1/91

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

2EX

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : Génie-Chimique

PROJET DE FIN D'ETUDES

SUJET

Contribution à l'élaboration
d'un logiciel
de simulation de procédés

Proposé par : Mr. M. BENIDDIR

Etudié par : Melle L. BENABBAS
Mr T. M. M. OUNANE

Dirigé par : Mr. M. BENIDDIR

PROMOTION : JUIN 1991

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT :

PROJET DE FIN D'ETUDES

SUJET

Contribution à l'élaboration
d'un logiciel
de simulation de procédé

Proposé par : **Etudié par :** **Dirigé par :**

PROMOTION :

E.N.P. 10. Avenue Hacene Badi – EL-HARRACH – ALGER

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

SOMMAIRE

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

ملخص :

ABSTRACT : Our work consists in a small contribution to develope a software package designed specifically for chemistry engineering processes. We developed library permits the dynamic simulation of chemical process and a user hightly interactive package for parameters introduction and graphics.

Resume :

Notre travail est une modeste contribution au developpement d'un logiciel de simulation de procedes chimique. Nous avons developpe une librairie d'utilitaires permettant la simulation de procedes chimiques ainsi qu'une interface utilisateur interactive facilitant l'introduction des parametres et la presentation graphique ou dans un fichier des resultats de la simulations.

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

INTRODUCTION

L'évolution de l'outil informatique et des méthodes numériques, a donné une nouvelle allure au développement scientifique et technologique. De nos jours la modélisation et la simulation sont enseignées dans tous les domaines de l'Engineering (Génie-chimique, Génie-civil, Hydraulique, Electronique ...).

La recherche de modèles à partir de données expérimentales (modèles de représentation) ou à partir d'équations mathématiques et de connaissances physiques (modèles de connaissances), et la simulation, qui est la manipulation de ces modèles, sont des activités fondamentales en sciences expérimentales (physique, chimie, biologie ...) et en Engineering. Cette activité traditionnelle connaît un essor important depuis la généralisation de l'outil informatique, associant la puissance, la rapidité de calcul et la maîtrise de l'information à la commodité de la représentation graphique. Actuellement, on forme les opérateurs sur des logiciels de simulation; la recherche et la conception de procédés sont faites à l'aide de logiciels de C-A-O utilisant la simulation; les unités pilotes optimisant le fonctionnement des usines réelles sont remplacées par des logiciels. (Gain de temps, réduction de coût et élimination de risque).

Il existe, actuellement, des stations de travail " Work-station" offrant une grande puissance de calcul et des possibilités accrues en entrées / sorties, (haute résolution graphique : 1280 x 1024 pixels) et une grande capacité de stockage des informations (jusqu'à 400 Mo en disque dur à accès très rapide). Il existe aussi de nombreux logiciels de simulation en mécanique des fluides et dans le traitement des phénomènes de transport; simulation des procédés dynamiques, systèmes experts,bases de données, C-A-O, D-A-O, gestion des projets ...

Notre but est de contribuer à la réalisation d'une bibliothèque d'utilitaires de simulation de procédes chimiques,dans un langage de développement permettant l'extension et l'utilisation de toutes les ressources de la machine. Et la contribution à la réalisation d'un logiciel de simulation modulaire, interactif.

Ainsi, notre Projet de Fin d'Etude s'articule comme suit :

* Présentation de quelques généralités et quelques définitions.
* Adaptation des procédures du DYFLO [1] au langage C et réalisation d'une librairie d'utilitaires pour la simulation en génie-chimique.
* Réalisation d'une procédure de simulation d'une colonne de distillation selon le modèle proposé par DYFLO que nous avons généralisé.
* Réalisation d'une interface utilisateur interactive pour la manipulation des paramètres et pour la réalisation d'un simulateur de procédés utilisant une approche modulaire simultanée.

I. GENERALITES .

I.1. Introduction :

Les objectifs du génie des systèmes et procédés sont multiples :

* Conception et dimensionnement des systèmes.
* Extrapolation et interpolation des modèles.
* Optimisation des installations existantes par une meilleure compréhension de leur fonctionnement. Cette démarche nécessite couramment en préalable, une modélisation,une simulation,voire une identification de paramètres.
* Conduite optimale des procédés. Dans ce cas, on utilise les techniques d'automatisation et de commande permettant la maîtrise de l'évolution dans le temps du procédé.

I.2. Théories de modèles :

La théorie des modèles vise à la prévision des performances de composants, de machines, de systèmes et de procédés.

I.2.1. Définition d' un modèle .

Un modèle est la représentation d'un système ou d'un procédé par un ensemble d'équations ou par un montage expérimental permettant la simulation des conditions de fonctionnement et conduisant à l'établissement de lois prévisionnelles. Dans les cas les plus difficiles, ces lois ne seront que des corrélations empiriques entre les grandeurs de sortie (rendement...) et les paramètres d'entrée (conditions de fonctionnement...)

I.2.2. Types de modèles :

a. Modèles mathématiques :

Ils décrivent le procédé par un ensemble d'équations qui résultent des lois traduisant les effets mécaniques, thermiques, physico-chimiques..., couplés dans le processus modélisé. Dans certains cas, les équations se résument à des relations empiriques.

Les modèles mathématiques expriment couramment :

* Les bilans de matière, de quantité de mouvement, d'énergie , d'éxergie.
* Les couplages entre phénomènes de transfert, réactions chimiques et transformation de l'énergie.

On peut distinguer deux types de modèles mathématiques.

1. Modèle statique .

Un modèle statique ne prend pas en considération les variations dans le temps des paramètres du processus. L'élaboration d'un modèle statique du processus est précédé de l'analyse de sa nature physico-chimique, du but auquel il est destiné et des équations fondamentales qui décrivent ses particularités en tant que processus type.

On met ensuite en évidence les paramètres d'entrée et de sortie du processus.

Le modèle statique d'un processus type doit être construit en tenant compte de tous les régimes technologiques de travail possibles concernant le système type.

2. Modèle dynamique

L'élaboration d'un modèle dynamique se ramène à obtenir les caractéristiques dynamiques du processus, c'est à dire à établir les relations entre ses variables temporelles fondamentales.

Les caractéristiques dynamiques peuvent être obtenues soit par la théorie, soit par l'expérience, soit par la combinaison des deux méthodes.

L'obtention, par l'expérience, des caractéristiques dynamiques est fondée sur une expérimentation consistant à causer une perturbation à l'entrée du système étudié et à analyser dans le temps, à sa sortie, le passage de cette perturbation à travers le système.

Les limitations de la modélisation mathématique proviennent de la difficulté d'identifier les effets régissant le processus ou dans la convergence des méthodes iteratives de calcul (expérience mathématique) ou dans le manque de données fondamentales par exemple les données physico-chimiques.

b. Modèles analogiques

Ces modèles donnent du procédé une représentation dans un domaine scientifique voisin par transposition de variables. Il existe de nombreux types d'analogie : mécanique, électrique, hydraulique, mais les analogies électriques sont les plus courantes.

c. Modèles homologiques :

Ces modèles substituent, couramment, aux fluides réels des fluides modèles. Les lois de similitude totale ou partielle permettent alors une expérimentation sur maquette, dans des conditions aisément maitrisables de température ou de pression. La maquette est souvent un instrument d'étude et d'acquisition de données qui permet la visualisation du phénomène, la la mise en exergue d'un aspect précis du procédé, la mise au point du modèle. Cette pratique est fréquente dans le cas des études hydraudynamiques et souligne le caractère spécifique des études sur maquette qui privilégient souvent un aspect particulier du phénomène étudié, par exemple l'examen des conditions d'écoulement.

I.2.3. Identification de paranètres .

Dans l'étude des systèmes et des procédés intervient la représentation mathématique supposée connue. La connaissance de la linearité ou non linearité du modèle est importante mais il faut de plus connaître les valeurs numériques des coefficients qui interviennent dans le modèle.

Pour que les résultats théoriques obtenus à partir du modèle coïncident au mieux avec ceux fournis par le système réel (validation du modèle), il faut que le modèle soit le plus vraisemblable possible. La fidélite de la représentation est aisée pour des systèmes simples àparamètres bien localisés et facilement mesurables.

Pour les systèmes plus complexes, la concordance entre les résultats théoriques et les résultats expérimentaux n'est acceptable que sur un domaine limité de variations des paramètres d'entrée. Si on veut changer le domaine de validité du modele, il faut au moins ajuster les valeurs numériques de différents coefficients.

Dans certains cas, l'établissement du modèle mathématique est realisé à partir d'une " boite noire " en utilisant des hypothèses et en s'aidant d'observations qui permettent de donner une représentation des phénomènes. L' identification permet alors de calculer des paramètres d'entrée du modèle pour avoir coïncidence avec les observations.

I.2.4. Conclusion .

L'utilisation du modèle en simulation est souvent une étape indispensable en vue de valider le modèle; puis la simulation permet aussi de faire des études de sensibilité par rapport à divers paramètres ou à des variables d' état. L' optimisation du système ou du procédé devient alors possible.

I.3. GENERALITES SUR LA SIMULATION.

On distingue deux types de simulation : la simulation physique et la simulation mathématique.

I.3.1. Simulation physique .

Dans la simulation physique, l'étude d'un phénomène donné s'opère en le reproduisant a des échelles différentes et en analysant l'influence de ses particularités physiques et de ses dimensions lineaires. L'expérience est directement conduite sur le processus physique étudié.

La simulation physique se ramène à reproduire la constance des critères de similitude caractéristiques, et dans le modéle et dans le système étudié.

I.3.2. Simulation mathématique .

C'est la manipulation du modèle mathématique avec différentes excitations ou sollicitations. Elle évite le recours systématique à l'expérimentation pour des raisons évidentes de coût, mais aussi de sécurité.

Seul ce type de simulation sera developpé ci-après.

I.3.3 Simulation de procédés .

La simulation de systèmes ou procédés consiste à calculer les variables de fonctionnement à partir de la description mathématique fournie par la modélisation. Généralement, le problème consiste à résoudre un système d'équations simultanées linéaires ou non.

La simulation de procédés est née et s'est développée dans les industries de la chimie et de la pétrochimie. Les caractéristiques et les réactions des produits utilisés étant souvent relativement bien définies; les procédés mis en oeuvre dans le secteur se sont, en effet, prêtés à une mise en équation des phénomenes observés et à un traitement automatisé des calculs.

D'autre part, la nature très souvent séquentielle de ces procédés conduit naturellement a des modèles dont les équations sont très peu interdépendantes et donc facilement solubles au moyen de méthodes itératives. Celle ci présentent l'avantage d'être à la fois bien adaptées à la programmation et de conception assez simple et se sont donc développées les premières.

Les procédés à simuler peuvent être très variés et la complexité de la simulation dépend bien évidemment de la structure des modèles obtenus lors des phases de modélisation et d'identification.

Néanmoins, trois caractéristiques générales s'en dégagent :

- Leur structure algèbro-différentielle linéaire ou non.
- Leur grande dimension due au nombre élevé de variables, ainsi qu'au nombre d'équations liant ces variables.
- La présence éventuelle d'équations à retard pur et d'équations logiques.

I.3.5. Différents types de simulation mathématique .

Selon la nature du système d'équations algèbro-différentielle décrivant le procédé étudié, on distingue :

a. Simulation statique .

On dit qu'une simulation est statique si le modèle représentant le système est du type statique, défini dans le chapitre précédent.

b. Simulation dynamique :

On dit qu'une simulation est dynamique si le modèle représentant le système est du type dynamique, défini dans le chapitre précédent.

c. Simulation continue :

On dit qu'on a une simulation continue si la méthode de résolution, choisie, du système algèbro-différentie nous permet d'avoir des résultats continus et non en paliers.

II.4. Conclusion :

L'étude des procédés industriels, après leur modélisation et leur identification, nécessite souvent d'en effectuer une simulation pour mettre en évidence et résoudre un certain nombre de difficultés: validité du modèle, stabilité, régulation, commande.

En outre, la simulation peut être une démarche pédagogique pour apprendre à un "opérateur" à maitriser le comportement d'un procédé: sur le procédé simulé, l'instructeur peut appliquer certaines configurations de signaux d'excitation pour tester les réactions de l'opérateur en face d'une situation qu'il ne connait pas encore, et l'inviter ainsi à une reflexion sur les actions à prendre face à une telle situation.

II. PRESENTATION DU DYFLO .

Le DYFLO est un ensemble de procédures développées , en FORTRAN IV sur UNIVAC 1108 , par R.G.E. FRANKS en 1971 aux U.S.A . Ces procédures permettent la simulation des opérations de procédés courants (évaporation , condensation , accumulation , . . .) . Elles forment une librairie de fonctions facilitant la programmation pour la simulation de procédés complexes .

Les modèles adoptés par chaque procédure vont être présentés. L'utilisateur peut modifier , adapter ces modèles selon ces besoins et enrichir cette librairie de procédures .

La résolution des équations algébriques et différentielles, constituant le modèle , est assurée par un ensemble de procédures numériques présentees dans un autre programme , nommé INT .

Les listings, en FORTRAN IV , de toutes ces subroutines constituent le DYFLO, présenté par R.G.E. FRANKS [1] .

II.1. Procédures numériques de INT .

II.1.1. Résolution d'équations algébriques .

a . Par la méthode de substitution partielle .

La procédure utilisant cette méthode est nommée CPS .

exemple d'utilisation : pour resoudre l'équation

```
X=(5*Y**2+3*SQRT(X)-8*X**.8)**.36
```

L'appel en FORTRAN se fera comme suit :

```
      X=5.0
5     CONTINUE
      XC=(5*Y**2+3.*SQRT(X)-8*X**.8)**.36
      CALL CPS(X,XC,.5,NC)
      IF (NC.NE.1) GOTO 5
6     CONTINUE
```

b . Par la méthode de WEGSTEIN .

La procédure utilisant cette méthode est nommée CONV .

Exemple d'utilisation : pour résoudre l'équation de l'exemple précédent l'appel se fera comme suit :

```
      X=5.0
5     CONTINUE
      XC=(5*Y**2+3.*SQRT(X)-8*X**.8)**.36
      CALL CONV(X,XC,1,NC)
      GOTO (6,5),NC
6     CONTINUE
```

II.1.2. Résolution d'équations différentielles .

a . Par la méthode de RUNGE-KUTTA .

C'est la méthode la plus utilisée en simulation grâce à sa stabilité. Elle est representée par les deux subroutines INT et INTI. La première pour l'intégration proprement dite et la seconde pour le choix du pas et de l'ordre d'intégration (à chaque appel). Elles regroupent les quatres ordres d'intégration : RUNGE-KUTTA d'ordre 1 (EULER) ; d'ordre 2,3 et 4.

Mode d'utilisation : Pour résoudre l'équation différentielle :

d(N Xc)/dt = Fr Xcr + R - V Yc .

On traduit en écriture FORTRAN par : **DNXC=FR*XCR+R-V*YC**

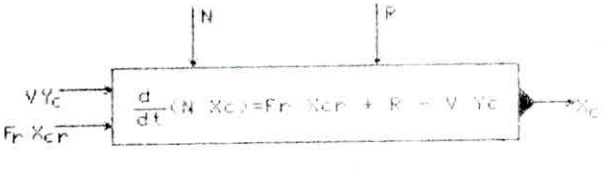
Et on appelle INT comme suit : **CALL INT(NXC,DNXC)**

ou **XC=NXC/N**

L'appel de INTI, pour chaque pas,peut se faire comme suit :

CALL INTI(T,DT,2)

Ceci est représente dans le schéma du modèle par :



SCHEMA DU BLOC D'INTEGRATION

II.1.3. Génération de la dérivée d'une variable .

Ceci est realisé par la subroutine DER .

Mode d'utilisation : Y=F(X)
CALL DER(Y,X,DYDX,I)

ou I representé le nombre d'appels .

N.B : Le programme INI comporte d'autres procédures numériques qui n'ont pas été présentées :

FUN1 : Interpolation d'une fonction à 2 dimensions.

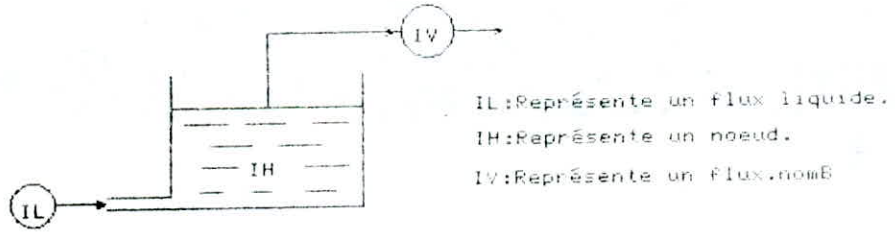
FUN2 : Interpolation d'une fonction a 3 dimensions.

PRNTF; PRNTRS : Sorties des résultats.

II.2. Structure du DYFLO :

En simulation de procédés on a affaire,généralement, à des flux et à des noeuds. Un flux peut être liquide ou vapeur, rarement solide. Un noeud est une quantité de matière stagnante (qui n'est pas en mouvement).

Exemple :



Les flux et les noeuds ont certaines propriétés extensives : telles que la composition et la température ; le débit pour un flux ou la quantité pour un noeud, ... etc.

Il est nécessaire de numéroter les flux et les noeuds et d'avoir pour chaque numéro, de flux ou de noeud, une matrice de propriétés. Pour cela on a défini une matrice SIRM(IS,IP) ou IS est le numéro du flux ou du noeud (IS=1,...,MaxFlux; MaxFlux=300), et IP le numéro de la propriété :

IP=1,..,20 : **Composition** X1,..,X20, généralement en fraction molaire.

IP=21 : **Debit** en (moles/min) ou retenu ,pour un noeud,en(moles).

IP=22 : **Temperature** en (C).

IP=23 : **Enthalpie** (PCU/MOLE).

IP=24 : **Pression** (atm).

Nous avons défini ainsi une matrice de flux SIRM(300,24), ses dimensions peuvent être modifiées selon nos besoins.

Dans un flux, il y a un certain nombre de composés dont on doit connaître les propriétés. C'est pourquoi, on a défini une matrice de données DATA(IC,ID) où IC est le numéro du composé (IC=1,..,20 correspondant a IP=1,...,20 des compositions de SIRM) et ID l'indice de la propriété :

ID=1 : Coefficient d'Antoine C1.

ID=2 : Coefficient d'Antoine C2.

ID=3 : Coefficient d'Antoine C3.

ID=4 : Coefficient d'enthalpie vapeur Av.

ID=5 : Coefficient d'enthalpie vapeur Bv.

ID=6 : Chaleur latente a 0 C.

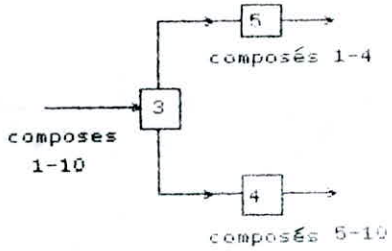
ID=7 : Coefficient d'enthalpie liquide Al.

ID=8 : Coefficient d'enthalpie liquide Bl.

ID=9 : Activité Nu.

Dans un flux donné , nous avons besoin de savoir quels sont les composés qui interviennent parmi ceux définis dans un système (dans DATA). Pour cela on définit deux variables NCf et NCL représentant l'indice du premier et du dernier composé , respectivement , intervenant dans un flux.

Exemple :



Pour simuler le nœud 3 on a NCF=1;
NCL=10;

Pour simuler le nœud 4 on a NCF=5;
NCL=10;

Pour simuler le nœud 5 on a NCF=1;
NCL=4;

II.3. Procédures du DYFLO .

II.3.1. Subroutines ENTHL(I) et ENTHV(I) .

Parfois , nous avons besoin de calculer l'enthalpie molaire (liquide ou vapeur) d'un flux dont nous connaissons la température et la composition, grâce aux deux subroutines correspondantes.

La capacité calorifique varie linéairement avec la température :

$$Cp = a + b\,T$$

où a et b sont des constantes.

L'enthalpie s'exprime, en fonction de la température, par l'équation :

$$H = \int_0^T Cp\,dT = \int_0^T (a + b\,T)\,dT \qquad d'ou \qquad H = a\,T + b/2\,T^2 + c$$

Où c est la constante d'intégration.

- c=0 pour les liquides ;
- c=Lambda pour les vapeurs (Chaleur latente a 0 C).

L'enthalpie vapeur : **Hv = Lambda + (Av + Bv T) T** (PCU/mole)

L'enthalpie liquide : **hl = (Al + Bl T) T** (PCU/mole)

Ou :

A : Capacité calorifique a 0 C.

B : 1/2 du Coefficient de température dans la capacité calorifique.

Pour un mélange idéal liquide : **hl = T SUM Xi (Ali + Bli T)** .

Pour un mélange vapeur : **Hv = T SUM Xi (Avi + Bvi T) + Lambda** .

Ceci est traduit par les deux procédures ENTHL(I) et ENTHV(I). Ces procédures peuvent être schématisées comme suit :

II.3.2. Subroutine TEMP(I,L) .

Cette subroutine permet de déterminer la température à partir de l'enthalpie et de la composition.

Hv=T SUM Xi(Avi+Bvi T) +Lambda d'ou $$T = \frac{Hv - SUM\ Lambdai\ Xi}{SUM\ Avi\ Xi + T\ SUM\ Bvi\ Xi} \quad (1)$$

ou bien

hl = T SUM Xi (Ali + Bli T) d'ou $$T = \frac{hl}{SUM\ Ali\ Xi + T\ SUM\ Bli\ Xi} \quad (2)$$

La résolution de l'équation (1) ou bien (2) se fait avec la subroutine CONV.

II.3.3. Subroutine ACTY(I) .

Pour un mélange non idéal on a besoin des activités . Dans le DYFLO les activités sont calculées par la subroutine ACTY, seulement, cette subroutine n'est pas generalisée et il faut l'adapter pour chaque cas.

Exemple : Pour un mélange ternaire on a les corrélations suivantes :

$$\ln Nu1 = A12\,X2^2 + A13\,X3^2 + X2\,X3\,(A12 + A13 - A23)$$

$$\ln Nu2 = A23\,X3^2 + A21\,X2^2 + X3\,X1\,(A23 + A21 - A31)$$

$$\ln Nu3 = A31\,X1^2 + A32\,X2^2 + X1\,X2\,(A31 + A32 - A12)$$

Où les Aij sont des coefficients dépendants de la température.

Pour ce cas particulier le listing de la procédure ACTY est comme suit :

```
1*      SUBROUTINE ACTY(N)
2*      COMMON/CO/STRM(300,24),DATA(20,10),RTL(22),NCF,NCL,LSTR
3*      DATA(DATA(J,9),J=1,20)/20*1./
4*      DATA SA67,SA68,SA78,B67,B68,B78/-.62,-.8,-.3,370.,380.,170./
5*      DATA SA76,SA86,SA87,B76,B86,B87/-.6,-.7,-.4,365.,375.,185./
6*      TK=STRM(N,22)+273.
7*      A67=SA67+B67/TK
8*      A68=SA68+B68/TK
9*      A78=SA78+B78/TK
10*     A76=SA76+B76/TK
11*     A86=SA86+B86/TK
12*     A87=SA87+B87/TK
13*     X6=STRM(N,6)
14*     X7=STRM(N,7)
15*     X8=STRM(N,8)
16*     DATA(6,9)=EXP(A67*X7**2+A68*X8**2+X7*X8*(A67+A68-A78))
16*     DATA(7,9)=EXP(A78*X8**2+A76*X6**2+X6*X8*(A78+A76-A86))
16*     DATA(8,9)=EXP(A86*X6**2+A87*X7**2+X6*X7*(A86+A87-A67))
19*     RETURN
20*     END
```

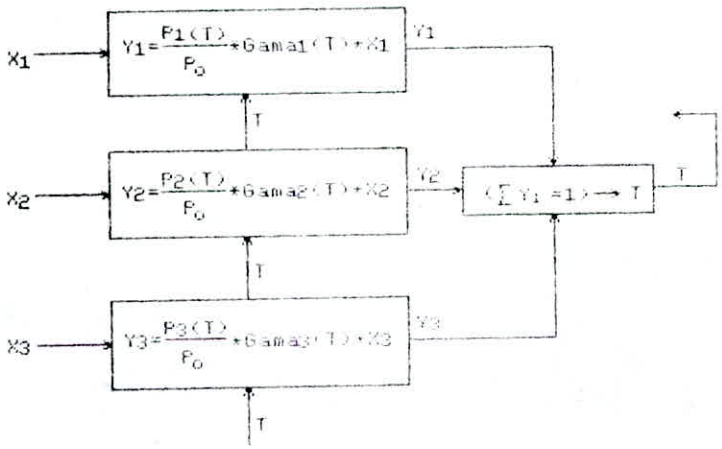
Si nous nous limitons au cas ideal , cela nous permettra de simplifier et d'eviter de modifier la procédure ACTY pour chaque cas.

```
1*      SUBROUTINE ACTY(N)
2*      COMMON/CD/STRM(300,24),DATA(20,10),RTL(22),NCF,NCL,LSTR
3*      DATA(DATA(J,9),J=1,20)/20*1./
4*      RETURN
5*      END
```

Sinon , nous pourrons introduire des valeurs estimées des activités.

II.3.4. Subroutine EQUIL(IL,IV) .

Pour calculer la composition en vapeur Y qui est en equilibre avec la composition X , il faut s'assurer que la pression ambiante est connue. Il est aussi nécessaire de calculer la température d'ébullition.Le modèle utilisé est representé dans le schéma suivant:



MODELE GENERAL DE L'EQUILIBRE LIQUIDE-VAPEUR.

Soit un mélange de liquide et de vapeur en équilibre. A l'équilibre, les potentiels chimiques sont égaux :

FIi Yi P = Xi Fi NUi

ou

Xi : Fraction molaire liquide du compose i.

Yi : Fraction molaire vapeur de compose i.

Fi : Fugacité du composé i pur, à la température d'ébulition T.

FIi : Fugacité du composé i dans le mélange.

P : pression totale.

NUi : Activité du composé i dans le mélange.

A une pression modérée et pour un mélange idéal, la relation précédente peut être simplifiée; nous obtenons ainsi :

la loi de Raoult

P Yi = Xi Pi (T) .

ou **Pi (T)** : Tension de vapeur.

Pour l'appliquer à un mélange non ideal, un coefficient **Nui** (activité) doit être introduit dans la relation d'équilibre :

Yi = Xi Pi(T)/P Nui

La méthode de calcul est itérative, il y a plusieurs moyens d'accélérer la convergence :

* En utilisant **CONV** :

```
E=1-SUM Yi
T1=T+G*E
CALL CONV(T,T1,1,NC)
```

L'introduction d'un gain G qui varie d'un cas à un autre, constitue l'inconvenient de cette méthode.

* En utilisant la méthode analytique de NEWTON-RAPHSON :

On fait appel à l'équation d'ANTOINE.

Pi(T) = EXP (C1 + C2/(T+C3)) ; T en (C).

En introduisant la relation d'équilibre développée précédemment, la variation de la composition vapeur peut s'exprimer par l'équation:

$$\frac{d}{dt}(Yi) = \frac{d}{dt}\{ EXP(C1i + C2i/(T+C3i)) * NUi * Xi/P \} .$$

Si on néglige la variation de l'activité en fonction de la température, on aura :

$$dYi / dt = - Yi * C2i / (T+C3i)^2$$

Pour N composants, la variation de Yi avec la température sera : SDY = SUM (dYi/dT) . L'erreur s'exprimera par : YER=1. - SUM Yi .

Le prochain choix de la température sera : T = T + YER/SDY .

La boucle est répétée jusqu'à la réduction de YER à moins de 0.001.

Ce calcul est exécuté par la subroutine EQUIL.

II.3.5. Subroutine DEWPT(IV,IL) .

Elle calcule le point de rosée d'un flux. L'équation d'équilibre est :

$$Xi = \frac{P\ Yi}{Pi(T)\ NUi} .$$

Pour appliquer la méthode de NEWTON-RAPHSON :

$$\frac{d\ Xi}{dt} = Xi\ C2 / (T+C3)^2 .$$

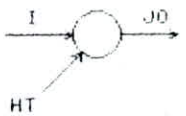
On utilise la même méthode que précédemment.

II.3.6. Subroutine HTEXCH .

Elle peut être utilisée dans les opérations unitaires suivantes:

- Réchauffeur,
- Refroidisseur,
- Super-réchauffeur,
- "Desuperheater".
- Rebouilleur,
- Super-rebouilleur,
- Condenseur total,

Suivant la valeur de HI .



I : Numéro du flux d'entre.
JO : Numéro du flux de sortie.
HT : Chaleur apportée + ou - .
L : Phase;3 liquide;0 vapeur.

SCHEMA DES FLUX DE HTEXCH(I,JO,HT,L).

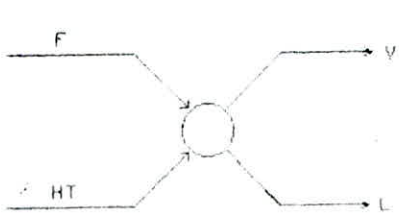
Cette procédure n'est pas adéquate pour le calcul d'un rebouilleur produisant de la vapeur saturée. Elle est valable pour un super-évaporateur (Tout s'évapore dès que HT est suffisante) . Le flux de sortie est soit liquide soit vapeur .

Tableau Classification des opérations unitaires avec changement d'enthalpie

Flux entrant	Chaleur ajoutée	Flux sortant	Opération unitaire
1. Liquide	Positive	Liquide seulement	Réchauffeur
2. Liquide	Négative	Liquide seulement	Refroidisseur
3. Vapeur	Positive	Vapeur seulement	Super-réchauffeur
4. Vapeur	Négative	Vapeur seulement	Desuperheater
5. Liquide	Nulle	Vapeur & liquide	Adiabatique flash
6. Liquide	Positive	Vapeur & liquide	Rebouilleur/évaporateur
7. Vapeur	Négative	Vapeur & liquide	Condenseur partiel
8. Vapeur	Négative	Liquide seulement	Condenseur total
9. Liquide	Positive	Vapeur & liquide	Echangeur de chaleur de type flash

II.3.7. Subroutine FLUSH .

Cette subroutine est utilisée pour le cas des opérations unitaires ou le flux d'entrée est converti en deux flux de sortie vapeur-liquide en équilibre.



I : Flux d'entrée.
JV : Flux vapeur de sortie.
JL : Flux liquide de sortie.
HT : Chaleur apportée.

SCHEMA DES FLUX DE FLASH(I,JV,JL,HT).

Si le flux de chaleur est spécifié, il est nécessaire de déterminer la température d'équilibre de sortie. (Sauf pour le cas particulier du condenseur total ou la température est specifiée).

Si F est un flux vapeur et HT négative on a un condenseur partiel ou total, si F est un flux liquide et HT est positive on a un rebouilleur rapide (flashing boiler).

BILAN DE CHALEUR : chaleur entrante = chaleur sortante.

$$F.hf + HT = U.Hv + L.hl$$

ou F;U et L sont les débits en (mol/min), HT flux de chaleur et Hv; hl; hf sont les enthalpies molaires des différents flux.

Soit $R = U / F$, le bilan de chaleur peut s'écrire :

$$R = \frac{(hf + HT/F - hl)}{(Hv - hl)}$$

BILAN DE MATIERE : $F.XFi = U.Yi + L.Xi$. Soit $Hi=Yi/Xi$.

La composition vapeur peut être établie à partir du bilan de matière en utilisant R.

$$Yi = XFi \frac{Hi}{1. + R (Hi - 1)}$$

EQUILIBRE :

$Hi = Pi(T)/P\ NUi$ ou $Pi(T) = EXP(C1 + C2/(T+C3))$.

NUi = DATA(i,9) Activité.

P=STRM(L,24) Pression totale.

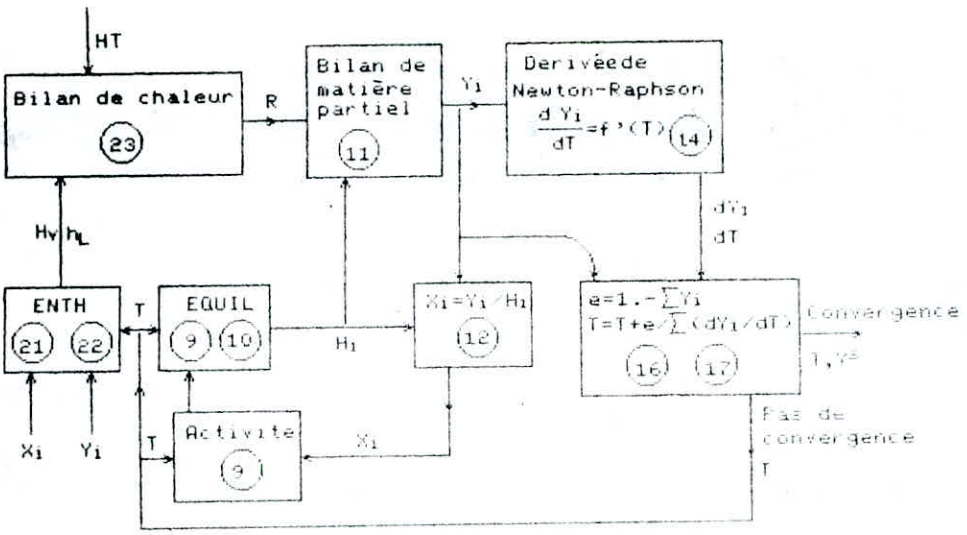
COMPOSITION LIQUIDE : $Xi=Yi/Hi$.

TEMPERATURE : On utilise la méthode de NEWTON-RAPHSON (utilisée précédemment).

$$\frac{dYi}{dT} = - Yi\ C2 / (T + C3)^2 \qquad \text{et} \qquad T = T + SUM \frac{YER}{(dYi/dT)} .$$

Ou $YER = 1 - SUM\ Yi$.

Ce calcul est représenté dans le schéma suivant :



SCHEMA DU FLUX D'INFORMATIONS DE FLASH.

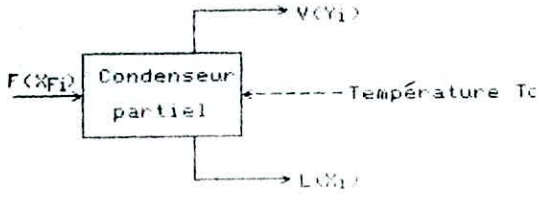
Dans la subroutine FLUSH,si R>1 ou R<0 on a un super-évaporateur ou un condenseur total, respectivement. On peut aussi l'utiliser pour un flush adiabatique dans le cas d'un flux liquide qui passe à travers un orifice ou une vanne de contrôle, et passe à une pression plus basse. Les enthalpies des flux entrant et sortant sont presque les mêmes. Il y a une évaporation due à une grande dépressurisation. Ceci est traduit par FLUSH avec HI = 0.

II.3.8. Subroutine CUBOIL .

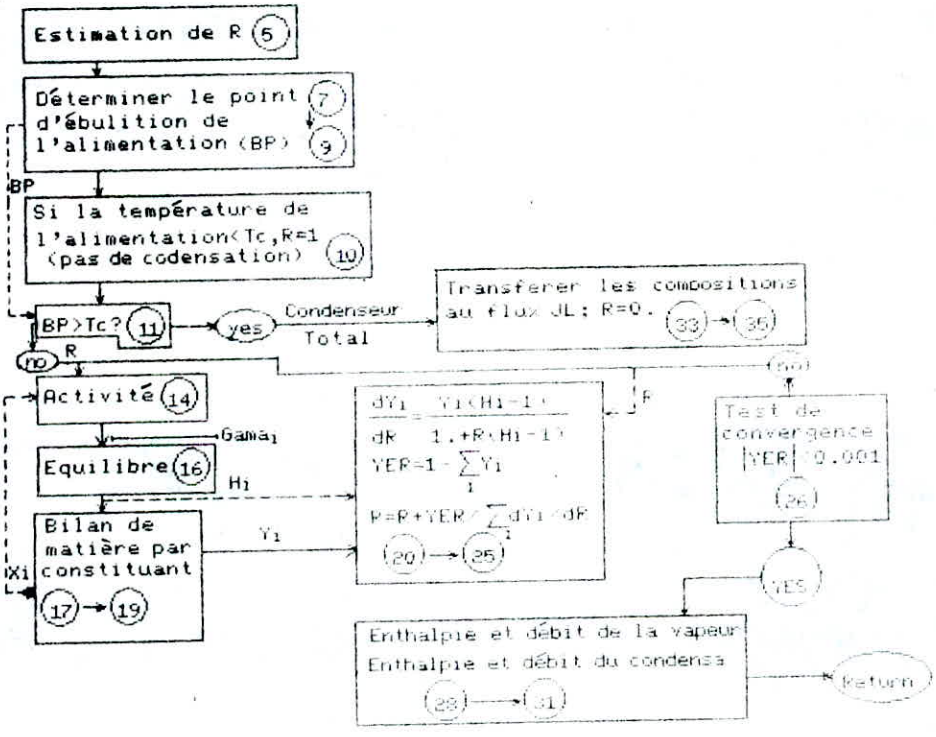
Plusieurs procédés unitaires, comme un reboilleur ou un évaporateur, opèrent, avec un contrôle aproprié, selon le principe que le flux de chaleur détermine le taux d'évaporation.Pour le cas où la rétention peut être négligée. Le débit rentrant est égal au débit sortant. La procédure CUBOIL a été développée (dans le cas d'un rebouilleur à volume constant) pour simuler cette situation.

II.3.9. Subroutine PCON :

Simule un condenseur partiel . (La séparation d'un flux vapeur en un flux liquide et un flux vapeur , en précisant la température). Dans cette subroutine on utilise aussi la méthode de NEWTON-RAPHSON. Elle peut être représentée schématiquement par :



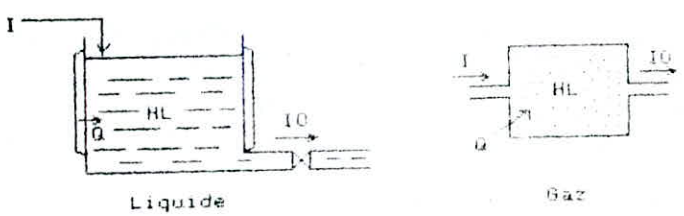
SCHEMA REPRESENTANT PCON.



SCHEMA DU FLUX D'INFORMATIONS DE LA SUBROUTINE PCON .

II.3.10. Subroutine HLDP :

Dans toutes les subroutines vues jusqu'à présent, il n'y a pas stockage ou accumulation de matière ou d'énergie. Pour rendre la librairie de procédures, développées dans le DYFLO, capable de simuler des situations dynamiques, quelques procédures qui vont suivre incorporent un retenu dynamique. En commencant par un réservoir de stockage ou simplement un volume monophasique avec une entrée et une sortie.



RESERVOIR MONOPHASIQUE (HLDP).

On considère que le reservoir est agité et les propriétés du flux de sortie sont les mêmes que dans le réservoir. Dans le bilan de matière il faut faire intervenir un terme representant les pertes dues aux réactions, s'il y a lieu, sinon le terme est nul. Ce terme est représenté par la matrice de réaction RCI.

BILAN : Accumulation = entrée - sortie .

BILAN MOLAIRE PARTIEL :

$$\frac{d}{dt} Xn = \frac{Fin\ Xin - Fout\ Xn + RCI(n) - Xn\ dHL/dt}{HL}$$

BILAN MOLAIRE IOTAL :

$$\frac{d}{dt} HL = Fin - Fout + RCT(21)$$

d' ou :

$$\frac{d}{dt} Xn = \frac{Fin\,(Xin - Xn) + RCT(n) - RCT(21)\,Xn}{HL}$$

de même le bilan d'énergie est :

$$\frac{d(ENo)}{dt} = (Fin\,ENi - ENo\,(Fout + dHL/dt) + Q + RCT(22))/HL$$

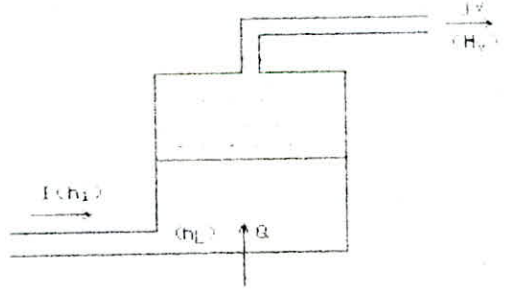
ou ENo : Enthalpie molaire

Ces équations sont programmées dans la subroutine HLDP.

II.3.11. Subroutine VVBOIL.

Cette subroutine simule un rebouilleur à volume variable (avec accumulation ou rétention). La subroutine CUBOIL présentée précédemment permet la simulation d' un rebouilleur à volume constant,ou le flux d'entrée est instantanément transforme en vapeur. Nous avons vu également , la subroutine HLDP permetant la simulation d'un réservoir à volume variable (avec accumulation). Le mixage de ces deux subroutines donne VVBOIL. La figure suivante décrit le système à simuler, constitue d'un noeud IHL d'un flux d'entrée I, et d'un flux de sortie JV. Le rebouilleur reçoit un flux de chaleur Q. Il peut y avoir une réaction dont les données sont dans la matrice RCT. Les équations de base de ce système sont similaires à celles de la subroutine HLDP, décrite précédemment.

REBOUILLEUR A VOLUME VARIABLE



Bilan molaire :

$$\frac{dHL}{dt} = F_I - F_{JV} + RCT(21)$$

ou HL : Quantité en (moles).

F : Débit des flux.

Bilan de chaleur :

Enthalpie totale ajoutée = Flux de chaleur + Enthalpie de réaction + Enthalpie du flux entrant

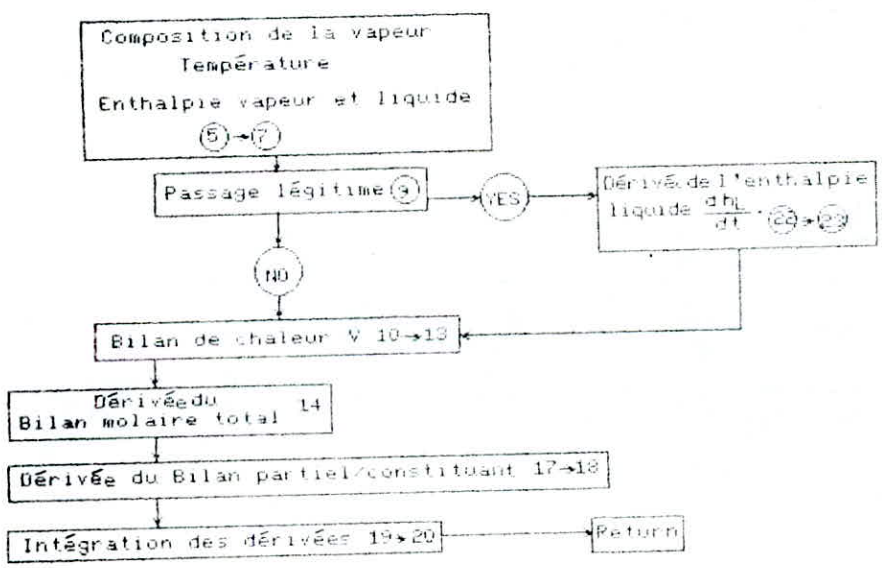
$$QP = Q + RCT(22) + F_I H_I$$

Débit d'évaporation :

$$F_{JV} = \frac{QP - hL\, d\,HL/dt - HL\, d\, hL/dt}{Hv}$$

où hL et Hv : Enthalpies molaires.

Le terme d hL/dt représente l'accumulation de chaleur dans le volume liquide, toujours petite, mais on doit en tenir compte ici.



SCHEMA DU FLUX D'INFORMATION DE VVBOIL.

La dérivée de l'enthalpie liquide hL est obtenue de même la manière que dans la subroutine DER (présentée précédemment).

Bilan partiel du constituant N :

Flux d'entrée : $FNI = F_I X_{NI} + RCT(N)$.

Accumulation = entrée - sortie .

$$\frac{d}{dt} \langle HL\ XN \rangle = FNI - FJU\ YN \ .$$

Cette équation peut être écrite sous la forme :

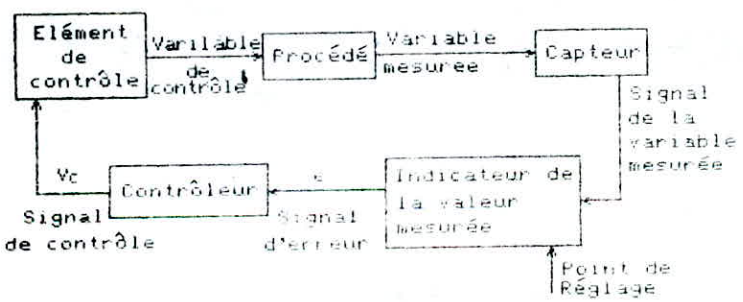
$$DERN = \frac{d\ XN}{dt} = \frac{FNI - FJU\ YN - XN\ d\ HL\ /dt}{HL} \ .$$

Le schéma du flux d'information est présenté dans la figure précédente.

II.3.12. Subroutines de contrôle .

Le DYFLO propose un ensemble de procédures pour le contrôle de procédés. Quelques unes sont brièvement présentées par la suite.

La figure suivante montre un schéma type d'une boucle de régulation.



SCHEMA TYPE D'UNE BOUCLE DE REGULATION.

Dans ce qui suit, nous présentons des procédures simulant l'action d'élments de mesure et celle de régulateurs.

a. Eléments de mesure .

procédure IFN1

Cette procédure simule l'action des éléments de mesure, pouvant être décrits par une fonction de transfert du premier ordre.

XIN -----> OUT/XIN=GAIN/(TC*s+1) -----> OUT

ou XIN = entrée.

OUT = sortie.

GAIN = gain.

TC = constante de temps.

s = variable de LAPLACE .

L'équation différentielle équivalente s'écrit :

d(OUT)/dt = (GAIN*XIN - OUT)/TC

Le listing de cette procédure est présenté en annexe.

procédure IFN2 .

La forme générale d'une fonction de transfert du second ordre est la suivante :

XIN -----> OUT/XIN=GAIN/(TC2*s**2+2*ksi*TC*s+1) -----> OUT**

ou XIN = entrée.

OUT = sortie.

GAIN = gain.

TC = constante de temps.

s = variable de LAPLACE .

ksi = coefficient d'amortissement.

Cette expression définit la relation dynamique entre le signal d'entrée et le signal de sortie passant à travers un SL2.

L'équation différentielle équivalente à la fonction de transfert d'un SL2 s'écrit:

$$\frac{d^2 OUT}{dt^2} = (GAIN * XIN - OUT) A1 - A2 \, d(OUT)/dt$$

ou $A1 = 1/(TC^{**}2)$ & $A2 = ksi/TC$

La double intégration nécessaire dans ce cas est faite par la subroutine TFN2.

b. Régulateurs :

Dans ce qui suit nous présentons les procédures simulant l'action des trois principaux types de régulateurs à savoir :

* régulateur à action proportionnelle RP
* régulateur à action proportionnelle et intégrale RPI.
* regulateur a action proportionnelle intégrale et dérivée RPID.

Subroutine CONTR1 .

Cette subroutine simule l'action d'un RP.

Subroutine CONTR2 .

Cette subroutine simule l'action d'un RPI.

Subroutine CONTR3 .

Cette subroutine simule l'action d'un RPID.

c. autre procédure :

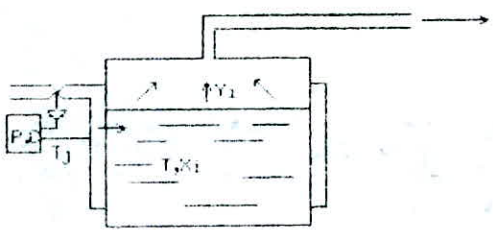
Subroutine IDL .

Cette subroutine permet de simuler un retard

N.B : Le DYFLO comporte d'autres procédures qui ne sont pas présentées ici. Ces subroutines concernent la cinétique réactionnelle, dynamique des fluide. ...

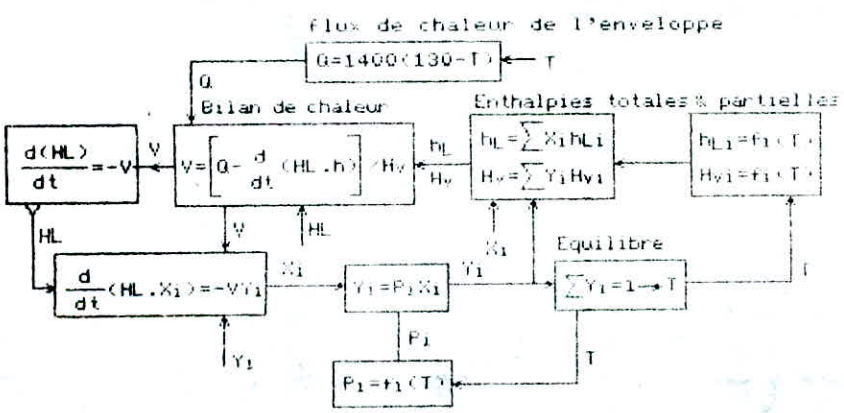
II.4. Exemple d'application du DYFLO .

Nous avons pris comme exemple d'utilisation du DYFLO celui présente dans le R.G.E FRANKS [1]. Il s'agit d'une distillation discontinue (Batch distillation). Le système à simuler est présenté dans la figure suivante :



DISTILLATION BATCH

Le modèle est présenté dans le schéma suivant :



MODELE MATHEMATIQUE DE LA DISTILLATION BATCH.

Le listing du programme simulant ce système suivant DYFLO est :

```
 1* C **  DONNEES  **
 2*    COMMON/CD/STRM(300,24),DATA(20,10),RET(21),NCF,NCL,LSTR
 3*    LOGICAL LSTR,NF
 4*    DATA(DATA(1,N),N=1,8)/13.46,-5210.,273.,8.,.01,9020.,20.,.02/
 5*    DATA(DATA(2,N),N=1,8)/15.2,-6050.,273.,12.2,.02,11500.,32.,.01/
 6*    DATA(DATA(3,N),N=1,8)/15.4,-5312.,273.,6.5,.01,7500.,16.,.03/
 7* C ** INITIALISATION **
 8*    DATA(STRM(2,N),N=1,3)/.43,.31,.26/NCF,NCL/1,3/
 9*    DATA(STRM(2,N),N=21,24)/350.,95.35,0.,1./
10*    LSTR=.TRUE.
11*    CALL EQUIL(2,3)
12* C ** SECTION DE DERIVATION **
13  7 Q=1400.*(130.-STRM(2,22))
14*    CALL VVBOIL(1,3,2,Q)
15*    CALL PRL(10.,60.,NF,2,3,0,0,0,0,0,0,0,0,0,0)
16*    IF (NF) GOTO 10
17* C ** SECTION D'INTEGRATION **
18* 5 CALL INTI(TIM,1.,4)
19*    GOTO 7
20*10 STOP
21*    END
```

Les résultats de simulation présentés par ce programme seront :

```
TIME =  .0000
STRM NO          2              3
FLOW         .3500+03       .7291+01
TEMP         .9535+02       .9535+02
ENTHAL       .2340+04       .8994+04
PRESS        .1000+01       .0000
COMP 1       .4300+00       .2169+00
COMP 2       .3100+00       .9110-01
COMP 3       .2600+00       .6920+00

TIME =  .1000+02
STRM NO          2              3
FLOW         .2912+03       .5291+01
TEMP         .1003+03       .1003+03
ENTHAL       .2539+04       .9326+04
PRESS        .1000+01       .0000
COMP 1       .4668+00       .2040+00
COMP 2       .3508+00       .1282+00
COMP 3       .1824+00       .5878+00

TIME =  .2000+02
STRM NO          2              3
FLOW         .2440+03       .4216+01
TEMP         .1056+03       .1056+03
ENTHAL       .2748+04       .9752+04
PRESS        .1000+01       .0000
COMP 1       .4947+00       .3657+00
COMP 2       .3894+00       .1783+00
COMP 3       .1160+00       .4560+00
```

II.5. Conclusion sur le DYFLO .

Les problèmes de dynamique des procédés, pendant les années 1955-1965, étaient résolus par des calculateurs analogiques. Durant les années 1965-1970, les calculateurs numériques sont devenus les outils favoris pour la simulation de procédés chimiques, avec ce qu'ils offraient comme rapidité de calcul, capacité de mémoire, possibilité de développer une librairie ... etc.

Le DYFLO a été concu à cette époque, en FORTRAN IV. Depuis, l'informatique a fait un grand chemin, avec l'apparition de la micro-informatique, des stations de travail avec de grandes possibilités d'entrée-sortie (- interactivité et convivialité; - graphisme très haute résolution; - contrôle et commande; ...etc), et l'apparition de la programmation object (système expert; inteligence artificielle). Le FORTRAN, qui a été longtemps le language scientifique le plus utilisé, n'est plus en mesure d'offrir toutes ces possibilités (du moins, pas aisément.). Le DYFLO en FORTRAN ne peut pas suivre cette évolution et ne peut pas exploiter toutes les ressources qu'offre le matériel informatique de nos jours. De plus, sa librairie permet de simuler des opérations unitaires élémentaires, et il n'y a pas de procédure pour simuler un appareil de génie chimique. Pour cela il faut écrire tout un programme, ce qui rend difficile la généralisation et la simulation de procédés complexes.

III. REALISATION DU LOGICIEL DE SIMULATION .

III.1. Introduction .

Vu les inconvenients et les limites que présente le DYFLO en FORTRAN IV, nous avons traduit et adapter la librairie du DYFLO au langage C. Nous proposons, aussi, de développer une librairie pour simuler des appareils de génie chimique avec des modèles proposés, et une façon de relier ces appareils. Nous avons réalisé une interface utilisateur interactive, offrant des possibilités graphiques et facilitant la manipulation de ces appareils, l'introduction des paramètres et la présentation des résultats de la simulation.

III.2. Adaptation des procédures du DYFLO au langage C .

III.2.1. Pourquoi le langage C ?

Le langage C est aujourd'hui le langage de développement le plus répandu, il le doit à ses qualités, car, contrairement à FORTRAN qui s'imposa grâce au support actif d'IBM, C a dû convaincre les développeurs petit à petit. Il est souvent appelé "langage assembleur évolue". Il est structuré, très "près" de la machine, déclaratif, compilable, récursif, portable ...
Il a permis de reprogrammer le noyau d'UNIX et des utilitaires accompagnant ce système d'exploitation.

Le langage C a eu des débuts plutôt élitistes. Il ne fût guère disponible que sous UNIX avant 1977, et les programmeurs C acquirent, rapidement, une réputation fort justifiée de "hacker", qu'ils se forcèrent d'entretenir.

Avec l'apparition de version du C sur micro-ordinateur il a permis au développeurs professionnels de travailler chez eux vu sa portabilité. L'apparition d'E.D.I (Environnement de Développement Intégré) qui comporte à la fois éditeur, compilateur, linker et débogueur a facilité grandement la programmation en C (Turbo C; Quic C.).

IL permet l'exploitation de toute les ressources de la machine. Il peut être utilise dans la commande et le contrôle.De plus il a une bibliothèque mathématique très riche. Il permet de développer des applications sophistiquées et de mener de grands projets informatiques. Il facilite le travail d'équipe. Il peut manipuler des objets. (Dernièrement, il y a eu l'apparition du C++ qui est une version du C orienté OBJET). De nos jours programmer en C, c'est s'ouvrir les portes de, pratiquement, toutes les machines existantes.

III.2.2. Structure adoptee .

Nous avons adopte la meme structure que le DYFLO, C'est la structure adoptee, generalement, par tous les logiciels de simulation de procedes, en definissant des flux numerotes. Cette structure est aussi adoptee dans la simulation modulaire. La structure est presentee en detail au paragraphe .II.2. .

III.2.3. Présentation des procédures du DYFLO traduites en C .

Nous avons regroupé l'ensemble des procédures en cinq fichiers "headers":

* NUMERIQU.C : Fichier contenant les traductions en C des procédures numériques déjà présentées au paragraphe (II.1.) Les prototypes de ces procédures sont :

```
float INI(float *px,float dx);
float INII(float *ptd,float *pdtd,int iod);
int CONV(float *px,float *py,int nr);
int CPS(float *px,float *pxc,float r);
int DER(float y,float x,int i);
```

Le listing complet de NUMERIQU.C est présenté en annexe (1).

* ETAT.C : Fichier contenant les procédures concernant l'état thermodynamique des flux. Ces procédures sont les traductions en C de procédures déjà présentées en FORTRAN au paragraphe .II.3. . Les prototypes de ces procédures sont :

```
void ENTHL(int i);
void ENTHV(int i);
void TEMP(int i,int I);
void ACTY(int n);
void EQUIL(int il,int iv);
void DEWPT(int iv,int il);
```

Le listing complet de ETAT.C est présenté en annexe (2).

* UNITETAT.C : Fichier contenant les procédures des opérations unitaires concernant l'étude de l'état et des transformations d'état. Ces procédures sont les traductions en C de procédures déjà présentées en FORTRAN au paragraphe .II.3.

Les prototypes de ces procédures sont :

```
void SUM(int i,int j,int k,int l);

void SPLIT(int j,int k,int m,float rkj);

void HTEXCH(int i,int jo,float ht,int l);

void CSHE(int isi,int iso,int iti,int ito,float ua,int ips,
          int ipt,int nsf,int nsi,int ntf,int nti);

void FLUSH(int i,int jv,int jl,float ht);

void CVBOIL(int i,int jv,float ht);

void PCON(int i,int jv,int jl,float tc);
```

Le listing complet de UNITETAI.C est presenté en annexe (3).

* UNITDYN.C : Fichier contenant les procédures des opérations unitaires dynamiques. Une seule procédure (HLDP) a été décrite au paragraphe .II.3. Les autres procédures vont être présentées par la suite . Les prototypes de ces procédures sont :

```
void REB(float a,float h,float cv,float wc,int jf,int jb);

void BOT(int li,int lo,int iv,float q,float hl);

void STAGE(int i1,int i2,int il,int iv,float h,float hl,float htc);

void STGF(int i1,int i2,int i3,int il,int iv,float h,float hl,
          float htc);

void STAGE(int i1,int i2,int il,int iv,int is,float h,float hl,
           float htc);

void HLDP(int i,int io,int l,float hl,float q);
```

Le listing complet de UNITDYN.C est présenté en annexe (4).

* CONTROL.C : Fichier contenant des procédures utilisées dans le contrôle. Les prototypes de ces procédures sont:

Le listing complet de CONTROL.C est présenté en annexe (5).

L'une des manières d'exploiter le travail qui a été fait est d'appeller les procédures précédentes à partir d'un programme en C (ou même en TURBO PASCAL) pour simuler un système de génie chimique donné. En C, pour utiliser l'une de ces procédures, il suffit d'inclure, dans le programme, le fichier ou elle est présentée.

III.2.3. Conclusion .

Jusqu'ici, nous avons adapté les procédures du DYFLO, faites en FORTRAN IV, au langage C. A ce niveau, nous pouvons réaliser des programmes de simulation de procédés en C, en procédant comme avec le DYFLO et en faisant appel a la librairie développée. En plus, nous bén ficions de la commodité d'utilisation de librairies en C (En incluant uniquement les procédures utilisées, ce qui réduit la taille du programme exécutable), de la librairie mathématique très riche du C, de sa rapidité de calcul et de la librairie graphique du C.

III.3. Réalisation d'une procédure générale simulant un appareil de génie chimique .

III.3.1. Introduction :

La librairie du DYFLO ne contient pas de procédure pour simuler un appareil. Elle se limite à un ensemble de procédures simulant des opérations unitaires élémentaires auxquelles il faut faire appel, dans un programme, pour simuler un appareil.

Nous nous proposons d'ajouter à notre librairie en C des procédures pour simuler des appareils entiers, selon des modèles proposés,qui peuvent être modifiés ou améliores. Pour cela nous nous limitons à un seul appareil Nous avons choisi le procédé unitaire le plus courant, mais aussi le plus complexe dans l'industrie chimique et pétrochimique, qui est la colonne de distillation. La complexité de la distillation dans une colonne varie d'une simple séparation d'un mélange binaire à la séparation d'un mélange complexe à plusieurs constituants, non idéale, avec réaction, avec plusieurs alimentations et plusieurs soutirages

Le DYFLO propose une subroutine simulant un étage normal,SIAGE, un étage avec alimentation,SIGF, et un étage avec soutirage SIGS . Ces procédures sont présentées dans la suite et ont été adaptées au langage C.

III.3.2. Réalisation d'une procédure générale simulant une colonne de distillation selon un modèle .

a. Présentation du modèle .

La première approximation, qui est faite dans la plupart des simulations de colonnes de distillation, est de décrire la colonne en termes d'étages théoriques. Une colonne de 20 étages réels avec une efficacité de 70% peut être correctement représentée par une colonne à 14 étages théoriques d'équilibre, chaque étage ayant un retenu de (1/0.7) fois le retenu par plateau de la colonne réelle

Le cas le plus général de distillation est la séparation de mélange complexe à plusieurs constituants. Pour chaque constituant i et pour chaque plateau n, le bilan massique sera :

$$(I) \qquad \frac{d}{dt}\left(X_n^i \right) = L_{n+1} X_{n+1}^i + V_{n-1} Y_{n-1}^i - L_n X_n^i - V_n Y_n^i .$$

La relation d'équilibre liquide-vapeur pour chaque constituant est :

$$(II) \qquad Y_n^i = \left[Ki(Tn)/Pn \right] \left(NU_n^i X_n^i \right) .$$

ou Ki(Tn) : Pression de vapeur du composant pur

NUi : Activité

Pn : Pression totale dans le plateau n.

Le bilan massique du constituant i (I) donne la composition liquide, et la relation d'équilibre (II) est utilisée pour déterminer la composition vapeur et la température Tn dans le plateau sachant que :

$$\sum_i Y_n^i = 1 . \qquad (III)$$

donc (I) --> X_n^i ; d'ou Y_n^i & Tn .

Le bilan de chaleur est :

$$(IV) \qquad \frac{d}{dt}\left(Hn\ Tn \right) = Q_{n-1} + q_{n+1} - Q_n - q_n .$$

ou Q : Enthalpie de la vapeur

q : Enthalpie liquide.

Hn : Capacité calorifique dans le plateau n.

L'accumulation **d(Hn In)/dt** est négligeable par rapport au flux de chaleur Q entre les Plateaux. Ce terme peut donc toujours être annulé.

Les enthalpies des courants vapeur et liquide sont définis, en termes de composition et de température comme suit :

Q = V SUM Qi Yi ; Qi = alfai + betai T .

Ou V : Débit du flux vapeur .

Qi : Enthalpie du constituant i .

Le modèle complet d'un étage, pour un mélange à quatre constituants, est donné dans la **fig -.III.1-** .

Les Bilans massiques partiels établissent les compositions liquides et les bilans massiques totaux établissent les débits liquides (qui passent par un retard représentant l'approximation de la dynamique des fluides de l'étage) L'équation d'équilib. liquide-vapeur établit les compositions vapeurs, et le bilan de chaleur définit les débits vapeur Vn quitant le plateau n.

Nous avons déjà une série de procédures capables de simuler des opérations simples, nous avons besoin d'autres procédures qui peuvent être assemblées pour simuler une colonne de distillation. L'objectif sera de développer un programme capable de simuler le fonctionnement dynamique d'une colonne qui, à l'état d'étude, permet de prévoir les profils de composition, de température, et de débits le long de la colonne. Le programme utilisant le modèle de l'étage présenté à la **fig -.III.1-** satisfait plusieurs situations, mais pourrait être ineficace pour d'autres cas à étudier.

b. Stabilité de la simulation .

Les études faites dans le domaine de la simulation dynamique numérique d'une colonne de distillation, ont révélé que les calculs ont une grande tendance à devenir instable.Ceci est dû à la présence, dans le système, d'équations différentielles ayant une constante de de temps très petite comparée à la constante de temps dominante du système. De plus il y a des constantes de temps qui varient avec les paramètres et les conditions. Pour palier à ce problème, nous devons soit prendre un pas d'intégration très petit (Ce qui satisfait toutes les situations mais cela demande un temps de calcul très long.), soit adopter la méthode suivante .

Les principales caractéristiques dynamiques des colonnes de distillation sont dues à l'accumulation et au bilan de tous les constituants et pour chaque étage. Un programme général consiste à résoudre à chaque étage l'équation différentielle suivante :

$$(U)\qquad \frac{dX_n}{dt} = \frac{(U_{n-1}Y_{n-1} + L_{n+1}X_{n+1}) - U_nY_n - L_nX_n}{H}$$

Où H : Retenu (Holdup) .

X : Composition liquide .

Y : Composition vapeur .

En définissant le rapport d'équilibre :

$$K_n = Y_n/X_n ,$$

et en substituant dans (U), on aura :

$$\frac{dX_n}{dt} = \frac{(U_{n-1}Y_{n-1} + L_{n+1}X_{n+1}) - (U_nK_n + L_n)X_n}{H}$$

Cette dernière équation peut être écrite sous la forme :

$$\frac{dXn}{dt} = \frac{1}{to}\,(Fi - Xn) \quad .$$

Où la constante de temps to sera : $to = H/(Un\,Kn + Ln)$

Cette constante de temps établit la taille du pas critique de l'équation différentielle. C'est en fonction du retenu par étage H, du débit de la vapeur qui monte Un et du débit liquide Ln. Ce sont des grandeurs communes pour tous les composants dans la colonne. C'est aussi en fonction du rapport de composition vapeur/liquide Kn qui est spécifique pour chaque constituant.

$$Max\,(to) = \lim_{Kn \to 0} (H/(Un\,Kn + Ln)) = H / Ln \quad .$$

$$\lim_{Kn \to inf} (to) = 0 \quad .$$

Donc une grande valeur de K rend la constante de temps très petite ce qui impose le choix d'un pas d'intégration aussi petit. (K peut être très petit pour une évaporation rapide ou très grand pour une évaporation lente.).

La solution ne consiste pas à adopter un pas d'intégration très petit; mais l'équation différentielle qui demande un pas très petit n'est pas intégrée, elle est résolue algébriquement, comme suit :

1 Calculer Ki pour le constituant :,(Ki=Yi/Xi a l'etage n).

2 Si Ki<5, calculer la dérivée et intégrer.

3 Si Ki>5 résoudre algébriquement pour Yi et obtenir Xi par équilibre.

(pour Ki=5, la méthode est choisie arbitrairement)

La résolution algébrique pour Yi est simple. L'étude d'état du bilan partiel, au plateau n, est :

$$\text{Entrée} - \text{Sortie} = 0 \quad .$$

$$\langle L_{n+1} X_{n+1} + V_{n+1} Y_{n+1} \rangle - \langle L_n X_n + V_n Y_n \rangle = 0 \quad .$$

Remplacons par Kn=Yn/Xn, on aura :

$$Y_n \langle L_n/K_n + V_n \rangle = L_{n+1} X_{n+1} + V_{n-1} Y_{n-1} \quad .$$

Finalement :

$$Y_n = \langle L_{n+1} X_{n+1} + V_{n-1} Y_{n-1} \rangle / \langle L_n/K_n + V_n \rangle \ \& \ X_n = Y_n/K_n \quad .$$

Physiquement ces équations expriment que dans le cas de composé relativement volatil, la plus grande partie du liqude s'écoulant du plateau (L n+1 X n+1) s'évapore et passe à l'étage n+1, dans la vapeur (Vn Yn). Cette situation est schématisée dans la **figure -III.2-**.

c. Procédure simulant un étage normal STAGE .

Cette procédure représente le programme du modèle présente pour un étage dans la **fig -.III.1-** . Pour des raisons de commodité les modifications suivantes sont apportées aux équations de base. Le retenu molaire total HL est considéré constant, mais la différence entre le débit d'entrée et le débit de sortie est prise en compte dans les bilans de matière et de chaleur.

$$\frac{d\,HL}{dt} = DHL = FLIN - Vn - Ln \quad . \qquad \langle VI \rangle$$

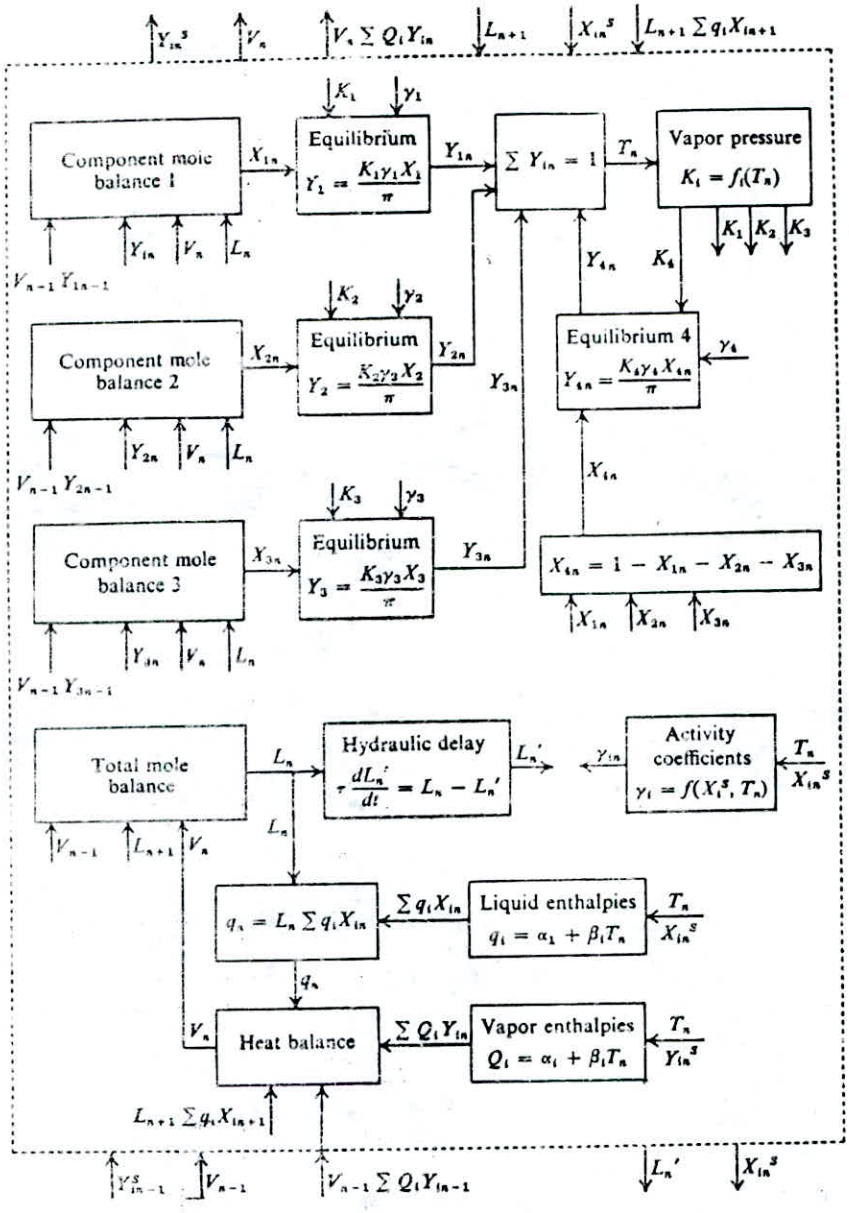


FIG-III.1- SCHEMA DU MODELE D'UN ETAGE (PROCEDURE STAGE).
[13]

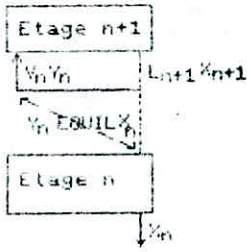


FIG -III-2- CAS DES COMPOSANTS VOLATILS.

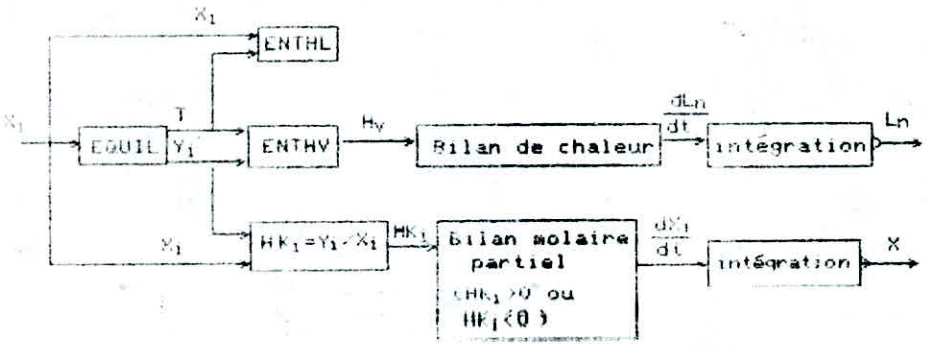
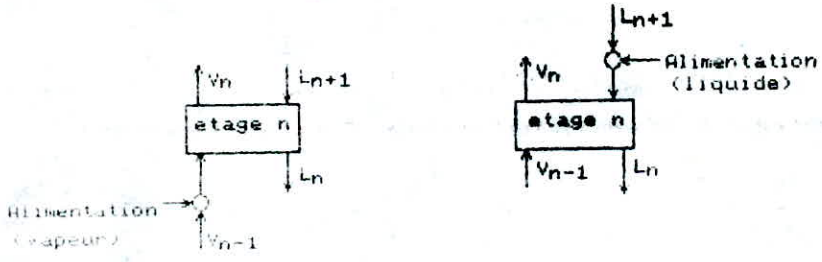


FIG -III-3- FLUX D'INFORMATION DE LA SUBROUTINE STAGE . [1]



Alimentation vapeur

Alimentation liquide

FIG -III-4-

Flux liquide de sortie : $\frac{d}{dt}(Ln) = DHL / HTC$.

Où $FLIN = L_{n+1} + U_{n-1} + RCT(21)$: Flux d'entrée total

HTC : Constante de temps hydraulique (retard) .

Bilan de chaleur :

La chaleur entrante est définie par :

$$HIN = (U\ Hu)_{n-1} + (L\ hl)_{n+1} + RCT(22) + H .$$

Où H est un flux de chaleur supplémentaire en cas de perte ou de gain.

Le bilan de chaleur à l'étage n sera :

Accumulation = Entrée - Sortie

$$\frac{d}{dt}(HL\ hl) = HIN - (U\ Hu + L\ hl)n .$$

Où HL : retenu liquide à l'étage (moles) .

En différenciant, on obtient :

$$HL \frac{d\ hl}{dt} + hl \frac{d\ HL}{dt} = HIN - (U\ Hu + L\ hl)n .$$

Or $\frac{d\ HL}{dt} = DHL$, déjà défini.

En négligeant le premier terme $HL \frac{d\ hl}{dt}$

on aura : $Ln = \frac{HIN - FLIN * hl}{Hu - hl}$.

Cette dernière équation est programmée dans la procédure.

(Le fait de négliger HL d(hl)/dt est justifié si la variation de l'enthalpie liquide est petite dhl/dt)

Bilan molaire d'un constituant : Accumulation = Entrée - Sortie

$$\frac{d}{dt}(HL\ X)_n = (L\ X)_{n+1} + (U\ Y)_{n+1} + RCT(N) - (L\ X)_n - (U\ Y)_n$$

En différenciant et en substituant d(HL)/dt par DHL on aura :

$$\frac{d\ Xn}{dt} = \frac{(L\ X)_{n+1} + (U\ Y)_{n-1} + RCT(N) - FLIN\ X_n - U_n\ (Y_n - X_n)}{HL}$$

Dans le listing de la subroutine SIAGE (présenté en annexe (4)), les 4 premiers termes sont sommés dans CHIN

H représente un flux de chaleur dû à une perte ou a un gain par transfert, autre qu'une chaleur réactionnelle qui est prise en compte par RCT(22)

Le schéma du flux d'informations de cette procédure est représenté dans la figure -III.4- .

d. Procédure simulant un étage avec alimentation SIGF .

Un plateau d'alimentation peut être simule en ajoutant au flux vapeur entrant dans l'étage la vapeur d'alimentation, ou en ajoutant un flux liquide au flux liquide entrant dans l'étage (figure-III.3-).

Ceci peut être réalise par la procédure SUM dont le listing est en annexe (3). Il serrait plus élégant de développer une procédure, similaire à SIAGE incluant une alimentation. (Cette procédure est représentée par SIGF dont le listing est en annexe (4)).

Le flux d'alimentation, dans la procédure STGF, peut être un liquide une vapeur, ou un méiange liquide-vapeur en équilibre. Ceci est détermine par les données suivantes: enthalpie, composition et pression, contenues dans la matrice STRM. Pour cette raison avant d'appeler STGF on doit spécifier toutes les données du flux d'alimentation.

e. Procédure simulant un étage avec soutirage STGS .

Comme pour l'étage d'alimentation, un étage avec soutirage peut être realisé avec la procédure SPLIT (présentée en annexe (3)), mais il est préférable d'utiliser une procédure similaire à STAGE avec soutirage STGS. Le listing de STGS est présenté en annexe (4).

f. Procédure simulant le bas de la colonne BOT .

La base de la colonne nécessite un traitement particulier pour les raisons suivantes :

1. Il y a un flux de chaleur entrant Q établissant le flux d'évaporation.
2. Le retenu (HOLDUP) est variable et considérable, la variation en enthalpie ne peut pas être négligeable.
3. Le flux liquide sortant du bas de la colonne est déterminé extérieurement (vanne) éventuellement par contrôle .

La variation du retenu en moles est :

$$\frac{dHL}{dt} = LI - LO - IV + RCT(21) = DHL .$$

Les flux sont représentés dans la fig -.III.5-

Bilan de chaleur :

$$\frac{d}{dt}(Hl\ hl) = Q - L0\ hl - IV\ Hv \quad .$$

En différenciant et en rearangeant on aura :

$$IV = \frac{Q - L0\ hl - HL\ d(hl)/dt - hl\ d(HL)/dt}{Hv} \quad .$$

En substituant d(HL)/dt=DHL on aura :

$$IV = \frac{Q - hl\ (\ L0 + DHL\ ) - DENL}{Hv} \quad .$$

Ou DENL = HL d(hl)/dt .

Ce calcul est effectué par la procédure BOI dont le listing est donné en annexe {4}.

g. Procédure simulant le rebouilleur REB .

Un design courant du rebouilleur d' une colonne de distillation est représenté dans la figure -III.6- .

Le liquide du bas de la colonne entre à la base de l'échangeur, au niveau des tubulures latérales, où il est partiellement vaporisé, puis le mélange des deux phases retourne dans la colonne, où il est séparé. L'écoulement du liquide du bas de colonne vers le rebouilleur, se fait automatiquement (Thermosyphon). La chaleur est apportée par un flux entrant par une vanne de contrôle. La vapeur se condense à l'extérieur du tube à une température (Tc) plus élevée que la température au bas de colonne (Tb). Ceci crée un gradient de température qui entraine le flux de chaleur H au bas de la colonne

La simulation d'une colonne de distillation doit inclure l'opération et les carracteristiques dynamiques du rebouilleur. Les deux principales car-actéristiques sont :

1. Le débit de la vapeur d'échauffement à travers la vanne de contrôle est :

$$W = A\ Cv\ SQRT(Ps (Ps - Pc)) \quad .$$

Ou A : Fraction d'ouverture de la vanne (%) .

Cv : Capacité de la vanne (moles/(min atm)) .

Ps : Pression de la vapeur d'échauffement à l'entrée de la vanne (atm)

Pc : Pression de la vapeur condensée (atm) .

Cette expression du débit est valable pour un écoulement subcritique (Pc > Ps/2 , c'est généralement le cas). La pression de condensation Pc est donnée par l'équation d' Antoine en fonction de la température :

$$Pc = EXP(C1 + C2/(Tc+C3)).$$

2 Dynamique du rebouilleur. Le rebouilleur peut être schématisé par la forme équivalente représentée dans la **fig -.III.7-** .

La température du tube est établie par l'équation différentielle suivante :

$$\frac{d}{dt}(Tm\ Uc) = (H1 - H2) \quad .$$

Ou Uc : Capacité calorifique du tube (PCU/C).

H1 : Flux de chaleur de la vapeur vers le tube (PCU/min).

H2 : Flux de chaleur du tube au fluide de bas de colonne (PCU/min).

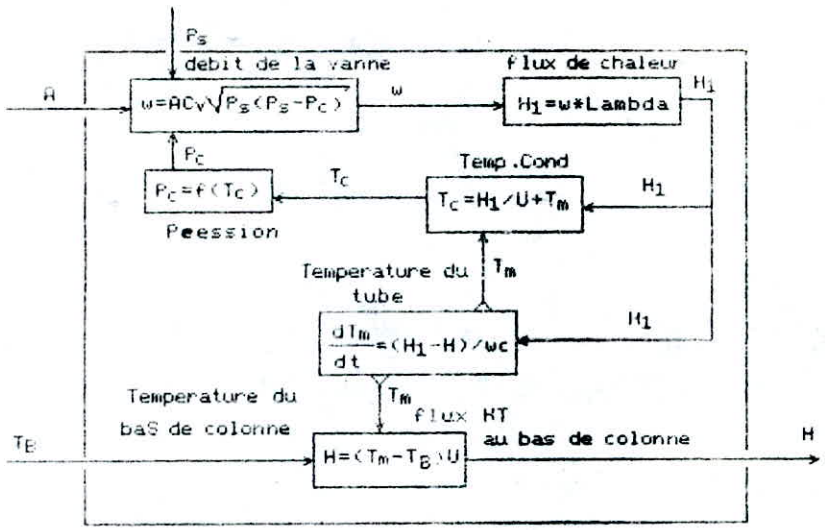
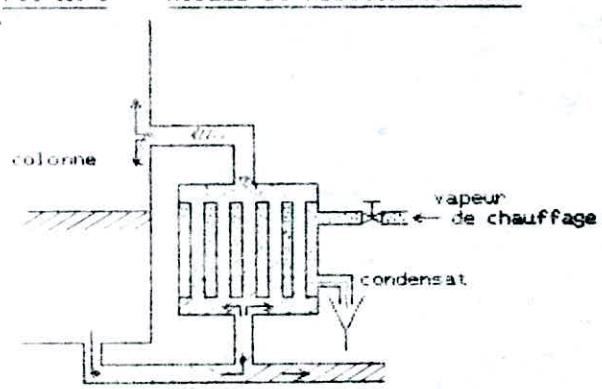
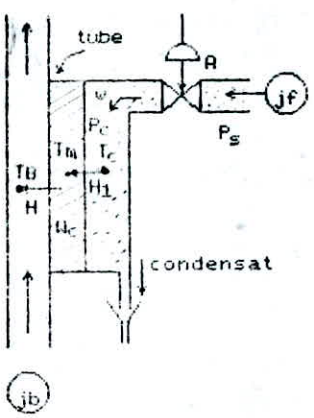


FIG III-8 MODELE DU REBOUILLEUR [1].



SCHEMA DU REBOUILLEUR DE LA COLONNE

FIG-III.6-



SCHEMA EQUIVALENT DU REBOUILLEUR

FIG-III.7-

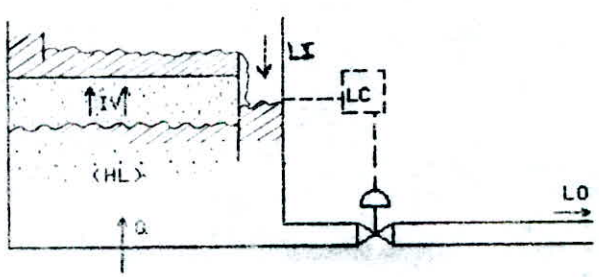


FIG-III.5- SCHEMA DU BAS DE LA COLONNE.

Comme première approximation, la résistance totale au transfert de chaleur est équivalente dans les deux côtés du tube :

$$H1 = U (Tc - Tm).$$

$$H2 = U (Tm - Tb).$$

Le modèle mathématique de ce rebouilleur est representé dans la figure -.III.8= .

Les variables d'entrée à ce modèle sont :

A : Le taux d'ouverture de la vanne (%)

Tb : La température du bas de colonne

Ps : La pression d'alimentation du rebouilleur en vapeur de chauffage (généralement constante)

Les variables de sortie seront :

H : Le flux de chaleur qui est utilisé par le bas de colonne puur générer le flux d'évaporation V.

Le flux de chaleur vers le tube H1 est considéré égal à la chaleur latente du flux vapeur : $H1 = V\ Lambda$ ou

Lambda : Chaleur latente molaire.

La température de condensation dérive de cette chaleur à partir de la température du tube Tm : $Tc = H1/U + Tm$

Ce modèle est représenté par la procédure REB dont le listing est en annexe (4).

h. Développement d'une procédure générale de simulation d'une colonne .

Les procédures décrites précédemment vont servir au développement de cette procédure. Le modèle de cette procédure est représenté dans la figure -III.9-. Dans le but de pouvoir ajouter des procédures pour d'autres apprareils et les utiliser dans un même système, nous avons adopté la structure suivante :

1. Définir un type nommé "Colonne" dans le header TIPE.H [6]. Ce type est une structuré contenant tous les paramètres définissant complètement une colonne, selon le modèle utilisé. Si, dans un programme , nous faisons la déclaration suivante :

Colonne Cptr;

Les paramètres définissant une colonne seront :

Cptr.n_pl : nombre de plateaux.

Cptr.etat_pl[i-1] : Etat du plateau numero i. (n : normal;
a : alimentation; s : soutirage.).

Cptr.h_pl[i] : Flux de chaleur avec l'extérieur au niveau du plateau i.

Cptr.pholdup[i] : Rétention au niveau du plateau i .

Cptr.htc[i] : Constante de temps hydraulique au niveau du plateau i . i=1,...,Cptr.n_pl.

Cptr.h : Flux de chaleur du rebouilleur vers le bas de colonne.

Cptr.Cv : Capacité de la vanne du rebouilleur.

Cptr.R : Fraction d'ouverture de la vanne du rebouilleur.

Cptr.wc : Capacité calorifique

Cptr.bholdup : Rétention du bas de colonne .

Cptr.tc : température du condenseur .

Cptr_holdup : Rétention du réservoir .

Cptr_h_holdup : flux de chaleur avec l'extérieur au niveau du réservoir de reflux

Cptr_ti[i] : Valeur initiale estimée des tempértures dans les flux liquides.

Cptr_hldpi[i] : valeur initiale estimée des flux liquides.

i=1,...,Cptr_n_pl;

Cptr_pcon : Pression à l'entrée du condenseur.

Cptr_Pbot : Pression au bas de colonne.

Cptr_deltatp : élévation de pression par étage.

Les autres paramètres sont utilisés dans la numérotation des flux (Ceci est commenté par la figure III 10-).

Tous param ètres doivent être introduits pour définir une colonne à simuler.

2 developper une procédure init_colonne permettant d'initialiser certaines variables pour pouvoir passer à la simulation. Le listing de init_colonne est présenté en annexe (6).

3. Développer une procédure permettant la simulation proprement dite de la colonne, dont le listing est inclus en annexe (6).

Des exemples d'utilisation de ce travail en mode programmation sont présentés en annexe (7).

Exemple: numerotation des flux
dans une colonne a 9 etages.

```
Cptr_n_pl=9;
Cptr.etat_pl="nnnasnnnn";
Cptr.i0=1;
Cptr.io=21;
Cptr.jf=25;
Cptr.i[11]=23;
Cptr.ia[1]=22;
Cptr.jc=24;
Cptr.jh=26;
Cptr.j0=1;
```

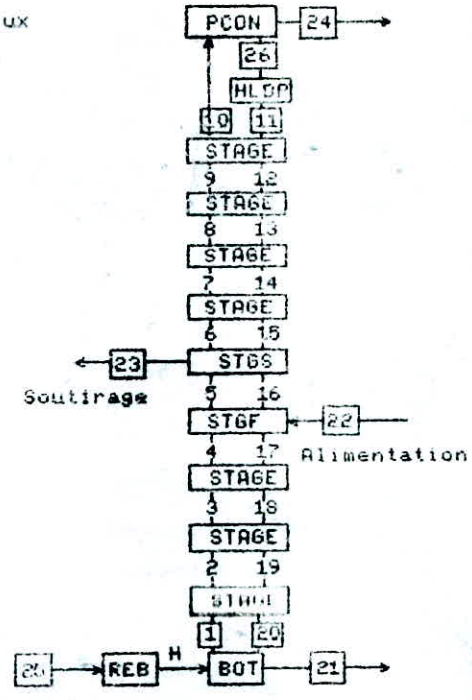


FIG -III-10- PROCEDURE EQUIVALENTE A UNE COLONNE A 9 ETAGE.

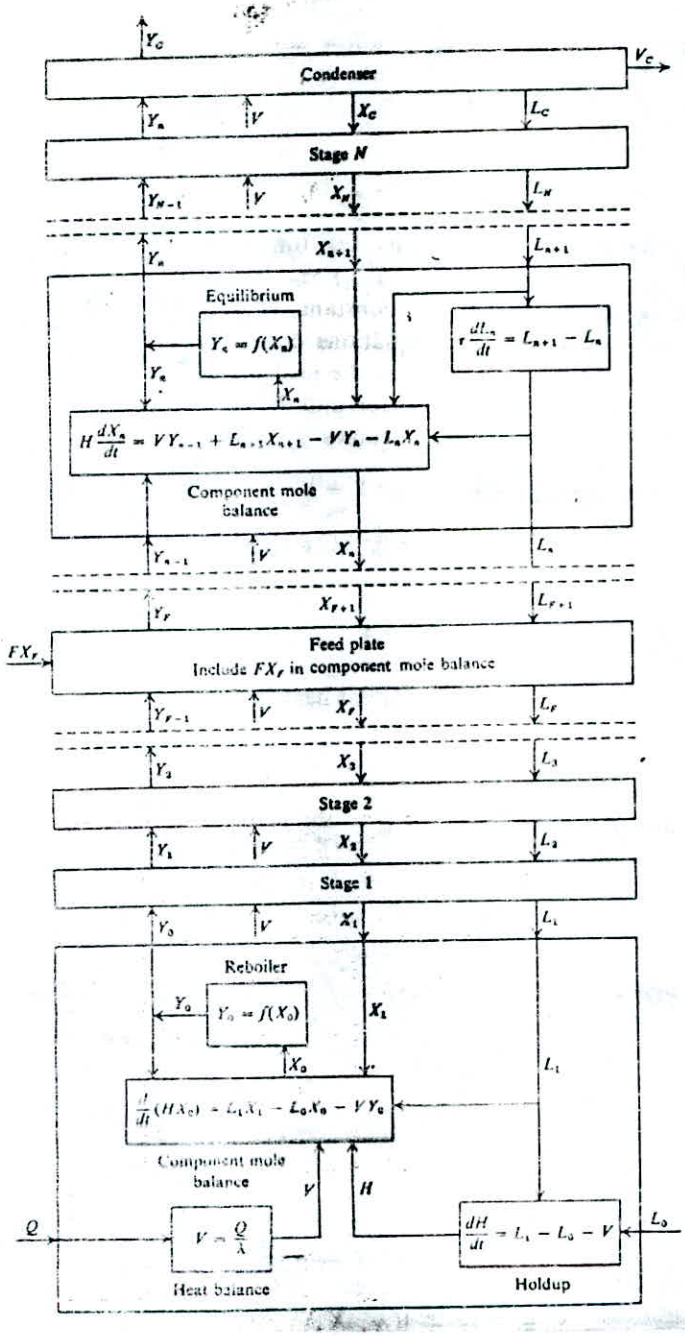


FIG III.9- SCHEMA OU MODELE D'UNE COLONNE A N ETAGE .

III.4. Réalisation d' une interface utilisateur interactive .

III.4.1. Introduction :

La librairie de procédures de simulation de procédés chimiques, réalisée précédemment, constitue un langage de simulation spécifique au génie chimique. Ces procédures peuvent être considérées comme des instructions. En utilisant le compilateur C ou même le compilateur Turbo PASCAL, on peut facilement réaliser des programmes de simulation de systèmes de génie-chimique (7).

Une librairie de procédures, faite dans un langage donné, n'est pas considérée comme un logiciel, mais comme un ensemble d'utilitaires dans un domaine donné. Donc, notre librairie est considérée comme un ensemble d'utilitaires en Turbo C pour la simulation en génie chimique. Pour donner à notre travail un aspect de logiciel, offrant toutes les possibilités que possèdent les logiciels actuels, nous avons réalisé une interface utilisateur interactive. Cette interface est réalisée en Turbo C, sans aucun utilitaire extérieur.

III.4.2. Possibilités de l'interface .

Cette interface utilisateur facilite l'introduction des paramètres d'un modèle d'appareil, et les données intervenant dans un système. Elle permet la manipulation de ces données dans des fichiers. La construction d'un système de génie chimique à simuler, se fait en reliant les symboles graphiques des appareils et en introduisant les paramètres et les données concernant ces appareils.

Une fois que le système est entièrement défini, on doit le sauvegarder. On peut rappe'ler un systeme sauvegarde, l'editer et le modifier. Le schém du système réalisé peut être imprimé. L'interface appelle un autre programme qui réalise la simulation proprement dite: SIMUL.EXE . MODULI.C et SIMUL.C ont une structure facilitant l'extension et l'introduction d'autres procédures simulant d'autres appareils. SIMUL.EXE permet aussi la présentation des résultats de la simulation graphique, numérique sur écran ou dans un fichier. Les résutats de la simulation peuvent être récupérés du fichier pour être traités ou imprimés.

III.4.3. Structures des programmes sources .

Pour réaliser un tel logiciel, en Turbo C, nous sommes obligé d'utiliser la gestion de projets qui est integrée dans l' E D I. La gestion de projets informatiques, proposée par Turbo C, est nécessaire quand le travail dépasse une certaine taille. Elle facilite grandement le développement d'application sérieuse. Le fichier projet MODULI.PRJ de l'interface est :

INTRO_PAR.C
G_GRP.C
G_GRP1.C
G_MENU.C
G_FICHIER.C

Nous avons 5 modules :

* G_GRP.C et G_GRP1.C : comportant certaines procédures graphiques.
* G_MENU.C : comportant les procédures de gestion de menus.
* G_FICHIER.C : comportant les procédures de gestion de fichiers.
* INIR_PAR.C : comportant les procédures d'introduction des paramètres, d'edition graphique et de numérotation automatique de flux.

Ces modules sont compilés chacun à part pour donner des modules objects (extension .OBJ), puis linker pour donner l'exécutable (.exe). Cette structure permet l'extension du logiciel en ajoutant d'autres procédures pour d'autres appareils et d'autres options.

La gestion de projet permet de reduire le temps de compilation durant le développement. Si une modification est faite au niveau d'un module, seul ce module est compilé, puis il est linké aux autres modules "objects" pour donner la nouvelle version exécutable du logiciel.

Les modules présentés précédemment utilisent d'autres fichiers headers avec l'extension .C, pour les procédures, ou .H, pour les prototypes de fonctions et certaines déclarations. Le listing de ces modules n'est pas presenté à cause de son grand volume. L'interface permet de définir un système, de stocker tous les paramètres et les données dans des fichiers, d'appeler le programme SIMUL.EXE,et lui transmettre tous ces fichiers. SIMUL.EXE fait tous les calculs et la simulation proprement dite et présente les résultats de la simulation.

III.4.4. Déscription de l'interface -

Le masque principal de notre interface utilisateur comporte un ensemble de menus (masque -I- (8)) . Pour accéder à un menu, il suffit d'appuyer sur la première lettre ou celle qui est en majuscule. Si ce choix comporte un sous-menu, on a une fenêtre qui apparaît comportant les autres choix (on a aussi des sous fenêtres), sinon, l'appui sur la touche exécute ce que l'utilisateur a choisi d'une manière interactive. Pour sortir d' un menue, il suffit d'appuyer sur ESC, pour quitter sur Q.

Certains menus apparaissant dans le masque, vont être développés ultérieurement. Pour les appareils il n'y a que la colonne de distillation, les alimentations et les capteurs qui sont opérationnels.

III.4.5. Exemple d'application avec l'interface utilisateur .

Pour expliquer l'utilisation de cette interface nous nous limiterons à un exemple.

* A partir du système, l'utilisateur lance l'exécution par :

```
a:\simul\moduli
```

Le masque -I- (8) apparaitra. Si nous voulons introduire dans notre système, une colonne de distillation nous devrons appuyer sur **A** pour se placer dans le menu Appareils, puis sur **C** pour choisir Colonne. Un curseur apparaitra,Nous positionnons ce dernier à l'aide des touches de déplacement à l'endroit ou nous désirons placer la colonne, et nous appuirons sur la barre d'espace.Le masqueII (8) apparaitra, nous pourrons alors soit introduire les paramètres de la colonne à partir du clavier ou choisir le menu Fichier en appuyant sur **Alt-F** et appeler un fichier existant. La liste des fichiers existants peut être consultée en tappant **L** de Liste. Une fois que tous les paramètres de la colonne sont définis, nous appuirons sur ESC. Un curseur apparaitra pour déterminer la taille du symbole graphique de la colonne. Nous appuirons sur la barre d'espace pour revenir au masque(I) avec le schéma de la colonne à l'endroit ou nous voulons le placer. Un curseur se déplace sur les flux internes au cas où nous voudrions placer des capteurs. Nous éditerons ensuite sur les entrées et les sorties, à l'aide du menu Edit R ou S respectivement. Nous définirons les alimentations et les soutirages (masqueIII (8)).

Les propriétés des constituants intévenant dans le système sont définies dans le menu Donnée (masqueIU {8}). Une fois que tout est défini, nous sauvegarderons (menu fichier) sous un nom donné. Pour simuler, nous taperons S du menu Simuler et nous introduirons le nom du système. Le programme SIMUL.EXE sera alors chargé. nous introduirons le temps maximal de simulation, le pas d'intégration, et nous choisirons le type de sortie des résultats. Cet exemple est illustre en annexe {8}.

Nous devons respecter un protocole durant l'utilisation de ce logiciel. Nous devons aussi respecter la structure du DYFLO presentée précédemment.

III.4.6. Conclusion

L'interface, présentée précédemment, n'est qu'une phase préliminaire à la réalisation d'un logiciel de flow-sheeting et de simulation. Elle présente les idées de base, et les algorithmes nécessaires à la realisation d'un tel logiciel . C'est une version 0.0 d'un logiciel,Il nè nous a pas été possible de faire une gestion d'erreurs et d'événement complète ni les messages d'aide. Il manque également le fichier d'aide nécessaire dans tout logiciel. Il n'y a pas de limites à la possibilité d'extension et d'amélioration d'un tel logiciel, on peut toujours ajouter des options, améliorer les procédures, du point de vue taille, rapidité et efficacité.

CONCLUSION

L'interet de l'informatique pour l'ingenieur de genie-chimique depasse le flow-sheeting pour atteindre la simulation des ecoulements et des phenomenes de transport complexes d'une part et apporter, des solutions technologiques tres interessantes aux problemes difficiles poses par le Genie-chimique surtout dans les technologies nouvelles. Les logiciels de simulation les plus repandus sur les PC sont : Process; Design II; ASPEN; BELSIM; CHEMCAD; PROSIM; HISYM-PETROLE ET GAZ; BIJACK-ECHANGEUR DE CHALEUR; ENSIM : REACTEURS CHIMIQUES; SPEEDUP; LSD ...

Avec la possibilite de les interfacer avec : dbase, lotus, autocad, base de donnees de corps purs estimation des proprietes.

Le developpement de tels logiciels est, generalement, realise par des equipes de recherche dans des ecoles et des universites.

Notre but etait de contribuer a cette revolution que connaissent toutes les branches de l'engineering.

Nous avons realise une librairie d'utilitaires (en langage C) pour la simulation en genie chimique. Cette derniere est representee par les cinq fichiers "headers" (NUMERIQU.C; ETAT.C; UNITETAT.C; UNITDYN.C; CONTROL.C). A partir de cette librairie, nous avons realise une procedure simulant une colonne de distillation suivant un modele propose par le DYFLO [1], qu'on a generalise. La structure de cette procedure se prete a la simulation modulaire simultanee, on peut realiser d'autres procedures pour d'autres appareils et les relier par la numerotation des flux.

Dans une seconde etape, nous avons developpe une interface utilisateur interactive permettant d'editer graphiquement le schema du systeme a simuler; d'introduire les parametres pour chaque appareil et de les manipuler dans des fichiers. Une fois que tout le systeme a simuler est defini on sauvegarde dans un fichier et l'interface appelle un autre programme qui fait le calcul et la presentation des resutats de la simulation. Durant le developpement nous avons rencontre des problemes de memoire et nous avons ete contraint de changer de structure, ainsi des programmes sont charges alternativement. Des exemples d'utilisation de la librairie, en mode programmation (colonne de distillation et compreseur) et en utilisant l'interface (colonne de distillation) sont presentes en en annexe {7}.

Nous avons tenu a ce que le travail realise se prete a l'extension et a l'amelioration dans le but d'elaborer un logiciel de simulation de procedes performant.

BIBLIOGRAPHIE

[1] ROGER G.E. FRANKS, Modeling and Simulation in Chemical Engineering, ED. WILEY-INTERSCIENCE 1972.

[2] M. FEIDT, Thermodynamique et Optimisation Energetique des Systemes Procedes ED. LAVOISIER 1987.

[3] V. KAFAROV, Methodes Cybernetiques et Technologie Chimique, ED. MIR 1974.

[4] S. DOMENECH, X. JOULIA, B. KOEHRET, Simulation et Optimisation en Genie des Procedes, Collection RECENTS PROGRES EN GENIE DES PROCEDES ED. LAVOISIER 1988.

[5] R. SIMIAN, M.PICARD, Logiciels et Materiel en Simulation des Procedes.

[6] J. GIL, X. JOULIA, A. BOUDEHEN, Analyse de Sensibilite et Mise en Evidence des Parametres Predominants d'un Procede.

[6] X. JOULIA, J.M. LELANN, B. KOEHRET, De la simulation Statique a la Simulation Dynamique.

[7] C. LOTTLER, G. BLOCH, J. RAGOT, Presentation d'un Logiciel de Simulation de Systemes Dynamiques.

[8] M. COEYTAUX, Simulation et Optimisation d'une Usine de Pate a Papier .

A N N E X E — 1 —

/* LISTING DE NUMERIQ.C

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

extern float t,dt,js,jn,dxa[500],xa[500];
float t,dt,js,jn,dxa[500],xa[500];
extern int   io,js4;
int   io,js4;
int nc=0;
extern float xt;
float xt;
float za[10],ya[10];


float INT(float *px,float dx)
{
 float x;
 x=*px;
  jn++;

 switch(io) {
             case 1: x+=dx*dt; break;
             case 2: switch(js) {
                                case 1: dxa[jn]=dx; x+=dx*dt;
                                        break;
                                case 2: x+=(dx-dxa[jn])*dt/2.
                                        break;
                                }
                                break;
             case 3:
             case 4: switch(js4) {
                                 case 1: xa[jn]=x; dxa[jn]=dx;
                                         x+=dx*dt; break;
                                 case 2:
                                 case 3: dxa[jn]+=2.*dx;
                                         x=xa[jn]+dx*dt;
                                         break;
                                 case 4: dxa[jn]=(dxa[jn]+dx)/6.;
                                         x=xa[jn]+dxa[jn]*dt;
                                         break;
                                 }
             }
  *px=x;
  return(x);
 }
```

```
float INTI(float *ptd,float *pdtd,int iod)
{
 float td,dtd,za[10],ya[10];
 td=*ptd; dtd=*pdtd; io=iod; jn=0;
switch(io) {
            case 1: js=2; dt=dtd; td+=dt; t=td; break;
            case 2: js++; if (js==3) js=1; if (js==2) break;
                    dt=dtd; td+=dt; t=td; break;
            case 3:
            case 4: js4++; if (js4==5) js4=1;
                    if (js4==1) { dt=dtd/2.; td+=dt; t=td;
                                  break; }
                    if (js4==3) { td+=dt; dt=2.*dt; t=td;
                                  break; }
            }
*ptd=td; *pdtd=dtd;
return(t);
}

int CONV(float *px,float *py,int nr)
{
 float x,y;
 x=*px; y=*py;
 if (fabs((x-y)/(x+y))<.0001) { x=y; nc=1; }
 else {
        if (nc<=1) { za[nr]=x; ya[nr]=y; x=y; nc=2; }
        else {
              xt=(za[nr]*y-ya[nr]*x)/(za[nr]-x+y-ya[nr]);
              za[nr]=x; ya[nr]=y; x=xt;
              }
      }
*px=x; *py=y;
return(nc);
}

int CPS(float *px,float *pxc,float r)
{
 int nc=2; float x,xc;
 x=*px; xc=*pxc;
 if (fabs((x-xc)/(x+xc))<.0001) nc=1;
 x+=(xc-x)*r;
 *px=x; *pxc=xc;
 return(nc);
}

float DER(float y,float x,int i)
{
 float ay[10],ax[10],dydx;
 if (t<=0.) { ay[i]=x; ax[i]=x; }
 else {
       if ((js4==4)||(js==2)) { dydx=(y-ay[i])/(x-ax[i]);
                                ay[i]=x;
                          ax[i]=x; } }
 return(dydx);
```

A N N E X E - 2 -

```
/*                    LISTING DE ETAT.C                              */

#include "numeriqu.c"

int ncf,ncl;
int lstr;
float strm[300][24],data[20][10],rct[22];

void ENTHL(int i)
{
 int n; float hv=0.0;
 for (n=ncf; n<=ncl; n++)
              hv  strm[i][n]*(data[n][7]+data[n][8]*strm[i][22]);
 strm[i][23]=hv*strm[i][22];
 }

 void ENTHV(int i)
 {
  int n; float hv=0.0;
  for (n=ncf; n<=ncl; n++)
 hv+=strm[i][n]*((data[n][4]+data[n][5]*strm[i][22])*strm[i][22]
                  +data[n][6]);
 strm[i][23]=hv;
 }

 void TEMP(int i,int l)
 {
  int j4,j5,n; float sax=0.0,sbx=0.0,slm=0.0,t,td;
  j4=4+l; j5=5+l;
  for (n=ncf; n<=ncl; n++) { sax+=data[n][j4]*strm[i][n];
                             sbx+=data[n][j5]*strm[i][n];
                    if (l!=3) slm+=data[n][6]*strm[i][n]; }
  t=strm[i][22];
  do td=(strm[i][23]-slm)/(sax+t*sbx); while(CONV(&t,&td,1)==2);
  strm[i][22]=t;
 }

 void ACTY(int n)
 {
  int j;
  for (j=1; j<=20; j++) data[j][9]=1.;
 }

 void EQUIL(int il,int iv)
 {
  int n; float sum,sdy,dy,yer;
 do { sum=sdy=0.0; ACTY(il);
 for (n=ncf; n<=ncl; n++) {
 strm[iv][n]=exp(data[n][1]+data[n][2]/(strm[il][22]+
     data[n][3]))*strm[il][n]/strm[il][24]*data[n][9];
 sum+=strm[iv][n];
 dy=-strm[iv][n]*data[n][2]/((strm[il][22]+data[n][3])*
                             (strm[il][22]+data[n][3]));
 sdy+=dy; }
 yer=1.0-sum; strm[il][22]+=yer/sdy; } while (fabs(yer)>=.001);
 strm[iv][22]=strm[il][22];
 }
```

```
void DEWPT(int iv,int il)
{
 int n; float sum,sdx,pn,dx,xer;
 do { sum=sdx=0.0; ACTY(il);
 for (n=ncf; n<=ncl; n++) {
        pn=exp(data[n][1]+data[n][2]/(strm[il][22]+data[n][3]));
        strm[il][n]=strm[iv][n]*strm[il][24]/data[n][9]/pn;
        sum+=strm[il][n];
        dx=strm[il][n]*data[n][2]/((strm[il][22]+data[n][3])*
                                   (strm[il][22]+data[n][3]));
        sdx+=dx;          }
 xer=1.0-sum;
 strm[il][22]=strm[il][22]+xer/sdx; } while(fabs(xer)>=.001);
}
```

A N N E X E - 3 -

```
/*                LISTING DE UNITETAT.C                              */

#include "etat.c"

void SUM(int i,int j,int k,int l)
{
 int n; float enin;
 strm[k][21]=strm[i][21]+strm[j][21];

 for (n=ncf; n<=ncl; n++)
 strm[k][n]=(strm[i][21]*strm[i][n]+strm[j][21]*strm[j][n])/
                                                    strm[k][21];
 enin=strm[i][23]*strm[i][21]+strm[j][23]*strm[j][21];
 strm[k][23]=enin/strm[k][21]; TEMP(k,l);
}

void SPLIT(int j,int k,int m,float rkj)
{
 int n;
 strm[k][21]=strm[j][21]*rkj;
 strm[m][21]=strm[j][21]-strm[k][21];
 strm[m][22]=strm[j][22]; strm[k][22]=strm[j][22];
 strm[k][23]=strm[j][23]; strm[m][23]=strm[j][23];
 for (n=ncf; n<=ncl; n++) strm[k][n]=strm[m][n]=strm[j][n];
}

void HTEXCH(int i,int jo,float ht,int l)
{
 int n; float qf;
 qf=ht/strm[i][21];
 for (n=ncf; n<=ncl; n++) strm[jo][n]=strm[i][n];
 strm[jo][23]=strm[i][23]+qf;
 strm[jo][21]=strm[i][21];
 TEMP(jo,l);
}

void CSHE(int isi,int iso,int iti,int ito,float ua,int ips,
          int ipt,int nsf,int nsl,int ntf,int ntl)
{
 int j; float q,dt1,dt2,r,dta,qc;
 for (j=nsf; j<=nsl; j++) strm[iso][j]=strm[isi][j];
 for (j=ntf; j<=ntl; j++) strm[ito][j]=strm[iti][j];
 strm[ito][21]=strm[iti][21]; strm[iso][21]=strm[isi][21];
 q=strm[isi][21]*(strm[isi][23]-strm[iso][23]);
 do { ncf=nsf; ncl=nsl; HTEXCH(isi,iso,-q,ips);
      ncf=ntf; ncl=ntl; HTEXCH(iti,ito,q,ipt);
      dt1=strm[iso][22]-strm[iti][22];
      dt2=strm[isi][22]-strm[ito][22];
      r=dt1/dt2; dta=(dt1-dt2)/(log(fabs(r))); qc=ua*dta; }
 while(CPS(&q,&qc,r/(1+r))==2);
}
```

```
void FLASH(int i,int jv,int jl,float ht)
{
 int n; float qf,r,sum,sdy,h,dy,yer,r1;
 qf=ht/strm[i][21]; r=strm[jv][21]/strm[i][21]; ACTY(jl);
 do { do { sum=sdy=0.0; for (n=ncf; n<=ncl; n++) {
h=exp(data[n][1]+data[n][2]/(strm[jl][22]+data[n][3]))*
                                    data[n][9]/strm[jl][24];
strm[jv][n]=strm[i][n]*h/(1.+r*(h-1.));
strm[jl][n]=strm[jv][n]/h;
sum+=strm[jv][n];
dy=-strm[jv][n]*data[n][2]/((strm[jl][22]+data[n][3])*
                                   (strm[jl][22]+data[n][3]));
sdy+=dy; } yer=1.-sum; strm[jl][22]+=yer/sdy;
                                   } while (fabs(yer)>=.001);
strm[jv][22]=strm[jl][22]; ENTHV(jv); ENTHL(jl);
r1=(strm[i][23]+qf-strm[jl][23])/(strm[jv][23]-strm[jl][23]);
if (r1<=0.0) { HTEXCH(i,jl,ht,3); strm[jv][21]=0.0; return; }
if (r1>=1.0) { HTEXCH(i,jv,ht,0); strm[jl][21]=0.0; return; }
} while (CONV(&r,&r1,1)==2);
strm[jv][21]=r*strm[i][21]; strm[jl][21]=(1.0-r)*strm[i][21];
}

void CVBOIL(int i,int jv,float ht)
{
 int n;
 DEWPT(i,jv);
 for (n=ncf; n<=ncl; n++) strm[jv][n]=strm[i][n];
 ENTHV(jv); strm[jv][21]=ht/(strm[jv][23]-strm[i][23]);
 strm[i][21]=strm[jv][21];
}

void PCON(int i,int jv,int jl,float tc)
{
 float r,tmp,sdy,sum,h,den,dy,yer; int n;
 strm[jv][22]=strm[jl][22]=tc; r=strm[jv][21]/strm[i][21];
 if (r>.95) r=.95; tmp=strm[i][22]; EQUIL(i,300);
 strm[i][22]=tmp;
 if (tmp<=tc) r=1.;
 if (strm[300][22]>tc) {
             for (n=ncf; n<=ncl; n++) strm[jl][n]=strm[i][n];
             r=0.;
 ENTHV(jv); ENTHL(jl); strm[jv][21]=strm[i][21]*r;
 strm[jl][21]=strm[i][21]-strm[jv][21]; }
 else {
 do { sdy=sum=0.0; ACTY(jl);
 for (n=ncf; n<=ncl; n++) {
h=exp(data[n][1]+data[n][2]/(tc+data[n][3]))*
                      data[n][9]/strm[jl][24];
den=1.+r*(h-1.); strm[jv][n]=strm[i][n]*h/den;
                     strm[jl][n]=strm[jv][n]/h;
dy=-strm[jv][n]*(h-1.)/den; sum+=strm[jv][n]; sdy+=dy;
if (sdy>=0.0) sdy=-1.0; yer=1.0-sum; r+=yer/sdy;
                                   } while (fabs(yer)>=.001);
ENTHV(jv); ENTHL(jl); strm[jv][21]=strm[i][21]*r;
strm[jl][21]=strm[i][21]-strm[jv][21]; }
}
```

A N N E X E — 4 —

```
/*                LISTING DE UNITDYNE.C                                */

#include"unitetat.c"

void REB(float a,float h,float cv,float wc,int jf,int jb)
{
 float h1=0,tc=0,pc=0,w=0,dtm=0,tm=0,str=0,aaa=0; int nn;
 double aa=0;
 if (lstr==1) {
            w=h/strm[jf][23];
            pc=strm[jf][24]-w*w/(a*a*cv*cv*strm[jf][24]);
            tc=strm[jf][2]/(log10(pc)-strm[jf][1])-strm[jf][3];
            tm=(strm[jb][22]+tc)/2;
            strm[jf][20]=h/(tc-tm);
            strm[jf][21]=w;
               }   else    {
            do { h1=strm[jf][21]*strm[jf][23];
              tc=h1/strm[jf][20]+tm;
                 aaa=tc+strm[jf][3];
                 aa=strm[jf][2]/aaa;
                 aaa=strm[jf][1]+aa;
                 pc=exp(aaa);
                 w=a*cv*sqrt(strm[jf][24]*fabs(strm[jf][24]-pc));
                 str=strm[jf][21];
                 nn=CONV(&str,&w,1);
                 strm[jf][21]=str;
               } while (nn==2);

            aa=tm-strm[jb][22];
            aa=strm[jf][20]*(tm-strm[jb][22]);
            dtm=(h1-h)/wc;
            INT(&tm,dtm);
           }
 }

void BOT(int li,int lo,int iv,float q,float hl)
{
 float denl,qp,dhl,den,hk,fni,fno,dern,dn; int n;
 if ((js4==4) || (js==2) || (lstr==1)) strm[lo][20]=strm[lo][23];
 EQUIL(lo,iv); ENTHL(lo); ENTHV(iv);
 denl=(strm[lo][23]-strm[lo][20])*hl/dt;
 qp=q+strm[li][21]*strm[li][23]+rct[22];
 dhl=strm[li][21]-strm[lo][21]-strm[iv][21]+rct[21];
 den=strm[iv][23]-strm[lo][23];
 strm[iv][21]=(qp-strm[lo][23]*(strm[li][21]+rct[21])-denl)/den;
 if (strm[iv][21]<0.0) strm[iv][21]=0.0;
 if (lstr==1) return;
 for (n=ncf; n<=ncl; n++) { hk=strm[iv][n]/strm[lo][n];
 if (hk>5.) { dn=strm[iv][21]+strm[lo][21]/hk;
 strm[iv][n]=(strm[li][21]*strm[li][n]+rct[n])/dn;
 strm[lo][n]=strm[iv][n]/hk; dern=0.0;
 INT(&strm[lo][n],dern); INT(&hl,dhl); return; }
 fni=strm[li][21]*strm[li][n]+rct[n];
 fno=strm[lo][21]*strm[lo][n]+strm[iv][21]*strm[iv][n];
 dern=(fni-fno-strm[lo][n]*dhl)/hl;
 INT(&strm[lo][n],dern); } INT(&hl,dhl); return;
 }
```

```
void STAGE(int i1,int i2,int il,int iv,float h,float hl,
           float htc)
{
 int n; float flin,hin,dl,hk,cnin,dern;
 EQUIL(il,iv); ENTHL(il); ENTHV(iv);
 flin=strm[i1][21]+strm[i2][21]+rct[21];
 hin=strm[i1][23]*strm[i1][21]+strm[i2][23]*strm[i2][21]+h+
                                                    rct[22];

 strm[iv][21]=(hin-flin*strm[il][23])/
              (strm[iv][23]-strm[il][23]);
 if (lstr==1) return;
 dl=(flin-strm[iv][21]-strm[il][21])/htc;
 for (n=ncf; n<=ncl; n++) { hk=strm[iv][n]/strm[il][n];
 cnin=strm[i1][21]*strm[i1][n]+strm[i2][21]*strm[i2][n]+rct[n];
 if (hk>5.0) { strm[iv][n]=cnin/(strm[iv][21]+strm[il][21]/hk);
               strm[il][n]=strm[iv][n]/hk; dern=0.0;
               INT(&strm[il][n],dern); INT(&strm[il][21],dl);
               return; }
 dern=(cnin-flin*strm[il][n]-strm[iv][21]*
             (strm[iv][n]-strm[il][n]))/hl;
 INT(&strm[il][n],dern); }
 INT(&strm[il][21],dl);
 return;
 }


void STGF(int i1,int i2,int i3,int il,int iv,float h,
          float hl,float htc)
{
 int n; float flin,hin,dl,hk,cnf,cnin,dern;
 EQUIL(il,iv); ENTHL(il); ENTHV(iv);
 flin=strm[i1][21]+strm[i2][21]+strm[i3][21]+rct[21];
 hin=strm[i1][23]*strm[i1][21]+strm[i2][23]*strm[i2][21]+
                    h+rct[22]+strm[i3][21]*strm[i3][23];
 strm[iv][21]=(hin-flin*strm[il][23])/
              (strm[iv][23]-strm[il][23]);
 if (lstr==1) return;
 dl=(flin-strm[iv][21]-strm[il][21])/htc;
 for (n=ncf; n<=ncl; n++) { hk=strm[iv][n]/strm[il][n];
 cnf=strm[i3][21]*strm[i3][n];
 cnin=strm[i1][21]*strm[i1][n]+strm[i2][21]*strm[i2][n]+
                                          rct[n]+cnf;
 if (hk>5.0) { strm[iv][n]=cnin/(strm[iv][21]+strm[il][21]/hk);
               strm[il][n]=strm[iv][n]/hk; dern=0.0;
               INT(&strm[il][n],dern); INT(&strm[il][21],dl);
               return; }
 dern=(cnin-flin*strm[il][n]-strm[iv][21]*
             (strm[iv][n]-strm[il][n]))/hl;
 INT(&strm[il][n],dern);}
 INT(&strm[il][21],dl);
 return;
 }
```

```
void STGS(int i1,int i2,int il,int iv,int is,float h,
          float hl,float htc)
{
  int n; float flin,hin,dl,hk,cnin,dern;
  EQUIL(il,iv); ENTHL(il); ENTHV(iv);
  flin=strm[i1][21]+strm[i2][21]+rct[21];
  hin=strm[i1][23]*strm[i1][21]+strm[i2][23]*strm[i2][21]+
                                                h+rct[22];
  strm[iv][21]=(hin-flin*strm[il][23])/(strm[iv][23]-
                                          strm[il][23]);
  if (lstr==1) return;
  dl=(flin-strm[iv][21]-strm[is][21]-strm[il][21])/htc;
  for (n=ncf; n<=ncl; n++) { hk=strm[iv][n]/strm[il][n];
  cnin=strm[i1][21]*strm[i1][n]+strm[i2][21]*strm[i2][n]+rct[n];
  if (hk>5.0) { strm[iv][n]=cnin/(strm[iv][21]+
                (strm[il][21]+strm[is][21])/hk);
                strm[il][n]=strm[iv][n]/hk; dern=0.0;
                INT(&strm[il][n],dern); INT(&strm[il][21],dl);
                for (n=ncf; n<=ncl; n++)
                                          strm[is][n]=strm[il][n];
                for (n=22; n<=24; n++) strm[is][n]=strm[il][n];
      return; }
  dern=(cnin-flin*strm[il][n]-strm[iv][21]*
             (strm[iv][n]-strm[il][n]))/hl;
  INT(&strm[il][n],dern); }
                INT(&strm[il][21],dl);
                for (n=ncf; n<=ncl; n++)
                                          strm[is][n]=strm[il][n];
                for (n=22; n<=24; n++) strm[is][n]=strm[il][n];
  return;
}

void HLDP(int i,int io,int l,float hl,float q)
{
  int n; float dhl,hin,den,dfx,dx;
  if (lstr==1) { if (l==3) ENTHL(io);
            if (l==0) ENTHV(io); return; }
  dhl=strm[i][21]-strm[io][21]+rct[21];
  hin=strm[i][21]*strm[i][23]+q+rct[22];
  den=(hin-strm[io][23]*(strm[io][21]+dhl))/hl;
  for (n=ncf; n<=ncl; n++) {
                dfx=strm[i][21]*(strm[i][n]-strm[io][n]);
                dx=(dfx+rct[n]-strm[io][n]*rct[21])/hl;
                INT(&strm[io][n],dx); }
                INT(&hl,dhl);
                INT(&strm[io][23],den); TEMP(io,l);
                        return;
}
```

A N N E X E — 5 —

Listing de CONTROL.C

```
#include "etat.h"

float out,xin,tc,gain,span,vim,epc;

void TFS1(float *pout,float xin,float tc,float gain)
{float out;
    float dtf;out=*pout;
    dtf=(gain * xin - out)/tc;
    INT(&out,dtf);
    *pout=out;
}

void TFS2(float *pout,float xin,float a1,float a2,float gain)
{float out;
    int j1;
    float dtf;out=*pout;
    jn++;
    j1=jn;
    dtf=(gain*xin-out)*a1-xa[j1]*a2;
    INT(&out,xa[j1]);
    INT(&xa[j1],dtf);
    *pout=out;
}

float CONTR2(float vi,float zr,float rng,float sp,float axn,float pb,
              float rpt,float oi,float *pdi)
{ float co,di;
di=*pdi;
    span=fabs(rng-zr);
    vim=vi;
    if(vi>rng) vim=rng;
    if(vi<zr) vim=zr;
    epc=100.*axn*(vim-sp)/span;
    di=epc*rpt*100./pb;
    if(oi>100.) oi=100.;
    if(oi<0.)oi=0.;
    co=100./pb*epc+oi;
    if(co>=100.) co=100.;
    if(co<0.) co=0.;
*pdi=di;
    return(co);
}

CONTR1(float vi,float co,float zr,float rng,float sp,float axn,float pb,
       float oi)
{
    span=fabs(rng-zr);
    vim=vi;
    if(vi>rng) vim=rng;
    if(vi<zr) vim=zr;
    epc=100.*axn*(vim-sp)/span;
    co=100./pb*epc+oi;
    [illegible]
    [illegible]
```

Le régulateur est un RPI et la vanne a une fonction de transfert du second ordre.

Le listing du programme simulant le compresseur est donne en annexe. Le résultat de la simulation fournit les profils du débit W1 et de la pression P2 en fonction du temps, ainsi que celle d'autres paramètres.

ANNEXE - 6 -

```
/*     LISTING DES PROCEDURES init_colonne ET colone                    *

void colone(void)
{ int i=0,j=0,k=0,l,n;
  l=Cptr.i0+2*Cptr.n_pl+1;
  strm[Cptr.lo][21]=strm[l][21]-strm[Cptr.i0][21];
  REB(Cptr.a,Cptr.h,Cptr.cv,Cptr.wc,Cptr.jf,Cptr.lo);
  BOT(1,Cptr.lo,Cptr.i0,Cptr.h,Cptr.bholdup);
  for (n=1; n<=Cptr.n_pl; n++) { k=Cptr.i0+n-1;
       switch(Cptr.etat_pl[n-1])  {
case 'n':
case 'N':
STAGE(k,l-k,l+1-k,k+1,Cptr.h_pl[n],Cptr.pholdup[n],Cptr.htc[n]
break;
case 'a':
case 'A': j++;
STGF(k,l-k,Cptr.ja[j],l+1-k,k+1,Cptr.h_pl[n],Cptr.pholdup[n],
                                                    Cptr.htc[n]);
break;
case 's':
case 'S': i++;
STGS(k,l-k,l+i-k,k+1,Cptr.js[i],Cptr.h_pl[n],Cptr.pholdup[n],
                                                    Cptr.htc[n]);
break;
                                        }
                                        }
PCON(k+1,Cptr.je,Cptr.jh,Cptr.tc);
HLDP(Cptr.jh,k+2,3,Cptr.holdup,Cptr.h_holdup);
}


void init_colonne(void)
{ int j,jd,m,n;
  m=Cptr.i0+Cptr.n_pl+1;
  strm[Cptr.i0][21]=Cptr.es_vp_bupi;
  for (j=1 ; j<=Cptr.n_pl+1; j++) {
                          strm[m+j-1][21]=Cptr.hldpi[j];
                          strm[m+j-1][22]=Cptr.ti[j]; }
  strm[Cptr.lo][21]=Cptr.hldpi[j]; strm[Cptr.lo][22]=Cptr.ti[j
  for (j=Cptr.i0; j<=m; j++) { jd=2*m-j;
  for (n=ncf; n<=ncl; n++)
                          strm[jd][n]=strm[Cptr.ja[Cptr.j0]][n]
              strm[jd][24]=strm[j][24]=Cptr.pbot-j*Cptr.deltap
                              if (j!=m) EQUIL(jd,j);
                              ENTHL(jd);
                              }
  strm[Cptr.jh][24]=Cptr.pcon;
  EQUIL(Cptr.lo,Cptr.i0);
}
```

```
/*              LISTING DE LA PROCEDURE Simulation

void Simulation(void)
{
 do {
      for (i=1; i<=N_Appareils; i++) {
                                                switch(Appareils[i])
 case 1: if (lstr==1) init_colonne(); colone(); break;
/*       .  .  .              .  .  .              .  .  .
/*       .  .  .              .  .  .              .  .  .
/*       .  .  .              .  .  .              .  .  .
                                                                    }

                                        }
        sortie_resultat();
        INTI(&t,&dt,1);
        lstr=0;
      } while (t<tmax && kbhit()==0);
}
```

ANNEXE - 7 -

Exemple d'utilisation de la librairie

Le procédé à analyser consiste en un compresseur conduit par une turbine à vapeur, comprimant un gaz dans une large gamme de pression.

La partie la plus complexe dans le procédé est le compresseur . Il y a quatre variables associées au compresseur, à savoir:

- la vitesse de rotation N,
- la pression d'entrée de gaz P1,
- la pression de sortie de gaz P2,
- le débit W1 du gaz.

La vitesse et les pressions sont imposées , alors que le débit en découle.La relation numérique entre ces variables pour un compresseur particulier est , généralement , représentée sous forme d'une série d'abaques , montrant la relation entre la pression de refoulement P2 et le débit W1, à différentes valeurs de la vitesse N et pour une gamme de pression d'aspiration P1. Ces abaques peuvent être combinés en une seule courbe, dite " courbe caractéristique " du compresseur, reliant deux paramètres : W1/N et PR/N**2 où PR est le taux de pression P2/P1.

Quand on se rapproche du taux de pression critique, pour une vitesse N0, le débit diminue rapidement. Au delà du taux de pression critique le compresseur "cahote" : le débit W1 tombe à zero et garde cette valeur jusqu'à ce que la valeur de PR/N**2 rechute .

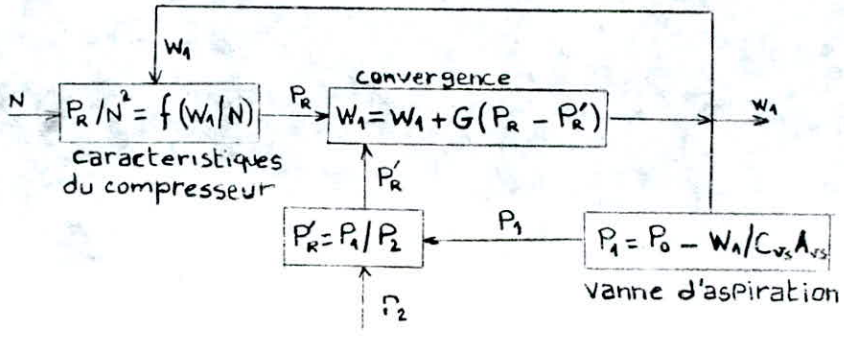
La pression d'aspiration P1 résulte d'une perte de charges à travers la vanne d'aspiration . En supposant que le gaz est, initialement, à la pression atmosphérique P0, la pression P1 sera:

$$P1 = P0 - [\ W1/(Cvs*Avs)\ ]^{**2}$$

Avec Cvs: coefficient de la vanne d'aspiration,

Avs: ouverture de la vanne d'aspiration.

En supposant que la vitesse de rotation N et la pression de refoulement P2, sont connues à partir d'autres sections du modèle mathématique, un schéma du flux d'information pour le compresseur et la vanne d'aspiration est représente dans la figure suivante:



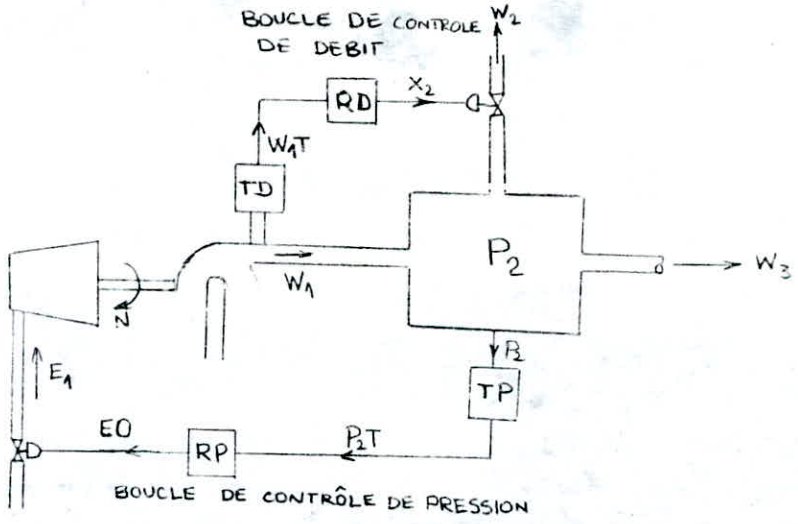
Il y a deux boucles avec W1 comme point commun. La boucle supérieure passe par la courbe du compresseur, avec comme entrée W1/N et comme sortie PR/N**2. Dans la boucle inférieure, l'équation de la vanne donne P1 en ayant W1 comme entrée. On peut alors calculer le ratio P'R=P1/P2. Les valeurs des ratios PR et P'R sont appariées en faisant des itérations sur W1 dans l'équation de convergence:

$$W1 = W1+G\ (\ PR-P'R\ ) \qquad \text{avec} \quad G=2000$$

La convergence est accélérée par la procédure CONV.

La turbine est conduite par un flux de vapeur, ayant une énergie E1. Le travail total E2 fourni par la turbine est le résultat de l'action simultanée de la vitesse N et de l'écoulement du gaz.

Le système de contrôle propose,est représenté dans la figure suivante:



Il comporte deux boucles : l'une contrôle la pression de la vapeur actionnant la turbine et l'autre, contrôle le débit, en agissant sur une vanne de bypass.

Dans la boucle de contrôle de débit, le transmetteur a une fonction de transfert du premier ordre soit:

W1T/W1 = 1/(tau * s + 1) avec tau=0.13 sec.

Ce qui correspond à l'appel de la procédure TFN1 dans le programme.

Le régulateur est du type 2 (RPI), ce qui nécessite l'appel de la procédure CONTR2.

La vanne a une fonction de transfert du second ordre:

X2/X2P = 1/(s**2 + 1.6 s + 1)

Dans la boucle de contrôle de pression, le transmetteur a une fonction de transfert du premier ordre:

P2T/P2 = 1/(0.25 s + 1)

```
/*                  Listing   de   COMPRES.C                          */


#include<stdarg.h>
#include<dos.h>
#include "control.c"
#include "procedl.c"
#include "colgrpl.c"


float n,w1,w2,w3,w1n,w1t,w1ts,p1,p2,p2t,p2ts,pr,prp,erp,w1c,e1,e2,eo,x2,x2p,
      pbx,pbe,rptmx,rptme,oix,oie,dix,die,av,dn,dp2,tfin;
float w1nx[15],w1ny[15],avx[15],avy[15],tprnt;
int nc1,nc,nc2,jj;

main()
{
tprnt=0.;
    DATA();
   printf("-------------------------------------------------------------------------------\n");
    printf("  t  |  p2  |   w2    |   w1    |   n   |  x2  |  x2p  |  w1t  |  e1  | p2t  \n");
    printf("-----|------|---------|---------|-------|------|-------|-------|------|------\n");
    do{
    do{
        w1n=w1/n;
        p1=14.7-pow((w1/(180000.*.5)),2.);
        prp=p2/p1;
        if((w1<=0.) && ((prp/pow(n,2))<(1.*pow10(-7))) &&
                       ((prp/pow(n,2))<(7.6*pow10(-8))))
        w1=55000.;
        else if((prp/pow(n,2)) < (1.*pow10(-7))){
          pr=pow(n,2)*FUN1(w1n,11,w1nx,w1ny);
          erp=pr-prp;
          w1c=w1+erp*2000.;
          nc1=CONV(&w1,&w1c,1); }
        else if(((prp/pow(n,2))>=(7.6*pow10(-8))) ||
                ((prp/pow(n,2))>(1.*pow10(-7))))
        { w1=0.;
          break; }
      } while(nc1==2);
    e2=pow(n,3)*1.*pow10(-10)*(6.+4.4*w1/n);
    if(t>5.) w3=20000.;
    x2p=CONTR2(w1t/*,x2p*/,0.,1.*pow10(5),w1ts,-1.,pbx,rptmx,oix,&dix);
    av=FUN1(x2,11,avx,avy);
    w2=666.7*av*p2;
    eo=CONTR2(p2t,0.,200.,p2ts,-1.,pbe,rptme,oie,&die);
    dn=(e1-e2)*19150./n;
    dp2=(w1-w2-w3)*8.*pow10(-4);
    nc2=PRNTF(.5,tfin,p2,w2,w1,n,x2,x2p,w1t,e1,p2t);
    switch(nc2){
              case 1: INTI(&t,&dt,2);
                      INT(&n,dn);
                      TFN1(&w1t,w1,.13,1.);
                      TFN1(&p2t,p2,.25,1.);
                      TFN2(&x2,x2p,1.,1.6,.01);
                      TFN2(&e1,eo,59.,10.75,50.);
                      INT(&oix,dix);
                      INT(&oie,die);
```

```
                    INT(&p2,dp2);
                    break;
            case 2: break;
            }
  }while((t<=tfin)&&(nc2!=2));


TA(void)

nt i;
1nx[1]=0.;w1nx[2]=1.;w1nx[3]=2.;w1nx[4]=3.;w1nx[5]=4.;
1nx[6]=5.;w1nx[7]=5.5;w1nx[8]=6.;w1nx[9]=6.5;w1nx[10]=7.;w1nx[11]=7.2;
1ny[1]=7.6;w1ny[2]=8.5;w1ny[3]=9.2;w1ny[4]=9.8;
1ny[5]=10.;w1ny[6]=9.8;w1ny[7]=9.2;
1ny[8]=8.2;w1ny[9]=6.3;w1ny[10]=2.8;w1ny[11]=0.;
or (i=1;i<=11;i++) w1ny[i]=w1ny[i]*(1.*pow10(-8));
vx[1]=0.;avx[2]=.1;avx[3]=.2;avx[4]=.3;avx[5]=.4;
vx[6]=.5;avx[7]=.6;avx[8]=.7;avx[9]=.8;avx[10]=.9;avx[11]=1.;
vy[1]=0.;avy[2]=.17;avy[3]=.225;avy[4]=.266;
vy[5]=.3;avy[6]=.374;avy[7]=.46;avy[8]=.56;avy[9]=.67;
vy[10]=.82;avy[11]=1.;
=8920.;
!=90.;
 =55200.;
 =40200.;
 0.;
 =.1;
 in=50.;
 e=120.;
 tme=.5;
 ts=90.;
 e=46.8;
 x=100.;
 tmx=.5;
-ts=55200.;
 x=20.;
=t=90.;
 =2340.;
=t=54250.;
 =.2;
```

ANNEXE - 8 -

MASQUE - I -

Appareils

Alimentations

Colonnes

Separateurs

Reacteurs

Vannes

Pompes

cOmpresseurs

reGulateurs

capTeurs

Derivation

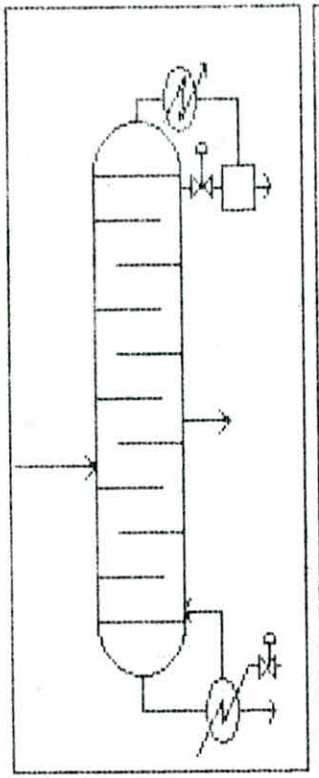
recYclage

Fichier | Construire | Verifier | Editer | Simuler | Données | Options

MASQUE - I -

Fichier

MASQUE - II -



Nombre de plateau=9
Etat des palateaux : Initialisation
Flux de chaleur avec l'exterieur au niveau du plateau=0
Holdup du plateau=5
Constante de temps hydrolique du plateau=0.1
Flux de chaleur du rebouilleur vers le bat de colone=3.75e+04
Capacite de la vanne du rebouilleur=2.5
Fraction d'ouverture de la vanne du rebouilleur=0.303
Capacite calorifique du tube du rebouilleur=1
Holdup du bas de colonne=40
Temperature de condensation=75
Holdup du reservoir du reflux=50
Flux de chaleur avec l'exterieur du reservoire de reflux=0
Pression initiale du flux de condensat a la sortie du condenseur=1
Valeures initiales estimees des temperatures dans les flux liquide=80
Valeures initiales estimer des flux liquides=3
Estimation du flux de vaporisation initiale=5
Pression au bat de colonne=1.11
Elevation de pression par etage=0.01 * /

Fichier

Nombre de composees=3
Coefficient d'Antoine C1=15.4 à
Coefficient d'Antoine C2=-5310 à
Coefficient d'Antoine C3=273 à
Coefficient d'enthalpie vapeur Av=6.5 à
Coefficient d'enthalpie vapeur Bv=0.01 à
Chaleur latente a 0C=7500 à
Coefficient d'Enthalpie liquide Al=16 à
Coefficient d'Enthalpie liquide Bl=0.03 à
Activite nu=1 *

MASQUE

MASQUE - III -

Fichier

Numero du premier composer=
Numeros du dernier composer=
Fraction molaire du composer=
Debit molaire=
Temperature=
Enthalpie=
Pression=

Fichier

Numero du premier composer=
Numeros du dernier composer=
Fraction molaire du composer=
Debit molaire=
Temperature=
Enthalpie=
Pression=

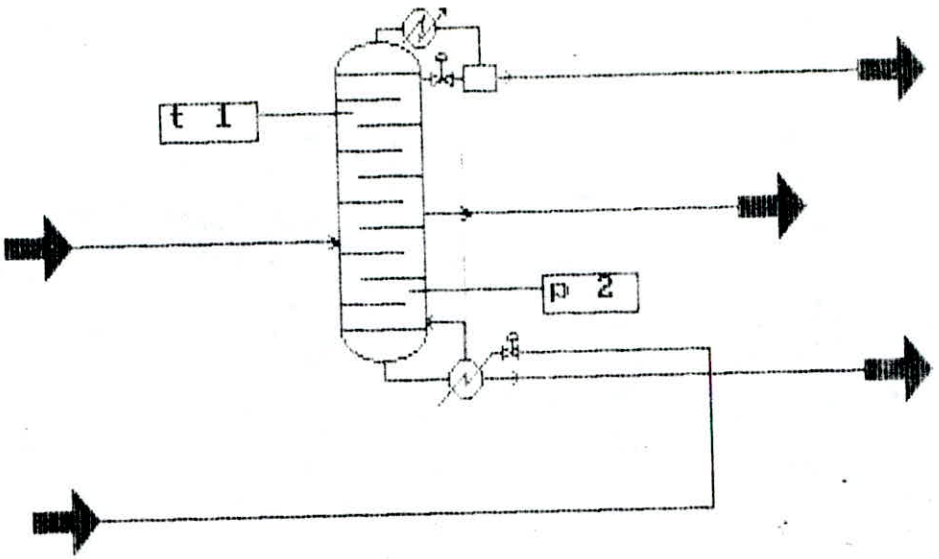
MASQUE - V -

Fichier | Construire | Verifier | Editer | Simuler | Donnees | Options

Appareils

- Alimentations
- Colonnes
- Separateurs
- Reacteurs
- Vannes
- Pompes
- cOmpresseurs
- reGulateurs
- capTeurs
- Derivation
- recYclage

SIMULATION D'UNE COLONNE DE DISTILLATION



Path : A:SYS3.DAT_