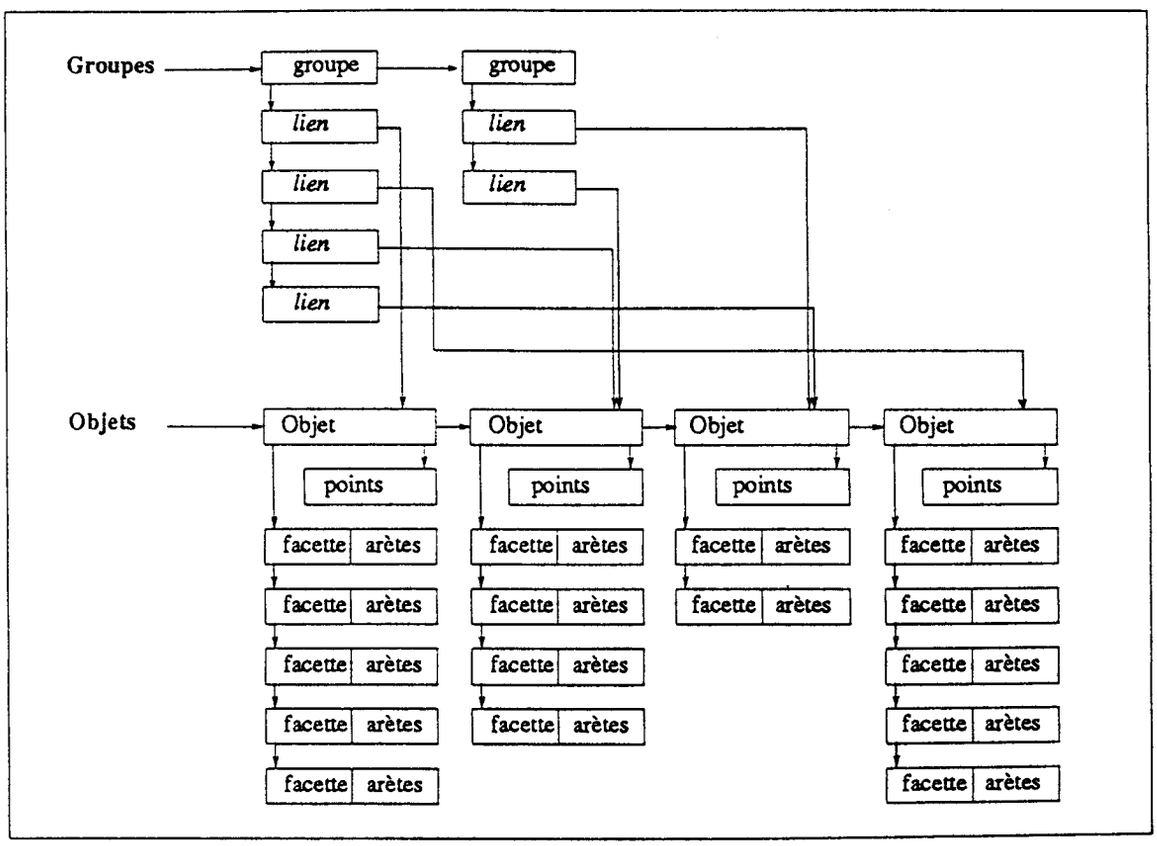
Le choix de la structure de données de Formes reflète ces exigences : simplicité de gestion d'une structure dynamique, minimisation de la place nécessaire à l'information. Les entités de plus bas niveau, points et arètes, seront gérées par blocs alloués dynamiquement, à l'intérieur de la description des entités de plus haut niveau, gérées par listes.

Les entités de base :

- un point est la description géométrique d'un sommet d'un objet
- une arète est un lien entre deux de ces sommets
- une facette est une suite ordonnée d'arètes
- un objet est une liste de facettes
- un groupe est une liste d'objets

Leur type de gestion :

- les sommets sont gérés par blocs à l'intérieur d'un bloc lié à l'objet
- les arètes sont gérées par blocs dans chaque facette
- les facettes sont liées en liste à un objet
- les objets et les groupes sont des listes dynamiques



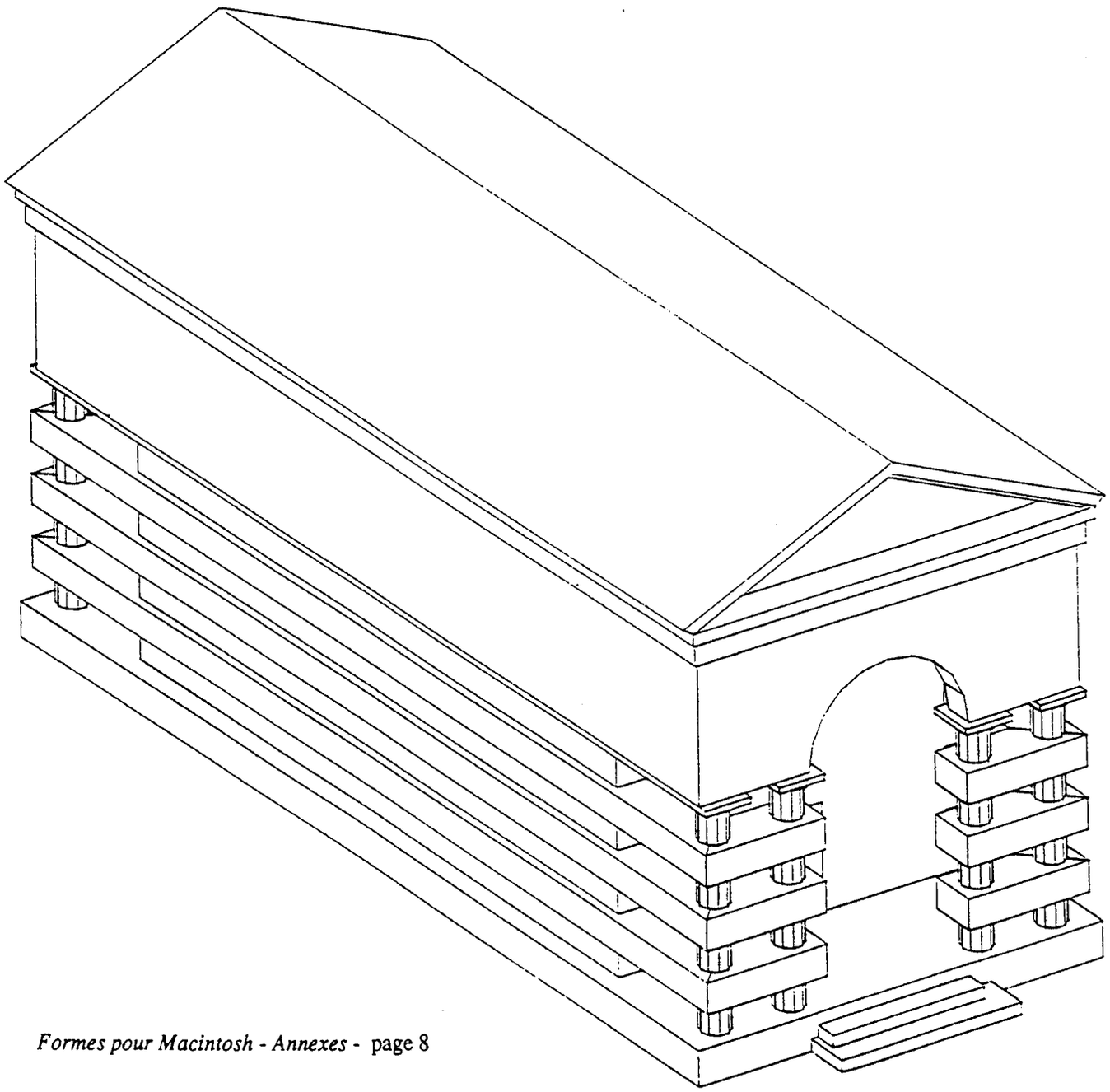
Structuration des données objets

Lors de la destruction d'une arète ou d'un point d'un objet, les indices et tableaux de points doivent être compactés. Cette tâche, pour ne pas alourdir la gestion des objets, a été dévolue à une procédure de "*garbage collecting*", qui récupère dès qu'elle le peut, ou

dès que la place mémoire disponible est trop faible, les objets inutilisés et les libère pour une utilisation, ultérieure.

Une base de donnée de ce type est très souple, puisqu'elle permet sans grandes modifications de gérer d'autres types d'objets :

- une hiérarchie à niveaux multiples, où un groupe d'objets peut comporter à son tour un groupe ;
- des calques, qui affichent des objets indépendants ;
- plusieurs fichiers d'objets en simultané, avec transfert des objets de l'un à l'autre.



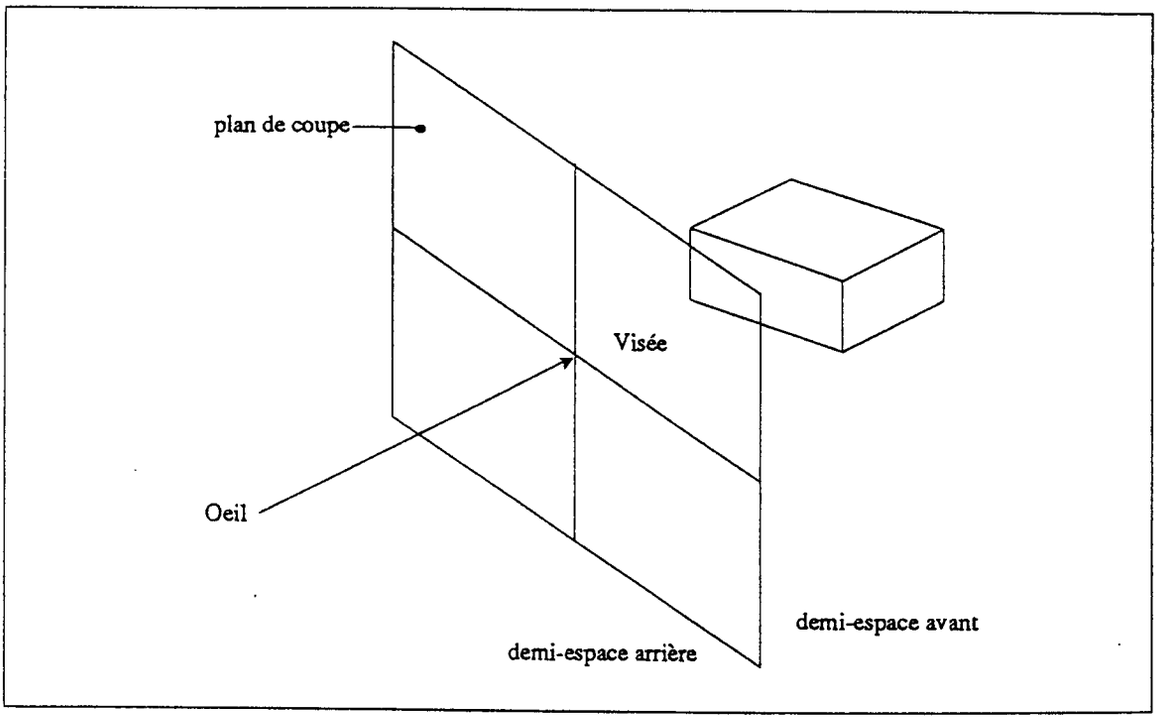
L'algorithme perspectif

L'algorithme de projection employé pour les représentations des objets manipulés a été choisi en fonction de critères de simplicité et de généralité de mise en œuvre.

C'est un algorithme de calcul matriciel, chaque fenêtre de représentation possédant sa propre matrice de projection 4 x 4. Ces matrices intègrent les données nécessaires aux projections perspectives ou parallèles, orthographiques, axonométriques ou obliques.

L'algorithme de "*clipping*" mis en œuvre consiste à tronquer lors de l'affichage les objets possédant un point au moins derrière l'oeil de l'observateur, divisant ainsi l'espace des objets en deux demi-espaces. L'utilisateur peut visualiser les objets sur 180°, donc en dehors du champ de vision défini par l'angle d'ouverture, par simple déplacement ou agrandissement de la fenêtre de représentation.

Une projection en coupe peut être obtenue par simple déplacement du plan définissant le demi-espace visible à la position du point de visée. En modulant sa position, l'utilisateur positionne la profondeur du plan de coupe par rapport à l'oeil.



L'espace de projection

L'utilisateur a accès à :

- position de l'oeil
- position du point de visée
- angle d'ouverture
- angle de roulis
- projection perspective / parallèle, avec / sans coupe.

L'algorithme utilisé est donc extrêmement général, et permet d'obtenir l'essentiel des projections couramment utilisées [1]. Pour une description de ses fondements, on se reportera au didacticiel *Perspectiva*, développé conjointement avec le département "Métiers de l'Histoire de l'Architecture" (EA Grenoble), dont le noyau provient de la version Mac de Formes.

L'essentiel de l'algorithme d'élimination des parties cachées est repris de la version BFM du logiciel. Il est basé sur celui décrit par Montanari-Galimberti, qui propose une méthode orientée traceur, seule apte à conserver une précision suffisante lors de sorties papier.

Son principe est, une fois la projection plane des objets calculée, de déterminer les parties visibles ou invisibles des arêtes de ces objets. Son cœur est un calcul d'intersections entre projections de facettes.

Nous l'avons étendu de façon à gérer aussi bien les facettes concaves que des facettes convexes, et traiter les percements d'objets. Les optimisations apportées à l'algorithme par rapport à sa première description, tant pour le précision des calculs que pour le gain de temps, offrent une grande qualité de résultats, absente de trop nombreux logiciels équivalents.

1. Il est évident qu'un algorithme aussi général n'est pas optimum pour certaines situations : par exemple, dans le cas d'une orthographie, où aucune transformation due à la perspective n'est nécessaire, une simple projection directe aurait suffit. L'algorithme possède néanmoins l'avantage de ne pas fixer de statut particulier ("plan", "élévation", "coupe", ...) à la représentation.

La gestion des tâches

Le processus de gestion des tâches mis en œuvre dans la version Mac de Formes a été introduit afin d'améliorer à la fois les performances du logiciel et le confort de l'utilisateur.

A aucun moment, le logiciel ne doit bloquer celui-ci. Toutes les tâches critiques, gourmandes en temps machine, doivent être le plus transparentes possibles, et, autant que faire se peut, se dérouler en arrière-plan.

Comme types de tâches critiques, nous pouvons citer:

- la mise à jour des représentations, après manipulation des objets ;
- la mise à jour des fenêtres, après agrandissement, déplacement, permutation ;
- les modifications des paramètres des projections ;
- ...

Toutes ces tâches doivent pouvoir être abandonnées si elles sont devenues inutiles, et exécutées si l'utilisateur ne commence pas une tâche plus prioritaire. Ainsi ne seront exécutées que les tâches pertinentes à un instant donné de la session de travail.

Par exemple :

- si l'utilisateur modifie les paramètres de la perspective, un nouveau dessin est calculé, mais, si intervient un événement qui a une influence sur ce dessin, il est interrompu, et recommencé en tenant compte des nouvelles données.
- en cas de déplacement d'une fenêtre, sa mise à jour - ou celle des fenêtres qui deviennent visibles - ne s'effectuera que lorsque l'utilisateur ne fera rien.

Interruptions des tâches

La difficulté de l'écriture de ce type de processus est, outre de déterminer si une tâche donnée peut devenir critique, d'écrire des procédures interruptibles mais aussi réentrantes, car l'événement qui les interrompt peut fort bien les rappeler : si l'utilisateur déplace une fenêtre et provoque sa remise à jour, celle-ci peut être interrompue par un nouveau déplacement, qui à son tour provoque une nouvelle mise à jour...

Le cœur du processus est une procédure *TreatEvents* [1] chargée de recevoir les événements, et, suivant leur type, de dérouter le programme sur la procédure charger du traitement :

```
TreatEvents
      répéter
            si un événement est en attente
                  suivant son type
                        appeler la procédure de gestion de ce type
      jusqu'à la fin de la session
```

Prenons en exemple de procédure de gestion d'un type d'événement particulier, la procédure *Redraw*.

1. L'anglais présente l'avantage de la concision...

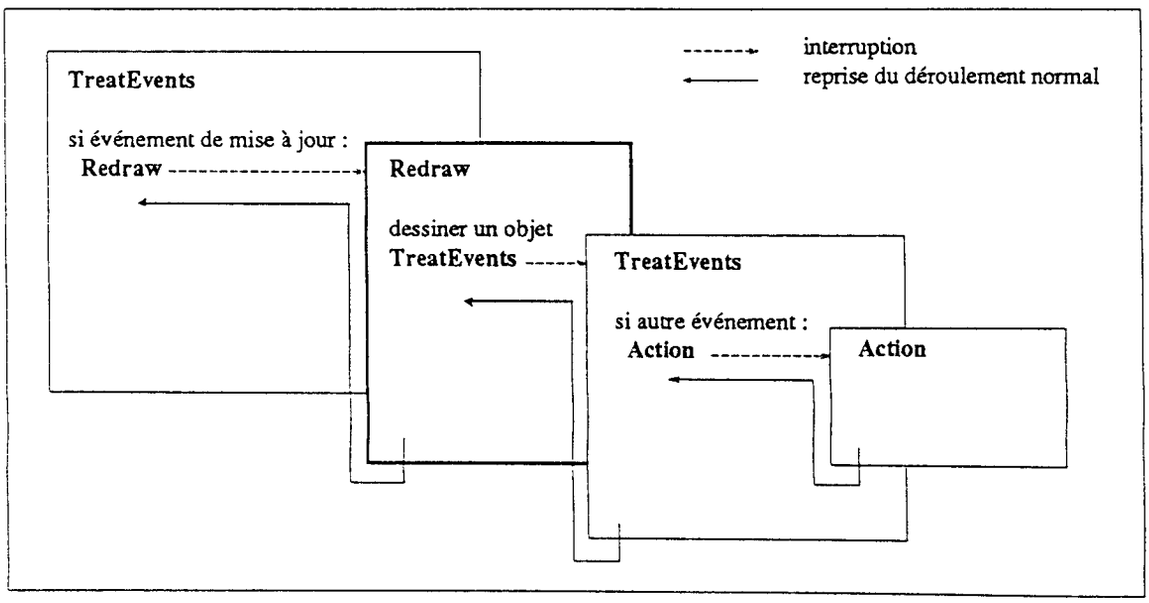
Cette procédure est chargée de tracer les projections des objets, soit à la suite d'une modification de ces objets, soit à la suite d'une mise à jour des fenêtres (déplacement, zoom, permutation de plans, ...), soit à la suite d'une modification des paramètres de projection. Elle reçoit en paramètre la fenêtre à retracer, et dessine pour cette fenêtre tous les objets visibles qu'elle contient, suivant le matrice de projection associée.

```
Redraw (aWindow)
      répéter pour chaque objet de aWindow
            calculer sa projection
            le dessiner
```

Dans cette version, la procédure Redraw n'est pas interruptible. Si elle est appelée par TreatEvents par un événement du type <modification des paramètres de projection>, provoqué par l'utilisateur, alors celui-ci devra attendre la fin du tracé pour enclencher une nouvelle modification.
Pour permettre l'interruption du tracé, nous scruterons à intervalles réguliers la liste des événements en attente, par exemple après le tracé de chaque objet élémentaire. Nous appellerons donc dans Redraw la procédure TreatEvents :

```
Redraw (aWindow)
      répéter pour chaque objet de aWindow
            calculer sa projection
            le dessiner
            appeler TreatEvents
```

Lorsque TreatEvents aura traité l'événement en attente, s'il existe, l'exécution du programme se continuera dans Redraw, qui reprendra le processus interrompu [1].

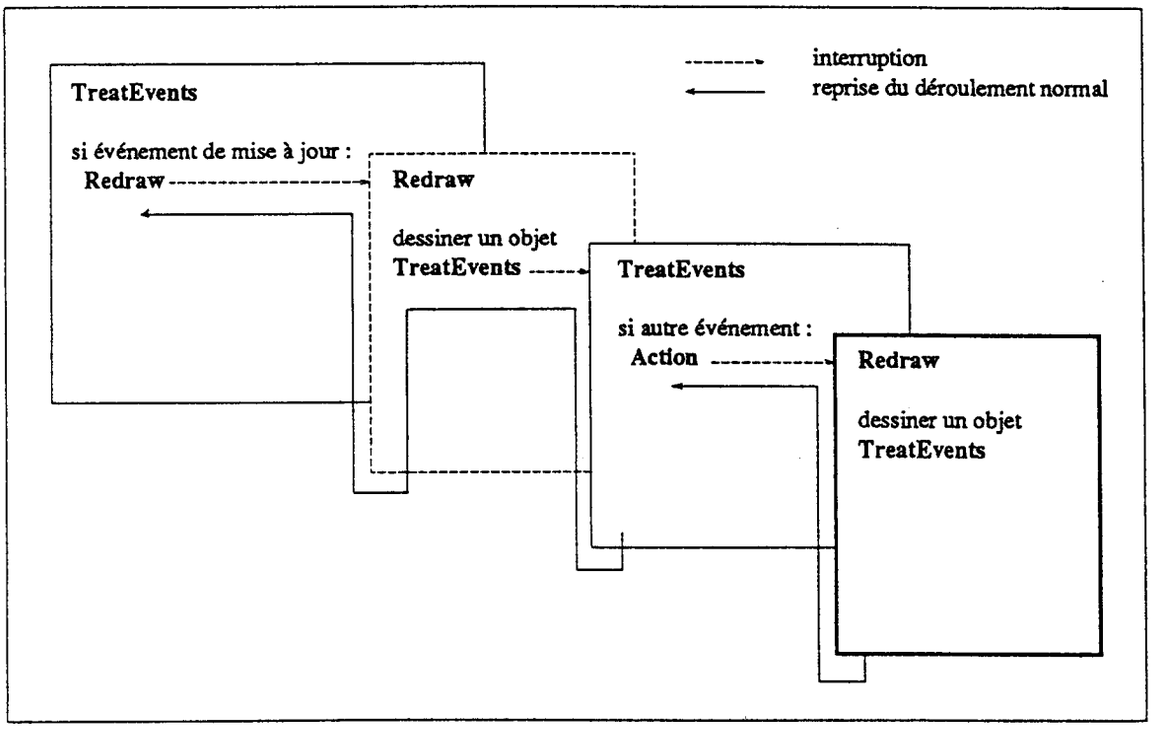


Processus d'interruption temporaire de tâche

Imaginons maintenant que l'action appelée par TreatEvents lors de l'interruption du dessin soit la procédure Redraw elle-même (par exemple, suite à une nouvelle modification des paramètres de projection). Dans ce cas, il est évident que la première exécution de Redraw - moins "actuelle" - ne doit pas reprendre une fois le nouvel appel

1. A première vue, ce procédé augmente le temps total mis par Redraw pour redessiner l'ensemble des objets. Mais pour l'utilisateur, le fait de pouvoir continuer à agir pendant l'exécution du tracé lui donne l'impression de ne jamais attendre. Son confort en est grandement amélioré, puisque ses actions sont prioritaires sur les tâches qui ne nécessitent pas son intervention.

terminé, faute de quoi les deux dessins - dont un non valide - seraient superposés. Chaque appel à un dessin complet de la fenêtre, une fois terminé, doit signaler que les appels précédents ne doivent pas reprendre, et doivent être définitivement interrompus.



Processus d'interruption définitive de tâche

```
Redraw (aWindow)
      répéter pour chaque objet de aWindow
            calculer sa projection
            le dessiner
            appeler TreatEvents
            si Redraw a été rappelée
                  sortir
```

Deux types de mise à jour coexistent : la mise à jour de la représentation, suite à une modification des paramètres ou des données, ou la mise à jour des fenêtres elles-même, suite à un déplacement ou agrandissement [1]. Ce dernier type de mise à jour fut plus délicat à mettre en œuvre, de par le fait que la gestion des fenêtres est dévolue au "*Window Manager*" de la boîte à outils du Macintosh. Les routines qui permettent cette gestion, passage obligatoire pour la mise à jour, ne sont ni réentrantes, ni interruptibles. Il a donc fallu réécrire les procédures de bas niveau.

Processus multi-tâches

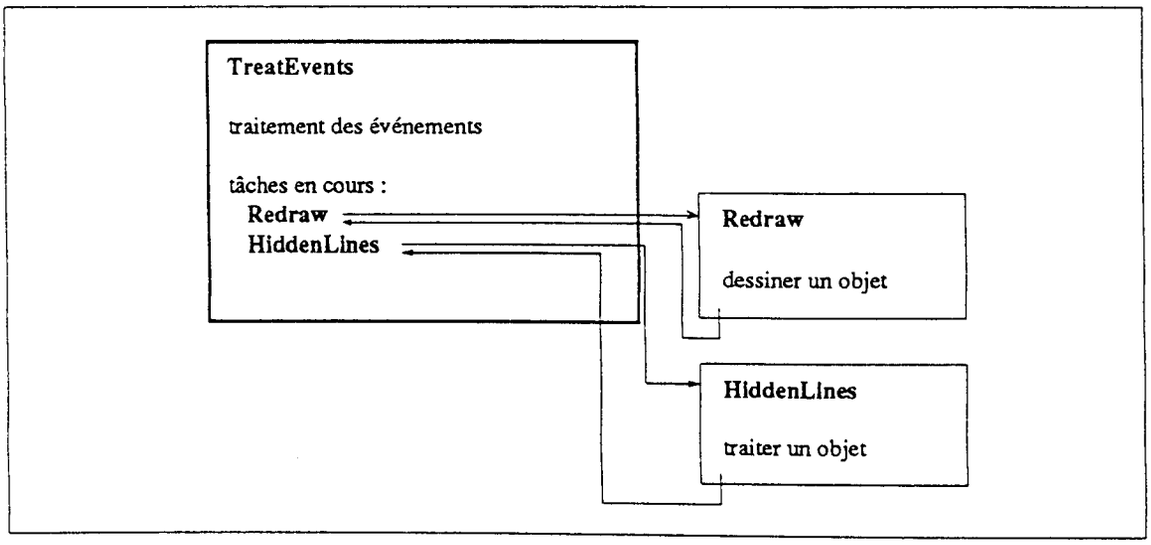
Dans la partie précédente, nous avons parlé de la nécessité, pour un logiciel demandant une part importante de calcul (en temps machine), de permettre l'interruption définitive ou temporaire de la tâche en cours.

1. En interface multi-fenêtres, l'utilisateur peut permuter les fenêtres (les faire passer d'avant-plan en arrière-plan, et vice-versa), les agrandir, les déplacer. Ces actions ont pour effet de faire apparaître des zones de fenêtres jusqu'alors masquées. Ces fenêtres ont alors besoin d'une "mise à jour".

Pour améliorer encore l'attrait et le confort du logiciel, il faut que ce type de fonctionnement - non bloquant - s'applique non seulement à la dernière tâche demandée, mais aussi que le logiciel soit capable de lancer plusieurs processus simultanément. Par exemple, que soit calculée dans une fenêtre une vue en faces cachées, pendant que l'utilisateur construit des objets dans une autre fenêtre.

Dans ce modèle de comportement, il faut que l'application fonctionne en mode multi-tâches. Sur une machine mono-processeur, la seule façon d'opérer est d'allouer à chaque tâche un temps machine fonction de son niveau de priorité, et du pourcentage de temps qu'elle désire.

Chaque routine critique sera appelée à intervalles réguliers par la routine de traitement des événements, chargée de ce partage du temps d'occupation. Celle-ci mémorise la liste des tâches en cours d'exécution, supprime celles arrivées à terme, interrompt celles dont le déroulement doit être stoppé par décision de l'utilisateur.
Chacune des routines critiques doit être écrite soit de façon à être interruptible - et rappeler la procédure de traitement centrale de façon régulière -, soit de façon modulaire - à chaque appel, elle ne traite qu'une partie des données, mémorise son contexte et rend la main à la procédure centrale.



Processus multi-tâches

Cette gestion des processus, si elle a pour inconvénient de les ralentir proportionnellement au nombre de tâches en cours, présente l'avantage de ne jamais contraindre l'utilisateur à attendre la fin d'une tâche pour intervenir sur le logiciel. A tout instant, il a le loisir de vérifier le déroulement de chacune des tâches distinctes (par exemple dans le cas d'un calcul d'image), et de les interrompre si nécessaire, ce qui n'est pas possible dans le cas d'un fonctionnement standard de process différé ou de traitement.

Comme pour le fonctionnement décrit plus haut, chaque tâche de fond doit être interruptible, si les données qu'elle va traiter deviennent incohérentes soit avec celles déjà traitées, soit avec le contexte : cas d'une modification ou destruction d'un objet, cas d'une modification des paramètres de projection d'une vue en faces cachées non terminée. L'utilisateur doit quand même pouvoir décider si les modifications intervenues depuis le commencement du processus doivent nécessiter son interruption.

Formes Mac : fonctionnalités

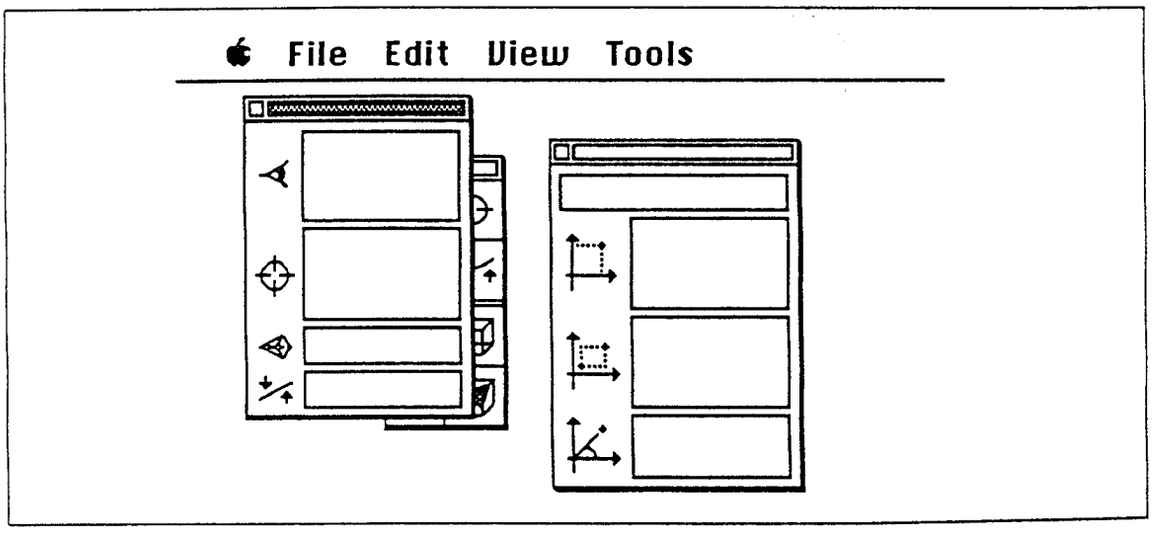
Avertissement : la version de Formes Mac qui accompagne ce rapport est majoritairement écrite en anglais, pour des raisons de concision et de simplicité d'écriture et de test (la totalité de la documentation technique disponible sur Macintosh est en anglais)

La recherche sur une version Macintosh "pure" de Formes a porté, comme nous l'avons vu, sur le design et l'ergonomie de l'interface utilisateur. La plupart des fonctions permettant la gestion des fenêtres décrites dans ce rapport ont été implantées.

La gestion des objets, même si elle s'appuie sur la structure et les routines décrites dans l'annexe "Base de données", est limitée au rappel de fichiers, à des fins de test de l'interface. Ces fichiers objets peuvent être créés au format Texte dans un logiciel de traitement de texte ou dans un tableur. Sur la disquette de démonstration est inclus un fichier "Init" comportant quelques objets simples générés sur Excel.

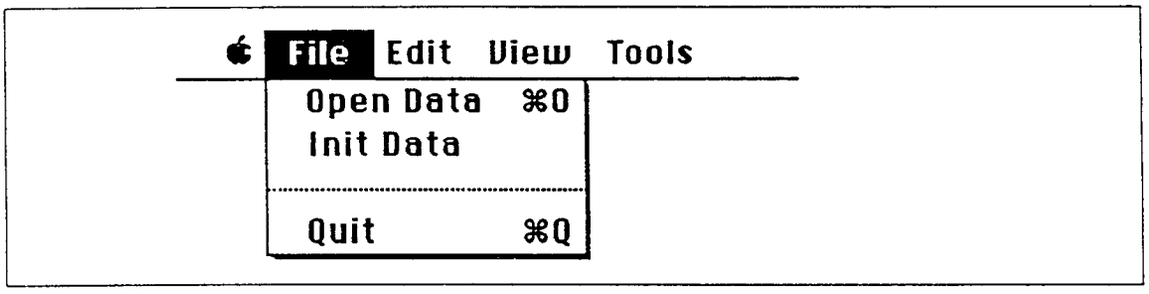
Les lignes ci-après décrivent les fonctionnalités de gestion des fenêtres et des projections actuellement implantés dans cette version de Formes.

Les menus



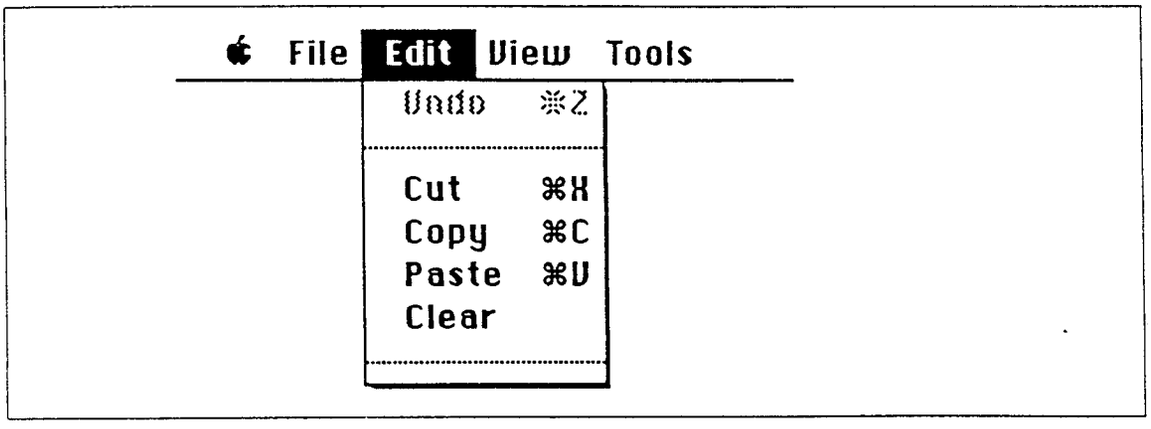
Les menus de Formes

- Le menu *File* permet de gérer les données, et de quitter le logiciel ;
- Le menu *Edit* permet, comme dans toutes les applications Macintosh, de transférer les données textuelles ou graphiques sélectionnées d'une zone à une autre, ou d'une application à l'autre ;
- Le menu *View* gère les fenêtres de document ;
- Le menu *Tools* gère les fenêtres de palette.



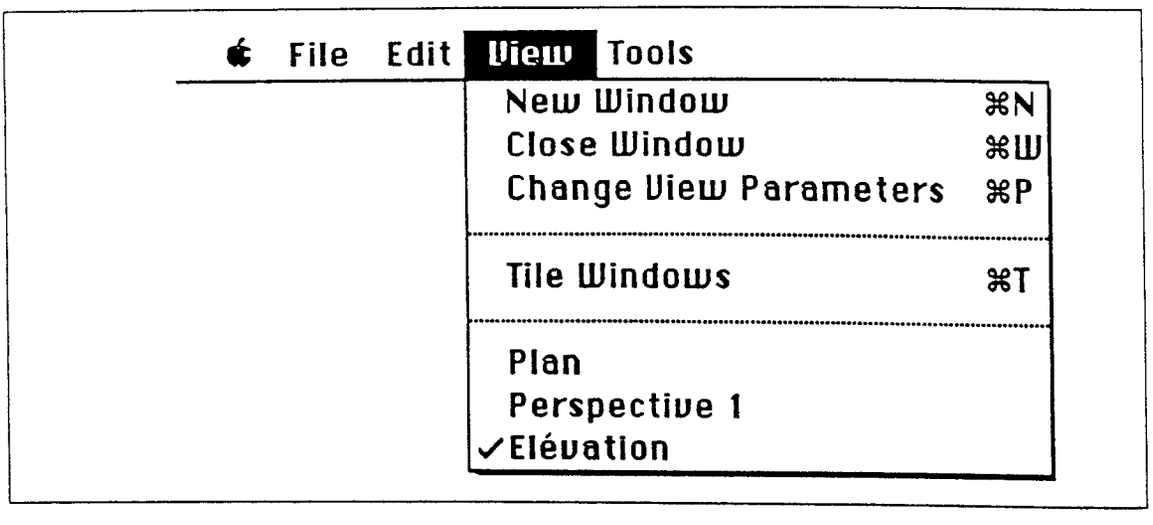
Le menu File

- *Open Data* permet de charger des données en provenance d'un fichier au format Texte, tel que le fichier Init. Aucun contrôle n'est effectué sur la validité des données. Les objets du fichier chargé sont rajoutés à ceux déjà existants.
- *Init Data* réinitialise les données de Formes.
- *Quit* permet de quitter le programme, et de revenir au Finder.



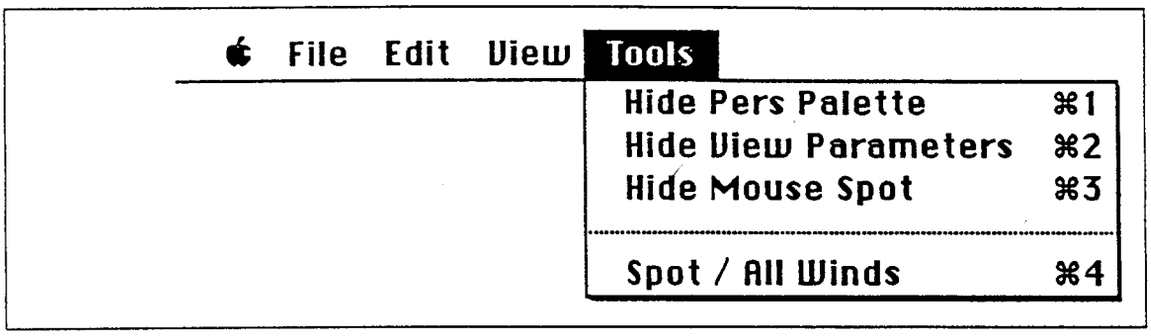
Le menu Edit

- *Undo*, qui dans les applications courantes Macintosh permet d'annuler les dernières opérations effectuées, n'est pas implémenté.
- *Cut*, *Copy*, *Paste* et *Clear* sont les articles standard Macintosh (Couper, Copier, Coller et Effacer). Dans cette version, ils ne permettent que de transférer des données numériques d'une zone des palettes à une autre zone (par exemple les coordonnées de l'oeil dans les coordonnées du point de visée). Dans les versions ultérieures, ce menu permettra le transfert de données textuelles ou graphiques entre applications.



Le menu View

- *New Window* permet l'ouverture d'une nouvelle fenêtre de représentation. Lors de la création d'une fenêtre, Formes demande le nom de la fenêtre, qui permettra de la repérer dans la liste. Si aucun nom n'est donné, alors les fenêtres sont numérotées. A l'ouverture, les paramètres de la représentation (position d'oeil et de point de visée, angle d'ouverture et roulis) sont donnés par défaut.
- *Close Window* supprime la fenêtre de premier plan de la liste des fenêtres de représentation. Cette commande a un effet différent d'un clic dans la case de fermeture de la fenêtre, qui ne fait que la masquer temporairement.
- *Change View Parameters* ouvre la fenêtre des paramètres de représentation, ou l'affiche en premier plan si elle existe déjà. Cette fenêtre de palette donne accès à la position de l'oeil, du point de visée, et à l'angle d'ouverture.
- *Tile Windows* range les fenêtres à l'écran, actuellement sans tenir compte de leur taille.
- A la fin du menu se trouve la liste des fenêtres de représentation, visibles ou non. C'est grâce à ce menu que l'on fait réapparaître une fenêtre précédemment masquée (par clic dans sa case de fermeture), ou que l'on sélectionne la fenêtre de premier plan. Ce même menu est affiché sous la souris si le clic se produit à l'extérieur d'une fenêtre, ou avec la touche Commande enfoncée.



Le menu Tools

- Les trois premiers articles gèrent les fenêtres de palette, et permettent de les masquer ou de les montrer.
- Par défaut, lorsque la souris est déplacée, les coordonnées affichées dans la fenêtre curseur sont relatives à la fenêtre survolée. *Spot / All Winds* et *Spot / A Wind* permet de commuter en coordonnées relatives à la fenêtre de premier plan. Cette facilité a été introduite pour autoriser la saisie simultanée dans plusieurs fenêtres.

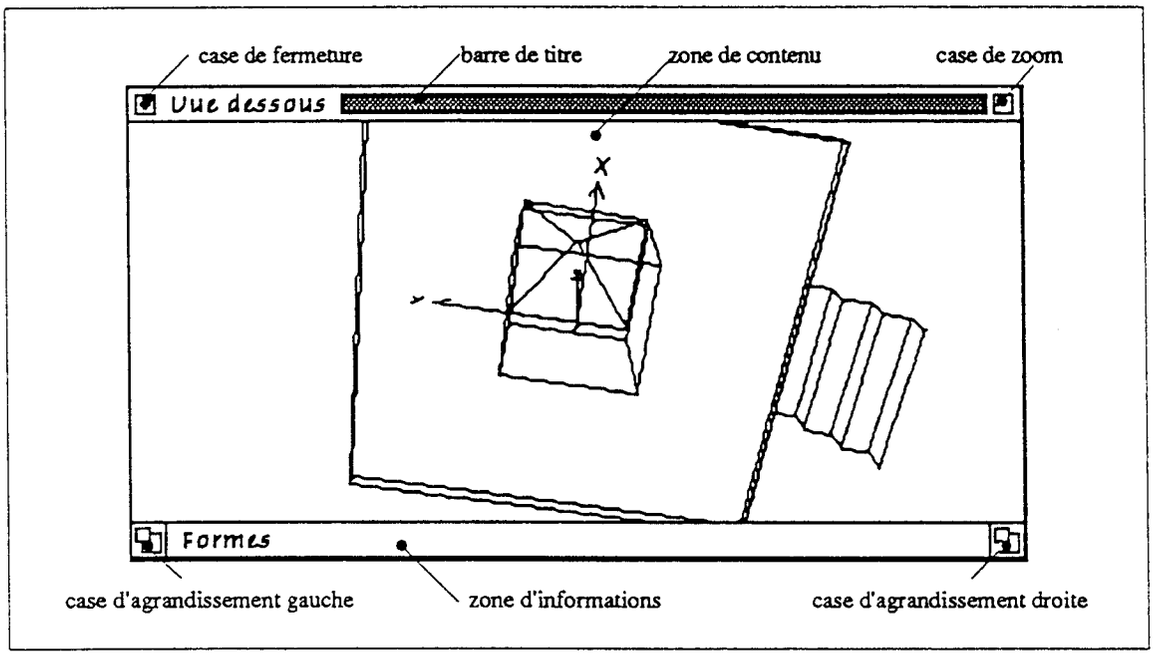
Les fenêtres de document

Comme nous l'avons vu précédemment, chacune des fenêtres de document est une représentation particulière des mêmes données. Actuellement, Formes ne gère qu'un fichier de données à la fois. Chaque fenêtre possède ses propres paramètres de projection.

Pour simplifier la manipulation de fenêtres multiples, toutes les opérations possibles sur une fenêtre de premier plan, de façon standard, ont été étendues aux fenêtres d'arrière plan. Ainsi, sans modifier l'ordre des fenêtres, l'utilisateur pourra-t-il agrandir, zoomer, déplacer, travailler sur une fenêtre d'arrière-plan . Pour manipuler une telle fenêtre sans l'activer, il suffit de maintenir la touche *Commande* enfoncée pendant l'action.

La touche *Option*, quant à elle, permet de modifier, lors d'une manipulation de fenêtre, la taille ou la zone visible de son contenu. Par exemple, l'agrandissement d'une fenêtre avec cette touche enfoncée agrandit en même temps la représentation qu'elle contient.

De façon générale, la touche Commande concerne les fenêtres d'arrière-plan, et la touche Option concerne la représentation contenue dans la fenêtre. Les deux touches sont combinables.



Une fenêtre de document

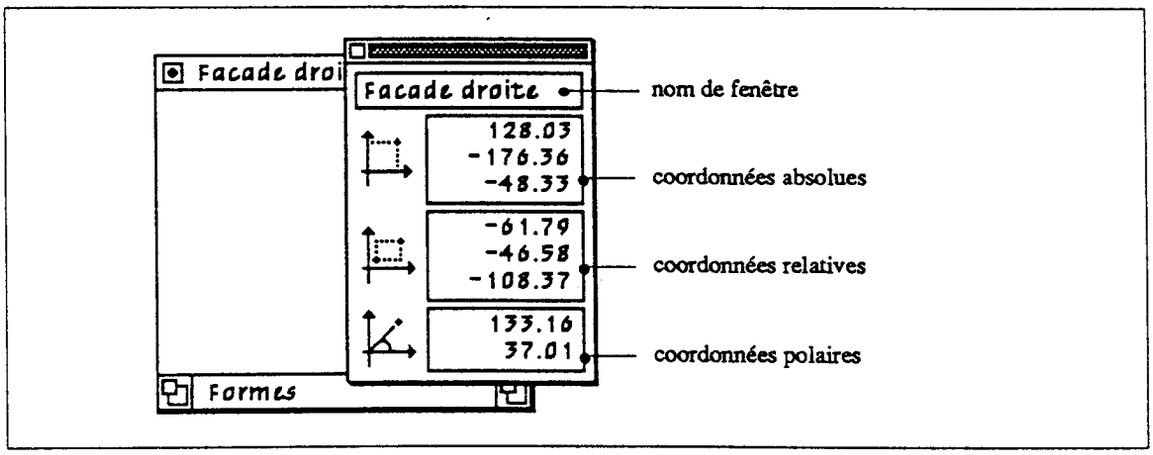
Ci-dessous sont listées les fonctions de chaque zone de fenêtre, active ou non, avec ou sans combinaison de la touche Option.

- La *barre de titre* permet le déplacement de la fenêtre. Ce titre peut être modifié dans la palette du curseur (cf chapitre suivant).
- La *case de fermeture* masque la fenêtre (à la différence de l'article *Close Window* du menu *View* qui la ferme définitivement). Elle peut être rendue visible en la sélectionnant dans le menu View.
- La *case de zoom* permute entre taille actuelle et taille plein-écran ; avec la touche Option, la représentation est agrandie ou rétrécie en proportion.
- Les *cases d'agrandissement* étendent ou rétrécissent la taille de la fenêtre ; avec la touche Option, la représentation est elle-même redimensionnée.
- La *zone d'information* est destinée à recevoir des messages textuels indiquant le statut courant du logiciel (opération longue, attente d'une donnée, ...) ; actuellement, elle donne le nom d'un objet de la fenêtre si l'on clique sur une de ses arètes.
- Un clic souris dans la *zone de contenu* avec la touche Option enfoncée permet de déplacer la représentation à l'intérieur de la fenêtre ; un double clic, toujours avec la touche Option, recadre la représentation.
  Avec la touche Commande, on obtient la liste des fenêtres de représentation, ce qui permet de sélectionner la fenêtre de premier plan sans se déplacer à la barre de menus. Le même résultat est obtenu en cliquant à l'extérieur de toute fenêtre (dans la région grisée du bureau).

Les fenêtres de palette

Dans la version actuelle, trois fenêtres de palette coexistent ; elles permettent d'accèder par un simple clic souris aux paramètres de la projection, et de visualiser le mode de projection et la position du curseur dans l'espace des objets. Certaines fonctions de ces palettes ne sont pas implantées. Dans des versions ultérieures seront rajoutées des palettes d'accès aux outils de création et de transformation d'objets. Le menu *Tools* permet de masquer ou d'afficher ces palettes, et d'en modifier le comportement.

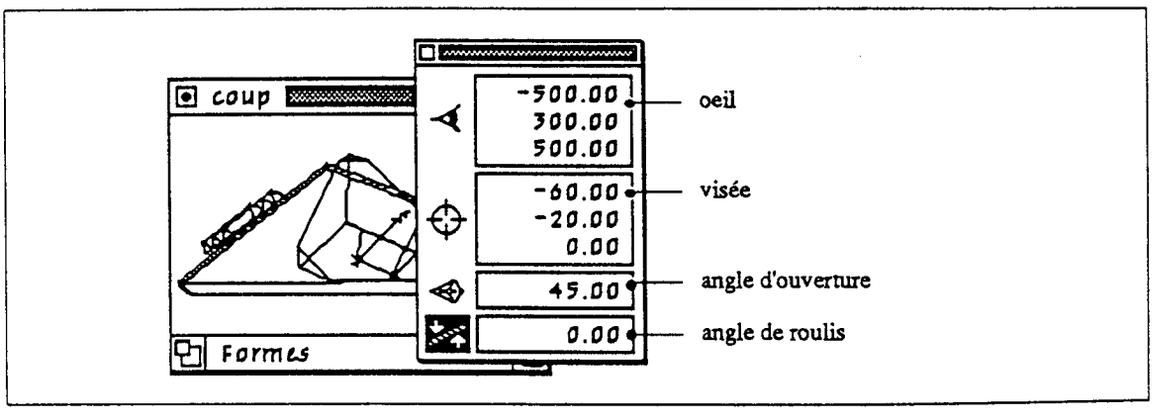
Certaines des informations affichées dans ces palettes sont redondantes, pour simplifier leur manipulation, et permettre de libérer l'écran en gardant l'accès aux fonctions essentielles. Par exemple, les paramètres de la projection sont accessibles numériquement dans la palette *Paramètres de la Vue* ; ils seront accessibles graphiquement à partir de la palette *Perspective*, qui visualise le mode de projection courant.



Palette du curseur

La palette de curseur affiche les coordonnées de la souris par rapport à l'espace des objets. Les coordonnées sont relatives à la fenêtre de représentation survolée, ou à la fenêtre de premier plan, si l'on a sélectionné l'article *Spot / A Wind* dans le menu *Tools*. Dans cette fenêtre de palette, on peut modifier le nom de la fenêtre active, et modifier directement les coordonnées du curseur, permettant une saisie numérique de coordonnées. Cette saisie s'opère en cliquant dans la palette avec la touche Option enfoncée, et se termine de la même manière ; cette façon de procéder permet de "masquer" la palette au curseur lors de ses déplacements.

Actuellement, le curseur se déplace dans un plan parallèle au plan de projection. La troisième coordonnée - la profondeur - est donnée par la dernière position relevée dans une fenêtre dont le plan de projection ne serait pas parallèle à celui de la fenêtre active. Par exemple, si deux fenêtres coexistent, l'une étant un "plan" - Z fixe -, l'autre une "élévation" - Y fixe -, la profondeur Z du curseur dans la fenêtre de plan est la dernière position en Z relevée dans la fenêtre d'élévation.

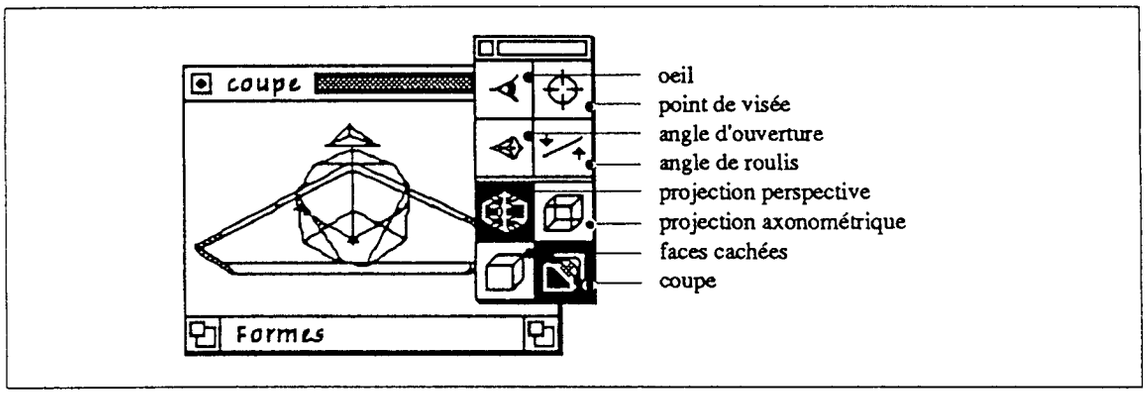


Palette de paramètres de vue

La palette de paramètres de vue permet l'affectation des différents paramètres de la projection de la fenêtre active (celle dont le nom est affiché dans la palette de curseur).

Ces paramètres sont : la position de l'oeil, la position du point de visée, définissant la direction du regard et l'éloignement du plan de projection par rapport à l'oeil, l'angle d'ouverture exprimé en degrés, et l'angle de roulis (inclinaison de la caméra par rapport à son axe de vision). Le roulis n'est pas pris en compte dans cette version.

Contrairement à la palette de curseur, la saisie numérique s'opère directement sans la touche Option. La modification graphique d'un paramètre pourra se faire dans les versions ultérieures par clic préalable sur son icône.



Palette de perspective

La palette de perspective permet deux types d'opérations: modifier de façon graphique les valeurs des paramètres de projection de la fenêtre active, et en sélectionner le mode de projection.

La saisie graphique des paramètres, redondante avec celle de la palette précédente, permet de masquer une des palettes tout en gardant l'accès aux paramètres de la projection. Ni la modification graphique, ni les faces cachées ne sont encore implémentées.

Les modes de projections axonométrie / perspective, faces cachées et coupe sont combinables. La vue en coupe se fait en sectionnant tous les objets par le plan de projection, parallèle au plan de la fenêtre et passant par le point de visée. La profondeur du plan de coupe est donc accessible en modifiant numériquement la position du point de visée dans la palette Paramètres de la Vue.

Formes BFM / Mac : fonctionnalités

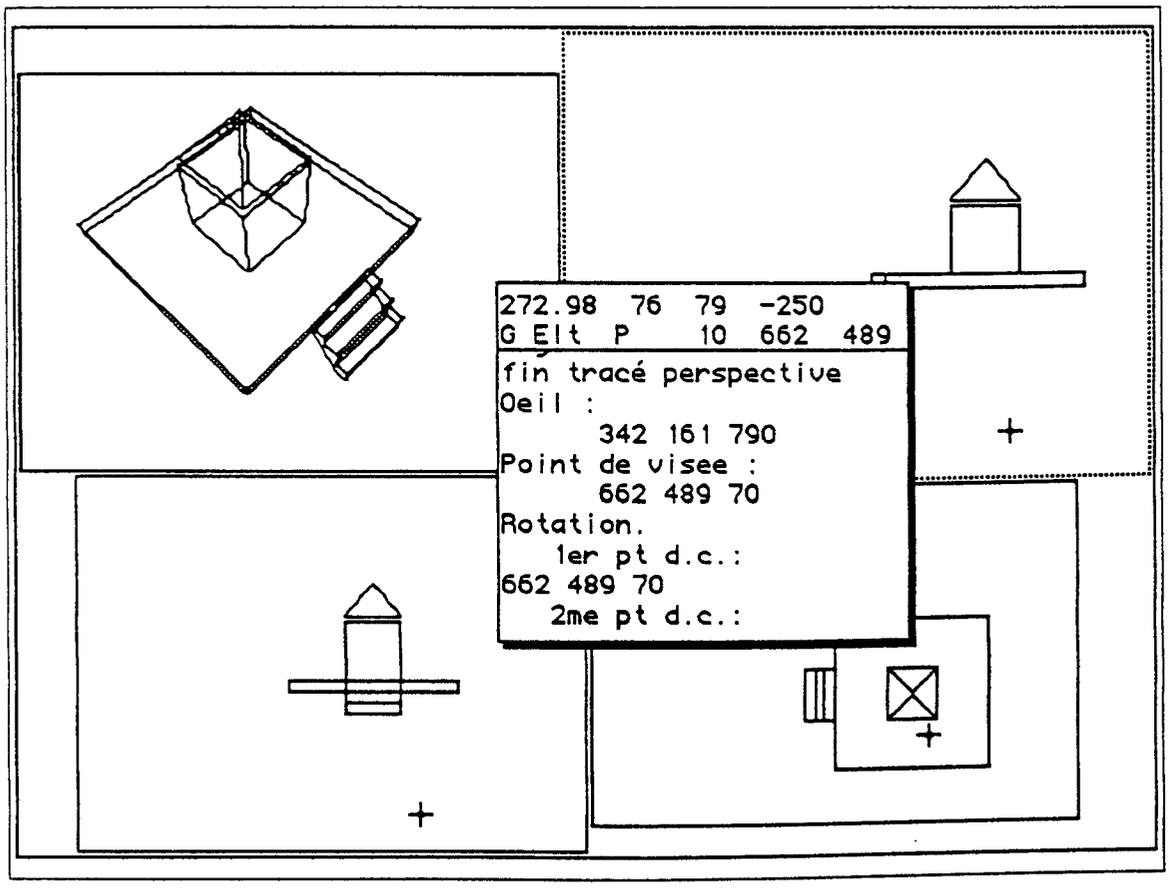
Ce chapitre décrit les quelques fonctions rajoutées à la version BFM de Formes, fonctions qui permettent un comportement "Macintosh" du logiciel. Elles concernent essentiellement la gestion des fenêtres et des fichiers, les déplacements du curseur, les transferts de données entre applications.

Les fonctionnalités de base reprennent à quelques détails près celles de la version originale, et sont décrites dans le manuel BFM de Formes. Elles ne seront donc pas décrites ici. Ont été principalement rajoutées une fonction de génération de B-splines, et une fonction de création de trajectoires.

Les fenêtres

Nous avons vu qu'il était obligatoire de créer des fenêtres pour contenir les éléments d'informations textuelles et graphiques. Formes BFM utilise deux fenêtres : l'une est la copie conforme de l'écran tel qu'il apparaissait à un utilisateur de la version originale sur BFM, et affiche les représentations graphiques. L'autre remplace l'écran texte du BFM, et est utilisée pour l'affichage des demandes de saisie, des réponses de l'opérateur, et du statut du logiciel.

D'autres fenêtres annexes, de type dialogue, sont utilisées pour la saisie de données numériques ou pour la sélection des objets à manipuler.



Les fenêtres de Formes version BFM

Le curseur

Comme dans la version originale, le curseur peut être déplacé grâce aux touches du clavier numérique, la sélection du plan de déplacement se faisant par combinaison des touches Commandes et 1, 2, ou 3. Le déplacement par souris a été rajouté, pour remplacer la saisie tablette ; cette option se sélectionne par menu.

Dans le cas d'un déplacement par souris, la trame sous-jacente peut être activée par combinaison avec la touche Majuscules. La touche Option permet de placer le curseur sur le point d'un objet le plus proche de la position actuelle de la souris.

Les transferts de données

Pour faciliter les transferts de données entre Formes et d'autres logiciels, qui pourraient effectuer des opérations spécifiques sur celles-ci, d'autres formats de données ont été spécifiés :

- Le format *PICT2*, standard d'Apple, qui permet de récupérer les représentations produites par Formes dans un logiciel de dessin vectorisé, à des fins de traitement, de coloration, ou d'impression ;
- Le format *TEXT*, qui permet de générer ou de récupérer des données de descriptions géométriques d'objets sous forme textuelle, et telles que les définit le didacticiel sur l'algorithme perspectif développé conjointement par les laboratoires de Lyon et de Grenoble. Ce format est également lisible par des logiciels de traitement de texte, ou des tableurs.

D'autre part, la sauvegarde d'éléments ou d'objets en bibliothèque a été remaniée, pour permettre de gérer des bibliothèques de composants plus étendues.

Bibliographie

AHO A. - HOPCROFT J. - ULLMAN J.
Structures de Données et Algorithmes
InterEditions *1989*

BRET M.
Images de Synthèse
Dunod *1988*

Collectif
La Synthèse d'Images
Hermès *1988*

FOLEY J.D. - VAN DAM A.
Fundamentals of Interactive Computer Graphics
Addison Wesley Publishing Company *1983*

INSIDE MACINTOSH VOL I..V
Addison-Wesley Company *1986-89*

LIEBLING T. - RÖTHLISBERGER H.
Infographie et Applications
Masson *1988*

MITCHELL WILLIAM J.
Computer Aided Architectural Design
Van Nostrand Reinhold Company *1982*

NEWMAN M. - SPROULL R.
Principles of Interactive Computer Graphics
Mc Graw Hill *1978*

ROGERS DAVID F. - ADAMS ALAN J.
Mathematical Elements for Computer Graphics
Mc Graw Hill *1976*

ROGERS DAVID F.
Algorithmes pour l'Infographie
Mc Graw Hill *1988*

RYAN DANIEL L.
Computer Aided Architectural Design
Marcel Dekker *1983*

SCHWEIZER PH.
Infographie I & II
Presses Polytechniques Romandes *1983*

WOODWARK J.
Calcul de Formes par Ordinateur
Masson *1988*