Etude des procédures informatiques conversationnelles d'aide à la conception et au calcul des structures architecturales

Jean-Marc Brun, Michel Maille, Michel Théron, Louis-Paul Untersteller

ETUDE DES PROCEDURES INFORMATIQUES CONVERSATIONNELLES D'AIDE A LA CONCEPTION ET AU CALCUL DES STRUCTURES ARCHITECTURALES

Jean-Marc BRUN
Michel MAILLE
Michel THERON
Louis-Paul UNTERSTELLER

AUPA n° 5 et LIMSI du CNRS

ETUDE DES PROCEDURES INFORMATIQUES CONVERSATIONNELLES D'AIDE A LA CONCEPTION ET AU CALCUL DES STRUCTURES ARCHITECTURALES

Jean-Marc BRUN
Michel MAILLE
Michel THERON
Louis-Paul UNTERSTELLER

AUPA n° 5 et LIMSI du CNRS

Cette étude à été subventionnée par le Comité de la Recherche pour la Direction de l'Architecture (CORDA).

Convention 75-73-013 00 202 75 01.

Jean-Marc BRUN
Michel MAILLE
Michel THERON
Louis-Paul UNTERSTELLER

La conception de structures architecturales nécessite la manipulation d'éléments architecturaux ; ceux-ci sont décrits à partir d'éléments géométriques et calculés à partir d'éléments finis.

Si l'architecte trouve sa spécificité dans la gestion des premiers, s'il maitrise bien les seconds, il considère généralement que les troisièmes ne le concernent pas.

Peut-on malgré l'informatique, et grâce à des procédures graphiques interactives, amener les éléments finis aux architectes sans obliger ces derniers à venir aux éléments finis.

Tel est le but de la présente recherche.

S O M M A I R E

CHAPITRE I

ARCHITECTURE ET INFORMATIQUE

QUELQUES POINTS DE REPERES

PRELIMINAIRES

Vouloir discuter des rapports entretenus par l'Architecture et l'Informatique peut paraitre relever de la simple provocation.
Tel n'est cependant pas l'objet de cette introduction dont le but est de clarifier des situations souvent ambigües et établies sur la base d'incompréhensions réciproques.

LE FAIT INFORMATIQUE :

La première chose à rappeler est que, qu'on le veuille ou pas, l'humanité, limitée sans doute pour le moment aux seules sociétés industrialisées, est en train de s'informatiser. Fait technologique, l'informatique est devenu un fait de société et si certains risques sont à craindre, par exemple dans le domaine des libertés individuelles, il ne faut cependant pas voir de façon systématique "BIG BROTHER" derrière chaque machine IBM. L'informatique n'est pas une science mais une simple technologie issue des progrès de l'Electronique, science et industrie à laquelle nous ne pouvons plus échapper.
L'Electronique est entrée dans les moeurs : du réveil électronique à la calculatrice de poche, du téléviseur aux guichets de la SNCF, du rasoir aux distributeurs de billets de banque, du péage d'autoroute au pilotage automatique des avions, de la machine à écrire à la machine-outil, l'Electronique est omniprésente et le sera de plus en plus.

L'industrie électronique a, en effet, la particularité de proposer des produits dont les coûts sont divisés par deux tous les deux ans et dont les performances dans le domaine informatique augmentent pratiquement dans les mêmes proportions.
Les premiers ordinateurs nécessitaient une salle entière climatisée ; à capacité égale et performance décuplées, ils tiennent aujourd'hui sur un petit bureau, voire dans une valise. Puissants calculateurs, mais longtemps sourds et muets, les ordinateurs sont devenus gratte-papier dans un premier temps, dessinateurs dans un second ; ils commencent à savoir parler et lire, savent entre eux correspondre par téléphone ; établir une liaison par satellite entre ordinateurs placés de chaque côté de l'Océan Atlantique est devenu un jeu d'enfant.
L'obsolescence des produits est telle qu'un sentiment de mystification est naturel ; il faut cependant bien s'en garder et toujours conserver à l'esprit

.../...

le fait que c'est l'homme qui maîtrise ces développements ultra-rapides et que le jour de la révolte des robots n'est pas encore arrivé.

LES ARCHITECTES DEVANT L'INFORMATIQUE :

Le deuxième préliminaire est relatif à l'échelle des réactions propres à l'architecte face à l'ordinateur. Cette échelle est immense. Il nous est arrivé de rencontrer à l'une des extrémités ce que nous appellerons l'architecte curieux et visionnaire. Il s'attend à tout de lui, entendons par là qu'il rêve de pouvoir un jour se croiser les bras pendant que "la machine" ferait son travail en lui laissant cependant la définition des buts et le choix des résultats.
Cet architecte n'a pas compris que l'ordinateur ne faisait que refaire, sans doute plus vite, ce que l'homme lui avait appris à faire et encore dans la seule mesure où les actions à entreprendre avaient pu être formalisées. La rencontre entre cet architecte et l'informaticien est toujours dramatique et le premier comprend mal qu'il n'existe pas déjà un bouton sur lequel il suffit d'appuyer pour avoir la solution au problème posé. Déçu dans son premier contact avec l'Informatique, cet architecte là est perdu à jamais pour l'ordinateur dont il aura déclaré une fois pour toute qu'il ne sert alors à rien.
A l'autre extrémité de l'échelle, on peut rencontrer l'architecte-artiste mais homme d'affaires ayant réussi.
Celui-ci pense avoir le monopole de la conception et remodèle les villes comme les enfants la pâte à modeler.
Il vit très mal l'arrivée de la machine sur le marché et pense que l'ordinateur fait partie d'un vaste complot destiné à le déposséder de son domaine de compétence.
En permanence sur ses gardes, inquiet, il mènera son combat d'arrière-garde coûte que coûte avec des arguments d'autant plus fallacieux qu'il aura réussi à imposer "son architecture".
Lorsqu'on lui demande quels services il attend de l'informatique, il vous répond qu'il n'a pas besoin, lui, de cette "béquille".

Heureusement pour tous l'échelle n'est pas vide en dehors de ses extrémités et l'on a pu dès à présent, trouver des passerelles entre l'art de l'architecte et celui de l'informaticien.

Nous voudrions aborder enfin un dernier point avant d'entrer dans

.../...

le vif du sujet. Ce point est relatif à la confusion encore entretenue entre la Conception Assistée par Ordinateur et la Conception Automatique.

CONCEPTION AUTOMATIQUE OU CONCEPTION ASSISTEE ?

La Conception Automatique est un mode d'utilisation de l'ordinateur dans lequel, à partir de données sélectionnées par l'homme, la machine fournit LA solution, parfois LES solutions. Cette ou ces solutions sont issues de choix successifs faits par la machine à partir d'algorithmes et de critères pré-définis par le programmeur. La richesse des choix augmente avec l'habileté de ce dernier beaucoup plus qu'avec le savoir-faire du concepteur ; la qualité des choix ne saurait être remise en cause puisque "c'est la machine" qui a fait ces choix suivant des évaluations pré-programmées qui ne peuvent par définition, qu'être les meilleures.
Un certain nombre de bâtiments à usage de bureau ont été ainsi construits aux Etats-Unis. Le programme de Conception Automatique intégrait tous les paramètres nécessaires (plusieurs milliers) à la détermination des plans d'exécution, des plans de charge, du bon fonctionnement ultérieur, des coûts de fonctionnement. Dans la pratique le client donnait quelques paramètres et tous les autres étaient choisis par défaut. Les résultats ont été catastrophiques au plan architectural.
La Conception Assistée par Ordinateur est tout au contraire le droit à l'erreur. Elle est basée sur un dialogue homme-machine et sur une répartition des tâches correspondant à la meilleure utilisation des compétences et des capacités de l'un et de l'autre. Les choix sont faits par le concepteur sur la base de son expérience et de son savoir-faire ; la machine se contente de déterminer les conséquences de ces choix et les cohérences avec des choix précédents ; l'homme conserve la liberté de les évaluer. Les algorithmes de détermination des conséquences ne sont pas nécessairement "uniques" et peuvent être construits à partir de la connaissance que possède le concepteur du problème à résoudre. Il est alors bien évident que la qualité du produit conçu augmente avec les compétences du concepteur et non avec l'adresse de l'informaticien qui a mis au point le système électronique de base.

.../...

LIAISONS ARCHITECTURE - INFORMATIQUE :

Ceci posé, on peut maintenant essayer de passer en revue quels sont les points d'ancrage des liaisons architecture-informatique. Ces points sont nombreux et variés, d'autant plus variés qu'il est très difficile aujourd'hui de savoir ce qu'est l'architecture en regard du nombre des intervenants dans le processus de production du cadre bâti. Aussi, prendrons-nous le mot "architecture" dans son sens le plus large en regrettant que bien souvent l'architecte range hors de son domaine de compétence tant l'analyse qui se trouve en amont de la phase purement conceptuelle que le contrôle et la gestion du bâtiment qui se situent en aval. Pour la clarté de l'exposé, nous aborderons les points successifs dans le sens qui va de l'expression du besoin à la gestion du produit fini.

ALEXANDER :

Il est impossible de parler architecture et informatique sans une pensée émue pour Christopher ALEXANDER qui, dans les années 60 à 65, a l'un des premiers, posé le problème des données pour la conception. Constatant en effet le nombre de précautions et la masse des informations accumulées avant la mise en fabrication du moindre objet "industriel", certains architectes ont alors ressenti la nécessité de collecter un maximum de données avant d'entamer la phase de conception proprement dite. On peut en effet raisonnablement penser qu'une bonne connaissance du contexte évitera ou limitera les erreurs, erreurs qui apparaissent toujours trop tard. Ce besoin est ressenti de façon d'autant plus impérieuse en architecture que la mise en chantier n'est jamais précédée par la fabrication d'un prototype. Les essais ont toujours lieu sur une maquette à échelle 1 livrée et prête à être expérimentée en grandeur nature par les usagers eux-mêmes, et sans retour en arrière possible. Toutes les machines à se raser, les machines à se déplacer sont testées avant d'être livrées au public. Comme ces tests et les simulations sont impossibles dans le processus de création des machines à habiter, on conçoit qu'il faille prendre plus de précautions encore en ce domaine.

.../...

Les données sont d'autant plus nombreuses que l'objet à concevoir est important et complexe ; concevoir une habitation, à fortiori une ville, est, à priori, plus délicat et plus lourd de conséquences que la conception d'un briquet à gaz. La masse des informations à prendre en compte est sûrement plus importante et l'affaire ne peut être que compliquée par leurs natures très différentes : expression du besoin des usagers, expression des contraintes de la collectivité, données historiques, données socio-économiques, données géographiques, données formelles,... pour n'en citer que quelques-unes.
Le traitement d'une telle masse d'information implique à priori le recours à l'informatique et l'on connut voici quelques années la grande vogue de l'analyse des données en architecture et en urbanisme.

Pour compléter le tableau, il faut dire quelques mots de la problématique exposée par Alexander. Pour lui, un projet d'architecture peut être décrit par une liste d'exigences à satisfaire, une bonne solution architecturale étant un bon compromis entre ces exigences souvent contradictoires. Constatant que l'esprit humain a beaucoup de difficultés à concilier les quelques cinq ou six contraintes exprimées à travers un problème simple, Alexander propose de décomposer le problème de la conception architecturale en sous problèmes hiérarchisés correspondant à des sous-ensembles d'exigences : ces sous-problèmes doivent être définis autour d'exigences intimement liées permettant la résolution d'un système homogène de contraintes et doivent présenter le maximum d'indépendance entre eux. Chaque sous-problème correspond à une solution architecturale partielle et l'art du concepteur consiste à reconstituer une solution globale la plus satisfaisante possible à partir de solutions partielles que l'on croit correctes. Cette approche par décomposition hiérarchique pose elle aussi quelques solides difficultés opératoires et le recours à l'informatique est là encore obligatoire.

La mode du recours systématique à l'analyse mathématique informatisée est aujourd'hui sinon passée, du moins en nette régression et ses plus brillants zélateurs se sont parfois transformés en violent détracteurs. C'est oublier un peu trop vite que ces techniques d'analyse étaient issues d'autres domaines qui ne les appliquaient avec succès que dans la mesure où les objets manipulés avaient été parfaitement formalisés.

.../...

Ceci est rarement le cas en architecture et il semble que seules les analyses ayant eu pour finalité la mise au clair de typologies de bâtiments ou d'éléments architecturaux aient connu quelques succès. Cet échec relatif n'est pas à mettre au compte de l'ordinateur mais plutôt sur la méconnaissance de deux facteurs importants :

- ne sont susceptibles de traitements scientifico-mathématiques que les données qui en ont le statut, c'est-à-dire qui sont formalisables et formalisés.
- collecter des données n'a d'intérêt que si l'on sait ce que l'on veut en faire ; ceci implique que les données doivent être pertinentes pour la problématique envisagée. Tel n'est pas toujours le cas et la formule "calculer, calculer encore , il en sortira toujours quelque chose " ne parait ni juste, ni saine.

Ces deux remarques ne remettent pas en cause l'un des usages possibles de l'ordinateur mais visent à préciser l'attitude souhaitable avant d'en introduire l'usage dans un domaine nouveau. La porte reste ouverte et pourrait ne jamais se refermer si les architectes et les urbanistes poursuivent leurs efforts de réflexion et de formalisation sur les objets et concepts qu'ils manipulent.

COMBINATOIRE ARCHITECTURALE :

Le deuxième créneau dans lequel se sont engouffrés certains architectes a été celui de la combinatoire de l'espace architectural. Si certains pensent avoir tout exploré à partir des quelques règles classiques de la "composition" et de quelques essais manuels, nombreux sont les architectes conscients de la relative pauvreté de telles méthodes et des richesses de l'exploration systématique enfin rendue possible par la capacité fabuleuse à calculer et à mémoriser des ordinateurs.
Dans ce domaine comme dans d'autres on est passé par plusieurs stades.
Une première voie a consisté à explorer toutes les configurations possibles avec un minimum de contraintes. Le minimum était justifié par la volonté de ne pas ignorer à priori un trop grand nombre de solutions.
Cette attitude a eu deux conséquences sur les architectes qui ont expérimenté cette approche :

- leur faire prendre conscience de leur manque relatif d'imagination quand ils se contentent de faire appel à l'intuition ;
- leur fournir du papier brouillon pour jusqu'à la fin de leur vie tant étaient nombreuses et inexploitables les solutions proposées.

.../...

Les architectes avaient rencontré là un travers devenu classique et dont tout le monde n'a pas encore pris conscience : il faut chercher des outils et des méthodes adaptées aux méthodes de conception et non pas modifier ces dernières pour pouvoir utiliser à tout prix l'outil.

Trop de contraintes, pas de solutions ; peu de contraintes, trop de solutions. De ce dilemme est née une nouvelle approche combinatoire communément appelée "allocation spatiale" dans laquelle le rôle du concepteur allait devenir plus actif.
Cette approche est fondamentalement différente de la précédente dans le mesure où elle ne fournit pas toutes les solutions mais en élabore une petit à petit en fonction de contraintes et de formules d'évaluation librement choisies par le concepteur.
A chaque étape, il existe toujours au moins une solution particlle qui, si elle n'est pas toujours la meilleure, est du moins la moins mauvaise. Si le système est bien fait, il permet une certaine flexibilité et permet au concepteur d'intervenir au cours du processus, en particulier si plusieurs solutions sont jugées équivalentes par la machine, le concepteur peut alors "prendre la main", choisir celle qui lui semble la meilleure et sur les bases de laquelle l'ordinateur poursuivra son élaboration.
Cette approche a déchainé les passions mais, là encore, et c'est normal, des limites sont vite apparues ; une telle approche en effet est basée sur des fonctions d'évaluation qui, comme les contraintes, n'ont de sens que si elles s'appliquent à des concepts formalisables et formalisés. Dans la pratique, on se limite aujourd'hui à prendre en considération les contraintes liées aux proximités et circulations. Ceci fait dire à certains que la conception architecturale assistée par ordinateur ne saurait qu'être fonctionnelle, mais ceci est un autre débat et comme l'aspect fonctionnel ne saurait être complètement négligé, nous avons tendance à penser que "c'est toujours celà de pris".
Que les espoirs fous du début aient fait place à un certain découragement, quoi de plus naturel ! Les quelques applications qui ont été menées à terme devraient cependant permettre d'envisager l'avenir avec une relative sérénité.

.../...

VISUALISATION

Depuis que les ordinateurs sont capables de piloter des machines à dessiner automatiques, le lieu privilégié des rencontres entre l'informatique et l'architecture relève du domaine graphique. On peut reconnaitre trois grands niveaux de dessins dans ce qui est manipulé par l'architecte :

..les esquisses et dessins de travail
. les dessins d'exécution et plans de détail
. les dessins de "prestige" parmi lesquels nous rangerons les perspectives destinées à éclairer le promoteur, les collectivités locales ou le public.

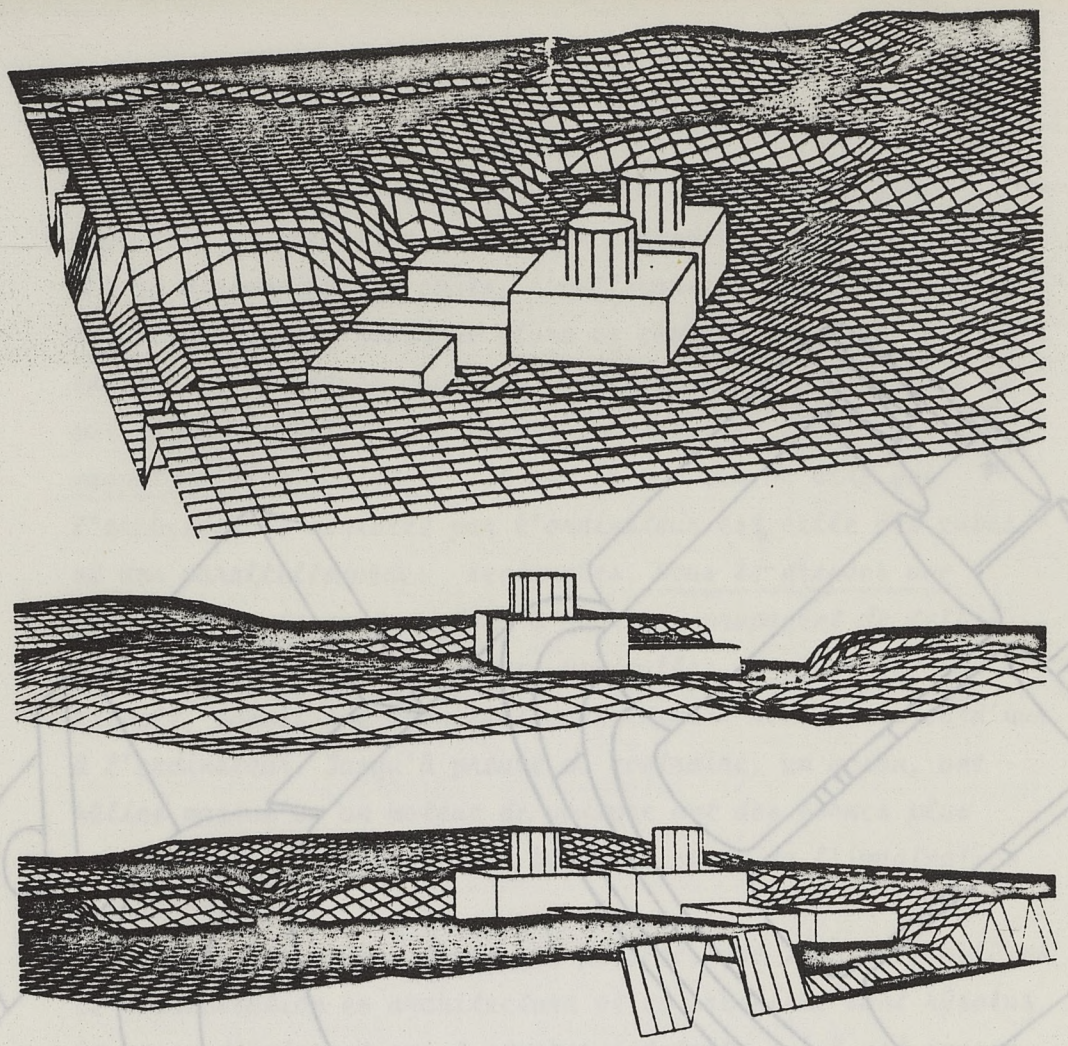
L'usage de l'ordinateur dans les deux derniers niveaux est maintenant entré dans les moeurs ; c'est de loin ce qu'il y a de plus spectaculaire.

Partant du fait que près de 70% du travail d'un bureau d'études consiste à reproduire plusieurs fois tout ou partie d'un même plan, que 20% du travail consiste à dessiner proprement de nouveaux plans et 10% seulement à des tâches plus créatives, c'est-à-dire à les élaborer, on conçoit le succès que peut rencontrer le dessin automatique. L'ordinateur sait parfaitement aujourd'hui reproduire autant de fois que nécessaire le même détail sur un plan de façade, changer automatiquement d'échelle, enregistrer des modifications ou des paramètres permettant la sortie de gamme d'éléments.

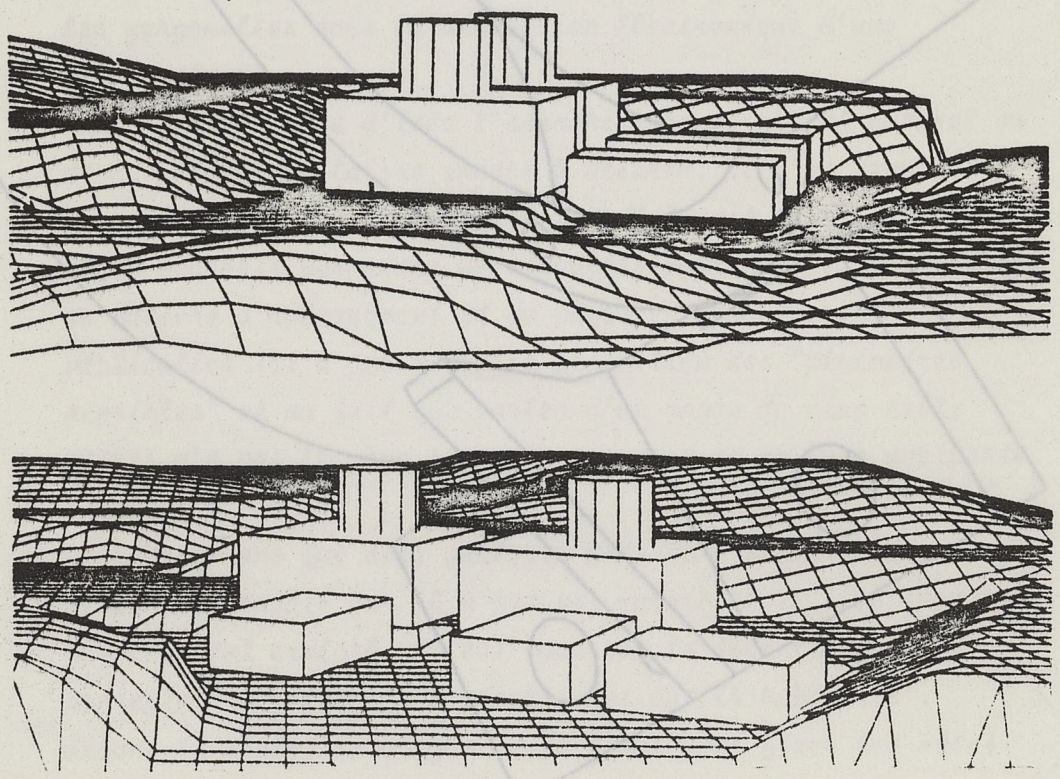
L'ordinateur sait de même parfaitement calculer et dessiner une perspective quelconque avec ou sans élimination des parties cachées, fournir la perspective qui permet le photomontage parfait du bâtiment conçu dans son futur site, fournir un jeu de perspectives légèrement décalées et suffisamment précises pour simuler un cheminement.

Longtemps utilisée comme outil d'aide à la conception, la perspective depuis son invention a peu à peu perdu ce rôle pour se confiner dans celui du faire valoir. Les architectes ne faisaient plus faire que les quelques perspectives avantageuses susceptibles, convenablement habillées, d'aider le promoteur à vendre sa marchandise.
La disparition progressive de projeteurs de haut niveau et l'augmentation du coût de la main d'oeuvre ont accéléré ce processus. On peut espérer rendre à la perspective son statut initial dans la mesure où l'ordinateur est capable, à très bon marché, de fournir au concepteur un grand nombre d'images, justes et de toutes tailles sur lesquelles il peut travailler.

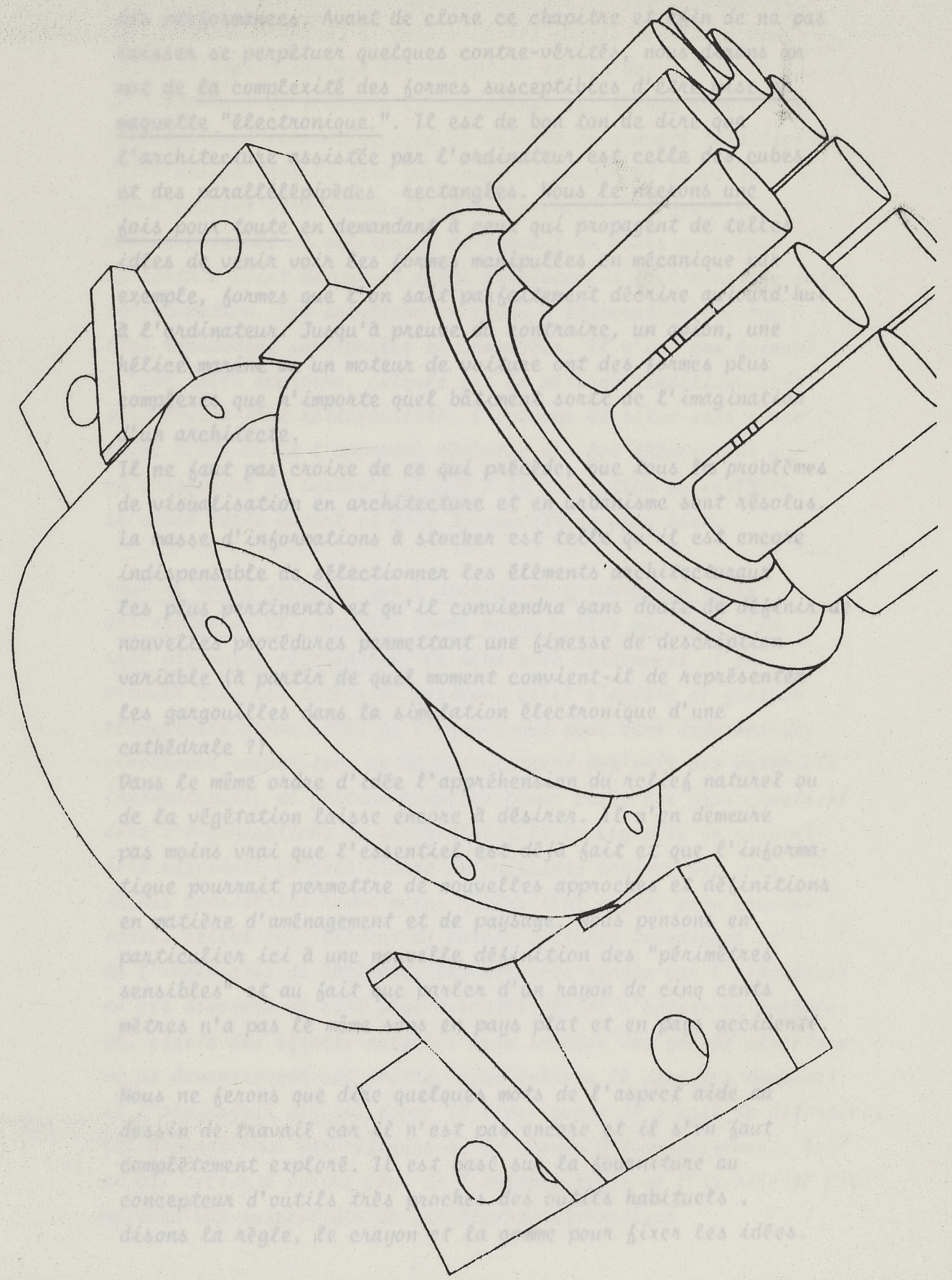
.../...



Implantation d'une centrale nucléaire dans un site côtier.
Etude réalisée à l'aide du programme EUCLID



Etude du terrassement nécessaire à l'installation d'une centrale nucléaire. Vues perspectives du projet réalisées à l'aide du programme EUCLID.



L'ordinateur ne connaît que les cubes ! EUCLID. LIMSI du CNRS

La plupart des ressources disponibles ont été à ce jour explorées et l'on peut surtout en attendre la généralisation par la diminution des coûts, l'amélioration des matériels et des performances. Avant de clore ce chapitre et afin de ne pas laisser se perpétuer quelques contre-vérités, nous dirons un mot de la complexité des formes susceptibles d'être mise en maquette "électronique". Il est de bon ton de dire que l'architecture assistée par l'ordinateur est celle des cubes et des parallélépipèdes rectangles. Nous le nierons une fois pour toute en demandant à ceux qui propagent de telles idées de venir voir les formes manipulées en mécanique par exemple, formes que l'on sait parfaitement décrire aujourd'hui à l'ordinateur. Jusqu'à preuve du contraire, un avion, une hélice marine ou un moteur de voiture ont des formes plus complexes que n'importe quel bâtiment sorti de l'imagination d'un architecte.

Il ne faut pas croire de ce qui précède, que tous les problèmes de visualisation en architecture et en urbanisme sont résolus. La masse d'informations à stocker est telle qu'il est encore indispensable de sélectionner les éléments architecturaux les plus pertinents et qu'il conviendra sans doute de définir de nouvelles procédures permettant une finesse de description variable (à partir de quel moment convient-il de représenter les gargouilles dans la simulation électronique d'une cathédrale ?).

Dans le même ordre d'idée l'appréhension du relief naturel ou de la végétation laisse encore à désirer. Il n'en demeure pas moins vrai que l'essentiel est déjà fait et que l'informatique pourrait permettre de nouvelles approches et définitions en matière d'aménagement et de paysage. Nous pensons en particulier ici à une nouvelle définition des "périmètres sensibles" et au fait que parler d'un rayon de cinq cents mètres n'a pas le même sens en pays plat et en pays accidenté.

Nous ne ferons que dire quelques mots de l'aspect aide au dessin de travail car il n'est pas encore et il s'en faut complètement exploré. Il est basé sur la fourniture au concepteur d'outils très proches des outils habituels, disons la règle, le crayon et la gomme pour fixer les idées.

.../...

Il requiert des moyens graphiques souples, des temps de réponse rapide et des moyens de communication homme-machine en cours de développement mais non opérationnels à ce jour. Cette voie est intéressante dans la mesure où les outils sont bien des outils d'aide directe à la conception et non des outils se contentant de soulager le concepteur des tâches répétitives. La différence esentielle avec la méthode de travail traditionnelle réside dans le fait que coups de crayon et coups de gomme peuvent être mémorisés et hiérarchisés. Il est à tout moment possible de superposer le travail en cours pris sous un certain point de vue et celui précédemment fait avec une autre préoccupation. Les solutions en cours d'élaboration peuvent être rapidement superposées et comparées. Il n'est pas évident qu'une telle approche, calquée sur la méthode traditionnelle, soit tout-à-fait satisfaisante. L'optimum se situe sans doute entre celle-ci et l'approche élaborée par les théoriciens qui ne conçoivent de systèmes d'aide à la conception qu'intégrés. Le débat reste ouvert et nous y reviendrons plus loin.

Faut-il enfin rappeler que l'architecte a souvent à faire des tâches qu'il considère comme moins nobles mais qui n'en sont pas moins nécessaires. Ce sont bien évidemment et en premier lieu les tâches de gestion, qu'il s'agisse de gestion élémentaire comme celle de l'agence qui peut être informatisée simplement compte-tenu de la décroissance des prix des matériels informatiques, de gestion complexe comme celle des gros chantiers dont l'organisation, la surveillance et le contrôle ne peuvent être effectués qu'avec l'aide de puissants ordinateurs. Ces tâches considérées comme moins nobles sont enfin, bien souvent toutes celles qui relèvent du bureau d'études techniques et de l'ingénierie du bâtiment. Les services rendus par l'ordinateur en ces domaines ne sont plus discutés. Qu'il s'agisse de calculer les effets des efforts auxquels sera soumise une grande structure ou de dimensionner une poutre, nous entrons là dans les domaines où l'ordinateur est roi comme il l'est quand il s'agit de déterminer les déperditions calorifiques ou le réseau d'assainissement. Tout ce qui relève à priori plus de l'ingénieur que de l'architecte est aujourd'hui susceptible de traitement informatique .

.../...

Ceci pose deux problèmes de nature différente : celui de l'intégration d'une part, celui des rapports entretenus par les architectes et les ingénieurs d'autre part.

Avant de revenir sur ces deux points, nous rappellerons que l'ordinateur permet la constitution de banques de données urbaines ou technologiques et que les possibilités du télétraitement permettent dès à présent d'envisager l'informatisation partielle de l'agence d'architecture. Nous rappellerons aussi que la gestion et la maintenance des grands édifices publics n'est elle aussi possible que par l'automatisation des processus de contrôle et que l'existence de tels processus peut ne pas être sans influence sur la conception.

Le problème de l'intégration déjà évoqué plus haut est le suivant : la conception d'un bâtiment requiert des informations, des manipulations et des calculs de nature très différentes. Tous ont au demeurant trait au même objet et s'il est redondant d'en donner autant de descriptions que de manières de voir, il est sans doute impensable de n'en envisager qu'une seule. Comme il n'existe en outre pas d'homme capable d'appréhender tous les problèmes, peut-on imaginer une machine qui stockerait toutes les informations, en assurerait la cohérence et qui serait la source unique dans laquelle chacun puiserait à sa convenance pour résoudre avec son langage propre le problème relevant de sa seule spécificité ? Divers systèmes ont été mis au point ou sont en cours d'élaboration. Nous ne pensons pas qu'il en existe de parfait aujourd'hui, mais on peut toujours rêver et se demander pourquoi ce que l'on a su faire pour les industries aéronautiques et spatiales ne serait pas reproductible après adaptation à l'industrie du bâtiment. Et que l'on ne vienne pas dire que celà implique l'industrialisation de la construction car, si la construction aéronautique est du ressort de la fabrication en série, la construction spatiale reste quant à elle tout-à-fait artisanale.

Si enfin, nous avons parlé des rapports entretenus par les architectes et les ingénieurs, c'est que l'ordinateur parait bien souvent aux yeux des premiers comme l'allié inconditionnel des seconds.

L'ARCHITECTE ET LE CALCUL

Le problème est alors posé de savoir si l'ordinateur précipite la disparition de l'architecte ou si, au contraire l'ordinateur peut apparaitre comme un outil susceptible de rendre l'architecte plus indépendant de l'ingénieur en limitant les demandes d'intervention et renverser la tendance actuelle qui veut que la part de l'architecte dans la distribution des rôles soit une fonction décroissante du temps.

Nous sommes intimement convaincus, tout ingénieur ou scientifique que nous soyons, que pour le bien de la collectivité, il vaut mieux croire à la deuxième branche de l'alternative. La diminution des coûts de l'informatique, les améliorations notables de la communication homme-machine permettent d'y croire.

Ceci implique cependant et bien évidemment une politique hardie d'introduction de l'informatique dans les établissements d'enseignement de l'architecture, donc des moyens matériels et des méthodes pédagogiques modernes.

Les progrès de l'Electronique rendent l'ordinateur financièrement abordable et ceux du télétraitement permettent de partager à plusieurs les mêmes ressources ; les expériences déjà faites ont appris comment il fallait introduire l'ordinateur à l'Ecole et comment il fallait s'en servir pour qu'il ne soit pas considéré comme une solution de facilité mais comme un outil d'assistance à l'enseignement avant de devenir celui de l'aide à la Conception.

C'est dans cette optique que nous avons lancé l'étude des procédures informatiques conversationnelles d'aide à la conception et au calcul des structures architecturales. Notre ambition est <u>de montrer que les architectes et les étudiants en architecture peuvent contrôler par eux-mêmes les structures qu'ils imaginent</u> et que les techniques de calcul, même très sophistiquées comme celles qui font appel aux éléments finis peuvent être mises entre leurs mains aux conditions suivantes :

- la description des formes est assurée par de simples manipulations graphiques sur un écran de visualisation ;
- la description des cas de charges est aussi intuitive que possible.
- le calcul est complètement transparent pour l'utilisateur.
- les résultats sont automatiquement et directement visualisés.

CHAPITRE II

DU CHOIX DU SUJET ET DES SOLUTIONS ADOPTEES

L'IDEE D'EXPERIMENTATION DANS L'ENSEIGNEMENT DES STRUCTURES

L'enseignement analytique classique de la statique et de la Résistance des matériaux permet dans quelques cas simples d'aboutir au dimensionnement correct de chacun des éléments d'une structure ; il n'en fournit pas pour autant une appréhension et une connaissance globale de l'ensemble du sytème structurel.

Au stade de la conception, l'expérience acquise dans l'étude de cas voisins du cas étudié est plus utile qu'une analyse fine sur un objet non encore défini. Si cette expérience ne s'acquiert qu'au fil des années, on peut en accélérer le processus par l'expérimentation sur maquette. Malheureusement, si toute maquette est apte à mettre en évidence les défauts de stabilité, nous sommes bien obligés de constater que celles qui sont faites par nos étudiants dans le cadre des Unités Pédagogique d'Architecture, à savoir les maquettes en Bristol ou en Balsa, ne permettent pas d'améliorer la connaissance de la structure au delà d'un jugement par tout ou rien : la structure tient ou ne tient pas !

LA NOTION DE MAQUETTE VIRTUELLE

Les fabuleuses capacités à calculer des ordinateurs font que leur pratique la plus courante est celui du dimensionnement de projets déjà définis. L'informatique joue dans ce cas un simple rôle de vérification ou de contrôle. On peut en imaginer un tout autre usage et, avec les mêmes matériels, avec des logiciels (ou programmes) très peu différents, on peut envisager d'utiliser les sytèmes informatiques pour faire de l'expérimentation de structures. Ceci nécessite que les sytèmes de description, de manipulation et de visualisation d'objets tridimensionnels simulent complétement les structures à étudier, autrement dit que les structures de données ou informations numériques que sait traiter l'ordinateur représentent une véritable simulation tant de l'objet à étudier que des phénomènes dont il sera le siège ; c'est la notion de maquette virtuelle qui a été développée par les auteurs du programme EUCLID. Il faudra pour cela disposer de programmes permettant de décrire la structure, de la modifier, de la visualiser, mais aussi de programmes permettant de décrire les efforts auquels elle sera soumise comme de programmes de calculs permettant d'en simuler le comportement.

.../...

L'objet principal de cette étude est de vérifier l'éventuelle possibilité d'intégrer l'ensemble de ces programmes en un même sytème informatique afin de fournir à l'architecte et spécialement à l'étudiant un outil lui permettant de "voir" la première version de sa structure comme s'il avait fait une maquette classique, de lui appliquer l'analyse fine que permettent les codes de calcul, d'en observer le comportement comme si la structure avait été réalisée à l'échelle 1, d'en modifier éventuellement la géométrie ou certaines caractéristiques et de retourner au calcul afin d'élaborer ainsi de proche en proche une solution dont le comportement soit aussi voisin que possible de celui désiré.

Bien qu'il n'entre pas dans notre propos de comparer les mérites respectifs de l'expérimentation sur maquette classique et de celle sur maquette virtuelle, il peut être intéressant de remarquer que dans les deux cas le processus est décomposé en trois phases.

Dans un processus classique, l'étudiant devra :

1. réaliser la maquette et la mettre sous charge
2. mesurer des déplacements, des allongements ou des contraintes
3. dépouiller les résultats des mesures et en assurer leur représentation.

Dans le processus d'expérimentation sur maquette virtuelle, on distinguera de manière analogue :

a. une phase de création durant laquelle l'étudiant doit créer la maquette virtuelle, c'est-à-dire décrire des éléments de la structure, des liaisons, des charges ;

b. une phase de calcul durant laquelle l'ordinateur calcule les déplacements, allongements et contraintes.

c. une phase de restitution et de visualisation des résultats.

S'il est bien évidemment possible de visualiser le maquette virtuelle, éventuellement de la modifier avant calcul et d'en étudier

.../...

le comportement pour divers cas de charges, le processus normal d'expérimentation consiste à modifier la maquette au vu des résultats du ou des calculs précédents en itérant autant de fois que nécessaire sur les 3 phases a, b, c, pour obtenir une solution structurale satisfaisante et la réponse aux questions que l'on peut vouloir se poser à son sujet.

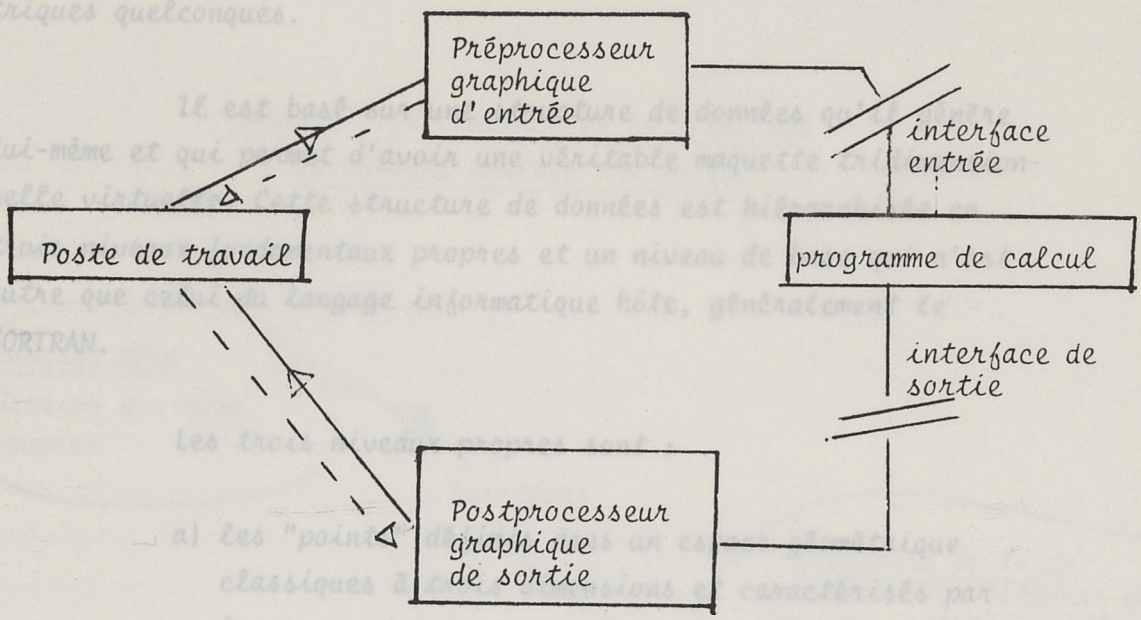
Si le problème du calcul (phase b) est classique et résolu automatiquement pour toute structure codée suivant les spécifications requises par les algorithmes utilisés, il en va autrement du problème de la description (phase a) puisque chaque projet est un cas d'espèce. Notre expérience nous a montré que dès que la forme était complexe un obstacle majeur se dressait devant nos étudiants soudain transformés en informaticiens : celui de la programmation. C'est pourquoi EUCLID a été conçu dès le départ comme un langage de description aussi proche que possible du langage propre de la géométrie, discipline dont on pouvait espérer que nos étudiants avaient quelques notions ; c'est pourquoi nous avons réalisé dans un deuxième temps un assouplissement de la syntaxe en nous affranchissant de certaines contraintes à priori imposées par l'informatique.

Il s'est avéré que l'obstacle majeur n'avait toujours pas été levé puisqu'il fallait toujours "programmer" c'est-à-dire tout prévoir avant que de commencer à faire exécuter quoi que ce soit. Les difficultés liées à la programmation sont supprimées dans un usage interactif de l'ordinateur, c'est-à-dire dans un usage basé sur le dialogue homme-machine. Un tel usage nécessite des procédures informatiques particulières dites conversationnelles afin que l'utilisateur puisse contrôler lui même le cheminement entre les 3 phases citées plus haut, comme il nécessite un raccourcissement des délais entre l'idée d'une modification et la visualisation de ses conséquences.

Il existe actuellement de nombreux programmes de calculs de structure. La description de la structure est généralement faite au préalable par un système indépendant. Ce système communique au

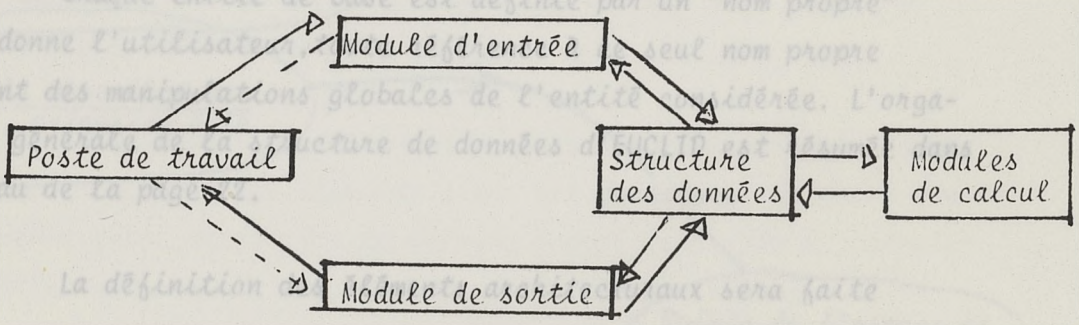
.../...

programme de calcul des informations codées de façon stricte. Ce dernier fournit par la suite à un troisième système de quoi visualiser les résultats. Le schéma en est classique :



Il est satisfaisant si l'on cherche à obtenir un dimensionnement en une seule passe a, b, c ; il a le défaut majeur d'appauvrir l'information à chaque passage aux interfaces et de ne pas offrir à l'utilisateur, sur l'image finale de l'objet qu'il vient de construire la possibilité d'intervenir comme il avait pu le faire sur l'image initiale quelques minutes auparavant.

Ceci nous a conduit à adopter un autre schéma dans lequel la structure des données est commune aux trois phases du processus :



Ceci a impliqué la réalisation d'un module de calcul opérant dans la structure de données générée par EUCLID.

Compte tenu de ce qu'EUCLID a déjà fait l'objet de nombreuses publications, nous nous contenterons d'en donner une brève description dans le chapitre suivant et de rappeler ici que c'est un langage de création, manipulation et visualisation d'éléments géométriques quelconques.

Il est basé sur une structure de données qu'il génère lui-même et qui permet d'avoir une véritable maquette tridimensionnelle virtuelle. Cette structure de données est hiérarchisée en trois niveaux fondamentaux propres et un niveau de base qui n'est autre que celui du langage informatique hôte, généralement le FORTRAN.

Les trois niveaux propres sont :

a) les "points" définis dans un espace géométrique classiques à trois dimensions et caractérisés par leurs coordonnées.

b) les liaisons ou "segments" qui assurent l'ordre dans lequel doivent être reliés les points pour constituer les éléments géométriques de base.

c) les éléments géométriques eux-mêmes qui peuvent être soit des lignes ouvertes, soit des lignes fermées et opaques appelées "facettes", soit des assemblages d'éléments de base constituant des "figures".

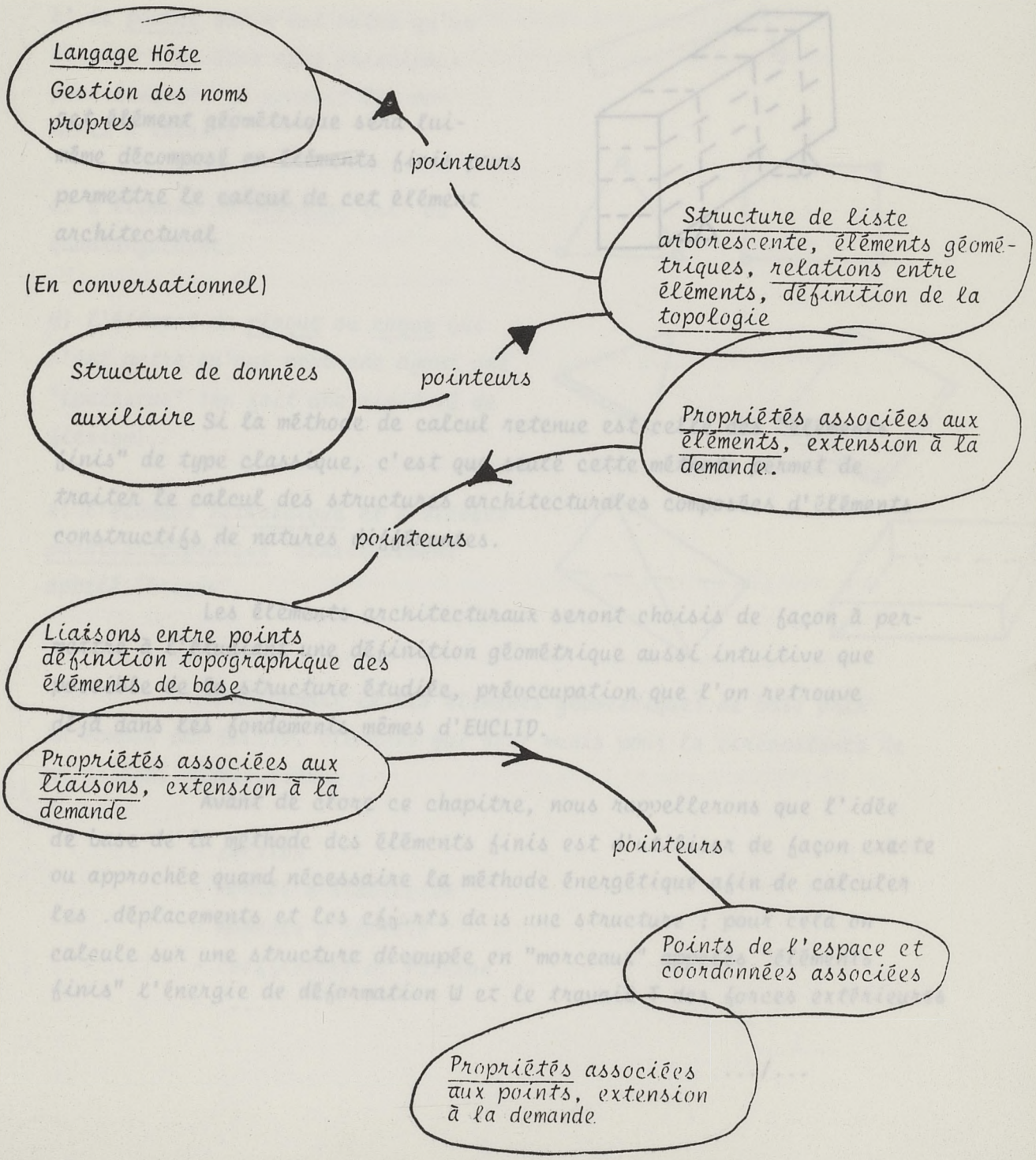
Chaque entité de base est définie par un "nom propre" que lui donne l'utilisateur, toute référence à ce seul nom propre permettant des manipulations globales de l'entité considérée. L'organisation générale de la structure de données d'EUCLID est résumée dans le tableau de la page 22.

La définition des éléments architecturaux sera faite

.../...

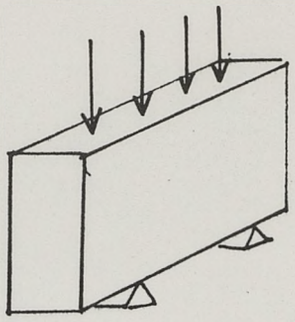
à partir de cette structure de données en associant à des éléments EUCLID, éléments qui ne sont que des éléments purement géométriques de description de formes, des éléments de calculs regroupés en éléments architecturaux qui peuvent être les seuls connus de l'utilisateur.

ORGANISATION GENERALE DE LA STRUCTURE DE DONNEES GENEREE PAR EUCLID

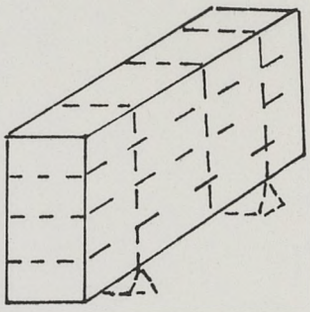


Ainsi à l'élément architectural "MUR" défini par des dimensions et un matériaux correspondra

un élément géométrique représentant cet élément architectural



cet élément géométrique sera lui-même décomposé en éléments finis pour permettre le calcul de cet élément architectural



Si la méthode de calcul retenue est celle des "éléments finis" de type classique, c'est que seule cette méthode permet de traiter le calcul des structures architecturales composées d'éléments constructifs de natures différentes.

Les éléments architecturaux seront choisis de façon à permettre à l'étudiant une définition géométrique aussi intuitive que possible de la structure étudiée, préoccupation que l'on retrouve déjà dans les fondements mêmes d'EUCLID.

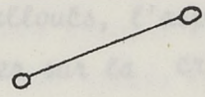
Avant de clore ce chapitre, nous rappellerons que l'idée de base de la méthode des éléments finis est d'utiliser de façon exacte ou approchée quand nécessaire la méthode énergétique afin de calculer les déplacements et les efforts dans une structure ; pour cela on calcule sur une structure découpée en "morceaux" appelés "éléments finis" l'énergie de déformation W et le travail T des forces extérieures

.../...

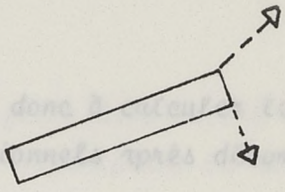
il suffit ensuite d'écrire que W-T est minimal pour trouver l'état d'équilibre de la structure.

Les éléments de calcul retenus seront du point de vue géométrique :

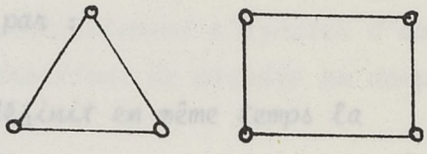
a) le segment auquel n'est attribué aucun axe privilégié ; c'est la barre ou le tube, articulé ou encastré à ses extrémités.



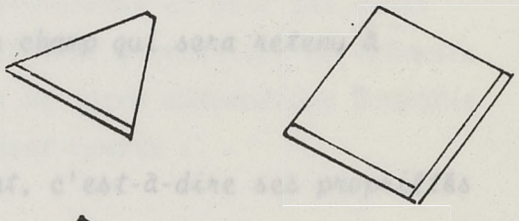
b) la poutre qui n'est autre qu'un segment avec deux axes principaux pour lesquels on connait les moments d'inertie.



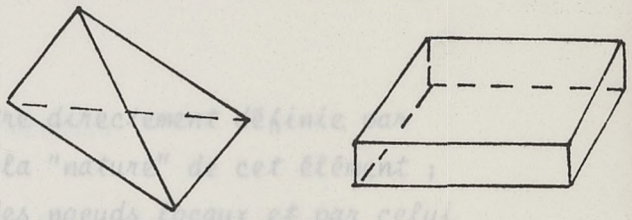
c) l'élément de membrane tri- ou quadrangulaire.



d) l'élément de plaque ou coque qui n'est autre qu'une membrane ayant une "épaisseur" (en fait une rigidité de flexion).



e) l'élément tétraédique et l'élément parallélipipédique classiquement appelé "brique".



On reconnait là les éléments géométriques de base pris en compte par EUCLID, éléments qui sont munis pour la circonstance de propriétés particulières :

- section
- module d'élasticité
- moments d'inertie
- coefficient de Poisson
- .....

.../...

Au plan théorique, il est bien évident que la prise en compte des éléments courbes tels que troncs de cône, cylindres, éléments de surface courbe ainsi que la prise en compte de milieux non isotropes ne pose pas de problèmes majeurs. Cependant notre sens des réalités nous incline à penser que les problèmes posés par ces configurations ne pouvaient raisonnablement être traités dans le cadre des douze mois effectifs de travail qui nous avaient été alloués, l'aspect de cette première étude devant essentiellement porter sur la création de la procédure et le test de sa faisabilité.

En règle générale on cherchera donc à calculer les champs de déformations linéaires bi ou tridimensionnels après décomposition des formes géométriques en éléments finis.

Un élément fini sera défini par :

- sa forme géométrique qui définit en même temps la décomposition du support du champ cherche ;

- le mode d'interpolation du champ qui sera retenu à l'intérieur de l'élément ;

- la pondération de l'élément, c'est-à-dire ses propriétés physiques.

La forme géométrique peut être directement définie par EUCLID. Le mode d'interpolation définit la "nature" de cet élément ; il est défini quant à lui par le choix des noeuds locaux et par celui des fonctions d'interpolation attachées à ces noeuds. Cette "nature" d'élément ne doit pas être confondue avec sa "forme" et nécessitera un codage spécifique ; il en est de même pour ce qui concerne la pondération.

Un assemblage d'éléments finis ou structure est donc défini par l'ensemble des éléments qui le compose et par le mode

.../...

d'assemblage ; ce dernier est déterminé par les liaisons entre les valeurs nodales attachées aux divers noeuds de chacun des éléments ; les liaisons sont définies par l'identité des valeurs nodales, c'est-à-dire finalement par l'identité des noeuds. Sauf cas très particuliers cette identité des noeuds n'est autre que leur identité en tant que points géométriques puisqu'un champ prend une valeur univoque en chaque point géométrique. Les assemblages d'éléments peuvent être définis très simplement par EUCLID avec génération des noeuds locaux et recherche des noeuds coïncidants.

L'implémentation des éléments finis dans EUCLID se fait donc par adjonction de codages spécialisés définissant la nature de l'élément et sa pondération.

A cette extension peuvent éventuellement s'ajouter d'autres extensions de la structure de données permettant de prendre en compte les liaisons spéciales.

Il est bien évident que l'utilisateur ne doit pas avoir à faire lui-même la décomposition de l'élément géométrique en éléments finis ; cette décomposition se fait donc de façon automatique à partir de considérations topologiques ; elle tient compte :

- de la finesse requise
- de la nature des éléments retenus
- des problèmes posés par les raccordements entre éléments de nature différente.

Les auteurs de ce rapport sont tous enseignants dans les Unités Pédagogiques d'Architecture ; ils ont à ce titre introduit une réflexion pédagogique dans l'élaboration du système proposé. Cette préoccupation se traduit par deux partis fondamentaux :

<u>simplicité des structures calculées</u> afin que l'étudiant puisse en un temps limité obtenir des résultats clairs ; d'où le choix :

- d'éléments architecturaux simples
- décomposés de façon standard

.../...

- assemblés aussi . automatiquement que possible.

<u>interaction étudiant-machine</u> afin que cette dernière non seulement fournisse au premier des résultats de calculs comme le ferait un bureau d'études mais l'aide en outre à comprendre les mécanismes fondamentaux de la RDM et des relations efforts-contraintes-déformations par manipulations nombreuses et rapides sur support graphique.

CHAPITRE III

SUR LE SYSTEME EUCLID

Comme il a été précédemment dit, il n'entre pas dans notre propos de donner ici une description complète de l'actuelle version d'EUCLID. Il nous est cependant apparu utile d'en rappeler l'essentiel.

1 - OBJECTIFS INITIAUX D'EUCLID

Ayant commencé à nous intéresser à la résolution de problèmes d'aéro ou d'hydrodynamique par calcul analogique et plus particulièrement par les méthodes rhéographiques, nous avions l'habitude de visualiser rapidement, sinon automatiquement et la forme d'un profil d'aile, par exemple, et les lignes de courant autour dudit profil. Nous nous sommes heurtés au problème de l'informatique graphique quand nous avons abordé la résolution numérique de nos problèmes et quand, au lieu d'obtenir après une demi-journée d'efforts un réseau d'équipotentielles, nous avons obtenu après une minute d'ordinateur un tableau de 20 000 nombres donnant les valeurs de la fonction à étudier au divers noeuds de notre domaine. Alors que notre oeil savait apprécier globalement et instantanément les résultats obtenus par la méthode analogique, nous restions par contre bien souvent perplexes devant les 100, 200 voire 400 pages de papier, pliées et recouvertes de nombres. D'où, notre intérêt soudain pour le dessin automatique par calculateur, ou, plus exactement pour le géométrie par calculateur, tant il est vrai que pour nous l'essentiel n'était pas de remplacer le "rapido" par un "spot" ou la "main" par un "mécanisme", mais bien de se doter et de donner au concepteur un outil d'emploi simple lui permettant de concevoir directement les formes spatiales, de les déformer, les déplacer ou les assembler avant que d'y mener des calculs et de les visualiser. D'où les objectifs fondamentaux :

I - Un langage de description géométrique et tridimensionnel proche du langage usuel, pouvant s'utiliser sans brochure à portée de la main, c'est-à-dire de façon instinctive et dont la partie visualisation n'est qu'un sous-produit.

.../...

2 - Un langage compatible avec l'ensemble des calculateurs, indépendants du système d'entrée-sortie, compatible surtout avec le calcul scientifique. Ainsi, EUCLID est écrit en FORTRAN standard et permet à l'utilisateur averti d'en utiliser toutes les ressources : boucles DO, tableaux indicés... tout en s'affranchissant par moment de certaines contraintes.

3 - Une grande souplesse structurelle permettant indifféremment et sans difficultés majeures :
- soit de travailler en "batch" avec programmation et table traçante
- soit de travailler en mode conversationnel avec écran de visualisation.
En aucun cas nous ne voulions être tributaires de tel ou tel matériel.

4 - Une grande souplesse de développement afin qu'à tout moment l'outil puisse être utilisable pour un type d'application quelconque.

II - ANALYSE DU LANGAGE GEOMETRIQUE USUEL

Le langage géométrique procède avant tout par création et nomination d'êtres géométriques. Les noms sont de deux types :
- les noms communs ou noms génériques définissent la classe de l'entité géométrique ;
- les noms propres qui permettent de reconnaitre l'être ou les êtres définis, toute référence ultérieure à ceux-ci se faisant par l'intermédiaire de ces noms propres.

Si ces derniers peuvent changer avec l'inspiration du moment, les noms communs sont quant à eux strictement codifiés et constituent l'ossature du langage géométrique. Mais un tel langage serait pauvre si on ne lui adjoignait des verbes pour caractériser les actions, des adverbes pour en préciser les modalités et des adjectifs pour qualifier les noms propres auxquels ils se rapportent ; bien souvent ces mots sont regroupés dans des locutions verbales.

Sans préciser plus ici, on conviendra que même réduit à ses éléments fondamentaux, le langage usuel de la géométrie est

.../...

encore trop vaste et trop flou pour en permettre la transcription directe dans un calculateur. Aussi, nous a-t-il été nécessaire de préciser encore le sens des éléments de la géométrie que nous avons pu regroupés comme suit :

- opérateurs primitifs ;
- opérateurs élémentaires ;
- opérateurs de concaténation ;
- opérateurs de transformation ;
- opérateurs de contraintes.

III - STRUCTURE DE DONNEES

Nous ne parlerons pas ici de la structure de données et nous contenterons de dire qu'elle combine une structure nominative (associative) à une structure de liste compacte avec bouclage unidirectionnel. Cette dernière permet de retrouver facilement sinon rapidement la filiation d'un élément de la liste, la structure nominative associant un point d'entrée au nom propre de l'élément créé.

IV - LES ELEMENTS DU LANGAGE

Toute la partie géométrique répond au principe du langage usuel :

⟨nom propre⟩ = ⟨nom commun⟩ (paramètres)

c'est-à-dire qu'elle utilise la "function" FORTRAN. Nous avons ainsi défini :

⟨A⟩ = POINT (x,y,z)
⟨B⟩ = POINT (N)
⟨S⟩ = SEGMEN (A,B)
⟨FACE⟩ = FACETE (A_1,...A_n) ou FACETE (N,A)

où A est un tableau de N points.

.../...

De la même manière, nous avons défini la ligne ouverte et la ligne fermée :

⟨L⟩ = OLIGNE ($A_1 \ldots A_n$) ou FLIGNE (N., A).

Les arguments peuvent éventuellement être des lignes déjà définies ;

⟨S⟩ = SURFAC ($L_1 \ldots L_n$) ou (N,L).

où $L_1 \ldots L_n$ sont des lignes brisées définies par le même nombre de points.

L'opérateur de concaténation est très simplement défini par :

⟨T⟩ = FIGURE ($E_1 \ldots E_n$)

ou encore FIGURE (N, E) si T comprend N éléments rangés dans le tableau E.

Ces éléments peuvent eux-mêmes être déjà des figures, le niveau d'imbrication n'étant théoriquement pas limité.

Il n'est pas bon que la création, le déplacement ou la déformation des êtres géométriques se fassent par des opérateurs aux noms rébarbatifs et à liste d'arguments trop longue. Nous nous sommes inspirés du langage géométrique usuel et nous avons décidé de faire appel aux facultés mnémoniques. Ainsi, les deux premières lettres précisent s'il y a création (CR) ou simplement déplacement ou déformation (DP) ; les trois suivantes précisent la nature de la transformation envisagée, la dernière lettre précisant si besoin est une direction privilégiée de la transformation géométrique.

Ces transformations sont :

CR éation	TRAN slation		(dz, dx, dy, F)
DP lacement	HOMO thétie		(C, R, F)
	ROT ation	A xe	(a, Axe, F)
	AFF inité	X	(a, F)
	SYM étrie	Y	(k, F)
		Z	
		P oint ou plan	(P, F)

.../...

Ainsi F étant une figure, on peut soit la déplacer par translation en écrivant par exemple :

F_1 = DPTRAN (F_1 , 1, 2, 1)

soit créer F_2 à partir de F_1 par la même translation en écrivant :

F_2 = CRTRAN (F_1 , 1, 2, 1)

IV BIBLIOTHEQUE DE BASE

Un certain nombre de formes géométriques utiles et complexes ont été définies pour faciliter la tâche de l'utilisateur ; la liste n'est pas limitative et chacun peut personnaliser sa bibliothèque. On notera pour mémoire toutes les surfaces prismatiques, surfaces de révolution et polyèdres plus ou moins réguliers comme toutes les surfaces réglées. Ainsi :

⟨nom⟩ = PRISMX (dx,S)

définit un profilé de sections S où S est soit une ligne, soit une facette (profilés creux, pleins...) et (dx) le vecteur translation faisant passer parallélement à l'axe x de la face avant à la face arrière (la section S peut être oblique).

V - VISUALISATION

Un grand nombre de sous-programmes permettent de visualiser simplement la ou les formes géométriques créés par EUCLID. C'est en particulier toutes les projections orthogonales, les axonométries et les perspectives- qu'elles soient centrales sphériques ou cylindriques. Ces parties cachées peuvent à la demande être éliminées, caractérisées ou tracées.

VI - GENERALITES SUR L'INTERACTIF D'EUCLID

On peut utiliser un ordinateur en "traitement par lots" (ou batch) ou en "interactif".

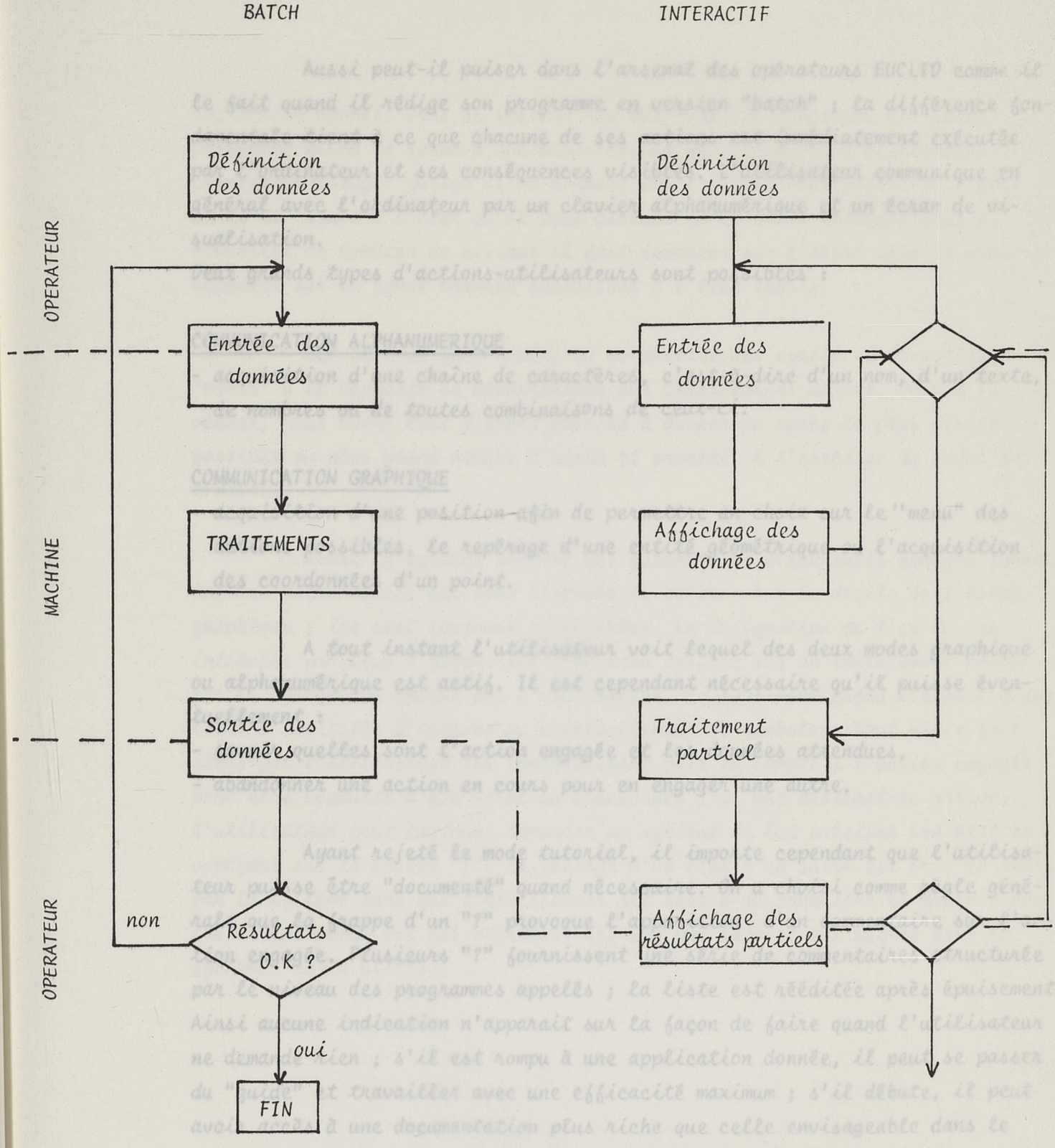
L'opposition fondamentale entre ces deux moyens de traitement est de même nature que celle qui oppose le texte et le discours. Si un texte écrit peut être relu ou corrigé avant d'être donné à lire, il ne reste plus qu'à attendre ensuite le verdict du lecteur.
L'orateur au contraire, s'il ne peut effacer ce qu'il a dit peut par contre faire évoluer son discours en fonction des réactions de l'auditoire. Ainsi le contenu du discours s'élabore-t-il en fonction du contexte.

Le traitement "batch" peut utiliser un langage de programmation fort strict ne permettant ni la moindre fantaisie, ni la moindre ambiguïté d'interprétation ; une erreur de codage autre que syntaxique n'est perceptible qu'au vue des résultats finaux ; et encore faut-il que ceux-ci soient très éloignés des résultats escomptés. Si un texte écrit peut être mûrement réfléchi, la langue parlée accepte tatonnements et redondances.

Bien qu'EUCLID ait pris dès l'origine l'aspect d'un langage de programmation comme le langage FORTRAN dont il est l'hôte, nous avons essayé d'y introduire une souplesse qui fait généralement défaut aux langages informatiques traditionnels. Cette souplesse tient essentiellement au fait qu'une "phrase EUCLID" s'efforce d'être proche du langage naturel et peut être diversement interprétée par l'ordinateur en fonction du contexte. Ainsi EUCLID laisse libre le nombre d'arguments auxquels fait appel l'opérateur-fonction utilisé, comme il laisse libre leur ordre ou leurs types.Cette souplesse d'utilisation se paye par le risque d'une interprétation erronée de l'ordinateur, erreur qui n'est détectée là encore qu'après exécution du programme et visualisation de la forme générée.

Dans un usage interactif, l'interprétation fonction du contexte est d'autant plus fondée qu'une mauvaise interprétation est immédiatement perceptible par l'utilisateur et qu'il est toujours à même de se reprendre. Aussi avons-nous délibérément refusé de construire le système sur un mode interactif tutorial dans lequel l'ordinateur demande à l'utilisateur les réponses à une liste de questions prévues d'avance.

.../.



En interactif, chaque bloc séquentiel du traitement batch est remplacé par un ensemble constitué :

- d'un bloc d'entrée de données, suivi
- d'un calcul partiel et d'un affichage partiel, suivis
- d'un choix de l'opérateur

Aussi peut-il puiser dans l'arsenal des opérateurs EUCLID comme il le fait quand il rédige son programme en version "batch" ; la différence fondamentale tient à ce que chacune de ses actions est immédiatement exécutée par l'ordinateur et ses conséquences visibles. L'utilisateur communique en général avec l'ordinateur par un clavier alphanumérique et un écran de visualisation.
Deux grands types d'actions-utilisateurs sont possibles :

COMMUNICATION ALPHANUMERIQUE

- acquisition d'une chaîne de caractères, c'est-à-dire d'un nom, d'un texte, de nombres ou de toutes combinaisons de ceux-ci.

COMMUNICATION GRAPHIQUE

- acquisition d'une position afin de permettre un choix sur le "menu" des actions possibles, le repérage d'une entité géométrique ou l'acquisition des coordonnées d'un point.

A tout instant l'utilisateur voit lequel des deux modes graphique ou alphanumérique est actif. Il est cependant nécessaire qu'il puisse éventuellement :
- savoir quelles sont l'action engagée et les données attendues,
- abandonner une action en cours pour en engager une autre.

Ayant rejeté le mode tutorial, il importe cependant que l'utilisateur puisse être "documenté" quand nécessaire. On a choisi comme règle générale que la frappe d'un "?" provoque l'apparition d'un commentaire sur l'action engagée. Plusieurs "?" fournissent une série de commentaires structurée par le niveau des programmes appelés ; la liste est rééditée après épuisement. Ainsi aucune indication n'apparait sur la façon de faire quand l'utilisateur ne demande rien ; s'il est rompu à une application donnée, il peut se passer du "guide" et travailler avec une efficacité maximum ; s'il débute, il peut avoir accès à une documentation plus riche que celle envisageable dans le mode tutorial où elle est imposée. Enfin et surtout, si l'utilisateur a été distrait de son travail, il peut le reprendre comme il l'entend.

Si la documentation à la demande permet de savoir ce qui est en cours, elle ne permet pas à l'utilisateur de trouver la manière d'engager

.../...

une action donnée. Dans un système non tutorial, le diagramme des états possibles prend la forme d'une arborescence dans laquelle l'utilisateur chemine en aveugle puisqu'il ne connait pas l'arbre. Pour passer de l'état où il se trouve à l'état qu'il veut atteindre, il devra tatonner sauf à connaître de combien de niveaux il doit remonter sur l'arbre afin de pouvoir repartir sur la bonne branche conduisant à l'état requis.

Cet état de fait limite la complexité des systèmes interactifs à "menus" à la capacité de mémorisation des cheminements utiles. Pour y remédier, nous avons tout d'abord cherché à donner un accès le plus direct possible au plus grand nombre d'états et procédé à l'ecrêtage du tronc de l'arbre.

Ainsi le système propose une quarantaine d'intitulés abrégés (quatre lettres mnémoniques) qui sont disposés en colonne sur la droite de l'écran graphique ; ils sont toujours accessibles. La désignation de l'un de ces intitulés provoque l'apparition d'un menu associé sur un emplacement prévu par le système ou choisi par l'utilisateur. Ce sous-menu prend l'aspect d'une liste d'intitulés d'actions ou éventuellement de symboles. Tout choix fait alors par l'utilisateur est immédiatement pris en compte ; l'action engagée peut être conduite à son terme ou abandonnée ; si une distraction arrive, l'utilisateur peut toujours demander au système de lui préciser son état en appuyant sur la touche "?". Le passage d'un état à un autre est pris en charge par le système sans que l'utilisateur ait besoin de connaître la manière dont les divers menus ont été organisés les uns par rapport aux autres dans le système qu'il utilise.

CHAPITRE IV

LES MODULES D'INTERACTION

I - PRESENTATION DES MODULES D'INTERACTION

Nous avons vu que l'application "calcul de structure" comportait trois grandes phases :

a) description de la structure et des charges
b) calcul de résolution du problème
c) restitution des résultats.

La phase a) comporte essentiellement des actions graphiques permettant de réaliser la description géométrique de la structure :

. Acquisition de position de points dans l'espace ;
. Désignation de ces points pour décrire les éléments de structure ;
. Désignation de ces éléments pour décrire en figure des sous-structures de celles à étudier.

Elle comporte aussi l'acquisition de valeurs numériques autres que celles liées aux seules coordonnées ; ce sont les propriétés physiques qui, associées à un élément géométrique, en font un élément calculable.

De la même manière, les charges sur les éléments sont décrites et représentées comme des entités géométriques associées à l'élément de structure. Elles sont par contre codées de façon spéciale.

Si la phase b) ne comporte guère d'interaction, (déclenchement du calcul à la demande), la phase c) qui traite de la visualisation des résultats nécessite elle aussi l'intervention de l'utilisateur. L'aspect conversationnel (ou interactif) réside alors dans le choix des résultats que l'on veut visualiser comme dans le choix des éléments que l'on veut osculter pour un examen plus approfondi. On retrouve là les interactions nécessaires à toute application géométrique :

- acquisition de points ou de lignes dans l'espace ;
- association d'éléments en figure;
- identification d'éléments ou de figures par des noms ;
- retrouver une entité déjà créée et connue par son nom ;
- retrouver une entité par désignation ;
- acquisition de valeurs numériques ;
- choisir une action à entreprendre.

.../.

Ces actions, que leur caractère géométrique assez général rend indépendantes de l'application "calcul de structure", sont suffisamment liées à la structuration tridimensionnelle des données pour qu'on ne puisse pas les traiter sans précautions sérieuses au niveau d'un "terminal intelligent" - c'est-à-dire muni d'une capacité locale de traitement -.

Il s'agit en effet de créer et manipuler des points dans l'espace, et non dans le plan de l'écran. La désignation d'entités géométriques peut ainsi utiliser les éléments visibles de deux sous-ensembles que le système local devra reconnaître comme deux représentations d'une même et unique entité dans l'espace.

Ces entités sont structurées en figures et sous-figures, structuration dont la connaissance doit être implicite dans le dialogue que l'on voudrait instaurer en local.

L'enchaînement dans le choix des intitulés d'actions suppose que l'on gère l'ensemble des modules d'interaction.

Le système enfin de "documentation à la demande" suppose que, même au cours d'une action locale, le système soit informé de l'état dans lequel se trouve l'action entreprise.

Nous avons été amenés, en conséquence, à réaliser un ensemble de modules situé à mi-chemin entre les fonctions d'application et l'échange direct avec le terminal, qui constituent la panoplie des outils permettant de réaliser une application interactive ; cet ensemble forme un langage de description des interactions au même titre qu'EUCLID forme un langage de description d'entités géométriques. En réalisant un découpage des enchaînements d'actions à un niveau de logique élémentaire, il est à la base même de la réalisation du dialogue entre machines, évoqué plus haut, sous forme de messages définis au niveau intermédiaire.

Notons enfin qu'il était nécessaire, pour simplifier l'apprentissage (ou simplement la connaissance) du système, que "la façon de faire" soit standardisée et qu'une même action élémentaire entraîne toujours les mêmes réactions du système.

.../.

La nomenclature des modules réalisés permet la mise en évidence de trois niveaux d'intervention de ces derniers :

niveau zéro nous n'en dirons rien sinon qu'ils assurent les échanges entre ordinateurs et périphériques. Les avoir regroupés permet de se faire une idée des problèmes d'implantation sur un matériel ou de changement de configuration d'un système donné.

niveau intermédiaire

nous distinguerons trois catégories :

1. Modules utilisables sans interaction, dont la réalisation a été rendue nécessaire par
   .. les possibilités d'un contexte interactif,
   .. l'application aux calculs de structures
2. Modules d'acquisition des interactions ;
3. Modules d'acquisition interactive d'entités élémentaires.

niveau bibliothèque

nous distinguerons deux catégories :

1. Modules généraux communs à toute application géométrique ;
2. Modules spécifiques de l'application "calculs de structures".

1. 1. Modules éventuellement interactifs, liés au développement des interactions

- Travail sur écran : tracés de vecteurs, sortie de dessins directement sur le terminal TEKTRONIX 4014.

- Dialogue alphanumérique : La manipulation de chaînes de caractères comme arguments de menus ou de diagnostics a conduit à réaliser la fonction TEXTE : liée à EUCLID, cette fonction en utilise l'opérateur ' FIGURE ' pour concaténer entre eux des tronçons de texte et , le cas échéant, des géométriques.
  Une entité "texte" a été retenue parmi les "paradoxaux" d'EUCLID : son code est le code "texte", elle a pour nombre le nombre de caractères qu'elle contient et pour numéro l'adresse du début du texte dans le tableau des Textes.

.../.

- Repérage : A ce niveau, il existe deux modules de repérage :
l'un, NREPNT, permet l'identification de points,
l'autre, QUEL, la recherche de ou des entités satisfaisant à un ensemble de contraintes.

NREPNT assure le repérage des points.
Pour une position (X, Y) sur l'écran il recherche parmi les points visualisés celui ou ceux dont la projection correspond à ce couple de valeur, à un facteur "ε" de précision près. Pour une vue donnée, les points possibles détectés sont classés par fonction de leur distance à l'oeil de l'observateur (le plus proche d'abord, ...).
Si à "ε" près, n points sont aussi proches, NREPNT retourne n comme valeur de la fonction, et l'ensemble des points possibles dans un tableau transmis comme seul argument.
Si, à "ε" près, plusieurs points provenant de vues différentes conviennent, on retourne arbitrairement le 1er trouvé, car il n'y a pas de classement possible. A l'utilisateur de séparer ses vues, ou de ne pas travailler dans une zone illisible !
L'utilisation de la touche DELETE permet d'explorer la liste des points possibles, du plus proche au plus lointain, en contrôlant à chaque fois le point concerné grâce à sa surbrillance ; en cas de doute, la touche [?] fait dessiner l'ensemble des arêtes aboutissant au point concerné, et permet une identification parfaite du dit point.

QUEL recherche des entités EUCLID définies par des contraintes.
QUEL est un module non-interactif de recherche d'éléments géométriques EUCLID, dans la structure de données EUCLID complétée de la structure de données conversationnelle, satisfaisant à l'ensemble des contraintes transmises en argument.
Ces contraintes sont :

CODE (K)	: EL doit avoir le code "K" (facette, ligne, cercle, figure...)
DERNIE (N)	: EL doit passer par les N derniers points repérés
PARTIE (TRUC)	: EL doit faire partie de TRUC
POSSED (BIDUL)	: EL doit posséder "CHOSE"

Il est prévu d'adjoindre quelques contraintes géométriques, telles :
PASSAN, GAUCHE, DROITE, DUCOTE, LOINDE, ...
Les éléments trouvés sont classés par ordre "hiérarchique", c'est-à-dire les éléments, puis les figures d'éléments, ... puis la figure du tout.

.../.

Dans un usage "BATCH", on peut écrire directement :
EL = FACETE (PARTIE (CAPOT), DERNIE (2))
Les fonctions EUCLID OLIGNE, FACETE, SURFAC, FIGURE, savent en effet appeler d'elles-mêmes la fonction QUEL lorsqu'elles détectent qu'on utilise comme argument une contrainte d'appartenance caractéristique du repérage.

1. 2. Modules traitant les interactions

K = KELTYP (is, x, y)
KELTYP acquiert une position (x, y), le caractère de validation de cette position (is), ainsi que le contexte de l'action, selon un code mis dans K (graphique, choix sur menu, colonne de droite, pointé sur dessin, etc ...)

I = MENU ('\POUTRE & BARRE\')
MENU acquiert un choix de l'opérateur, parmi une liste d'intitulés proposés.

KOIFER ('\ENTRER UN AXE\')
KOIFER est le module de définition de la documentation et de sa mise à jour.

MDIAG ('\TROP DE POINTS\')
MDIAG assure un dialogue en sens unique : système ⟶ opérateur. Il envoie des diagnostics ou des messages à l'opérateur.

1. 3. Modules d'acquisition interactive d'entités élémentaires

Tous ces modules relèvent de la syntaxe générale suivante :
ENTITE = NOM GENERIQUE ($ LABEL, liste de contraintes éventuelles)
ENTITE est donc une variable dont la nature est spécifiée par le nom générique de la fonction requise, éventuellement précisée par des contraintes.
En cas de refus ou d'impossibilité, le contrôle est transféré à l'adresse "LABEL" donnée en premier argument.
Ainsi : EL = CVLIGN ($ 13, PAPLUS (5))
renvoie dans EL une ligne de 5 points au plus, ou se branche en 13.

Les noms génériques disponibles sont les suivants :

.../.

ENTITE : le plus général : entier, réel, nom, ligne, figure repérée, etc ...
ENTIER : un nombre entier exclusivement
REEL : un nombre réel, ou un entier à partie décimale nulle
NOM : un alphanumérique ayant au plus 6 caractères
ELEMEN : une entité géométrique quelconque
REPERE : une entité géométrique existante

CV
- LIGN : une ligne (à créer, à composer, à repérer)
- AXE : un axe
- POL : un point, un centre
- ANGL : un angle, réel ou angle de deux éléments
- RAP : un rapport, réel ou rapport de deux longueurs
- RAY : un rayon, réel ou une distance

Il convient de faire ici deux remarques d'importance :

a - Ces fonctions peuvent s'utiliser de plusieurs manières

. Mode tutorial, documenté pas à pas

```
CALL KOIFER ('\AXE DE LA SURFACE DE REVOLUTION\')
AX = CVAXE ($ 13)
CALL KOIFER ('\MERIDIEN DE LA REVOLUTION\')
OL = CVLIGN ($ 14)
CALL KOIFER ('\NOMBRE DE MERIDIENS\')
N = ENTIER ($ 15)
CALL KOIFER ('\ANGLE DE REVOLUTION\')
ANG = CVANG ($ 16)
```

L'opérateur peut entrer ses différents arguments dans l'ordre qu'il souhaite. Il suffira alors, lorsque le système fera la demande de tel ou tel type d'argument, d'utiliser la fonction "IDEM".

. Mode bibliothèque, où la fonction bibliothèque demande l'acquisition d'une entité englobant toutes celles qui lui sont nécessaires, par exemple :
Surface de révolution axe, angle, nombre, méridien

```
EL = ENTITE ($ 13)
CALL SIAXE (EL, $ 10) EL est l'axe, sinon on continue
CALL SIBRIX (EL, $ 15) EL est le méridien, sinon ...
```

Comme pour les arguments des fonctions EUCLID, dont l'ordre est implicitement rétabli, il convient de ne pas avoir plusieurs entités non distinguables (parce que de même nature), à ceci près qu'un dialogue reste possible en cas de doute lors de l'exécution.

b - Des entités acquises étant stockées selon leur nature géométrique, il n'y a pas de différence entre :

(1) CVAXE ($ 13)

(2) CVLIGN ($ 13, PAPLUS (2)), qui peut éventuellement retourner un point unique, refusé par axe !

Néanmoins, l'axe acquis par (2) ne sera pas identifiable ultérieurement par "IDEM", à la requête d'un nouvel axe.

Pour préciser davantage l'emploi de ces modules, nous allons décrire brièvement les deux plus généraux : CVLIGN et REPERE.

CVLIGN

Ce module acquiert une ligne. Est-elle ancienne, nouvelle, simplement en cours ? L'utilisateur le sait, pas CVLIGN ! C'est donc la première intervention de l'utilisateur qui va permettre à CVLIGN de décider.
L'état "acquisition d'une ligne" est signifié à l'opérateur par l'apparition, en mode rafraîchi, de l'intitulé "LIGNE", un temps très bref : 3/10 de secondes, qui ne permet pas de lire, mais suffit à reconnaître l'intitulé.
La désignation dans la marge de l'intitulé "IDEM" prend comme ligne l'élément en cours, si ce dernier est effectivement une ligne.
La désignation d'un point d'une ligne déjà visualisée, accompagné de la frappe du caractère [R], indique la volonté de REPERER cette ligne ; un second pointé peut être alors nécessaire pour lever une ambiguïté due au fait qu'un même point peut appartenir à plusieurs lignes différentes.
La désignation d'un point déjà visualisé, en frappant le caractère [.], indique la volonté de réutiliser ce point, sans se soucier des éléments auxquels il est associé.
La création de nouveaux points utilise les significations que l'on a convenu d'associer à certains caractères de validation des pointés. Ainsi, par exemple :

: signifie pointé en descriptive (donc double)

H signifie pointé dans le plan Horizontal du précédent pointé

F signifie pointé dans le plan Frontal du précédent pointé

etc .....

.../.

La touche "DELETE" supprime le dernier point et permet une action récursive (n "DELETE" successifs suppriment les n derniers points). La revisualisation en "tiretés" du dernier vecteur de la ligne après gommage permet le contrôle de l'état de cette ligne.

REPERE

EL = REPERE ($ 12)

EL est l'élément qu'on cherche à repérer. Si rien n'est repéré, pour quelque raison que ce soit, on va se brancher directement en 12. REPERE est le programme interactif qui appelle "QUEL ", et propose un pointé.
Il visualise l'ensemble des éléments possédant le point repéré.EL contient alors la figure de ces éléments.
Un nouveau pointé permet alors de sélectionner l'un, ou plusieurs, de ces éléments, ou une figure contenant ces deux points.
Sans qu'il soit nécessaire de repérer alors de nouveaux points, on peut explorer la hiérarchie existante, soit en montant, soit en descendant, par l'utilisation des caractères :

, qui cherche la figure incluant l'élément déjà repéré
< qui permet le retour en arrière.

Le premier pointé de REPERE peut être fait dans la marge, sur l'intitulé "NOM". Il permet ainsi de repérer un élément déjà nommé en frappant au clavier le nom de cet élément.

2. 1. Modules généraux applicables à toute action géométrique

On dispose au niveau bibliothèque de modules généraux dont les intitulés suivent d'assez près la liste des fonctions de base d'EUCLID, et correspondent aux actions de type :

- création de lignes ou surfaces,
- manipulation et appel à des formes "bibliothèque",
- transformations ponctuelles ou topologiques,
- visualisations.

Sans vouloir entrer ici dans la description des modules concernés, on peut remarquer que se pose pour tout module bibliothèque générant une

.../.

entité EUCLID le problème du stockage du résultat. Dans un programme FORTRAN EUCLID BATCH, le compilateur FORTRAN réserve les mémoires nécessaires pour les variables et tableaux qu'on utilise. Lorsqu'on écrit :
AB = SEGMEN (A, B), la variable FORTRAN "AB" contient l'entité géométrique EUCLID générée par la fonction "SEGMEN", qu'on utilisera par la suite dans une figure ou une opération quelconque : C = FIGURE (A1, A2, AB, ...). Ce n'est qu'alors qu'elle sera incluse dans la structure de données EUCLID.

En mode interactif, le programme est écrit avant que soit connu l'enchaînement définitif des opérations. Il ne peut donc réécrire dans une mémoire unique ce qui concerne la même entité ; on ne pourrait jamais faire de figure ! Il faut donc gérer un tableau dit "BROUILLON" de l'interactif, dans lequel on range le résultat de toute création, par l'instruction :

CALL NOUVEL (TRUC).

L'appel à FIGURE remplace alors les n éléments concernés du brouillon par un seul mot, la figure des n éléments, lesquels sont recopiés à ce moment là dans la structure de données d'EUCLID. On trouvera donc dans cette structure annexe de données :

- les figures déjà faites;
- les éléments dernièrement créés (le brouillon).

En cas de débordement de cette structure annexe, on peut remplacer à l'insu de l'opérateur l'ensemble du brouillon par la figure des éléments qui le composent ; la place disponible est alors pratiquement illimitée.

2. 2. Modules spécifiques de l'application "calcul de structures"

2.2.1. Création des éléments structuraux

Comme on l'a vue dans le Chapitre 2, la création d'un élément structural se distingue de celle du géométrique qui le représente par l'adjonction d'un ensemble de valeurs numériques définissant des propriétés mécaniques et d'un codage éventuel (pour les poutres uniquement) du mode de liaison des extrémités de l'entité avec les éléments voisins.

2.2.1.1. Elément architectural - Elément fini

Pour assurer à l'algorithme de front de la phase "calculs" toute son efficacité, il est essentiel de l'alimenter en éléments finis et non en éléments architecturaux, trop liés les uns aux autres.

./.

Par exemple, un ensemble de poutres croisées a un front qui est toute la structure, alors que l'ensemble des tronçons de poutres qui comprend cette même structure a une largeur de front limitée au nombre des tronçons incidents à une seule des poutres.

L'utilisateur qui entre une suite de points, indiquant que chaque nouveau segment s'encastre sur le précédent décrit un élément architectural, arc (ou poutre courbe) des n tronçons consécutifs.

On peut le coder comme tel : ligne ouverte, avec codage des liaisons, mais, préalablement à tout calcul (en une passe), il faut générer la figure provisoire des n tronçons de poutre.

On a choisi de stocker directement cette définition dans le module d'acquisition des poutres, où elle se prête à la retouche ultérieure de tronçons isolés (introduction après coup d'une articulation, par exemple).

2.2.1.2. Description de la géométrie des liaisons

Ce module interactif utilise toutes les commodités communes aux modules d'entrées de points : pointé descriptif, plan particulier, etc .... Quatre caractères spécifiques de l'application complètent cette panoplie initiale et permettent le codage simultané de la nature des liaisons :

E pour ENCASTRE	O pour ARTICULE - FIXE
A pour ARTICULE	X pour ENCASTRE - FIXE

L'usage de l'un de ces quatre caractères utilise implicitement le mode d'entrée précédent (H, F, ...) et rend suffisant dans la majorité des cas un point unique. Si plusieurs informations sont nécessaires on utilise le signe +. Par exemple : [H] + [A] code le nouveau point "articulé, dans le plan horizontal du précédent".

.../.

Un point isolé peut bien évidemment être corrigé après coup, comme toute entité géométrique.

L'intitulé "RETOUCHES" permet de modifier ou corriger les codes liaisons des éléments, type "tronçon de poutre" , par repérage des extrémités validé par le nouveau code à mettre. On reboucle alors sur des couples de points, jusqu'à ce que l'opérateur désigne un autre menu

RETOUCHES

"désigner la première extrémité à corriger"

[E] ----> le nouveau code de cette extrémité sera ENCASTREE

" désigner la seconde"

[O] ----> le nouveau code sera ARTICULE - FIXE.

2.2.1.3. Regroupement des éléments structuraux

L'intitulé '\FIGURE\' permet de faire une figure du "brouillon interactif", munie du pointeur "latéral arrière" qui permettra de lui attribuer des propriétés mécaniques, ou de retrouver ses antécédents.

2.2.1.4. Interface : bibliothèque géométrique - figure d'éléments finis

Afin d'utiliser la bibliothèque EUCLID qui crée des "surfaces" (BOITE, SURFAC, PRISME, REVOL, POLYED, ...) pour générer des réseaux de poutres, on a réalisé un module effectuant l'opération topologique suivante : TRUC = ARETES (CHOSE) où TRUC est la figure d'arêtes de CHOSE, lui-même figure EUCLID quelconque. Une boite (6 facettes) devient ainsi une figure de 12 arêtes, une surface de révolution devient une figure de méridiens et de parallèles, ceci pour calculer une coupole réalisée avec des poutres croisées, la définition par facettes convenant parfaitement aux éléments finis de type "PLAQUE". Ce module ARETE, à caractère topologique et sans aucune interaction, est parfaitement utilisable en batch.

2.2.2. Description des charges

Pour décrire un cas de charge, il faut successivement :

- définir les échelles des valeurs,

.../.

- définir les couples : élément - charge,
- regrouper ces derniers en figures.

2.2.2.1. Introduction des échelles

Avant de commencer à dessiner des forces, il faut définir leurs échelles ; le système les réclamera de toute manière si l'opérateur tente de définir une charge sans avoir procédé d'abord à la définition de l'échelle correspondante.

Le système demande :
"Unité de force ?" et lit un réel, UF
"Représenté par ?" et acquiert un segment de longueur DL
(DL représente UF unités de force)
"Charge répartie sur ?" une deuxième longueur, DL2, précise qu'une charge représentée avec la convention UF/DL est répartie sur une longueur DL2.

Ceci permet l'ajustement des échelles relatives des diagrammes de moments et d'efforts.

Cette définition d'une échelle s'applique à toutes les acquisitions de charges, jusqu'à ce que l'opérateur renouvelle et modifie sa définition.

Elle s'applique également à la représentation des résultats, de sorte qu'on reviendra éventuellement sur cet intitulé du menu pour réajuster, après calcul, les échelles de représentation.

2.2.2.2. Introduction des charges

On a choisi de faire préciser, pour chaque charge, le noeud ou l'élément auquel elle s'applique, deux conventions ayant en effet été retenues pour le stockage, qui sont :
- les couples éléments - charge pour ce qu'on décrit par désignation directe ;
- les listes de points associés aux figures pour les charges définies par algorithme.

.../.

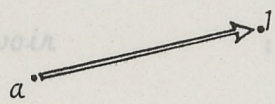
Dans le premier cas, le nombre d'éléments repérés est limité par le temps d'interaction, mais les retouches isolées sont fréquentes. Dans le second cas, le nombre de charges est très important, et un stockage plus compact s'impose : tous les éléments sont nécessairement chargés et une modification isolée d'un ensemble crée par algorithme n'est pas du tout réaliste.

Le nombre et les positions relatives des géométriques que sont les lignes décrivant les charges par rapport aux éléments chargés définissent l'interprétation qu'on leur donne :

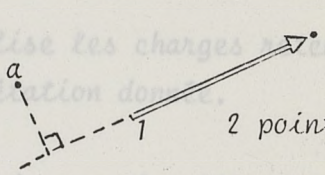
on note a, b, ... l'élément
1, 2, 3, ... les points décrivant la charge

on décrit des couples noeud/charge ou élément/charge.

NOEUD : un point pour le noeud, 1 ou 2 pour la charge.



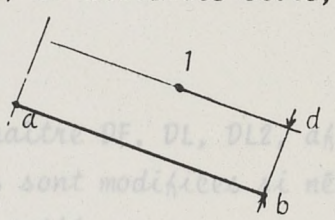
A point : force du noeud



2 points : Force (1 - 2)
Moment / a

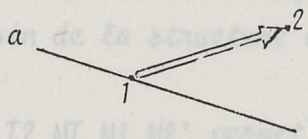
POUTRE : repéré par 1 pointé (a) si extrémité libre, ou 2 (a, b) sinon.

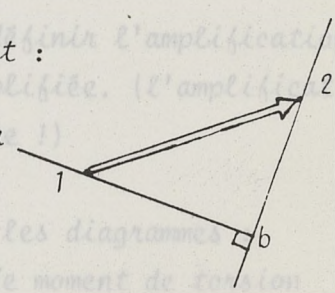
1/ Un point :



Charge répartie d'intensité définie par la distance de 1 à ab

2/ Deux points, dont un sur l'élément :

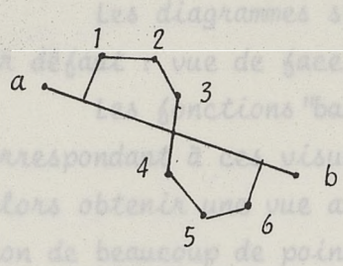


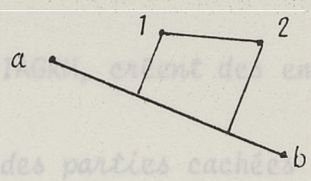


1 ou 2 sur l'élément; c'est une force appliquée en 1 ou 2 ... sauf si l'autre extrémité est sur la normale à "ab" en "a" ou "b"

.../.

3/ Deux points ou plus n'appartenant pas à l'élément :





La représentation immédiate des interprétations faites par le système (flèche pour force, facette pour charge répartie) évite tout malentendu.

CORRECTION : Les commandes suivantes permettent de corriger et contrôler son travail

annuler { le dernier : annule la dernière charge donnée
le tout : annule l'ensemble des charges données

les voir : visualise les charges retenues et l'interprétation donnée.

REUNION : On fait une figure des couples éléments-charges définis. Sauf repérage volontaire, cette figure est prise implicitement comme cas de charge.

2.2.3. Restitution des résultats

'UNITES' permet de faire apparaître DF, DL, DL2, afin de connaître les échelles des diagrammes. Ces échelles sont modifiées si nécessaire par l'intitulé "unité" du menu "charge" toujours actif.

'DEFORMEE-AMPLIFICATION' permet de définir l'amplification de la déformée, puis de tracer cette déformée - amplifiée. (l'amplification nulle donnant le dessin de la structure non déformée !)

'N T1 T2 MT M1 M2' permet d'obtenir les diagrammes :

N : d'effort normal MT: de moment de torsion
T1, T2: d'effort tranchant M1, M2: de moment de flexion
1 dans le plan principal d'inertie 2 dans le plan perpendiculaire.

.../.

'S I1 I2' permet la définition des propriétés mécaniques attribuées.

Les diagrammes sont dessinés suivant "la vue système", c'est-à-dire par défaut : vue de face, vue de dessus.

Les fonctions "batch" : DFORME, DIAGRM, créent des entités EUCLID correspondant à ces visualisations.
On peut alors obtenir une vue avec élimination des parties cachées nécessitant la création de beaucoup de points EUCLID. Ce n'est donc réellement utilisable que dans un contexte où les seuls programmes de vision sont présents, les résultats du calcul de structures ayant été relus sur mémoire de masse.

Il y a dans la pratique séparation entre l'interrogation de la structure, élément par élément, pour explorer les résultats et la représentation sèche des résultats finaux.

CHAPITRE V

LE CALCUL DE STRUCTURES

STRUCTURE DE DONNEES D'EUCLID

EUCLID a pour but de définir, manipuler, représenter des entités géométriques.

Ces entités sont de nature variées :

points, segments, facettes, figures.

et peuvent avoir des attributs variés :

opacité, connexité.

Ces entités sont construites comme une réunion d'autres entités crées précédemment. Cette structuration des entités géométriques est implémentée sous la forme d'une structure de listes réalisée par des tableaux contenant des éléments pointant sur des blocs d'éléments plutôt que sur les antécédents et suivants. Cette façon de procéder a été choisie de préférence à des procédés plus élaborés à base de pointeurs et d'indications de début et fin car l'exploration de la structure n'était envisagée que du général au particulier. Il s'est avéré ensuite que les modifications par tassements se révélaient largement suffisantes.
Les entités sont réparties en 2 tableaux. Le premier tableau pointant sur lui-même ou le second. Le deuxième pointant sur les tableaux de coordonnées.

PROGRAMMATION

Du point de vue du programmeur tout se passe (ou presque) comme si on avait ajouté au FORTRAN un type "géométrique" pouvant prendre des attributs variés.

CALCUL DES STRUCTURES - GENERALITES

La méthode utilisée est la méthodes des éléments finis. Rappellons-en brièvement le principe :
Pour le calcul des structures composées de barres et de poutres la méthode des éléments finis n'est qu'une généralisation de la méthode energétique bien connue des ingénieurs des bureaux d'études. En résumé, si l'on connait des déplacements des extrémités, on sait calculer l'énergie accumulée dans la barre ou la poutre. On sait aussi si les efforts ne sont

.../...

appliqués qu'en ces extrémités, calculer le travail des forces extérieures.

Un principe général de la mécanique affirme alors que la différence L entre l'énergie de déformation et le travail est stationnaire à l'équilibre. Il suffit donc de calculer cette fonctionnelle et de chercher son extremum pour avoir l'état d'équilibre défini par les déplacements des extrémités.

On sait alors calculer les efforts aux extrémités comme dérivées de l'énergie par rapport aux déplacements.

Remarquons que nous ne nous sommes pas préoccupés du caractère iso ou hyper statique de la structure. Or depuis longtemps les ingénieurs savent se ramener au cas statique pour faire des calculs plus simples à la main. qu'avons nous gagné ?
Pour le voir, regardons de plus près la démarche du calcul d'une structure hyperstatique : on choisit des inconnues hyperstatiques pour se ramener au cas isostatique et après on résoud un problème de compatibilité pour trouver la valeur des inconnues "hyperstatiques".

Or résoudre le problème de compatibilité c'est sur un sous-ensemble judicieux, résoudre le même problème que celui résolu par la méthode des éléments finis.

Les techniques utilisées sont donc les mêmes mais l'ingénieur utilise ses connaissances pour réduire au minimum les calculs nécessaires. Ceci est tout à son honneur mais malheureusement les machines savent très mal "choisir". La méthode des éléments finis leur évite ce choix et leur convient parfaitement. En plus il y a des problèmes où l'on ne peut plus "choisir". Prenons par exemple le calcul d'uen plaque quadrangulaire. Dans ce cas, la déformée n'est pas défini par les déplacements des sommets de la plaque mais dépend en toute rigueur d'une infinité de variables qui sont, disons pour simplifier, les déplacements des bords de la plaque. Il n'est donc plus question de choisir mais d'abord de limiter ! Dans la méthode des éléments finis, nous nous ramenons au problème précédent en approximant seulement la déformée par une fonction simple à calculer qui

.../...

est un polynôme et qui ne dépend plus que des valeurs des déplacements (et de quelques unes de leurs dérivées) aux sommets de la plaque.

Il n'y a plus dans ce cas de sous-structure isostatique et la généralité de la méthode des éléments finis est <u>la seule</u> qui soit satisfaisante dans ces cas.

EXPRESSIONS DE L'ENERGIE DE DEFORMATION EN FONCTION DES DEPLACEMENTS

Comme nous venons de le voir le point de départ est pour chaque élément fini de calculer les déplacements à l'intérieur de l'élément en fonction des déplacements des sommets de cet élément par interpolation polynomiale. Pour simplifier, considérons qu'il y a n sommets que l'on appelle des <u>noeuds</u> et soient $\vec{u_i}$ les vecteurs déplacements des sommets. Le déplacement $\vec{u}(x)$ au point x de l'élément sera défini alors par une formule dont le détail importe peu et qui sera en tout cas linéaire par rapport aux u_i

$$U_{i,j}(x) = \sum_{i=1}^{n} \varphi_{i,j}(x) \; U_{i,j} \qquad j = 1, 2, 3$$

Connaissant les trois composantes U.,1, U.,2, U.,3, du déplacement en $\vec{x}$ nous pouvons calculer les déformations

$$\varepsilon_{j\,k} = \frac{1}{2}\left(\frac{\partial U_k}{\partial x_j} + \frac{\partial U_j}{\partial x_k}\right)$$

Les déplacements $U_{i,j}$ des noeuds y apparaitront encore de façon linéaire.

Dans le cadre de l'élasticité linéaire, nous obtenons alors les contraintes $\sigma_{j,k}$ dans l'élément par :

$$\sigma_{j,k} = \frac{E}{1+\sigma}\left(\varepsilon_{jk} + \frac{\sigma}{1-2\sigma}\left(\sum_{l=1}^{3} \varepsilon_{el}\right)\delta_{jk}\right)$$

Finalement les contraintes $\sigma_{j,k}$ en chaque point $\vec{x}$ de l'élément seront encore une fonction linéaire des déplacements des noeuds comme les déformations ε_{jk}. Or on sait que l'énergie de déformation $\mathcal{E}$ est donnée par :

$$\mathcal{E} = \sum_{j,k} \iiint \sigma_{jk} \; \varepsilon_{jk} \; dv$$

Or l'intégrale est étendue au volume limitée par l'élément fini. Les contraintes σ_{jk} et les déformations ε_{jk} dépendant linéairement des déplacements des noeuds l'énergie $\mathcal{E}$ sera une fonction <u>quadratique</u> de ces mêmes déplacements.

.../...

L'énergie de déformation étant une grandeur additive en considérant toute la structure alors on ajoute les contributions de chaque élément et cette forme quadratique peut s'écrire :

$$\mathcal{E} = \frac{1}{2} \tilde{Q} K Q$$

où :

Q est le vecteur fermé des déplacements nodaux de tous les noeuds de la structure ;

K est une matrice symétrique définie positive appelée <u>la matrice de rigidité</u>

On peut écrire de la même façon le travail virtuel des forces extérieures sous la forme :

$$\tilde{F} Q$$

F vecteur des forces nodales généralisées.

On a donc à minimiser :

$$\mathcal{L}(Q) = \frac{1}{2} \tilde{Q} K Q - \tilde{F} Q$$

Un calcul simple montre que les déplacements nodaux Q minimisant cette expression sont solutions du système <u>linéaire</u>

$$KQ = F$$

<u>RECUPERATION DES CONTRAINTES</u>

La solution du système linéaire donne les déplacements des noeuds (et éventuellement les rotations etc...). Pour récupérer les contraintes dans l'élément, on remarque que pour chaque élément KQ, produit matriciel de la matrice de rigidité de l'élément par le vecteur des déplacements nodaux donne le vecteur des contraintes nodales. Calculer K, matrice de rigidité de l'élément, est donc fait deux fois :

- pour calculer les coefficients de K (appel par CALCOF)
- pour calculer les contraintes aux noeuds (appel par CALEFF).

Ensuite on peut interpoler le champ des contraintes et le représenter.

.../...

ALGORITHME DU FRONT

Pour calculer la solution de ce système linéaire, il faut :

1°) former la matrice K en calculant les contributions de chaque élément. Ceci revient à calculer d'abord la "petite"matrice de rigidité "locale" K_i de l'élément i puis à "éclater" cette matrice dans la "grande" matrice K. Cette opération s'appelle l'assemblage.

2°) on doit ensuite une fois la matrice K constituée résoudre le système linéaire. Cette résolution peut se faire de multiples façons. Comme dans tous les programmes d'éléments finis existants, nous avons choisi une méthode directe basée sur l'élimination (méthode de GAUSS).
Or on voit facilement que pour éliminer une inconnue dans le système linéaire il n'est pas nécessaire que ce système soit complètement écrit. Il suffit en effet que la ligne associée à cette inconnue soit complète ainsi que la colonne. Or les coefficients de cette ligne et de cette colonne sont obtenus en ajoutant les contributions des matrices de rigidité des éléments finis qui ont pour sommet le noeud correspondant à cette inconnue.
Or en calculant successivement les contributions de chaque élément, nous pouvoins, par un tableau auxiliaire par exemple, savoir quand un noeud est complètement "encerclé" et alors procéder à l'élimination.

Nous pouvons même orienter l'ordre dans lequel les éléments finis sont traités pour avoir le nombre le plus réduit possible de noeuds en "cours d'encerclement". L'ensemble de ces noeuds s'appelle le front et c'est cette façon de procéder dite algorithme du front que nous utilisons.

LES ELEMENTS FINIS UTILISES

Le but du projet étant essentiellement pédagogique nous n'avons pas cherché à faire utiliser des éléments finis de degré élevé qui permettent une meilleure précision mais qui sont complexes à se représenter et à utiliser. Sauf dans le cas des plaques et des poutres, l'interpolation utilisée est donc la plus simple possible c'est-à-dire linéaire. Nous avons comme éléments de ce type :

- la barre (traction - compression et torsion)
- l'élément triangulaire de membrane
- le tétraèdre.

.../...

Les inconnues nodales sont simplement les déplacements du noeud dans les 3 directions spatiales.
Pour les poutres en flexion on rajoute à ces inconnues les rotations dans l'espace du noeud et l'interpolation devient cubique dans chaque plan de flexion.

Enfin, les plaques posent des problèmes de raccordement qui imposent une interpolation de degré plus élevé.

EXTENSIONS POSSIBLES

En fait pour ajouter un nouveau type d'élément à calculer les seules modifications à faire sont :

1°) Prévoir ce type d'élément et le calcul des coefficients dans CALCOF/CALEFF

2°) Mettre à jour le nombre maximal de variables par noeuds (NMAX)

Sinon la démarche suivie et les programmes sont absolument généraux et peuvent s'adapter à d'autres applications.

EXTENSION DE LA STRUCTURE DE DONNEES D'EUCLID

Il s'agit d'attribuer à des entités purement géométriques les quelques propriétés nécessaires pour en faire des éléments structuraux. Une solution, testée sur un logiciel bi-dimensionnel interactif sur PDP 15, consiste à associer à chaque élément un numéro de type, parmi les 15 types définis préalablement. (Cela nécessitant une extension de 4 bits du "code élément").

Un peu couteuse en place, cette solution, qui permet la modification aisée du type d'un élément, demande de prévoir les groupes distincts ayant des propriétés identiques que l'on pourrait vouloir changer en bloc après coup, ainsi que d'avoir noté quels groupes avaient les mêmes propriétés, pour un éventuel changement global ...

Il manque, en un mot, à cette solution une structuration en figures et sous-figures qui permet de manipuler en bloc un ensemble ou un sous-ensemble indifféremment.

En raison de l'évident parallélisme entre la structuration que va instinctive-

.../.

ment donner l'utilisateur aux entités géométriques pour les besoins de leur propre génération, et celle qui est nécessaire aux propriétés associées, on a choisi la solution la plus cohérente : structurer les données à l'image de la structure géométrique pour y introduire des distinctions en sous-figures nécessitées par les seuls attributs ou, cas extrême, à faire quelques figures d'éléments uniques, pour certains éléments exceptionnels.

On convient ainsi d'associer à chaque niveau de figure un ensemble de propriétés, que tous les éléments constitutifs de cette figure partageront. Par ailleurs, il faudra pouvoir attribuer globalement des propriétés. Par exemple, une section pour un ensemble de membranes, ou un module d'élasticité pour une ossature complète. Cette ou ces propriétés, manquantes dans la liste des propriétés attribuées aux membranes, seront à rechercher dans la liste de la figure mère : l'ossature.

Enfin, il est parfois nécessaire de pouvoir attribuer en plusieurs fois les propriétés d'une même figure.

Filiation des propriétés

Pour réaliser ces spécifications, on a créé un élément EUCLID particulier choisi parmi les "paradoxaux".
Il comporte trois parties différentes :

- un code "paradoxal" qui permet son identification ;
- un pointeur "latéral" qui contient l'adresse, dans la table des propriétés associées, du début de la liste des propriétés ;
- un pointeur "arrière" qui renvoie, dans la taille des éléments EUCLID, sur la figure mère (donc nul pour la figure de l'ensemble).

Pour retenir sa place, on place ce paradoxal en tête de liste lors de la création d'une figure (qui possède donc, à l'insu de l'utilisateur, n + 1 éléments).

C'est également à la création d'une figure qu'on affecte les pointeurs arrières de chacune de ses sous-figures.

Il peut y avoir "conflit de paternité" si le pointeur arrière est déjà positionné. On considère en effet que c'est une erreur de conception que de vouloir simultanément affecter un élément à deux ensembles distincts aux propriétés

.../.

sans doute contradictoires !. Autrement dit, cette "filiation" des propriétés suppose une structuration en arbre.

Rangement des propriétés :

Les propriétés sont rangées dans une structure de données annexe, derrière un mot dit "descripteur" composé de deux parties :
- une zone de 20 bits décrivant combien et quelles propriétés sont spécifiées ;
- un pointeur "suite" permettant une extension, puisqu'un opérateur n'est pas obligé de préciser en une seule fois toutes les propriétés d'une sous-figure.
  Pour une application où 20 propriétés seraient insuffisantes, ce pointeur permettra un regroupement des propriétés en 2 groupes de 19, ou 4 de 18, ...

Syntaxe

C'est un module en fait indépendant de l'application qui attribue des propriétés connues par leur rang.
CALL ATTRIB (OBJET, liste des couples : n°/valeur)

où OBJET est nécessairement une figure (quitte à écrire :
OBJET = FIGURE (OBJET)).

Pour ne pas contraindre l'utilisateur à mémoriser que la section correspond à la quatrième propriété, le module d'élasticité à la troisième, etc .... (Convention arbitraire mais nécessaire), on peut remplacer le n° de propriété par un alphanumérique : 'E', 's' ....

L'usage interactif d'ATTRIB consiste à choisir le nom de la propriété sur un menu, et à donner sa valeur.

Les intitulés de type "BOIS", "ACIER", "BETON", ... , économisent la frappe de nombres toujours identiques.

Les intitulés de type "PROFIL", "CATALOGUE", ..., font appel à des modules de calcul géométrique de sections et de moments d'inertie en accédant à un dictionnaire de propriétés des éléments normalisés.

.../.

Résultats du calcul

Enfin pour le stockage des coëfficients de la matrice de rigidité, du second membre du système linéaire et de sa solution, nous avons défini une interface qui permet de s'affranchir des contraintes du système utilisé et d'autre part une implémentation sur le système.

Définition de l'interface

Tout ce qu'il est nécessaire de savoir sur l'interface est défini par les spécifications suivantes.
Pour chaque sous-programme on donne les paramètres et l'effet.
Le paragraphe but à valeur de commentaire.

- gestion des coëfficients de la matrice de rigidité
  fonction WRCOEF
  but : écrire un coëfficient
  paramètres : entiers IE et IV, réel V
  effet : erreur si IE et IV $\notin$ [1, NBVAR]
  paramètres inchangés

  fonction WACOEF
  but : additionner un coëfficient
  paramètres : entiers IE et IV, réel V
  effet : erreur si IE et IV $\notin$ [1, NBVAR]
  paramètres inchangés

  fonction RDCOEF
  But : lire un coëfficient
  paramètres : } idem, mais
  effet : }
  le paramètre V est changé de la manière suivante :
  1) si aucun appel précédent avec les mêmes valeurs de IE et IV, soit à WRCOEF, soit à WACOEF, V = 0
  2) la séquence : WRCOEF (IE, IV, V1); .... ; RDCOEF (IE, IV, V2) donne :
  V2 = V1
  3) la séquence : RDCOEF (IE, IV, V1) ; .... ; WACOEF (IE, IV, V2) ; ... ; RDCOEF (IE, IV, V3) donne :
  V3 = V1 + V2
- gestion du second membre et de la solution
  il existe des modules correspondant à ceux ci-dessus, avec des spécifications parallèles, il s'agit de :
  WRSM (I, V), WASM (I, V), RDSM (I, V).

.../.

CONCLUSION

Intérêt d'EUCLID dans le calcul des structures.

Les programmes d'éléments finis existants (NASTRAN, STRUDL, ASKA) avec lesquels nous n'avons nullement la prétention de rivaliser tant du point de vue de la généralité que de celui des performances - NASTRAN représente 150 000 instructions FORTRAN, contre 25 000 pour notre solution actuellement- ont tous des méthodes d' "entrée-sortie" fiables mais lourdes.

Pour définir une structure architecturale, il leur faut définir et numéroter les noeuds (fixés ou non), définir les barres comme couples de noeuds, leur affecter les propriétés nécessaires. Ce travail très lourd aboutit à des bordereaux de perforation sur lesquels on fini par ne plus rien voir, ce qui rend la vérification des données d'autant plus difficile.

Le système présenté ici ne présente pas ces inconvénients, puisqu'il dépend du système EUCLID qui permet la définition et la représentation d'objets géométriques. On crée les structures comme des constructions géométriques visualisables : les barres-poutres sont des segments qui joignent des points créés en perspective. On attribue ensuite à ces êtres géométriques des propriétés mécaniques et des charges immédiatement symbolisées par des dessins (avec une échelle numérique).

C'est encore EUCLID qui permet, après calcul, de visualiser les déplacements et les contraintes.

L'utilisateur n'a rien d'autre à fournir que ce qui est nécessaire à la définition de la structure:
- la topologie,
- les propriétés mécaniques,
- les charges,

et ce de façon interactive, sans codification préalable.

BIBLIOGRAPHIE SOMMAIRE

Principales publications de l'équipe

1/ Le langage EUCLID ou comment définir, manipuler et dessiner des corps tridimensionnels

Architecture Mouvement Continuité N° 24 - 1972

2/ EUCLID, langage graphique tridimensionnel

Journées graphiques de l'IRIA - Mai 1972

3/ Techniques de visualisation en architecture et urbanisme

Publication I.E. - 1974

4/ Automated aids for the design of mechanical parts : EUCLID

Congrès SME/ACM - Chicago - 1975

5/ Calcul des structures sur ordinateurs

CSFTA - 1978

Autres références de base

1. ALEXANDER : Notes on the synthesis of form - 1964
2. NEGROPONTE : Computer aids to design and architecture - 1975
3. EASTMAN : Spatial synthesis in Computer Aided Building design-1975
4. FRIEDMAN : Pour une architecture scientifique - 1971
5. NOTES METHODOLOGIQUES EN ARCHITECTURE ET EN URBANISME (Institut de l'Environnement)

   1 - Analyse des données

   2 - Allocation spatiale