MASTER EN SCIENCES INFORMATIQUES

Élaboration d'une architecture générale pour un système de résolution de problèmes, application au domaine des statistiques officielles

Lust, Frédéric; Wey, Alain

Award date:
1993

Awarding institution:
Universite de Namur

Facultés Notre-Dame de la Paix de Namur
Institut d'informatique

Elaboration d'une architecture générale pour un système de résolution de problèmes, application au domaine des statistiques officielles.

Lust Frédéric
Wey Alain

mémoire présenté pour l'obtention du diplôme
de licencié en informatique

Année Académique 1992-1993

Résumé

Ce mémoire concerne l'élaboration d'une architecture destinée à modéliser la résolution de problèmes statistiques (cette architecture peut cependant s'appliquer à des d'autres types de problèmes). Cette architecture est née suite à la demande par le Bureau du Royaume Uni des Statistiques Officielles de résoudre quelques procédures travaillant sur des séries statistiques. Après ces différentes résolutions demandées, nous avons essayé d'évoluer vers une généralisation de ce type de problème et ceci en nous basant sur l'architecture adoptée lors du projet EISI. Cependant, nous avons constaté que cette architecture n'était pas tout à fait adéquate pour la modélisation de problèmes concrets spécifiques, ceci étant dû au fait que ce projet étudiait en particulier non pas des données concrètes mais plutôt l'informations tournant autour de ces données. Nous avons dès lors établi une architecture annexe spécifique à notre problème tout en gardant en point de mire ce projet EISI. Ce mémoire reprend donc toutes ces étapes qui ont été nécessaires pour aboutir à une première architecture satisfaisante.

Abstract

This Memoire is concerned with the elaboration of a framework for modelling problem-solving in Official Statistics. The rationale for the work was based on problems identified by the United Kingdom Central Statistics Office with regard to estimate variances in economic time-series. The goal of the project was to find resolutions to instances of the problems and to generalise the solutions for eventual representation within the framework of the EISI system. It was realised that this framework was not completely sufficient, because of the emphasis of the EISI system on Statistical Meta-Information. It was therefore necessary to develop a parallel framework wtih the extensions necessary for supporting the specific problems. The design, contruction and operation of this framework is the subject of this Memoire.

Remerciements

L'élaboration et la rédaction de ce mémoire n'ont pas toujours été très aisées. En effet, parfois, l'énervement et la lassitude entraînaient un certain découragement. Heureusement la présence, les conseils et les encouragements de certaines personnes nous ont remonté le moral et remis dans le droit chemin.

Nous remercions donc, madame Noirhomme-Fraiture, pour son suivi, sa disponibilité et ses remarques pertinentes à propos de l'ensemble du mémoire.

Nous tenons également à remercier notre maître de stage, Monsieur de Vaney, pour la bonne humeur qu'il a su faire régner pendant et après notre stage. Merci également, pour toutes les bonnes idées et remarques dont il nous a fait part. Nous remercions vivement Monsieur Yao Chen pour nous avoir acceuilli au coeur de ses bureaux durant ces cinq mois de stage.

Remerciements également à Monsieur Leclercq, qui nous a conseillé à propos de la rédaction de tout le côté implémentation de notre solution.

Nous remercions particulièrement chaleureusement Madame et Monsieur Knepper-Hirt qui nous ont acceuilli durant cinq mois chez eux et qui nous ont fourni un logement de qualité ainsi qu'une bonne humeur permanente.

Finalement, merci à tous les membres de notre entourage ainsi qu'aux personnes ayant collaboré de près ou de loin à la mise-en-oeuvre de ce mémoire et dont nous ne pouvons citer malheureusement tous les noms.

Table des matières

CHAPITRE 1

Introduction

1.1. Synopsis.

Le but principal de ce mémoire est de concevoir et de présenter l'architecture d'un système destiné à la résolution de problèmes statistiques spécifiques (en l'occurence, ici, l'étude de séries statistiques biaisées). Cependant au fil de la lecture de ce mémoire, on peut noter un recul certain par rapport à ces problèmes statistiques particuliers.

Au-delà de la résolution pure et simple de ce problème, nous avons voulu pousser plus loin nos investigations afin d'envisager une architecture "générique" pour la résolution de problèmes appartenant à des champs d'applications plus vastes. Le principe de base de cette architecture repose sur l'un des concepts de base de l'informatique : la division des tâches. En l'occurence, dans ce travail, nous avons approfondi la notion de planification des tâches.

Nous avons donc élaboré une architecture s'articulant autour de ce concept de "planning". Cette architecture propose des mécanismes de résolution et de manipulation interne assez évolués. En effet, nous avons voulu intégrer une notion d'indépendance (nous n'utiliserons pas le terme d'intelligence car celui-ci est vaste, vague et trop vite cité dans de nombreux exposés) en ce sens que l'architecture proposée essaie de prendre du recul vis-à-vis du champ d'application pour lequel elle pourrait être utilisée et vis-à-vis d'un utilisateur potentiel.

L'apport de ce mémoire peut s'exprimer principalement de façon pratique grâce à l'élaboration d'un système prototype. D'un point de vue théorique, ce document permet de remettre à jour un sujet moins utilisé de nos jours, de l'étudier et de l'approfondir, tout ceci en parallèle avec la théorie existante dans ce domaine. Il nous permet également de voir quelles sont les tendances générales relatives à cette méthode de planification.

1.2. Le stage[1].

Le stage s'est déroulé en trois phases. Premièrement, les premières semaines ont été consacrées à l'étude du monde des statistiques officielles c'est-à-dire l'étude de différentes publications (enquêtes, méthodologies, etc...) et une réflexion sur les problèmes rencontrés dans cette branche des statistiques. Durant cette phase, nous avons également abordé le système EISI. Ce système est un outil "Case Based Reasoning" permettant aux statisticiens de disposer de l'analyse du contexte pour un problème particulier.

Dans une deuxième phase, nous avons été confronté à un problème relatif aux statistiques officielles : les biais d'estimation dans les séries statistiques. Notre but était d'analyser ce problème et de concevoir un outil graphique permettant la visualisation de ces biais grâce à un ensemble de graphismes. Cet outil permet aux statisticiens

[1] Ce stage s'est déroulé dans les batiments de World Systems, 28 rue Gillen, à Howald (Grand-Duché de Luxembourg) et ceci du 01/09/92 au 31/01/93.

responsables de se forger une idée quant à l'interprétation de ces "erreurs". Cette phase s'est déroulé de la façon suivante :

1. étude du problème dans son contexte
2. conception d'une solution
3. implémentation de cette solution

Finalement, lors de la troisième phase de notre stage, nous avons essayé de généraliser notre étude. En effet, nous avons élaboré une architecture capable de résoudre des problèmes statistiques comme celui étudié lors de la deuxième phase du stage. Cette architecture se base notamment sur EISI et également sur la théorie du planning. Suite à cela, nous avons réfléchi à propos de l'évolution future possible pour un tel système.

1.3. Organisation de ce document.

Le chapitre 2 expose en détails l'étude d'un problème concret (étude de séries statistiques biaisées). Il se décompose en trois grandes parties : la présentation du contexte du problème, l'énoncé du problème et la résolution du problème. Une quatrième partie relative à l'implémentation de la solution se trouve en annexe. Ce chapitre permet de bien appréhender un type de problème complexe auquel les statisticiens peuvent être confrontés et que nous allons essayer de généraliser.

Dans le chapitre 3, nous présentons les concepts généraux du système EISI. La présentation de ce système est intéressante car il a servi comme base à notre travail.

Le chapitre 4 concerne la théorie se cachant derrière la notion de "planning". On y découvre la méthodologie et la typologie des différents plannings. On y développe également une vue plus générale et plus théorique des problèmes statistiques. Finalement, un lien est établi entre ces deux concepts de "planification" et de "statistiques".

Le chapitre 5 développe en détails le système que nous avons élaboré pour la résolution automatique de problèmes dans le domaine des statistiques officielles. Dans celui-ci, on applique les exemples et la théorie vus dans les chapitres précédent. Il s'agit sans aucun doute du chapitre le plus difficile à lire du fait du niveau de détail assez élevé et également du fait de sa complexité. Cependant, il s'agit du centre nerveux de ce mémoire et, en tant que tel, il se doit d'être le plus complet possible.

Finalement le chapitre 6 se compose d'une critique du système, suivie d'une intégration future possible de celui-ci avec le système EISI et enfin de l'apport que pourrait y apporter un autre système, "MolGen" dont nous trouverons une présentation dans ce chapitre 6. L'objectif de ces intégrations est d'étudier des solutions permettant d'améliorer notre système.

Les annexes, quant à elles, développent tout le côté implémentation c'est-à-dire les descriptions détaillées, les algorithmes, les programmes et les résultats.

CHAPITRE 2

"Exposé d'un cas concret"

2.1. Introduction.

L'objectif final de ce mémoire est de développer un système qui soit capable de résoudre des problèmes comme ceux que l'on trouve dans le domaine des statistiques officielles. Pour bien comprendre l'utilité et la nécessité d'un tel système, il est important préalablement de se familiariser avec les types de problèmes que les statisticiens sont amenés à résoudre. Ces types de problèmes sont divers, allant de la simple comparaison de données jusqu'à des problèmes beaucoup plus complexes.

Dans ce chapitre, nous allons développer et présenter une méthodologie de résolution possible d'un problème complexe que les statisticiens peuvent être amenés à résoudre. Il s'agit de la recherche de biais dans des séries de variables nationales comptables périodiques. Pour bien cerner le problème, la suite du chapitre sera structurée comme suit : dans la première partie, nous allons présenter un exemple de variable nationale comptable périodique, le produit domestique brut. Cette variable a été choisie

car elle est bien représentative des autres variables. La section suivante sera constituée de la définition précise du problème et la dernière section contiendra la méthodologie suivie pour la résolution et l'implémentation du problème.

C'est suite à l'étude et à la résolution du problème que nous avons essayé de pousser plus loin nos recherches. Nous avons essayé de voir s'il n'était pas possible de sortir du contexte propre au problème afin de produire une architecture de résolution pouvant s'appliquer à un champ de problèmes plus vaste. Ce chapitre est relativement important par le fait qu'il nous permet de nous rattacher continuellement à un cas réel pratique. Nous y avons donc porté une réelle attention afin de le rédiger de la façon la plus complète possible.

2.2. Contexte relatif au problème.

Le problème développé est un problème réel, complexe et qui est intéressant en de nombreux points de vue. L'énoncé précis du problème sera donné dans la seconde section. Comme nous l'avons signalé dans l'introduction, il se rapporte au traitement des séries comptables nationales périodiques, et pour bien comprendre l'objectif poursuivi par la résolution du problème, il est important auparavant de comprendre ce qu'est une série comptable nationale périodique. Un bon exemple en est le produit domestique brut d'un pays (PDB) . Dans la section suivante, nous allons présenter le produit domestique brut, les différentes caractéristiques de ce produit et enfin les difficultés rencontrées lors de l'estimation de celui-ci.

Le choix du PDB comme exemple est motivé par le fait que celui-ci présente les principales caractéristiques des autres séries statistiques, et les problèmes rencontrés lors de la collecte de ses composants et de son estimation peuvent se retrouver dans les autres variables. L'étude du PDB permet de bien comprendre les données du problème ainsi que les raisons qui poussent à vouloir le solutionner.

2.2.1. Le produit domestique brut.

2.2.1.1. Définition du produit domestique brut

Le produit domestique brut d'un pays ("The Gross Domestic Product"), que nous noterons le "PDB", permet, pour une période donnée, de quantifier l'activité économique de ce pays durant cette période. En plus de quantifier l'activité économique, le calcul du "PDB" permet également aux statisticiens et aux économistes de calculer le taux de croissance de l'activité économique du pays. Le taux de croissance est calculé en comparant le niveau de la période concernée soit avec le niveau de la même période de l'année précédente, soit avec le niveau de la période précédent la période concernée.

Dans notre travail, nous nous intéressons au "PDB" pour le Royaume-Uni, mais le calcul dans ce pays est très proche du calcul dans de nombreux autres pays. Comme les estimations du "PDB" portent sur l'activité économique totale du pays, il est donc constitué d'un grand nombre de composants et ce n'est donc pas une chose facile que de calculer sa valeur sans erreurs. Dans la suite de cette section, nous allons développer les différentes manières de calculer le produit domestique brut. Dans la section suivante, nous expliquerons les différentes unités qui permettent d'exprimer les résultats, ensuite nous parlerons de la publication des estimations, et en dernière partie, nous développerons les problèmes rencontrés par les statisticiens pour effectuer ces estimations en limitant les erreurs, ainsi que la conséquence de ces problèmes.

2.2.1.1.1. Le calcul du "Produit Domestique Brut".

Le calcul du "PDB" peut s'effectuer de trois manières différentes: par la mesure des dépenses, par la mesure de rentrées ou par la mesure de la production.

a. La mesure des dépenses (PDB(D); D = dépenses)

Il s'agit du total de toutes les dépenses du pays pendant la période concernée, soit en consommant les marchandises ainsi que les services produits, soit en additionnant au capital le coût des importations. Cette mesure reprend les composants suivants:

1. Le total des dépenses des consommateurs résidents dans le pays pendant la période concernée.
2. Le total des dépenses du gouvernement pendant la période concernée.
3. La somme des dépenses sur le capital fixe.
4. La valeur d'ajout aux stocks des matériaux et travaux en progrès.
5. La somme des exportations moins la somme des importations.

b. La mesure des rentrées (PDB(R) ; R= Revenu)

Il s'agit du total de tous les revenus dérivés de la production dans le pays. Cette mesure reprend les composants suivants:

1. Le total des rentrées pour l'emploi.
2. Le total des rentrées pour les indépendants.
3. Les profits commerciaux des entreprises.
4. Les surplus commerciaux des entreprises du gouvernement général.
5. D'autres surplus commerciaux. ex: Les rentes.

c. La mesure des productions (PDB(P) ; P= Production)

Il s'agit de la somme des valeurs additionnées de toutes les activités qui produisent des biens et des services dans le pays. Cette mesure reprend les composants suivants:

1. Les sorties des industries de production.

2. Les sorties de l'agriculture et des activités du secteur primaire.
3. Les productions de construction.
4. Les sorties des entreprises de distribution et de service.

En principe, si les résultats étaient exacts, les trois résultats seraient égaux. Malheureusement, comme il est pratiquement impossible de calculer ces valeurs sans erreurs, et ce pour un certains nombre de raisons que nous développerons dans la suite du travail, on trouve généralement un résultat différent pour chacun des trois calculs. Pour remédier à ce problème, et pour avoir une solution générale, on calcule une quatrième valeur, qui est la moyenne du PDB(D), du PDB(R) et du PDB(P).Cette moyenne sera notée PDB(M) (M= Moyen).

2.2.1.2. Expression des résultats.

Les résultats sont des valeurs qui peuvent être exprimées dans l'unité monétaire du pays. Dans notre cas, il s'agit de l'unité monétaire du Royaume-Uni, c'est-à-dire la Livre Sterling. Cependant, afin de pouvoir comparer ensemble des résultats de périodes et d'années différentes, il est intéressant de pouvoir remettre toutes les valeurs sur une même base. La valeur d'une unité monétaire variant d'une année à l'autre à cause de l'index des prix, pour pouvoir faire cette comparaison, on transforme les résultats au prix d'une année de base fixe, ce qui permet d'éliminer les effets des changements de prix. Cette transformation n'est pas évidente car il n'existe pas de formule unique pour tous les composants. Généralement, on réévalue les composants principaux de chaque période grâce à un indicateur spécifique. Le choix de l'année de base est faite arbitrairement, mais depuis quelques années, l'année de base est changée tous les 5 ans et on choisit les années multiples de cinq.

On peut encore exprimer le résultat d'une troisième manière; On ne prend plus l'unité monétaire comme unité, mais un nombre indexé. Cette unité exprime la mesure pour la période concernée, comme un pourcentage de la mesure pour l'année de base. Pour cette mesure, on peut soit partir du résultat exprimé au prix courant, ou du résultat exprimé au prix de l'année de base. La valeur du produit domestique brut pour l'année de base sera égale à 100 et pour les autres années, proportionnelle à la valeur de l'année de base.

Exemple: Si la valeur pour l'année de base est 120000 et une valeur pour une autre année égale à 150000, L'expression du résultat en nombre indexé sera 100 pour l'année de base et l25 pour la seconde année.

Le fait de transformer les résultats dans différentes unités entraîne encore un risque d'erreurs ou d'imprécisions dont on discutera plus loin.

2.2.1.3. La publication des résultats.

Au Royaume-Uni, les estimations du produit domestique brut sont publiées entre autres dans la revue "ECONOMIC TRENDS" publiée tous les trimestres par le bureau central des statistiques (Le CSO). Les publications se font aux mois de janvier, avril, juillet et octobre et reprennent les estimations du produit domestique brut ainsi que d'autres résultats statistiques pour des périodes ou des années. Jusqu'à présent, nous avons parlé de périodes sans préciser la durée d'une période. Dans le revue "Economic Trends", une période est un trimestre, le premier trimestre d'une année reprenant les trois premiers mois de cette année et ainsi de suite. La première publication pour le produit domestique brut pour un trimestre apparaît généralement quatre mois après le fin de ce trimestre. Les résultats pour le premier trimestre d'une année seront publiés au mois de juillet de cette même année, les résultats pour le second trimestre au mois d'octobre, pour le troisième trimestre au mois de janvier de l'année suivante et enfin pour le dernier trimestre au mois d'avril de l'année suivante.

Après la première publication, les résultats pour une période sont republiés chaque trimestre. Il arrive fréquemment que lors des premières publications, les valeurs soient modifiées par rapport à la publication précédente. Cela est dû au fait que lors des premières publications les résultats ne sont pas connus avec exactitude, et qu'au court du temps, les résultats se complètent et se stabilisent. Les révisions sont généralement plus importantes aux premières publications et s'amenuisent par la suite. Les résultats sont généralement stabilisés après cinq ans, mais peuvent exceptionnellement s'étendre sur une plus longue période. Les raisons de ces révisions seront développées dans la section suivante.

En plus des publications dans la revue "The Economic Trends", le bureau central des statistiques met à la disposition des personnes les estimations sous forme interprétable par un ordinateur.

2.2.1.4. Les révisions du produit domestique brut

Dans les sections précédentes, on a souvent fait apparaître que les estimations du PDB sont difficiles à faire avec exactitude et que même après la première publication, elles étaient encore soumises à de nombreux changements. Dans cette partie, on développera les principaux problèmes rencontrés par les statisticiens responsables des estimations et on en déduira les conséquences sur l'exactitude du résultat ou sur le retard du résultat définitif. Cet énoncé reprend les difficultés principales, mais n'est certainement pas exhaustif. Il permettra cependant de comprendre pourquoi les résultats sont soumis à de nombreuses révisions et pourquoi la précision du résultat final est à considérer avec prudence.

Nous allons en premier lieu développer les raisons qui font que les résultats définitifs ne sont pas trouvés lors de la première publication, et ensuite nous donnerons plusieurs raisons qui font que la précision des résultats définitifs sont à prendre avec prudence.

2.2.1.4.1. Les révisions des estimations:

1. Lors des premières publications, certains des composants du produit domestique brut pour une période donnée sont incomplets et parfois même pas du tout connus.

Les économistes d'un pays veulent pouvoir accéder aux résultats aussi tôt que possible, c'est pourquoi la première publication du PDB pour un trimestre se réalise quatre mois après la fin de ce trimestre. Or, il arrive souvent que quatre mois après la fin d'un trimestre, on n'ait pas encore récolté toutes les données. Pour y remédier, on peut estimer les composants manquants sur base d'enquêtes auprès d'un échantillon

représentatif. Mais ces données ne sont pas toujours très précises, par exemple, si l'échantillon pris n'est pas suffisamment représentatif. D'autres mesures peuvent être simplement extrapolées ou interpolées sur base des dernières valeurs disponibles. Dans ce cas également il y a un risque d'imprécision si la formule utilisée pour faire ces interpolations ou ces extrapolations n'est pas fiable.

Exemple: Les résultats de certaines entreprises ne sont parfois pas disponibles pendant une longue période après la fin d'un trimestre. Dans ce cas, il faut l'estimer selon une des deux méthodes précédentes.

Lors de la seconde publication, les informations sont déjà plus complètes, et les informations supplémentaires obtenues peuvent être intégrées dans le total en remplacement des valeurs estimées. Après plus de trois trimestres suivants la première estimation, les informations sont plus ou moins complètes mais dans certains cas il faut attendre plusieurs années avant d'avoir les dernières valeurs.

Les révisions développées dans ce point proviennent du cheminement normal des révisions résultant du processus de collecte de données. Ces révisions peuvent être importantes, mais elles s'étendent généralement sur un petit nombre de périodes, sauf pour certaines exceptions.

2. Les changements méthodologiques.

Contrairement aux changements réguliers résultant de la collecte des informations, il existe des révisions irrégulières résultant dans la méthode de mesure. Ces changements peuvent être de nature diverse, les révisions en résultant sont moins importantes, mais elles s'étendent sur un plus grand nombre de périodes. Voici quelques exemples de changements méthodologiques:

- On peut découvrir qu'une source de données couramment utilisée est devenue inadéquate, et on est contraint d'en utiliser une autre qui fournira des renseignements plus proches de la réalité.

- Un composant du produit domestique brut, qui faisait partie d'un composant plus important peut avoir pris de l'importance et il faut le considérer comme un élément à lui tout seul.

- Il peut y avoir des changements de définition de certains composants, ce qui conduira à réviser ses valeurs en fonction de la nouvelle définition.

Remarque: Les changements méthodologiques peuvent apparaître tout le long de l'année, mais pour des raisons de facilité, ils ne sont généralement introduits dans les estimations qu'au mois d'octobre. On peut donc s'attendre à des changements plus importants lors de ce mois là. Certains changements importants qui ne peuvent pas attendre le mois d'octobre sont introduits au mois d'avril. Les changements lors de ce mois peuvent être plus importants que pour les publications du mois de janvier et de juillet.

3. Les effets du changement de l'année de base choisie comme référence pour les estimations du produit domestique brut à prix constant.

Tous les cinq ans, on change l'année prise comme année de base. Les rebasages sont nécessaires pour assurer que les comparaisons entre des valeurs du PDB ne soient pas déformées par l'utilisation d'une structure de prix d'une période trop ancienne. Lors d'un changement, il faut réévaluer les résultats antérieurs en fonction de cette nouvelle année de base. Cela a relativement peu d'effets pour les estimations à prix constant ou prix courant, mais cela peut conduire à un changement des poids relatifs de certains composants dans le calcul du produit domestique brut en nombre indexé. Un composant pourrait avoir un poids relatif de 6 par exemple et suite à un changement de base, passer à un poids relatif de 5.

2.2.1.4.2. Les problèmes de précisions des résultats:

1. Les estimations sont tirées de nombreuses sources différentes.

On ne sait malheureusement pas trouver tous les renseignements nécessaires dans la même source de données, il faut se servir d'un grand nombre de sources différentes et qui sont parfois totalement indépendantes l'une de l'autre. Cela entraîne un problème de consistance entre les données. Certaines données pourront être redondantes, ou au contraire oubliées. Certaines valeurs pourront se trouver dans une rubrique dans une source et être dans une autre rubrique dans une autre source.

Exemple:

- Pour l'estimation du PDB par la mesure des revenus, la source principale sont les données de "l'Inland Revenue", qui regroupe les analyses statistiques des enregistrements des taxes, mais la contribution des compagnies financières et des institutions n'est pas disponible par cette source, il faut donc les estimer par les données d'une variété d'autres sources.

- Le problème est encore plus important pour l'estimation du PDB par la mesure des dépenses, car contrairement à l'estimation des entrées on ne dispose pas de source de référence principale, mais d'un grand nombre de petites sources indépendantes. Il y a encore moins d'assurance d'obtenir une consistance interne valable.

2. De nombreuses sources qui servent pour les estimations sont des sources publiées annuellement mais pas trimestriellement. On ne dispose donc de ces sources qu'une fois par année d'où difficultés de faire les estimations chaque trimestre.

Pour connaître les valeurs entre deux années, on a différentes solutions:

- On effectue des enquêtes sur des échantillons représentatifs de la population étudiée. Pour cela, il faut être certains que l'échantillon soit bien représentatif, mais même dans ce cas, de petites erreurs peuvent apparaître.

Exemple: La publication de "L'Inland Revenue" est annuelle, mais cet organisme effectue des enquêtes trimestrielles sur les profits des entreprises les plus importantes.

- On effectue une interpolation ou une extrapolation en utilisant un certain nombre d'indicateurs comme mesure approximative des résultats, comme par exemple les dernières estimations reçues. Cela pose évidemment le problème de l'exactitude des données qui sont estimées. De plus, cette méthode ne prévoit pas le cas où un changement important intervient en cours d'année.

Exemple: On effectue une interpolation ou une extrapolation pour avoir une idée de l'activité des indépendants dans des industries particulières.

3. Par la mesure des dépenses, il peut y avoir des erreurs de timing pour plusieurs composants dans l'estimation du PDB .

Un grand nombre d'erreurs de timing peuvent survenir, parmi les plus importantes, on trouve:

- Les erreurs dûes au fait qu'une entreprise peut utiliser une comptabilité annuelle qui se termine à une autre époque que la comptabilité annuelle du gouvernement.

- Le moment auquel l'enregistrement d'une transaction doit être effectuée peut être important. Pour la vente d'un article par exemple, il peut y avoir une différence de temps entre le moment où la commande de l'article a été faite et le moment où celui-ci est payé. Le problème se pose de savoir à quel moment la transaction doit être enregistrée. Le principe adopté par la comptabilité nationale est autant que possible l'enregistrement de la transaction au moment de la commande, mais dans certains cas on enregistre la transaction au moment où le payement a été effectué. Il se pose alors un problème de réajustement.

Exemple: Les enregistrements des salaires se fait au moment de la paie. Les problèmes posés sont cependant limités.

- Difficultés pour l'enregistrement des dépenses de capital fixe (immobilisé).

Exemple: Dans la construction d'immobilisés, le travail peut s'étendre sur une période considérable et le problème consiste à mesurer la quantité de travail effectué sur une période et la quantifier financièrement.

- Difficultés sur les moyens d'enregistrer les importations et les exportations.

Cela pose le même problème que pour l'enregistrement de la vente d'un article sauf qu'en plus il faut connaître le moment où on enregistrera cet article comme importation ou exportation.

- Il existe une petite différence dans le timing de l'estimation d'un trimestre et certains comptes de résultats portant sur des petites périodes relatives à treize semaines, ce qui est légèrement différent de la longueur d'un trimestre.

4. Les estimations du produit domestique brut à prix constant sont moins fiables que les estimations à prix courant à cause des incertitudes entourant le processus de déflation par indice des prix.

Exemple: Pour l'évaluation du prix d'une certaine quantité de marchandise, on calcule en fonction du prix de cette marchandise à l'année de base, ce qui n'est pas toujours facile. Un autre problème intervient lorsqu'un produit nouveau apparaît.

5. Il existe des valeurs que l'on ne trouve dans aucune source et que l'on ne peut évaluer, parfois même parce que l'on en ignore l'existence.

Exemple : le travail en "noir".

2.3. Enoncé précis du problème.

Dans la section précédente, nous avons défini ce qu'était le produit domestique brut dans un pays comme le Royaume-Uni, ainsi que mis en évidence les difficultés que rencontraient les statisticiens pour en faire une estimation la plus fiable possible. Ces difficultés ne sont pas uniquement spécifiques à l'estimation du produit domestique brut, mais se retrouvent également dans l'estimation de nombreuses autres variables statistiques nationales. Si nous avons développé le problème principalement pour le produit domestique brut, c'est parce qu'il s'agit d'un bon exemple et nous pensons que la définition faite à la section précédente permettra de bien comprendre le problème qu'il nous a été demandé de résoudre par les personnes responsables des estimations des données de la statistique officielle.

Les données sur lesquelles nous allons travailler sont les séries comptables nationales périodiques. Avant d'énoncer le problème, il est bon d'en faire une définition précise:

> Une "variable comptable nationale périodique" est une variable statistique de la comptabilité nationale, c'est-à-dire en rapport avec l'activité économique du pays, dont on évalue la valeur chaque trimestre.

> Une "série comptable nationale périodique" est une suite d'estimations consécutives des valeurs d'une variable statistique pour une période (un trimestre) donnée.

> *Exemple: Une suite d'estimations de la valeur du produit domestique brut pour le second trimestre de l'année 1990.*

La première valeur de la série est la valeur de l'estimation pour cette variable statistique qui apparaîtra dans "L'Economic Trends" lors de la première publication de cette variable, c'est-à-dire quatre mois après la fin de la période concernée par cette variable. La seconde valeur de la série est la valeur de l'estimation pour cette variable qui apparaîtra dans "L'Economic Trends" lors de la seconde publication de cette variable, c'est-à-dire sept mois après la période concernée par cette variable, la valeur

suivante sera la troisième estimation de cette variable statistique pour le période concernée et ainsi de suite jusqu'à la dernière valeur de la série qui sera la dernière estimation publiée de la variable.

Exemple: En reprenant l'exemple du produit domestique brut pour le second trimestre de l'année 1990, on pourra avoir la série suivante:

PDB90/2 = val1, val2,... ,valn

val1 = la valeur de la première estimation publiée du produit domestique brut pour le second trimestre de l'année 1990.

val2 = la valeur de la seconde estimation publiée du produit domestique brut pour le second trimestre de l'année 1990.

Comme nous l'avons vu dans la section 2.2, la valeur d'une variable comptable nationale périodique pour une période n'est pas connue avec exactitude dès la première publication et peut mettre un certain temps avant de se stabiliser. Les responsables du bureau central de statistique du Royaume-Uni sont intéressés par la possibilité qu'il y ait un biais dans les premières estimations de certaines séries comptables nationales périodiques par rapport aux estimations finales. Par biais, on entend une différence significative entre les premières estimations et les dernières estimations d'une même variable statistique périodique.

Ils seraient intéressés par l'acquisition d'un système capable d'analyser les séries statistiques périodiques, donc les révisions à travers le temps des estimations des variables statistiques pour des périodes données, afin de retrouver l'existence ou non d'erreurs dans les premières publications des estimations par rapport aux estimations publiées plus tard. Le problème principal est de créer une méthode qui permettrait de mettre en évidence certains aspects des révisions des séries individuelles. La méthode permettra à des personnes relativement inexpérimentées de regarder rapidement des séries et identifier celles où des biais peuvent être présents.

Il serait également intéressant de pouvoir repérer les périodes où les révisions sont plus importantes pour l'ensemble des séries et les périodes où les révisions sont moins importantes.

2.3.1. Les Données.

Le programme traitera simultanément un ensemble de séries comptables nationales périodiques pour une même variable. Par "ensemble de séries", on entend l'ensemble des séries pour la variable qui sont situées dans un intervalle de temps précis.

Exemple: Les séries portant sur le produit domestique brut pour les périodes (trimestres) comprises entre le premier trimestre de l'année 1987 et le troisième trimestre de l'année 1992.

On peut se représenter les données comme un tableau à deux dimensions. Les colonnes représentant les périodes auxquelles se rapportent les séries. *exemple: La série pour le second trimestre de l'année 1990.* Les lignes représentant les périodes de publication des séries. *Exemple: La première publication de l'année 1990, c'est-à-dire la publication du mois de janvier.* Chaque élément du tableau représentera la valeur de l'estimation de la variable statistique pour la période correspondant à la colonne où il se trouve et publiée à la période correspondant à la ligne où il se trouve. Toutes les séries pourront être représentées dans un même tableau, mais certaines "cases du tableau" pourront être vides si la période de publication à laquelle l'élément se rapporte est antérieure à la période concernée par la série à laquelle se rapporte l'élément.

2.3.2. Le traitement.

On désire analyser pour chaque série s'il existe des biais dans les estimations et si c'est le cas évaluer la grandeur du biais. On voudrait également pouvoir voir quelles sont les périodes où les révisions sont les plus importantes. Pour cela, on demande d'effectuer certaines analyses sur le tableau de données et de faire apparaître les résultats tout en mettant en évidence les résultats remarquables. Voici un aperçu des

types d'analyses souhaitables, ces analyses ne sont pas les seules possibles et ne doivent pas être exactement les mêmes. Le concepteur décidera lui-même quelles sont les analyses nécessaires pour répondre au problème posé.

2.3.2.1. Une analyse des tendances des séries.

On analyse les séries du tableau afin de faire ressortir si les premières estimations ont été surestimées, sous-estimées ou bonnes par rapport aux estimations suivantes. On tentera de faire ressortir les résultats remarquables susceptibles de donner à l'utilisateur une idée de la tendance générale des séries. Ces résultats pourront être mis en évidence par l'aide de couleurs (voir point 2.5.).

2.3.2.2. La quantification de l'étendue des révisions.

Après avoir recherché les tendances générales des séries, on désire connaître une quantification de la grandeur des révisions. On analysera les séries afin de connaître l'étendue des révisions soit en pourcentage, soit en valeur réelle. On fera, comme pour le premier type d'analyse, ressortir les résultats remarquables, par exemple suivant différents intervalles de grandeur de révisions.

2.3.2.3. La présentation des résultats.

La présentation des résultats sous forme de tableau dans lequel on a fait apparaître les résultats remarquables en couleur est intéressante pour en déduire les conclusions désirées, mais la présentation à l'utilisateur d'un ensemble de chiffres peut parfois être lourde. C'est pourquoi on désire pouvoir visualiser les résultats sous forme graphique. Le choix des types de graphiques seront laissés à l'appréciation du concepteur qui les choisira en fonction de leur facilité de lecture, de leur utilité et en fonction du profil des utilisateurs, qui seront des statisticiens.

2.3.2.4. Les impressions des résultats.

Il est important de pouvoir imprimer les tableaux de données ainsi que les résultats. On voudrait des procédures qui permettent aussi bien l'impression des tableaux que des graphiques créés.

2.4. Méthodologie suivie

2.4.1. Le choix de l'outil

L'outil adopté pour la réalisation et l'exécution d'une solution au problème spécifié doit répondre à quelques impératifs :

1) il doit avoir une puissance de traitement suffisante afin de manipuler des tableaux de données assez conséquents.

2) il doit proposer la possibilité de manipulation de graphiques.

3) il doit permettre une interactivité aisée et performante entre lui-même et le futur utilisateur.

4) il doit être très convivial car il est destiné à des non-informaticiens.

Sur base de ces critères et de quelques autres critères de pure conception (temps de conception disponible,...), nous avons opté pour la technologie "tableur". En effet, les tableurs actuels proposent une gamme de possibilités très étoffées : simulation, graphismes, gestionnaire de dialogue, fonctions statistiques, fonctions arithmétiques, mise en page, ...De plus, grâce à la "guerre" des concepteurs de logiciels, on dispose maintenant d'outils de plus en plus puissants tout en restant très conviviaux.

Evidemment, nous pourrions également choisir un autre type d'outil tel qu'un langage de programmation "traditionnel", mais une telle option nous forcerait à

créer une solution complète (création d'un interface, gestion de la mémoire, gestion des périphériques,...) qui nous couterait un temps considérable et qui, en conséquence, nous détournerait de notre tâche première : une résolution *complète et de qualité* du problème présenté.

2.4.2. "Excel 4.0"

2.4.2.1. Pourquoi "Excel" ?

Le tableur Excel est un "standard" dans le monde des tableurs. Il possède un nombre important de qualités :

- Il fournit la possibilité d'importer des fichiers (ou plutôt feuilles de calcul) provenant d'une gamme étendue de tableurs concurrents.
- Son format de fichier est un standard et peut-être récupéré par d'autre tableurs.
- Il propose une ***gestion des graphiques très performante et complète***.
- Il dispose d'un ***langage pour "macro"***.
- Il s'exécute sous l'***environnement Windows*** et est, de par ce fait, ***convivial et assez facile d'utilisation*** (du moins pour une approche générale).
- Il est très ***puissant*** et propose une grande panoplie de fonctions auxiliaires (simulation, fonctions statistiques).
- Il dispose d'une documentation complète et relativement facile d'utilisation.

De plus, l'environnement Windows permet une communication dynamique (grâce au "Dynamic Data Exchange") entre plusieurs types d'applications différentes ce qui, pour une extension future, est un critère non-négligeable.

2.4.2.2. Le tableur : moyen de programmation

a) les concepts de base

la programmation par l'intermédiaire d'un tableur nécessite un vocabulaire spécifique qui est tout à fait différent de celui utilisé lors de la manipulation d'un pur langage de programmation (C, Pascal, Cobol, ...). On retrouve principalement six grands concepts:

- le ***tableau*** : c'est la zone de travail où l'on encode et stocke différentes données. C'est le concept fondamental du tableur. Le tableau se compose de ***lignes*** et de ***colonnes***. Ces lignes et ces colonnes déterminent des ***cellules*** qui sont en fait l'unité élémentaire du tableau. C'est dans ces cellules que l'on rentre l'information. Ces cellules sont "actives" au moment "t" si l'on travaille sur leur contenu au moment "t" et "non-active" dans le cas contraire. Pour déterminer une cellule bien précise, on utilise sa ***référence*** en stipulant la ligne et la colonne contenant cette cellule.

- La ***feuille*** : celle-ci peut-être de plusieurs types : calcul, macro, graphique. C'est en réalité équivalent à un fichier présenté sous la forme d'un tableau. Les opérations typiques portant sur ce composant du tableur sont des opérations d'ouverture, de fermeture, de modifications, ...

b) La programmation et les macro-commandes

- Les macros-commandes

Une macro-commande est une série de commandes qui accomplit une tâche désirée. Cette macro-commande est stockée dans un simple programme et peut être exécutée autant de fois que désiré. Les macro-commandes sont créées lorsqu'il est

nécessaire de répéter des séries complexes d'étapes. Grâce à cela, l'utilisateur dispose d'un gain de temps considérable et surtout facilite sa tâche.

Les macros-commandes peuvent également être créées afin d'augmenter l'efficacité d'une tâche: on crée un ensemble de macro-commandes permettant l'exécution de fonctions annexes pour cette tâche (aide, analyse, représentation automatique de graphiques,...).

- La programmation

La programmation de macro-commandes est quasiment identique à la programmation traditionnelle, le programme se déroule séquentiellement (avec possibilités de branchements conditionnels et d'appels de macros). Une particularité de ce "langage tableur" est que *le programme est lui-même écrit dans les cellules d'un tableau* (et pas sur un "simple" fichier émanant d'un éditeur) et, par ce fait, peut lui-même être traité comme un simple tableau de données c'est-à-dire que le contenu des cellules peut être modifié en cours d'exécution et que certains résultats intermédiaires peuvent être stockés sur la feuille de macro-commande elle-même (*tout en s'exécutant, le programme peut donc* ***très facilement*** *se modifier*) : le programme est vu comme une entité dynamique du fait de ce principe "d'auto-référence" (un des avantages de ce principe est que le programme peut simultanément jouer le rôle de programme et de fichier de données temporaires : on stocke certains résultats temporaires dans certaines cellules du programme ce qui accélère l'accès à ces données). Excepté cela, on retrouve dans la programmation "tableur" la même méthodologie utilisée dans les langages de programmation traditionnels (particulièrement les langages interprétés) appliquée aux concepts vus ci-dessus.

De plus, la programmation est simplifiée grâce à la puissance du logiciel. Cette puissance permet l'imbrication de nombreux appels de macros ("fct w(fct x(fct y(....)))") sans pour cela diminuer la vitesse d'exécution, bien au contraire.

2.4.3. Proposition d'une solution

A la lecture de l'énoncé du problème, on s'aperçoit qu'il y a une forte répartition des tâches : analyses de base, graphismes, analyses annexes, impressions. Il est donc possible d'envisager une approche fonctionnelle du problème c'est-à-dire la création de différents modules, chacun reprenant un type précis de fonctions (un module pour les analyses, un autre pour les graphiques et un dernier pour les impressions).

Ce qui est primordial dans cette conception, c'est principalement la "vue utilisateur". Nous voulons dire par cela que le système créé doit être totalement transparent (et ceci malgré toutes les possibilités offertes par Excel) afin de faciliter et d'alléger au maximum la tâche de l'utilisateur. Il faut "oublier" notre "connaissance" informatique afin de se mettre à la place d'un statisticien "novice" en informatique.

2.4.3.1. Choix des analyses

Face aux séries périodiques d'estimations, nous pouvons avoir deux vues différentes des analyses requises : une vue purement qualitative (superficielle) ou à l'opposé une vue quantitative (descriptive). En y regardant de plus près, on constate que ces deux vues s'enchaînent : la "qualitative" appelle la "quantitative" c'est-à-dire qu'après avoir vu le problème en gros, on désire souvent savoir le pourquoi, le détail. Mais comment envisager une vue qualitative des tableaux de données ?

Un des premiers désirs des statisticiens est de savoir si oui ou non, il existe des biais d'estimations dans les tableaux de données. Pour répondre à ce besoin, on peut envisager une *analyse qualitative* qui permet de visualiser la tendance globale de chaque série du tableau: "Y-a-t-il une réévaluation constantes des estimations? Un maintien ? Une dévaluation ?". Une fois que l'on a une réponse à ces questions, on peut alors passer à une analyse plus détaillée des données et c'est pour cela que l'on implémente des analyses quantitatives. Celles-ci doivent permettre de cerner d'un peu plus près le problème du biais d'estimation. Elles fournissent des résultats qui donnent des indications quant à l'importance du biais, quant à l'existence d'un biais constant, etc...

Pour cela, quatre analyses sont développées : une *analyse quantitative* qui permet de visionner les estimations positives, négatives et neutres de l'analyse qualitative mais exprimées, cette fois, sous forme de pourcentage, une *analyse proportionnelle* qui permet de voir l'évolution en pourcentage d'une estimation, une *analyse "différentielle"* qui permet de chiffrer en unités monétaires les écarts d'estimation et une *analyse de la variabilité* qui elle fournit une autre vue de l'évolution en pourcentage des estimations dans une série. Toutes ces analyses répondent de manières différentes aux questions posées ci-dessus, elles sont développées dans la section 2.5.

A côté de ces vues analytiques, il est également nécessaire d'avoir une approche "utilitaire" c'est-à-dire qu'il faut prévoir des fonctions permettant à l'utilisateur d'imprimer ou de visualiser les différents résultats produits.

2.4.3.2. Le choix des types de graphiques

La présentation des données, ou des résultats des analyses sur ces données, sous forme tabulaire est parfois lourde et difficile à interpréter. L'utilisateur est confronté à un tableau de nombres dans lequel il doit retrouver et analyser les éléments qui l'intéressent. Même si les données remarquables sont mises en évidence par l'intermédiaire de couleurs, ce n'est parfois pas suffisant. Afin d'avoir une vision plus globale et plus schématisée de certaines séries, il est nécessaire d'offrir à l'utilisateur la possibilité de visualiser les séries qui l'intéressent sous forme graphique. Ces graphiques permettront d'afficher les résultats des analyses d'une ou plusieurs séries statistiques.

Le logiciel Excel4 permet facilement la création de graphiques. Pour cela, il propose un grand nombre de types de graphiques différents. De plus, si les types de graphiques proposés, qui sont les types les plus répandus de graphique, ne suffisent pas, il est possible de construire ses propres graphiques grâce aux fonctions de dessin fournies par Excel.. Dans ce cas, il faut évidemment les construire de toute pièces, c'est-à-dire tracer les axes, calculer et tracer les graduations, tracer les graphiques pour la série...

Parmi les différents types de graphiques que propose Excel, nous devons effectuer un choix sur ceux qui sont les mieux adaptés aux données à visualiser. Le choix doit être fait selon certains critères:

1. Le type des données à afficher. Dans notre cas, il s'agit soit de nombres importants, soit de pourcentages. Le seul cas particulier survient lors de l'affichage des résultats de l'analyse qualitative. Les valeurs prises sont soit 1, soit 0, soit -1 et il est donc important de choisir un graphique bien adapté à l'affichage de valeurs discrètes et qui permet de bien faire ressortir les tendances des séries.

2. La lisibilité: Il faut que la lecture de ces graphiques soit claire et facile pour un utilisateur n'étant pas particulièrement familié avec une représentation graphique.

3. Le nombre de données par série: certains graphiques ont besoin d'un grand nombre de données pour être représentatifs alors que d'autres ne sont lisibles qu'avec un nombre limité de données.

4. Le nombre de séries à afficher simultanément: Plus le nombre de séries à afficher simultanément est important, plus cela pose un problème de lisibilité.

5. Il ne faut pas proposer un trop grand nombre de types de graphiques différents car cela alourdirait le problème de choix de l'utilisateur. De plus, on risque de choisir deux types de graphiques qui ne soient pas significativement différents l'un par rapport à l'autre et qui n'apporteraient pas une nouvelle vue de la situation.

6. Le profil cognitif de l'utilisateur: Dans notre cas, il s'agirait plutôt de statisticiens.

En fonction de ces critères, nous pouvons proposer à l'utilisateur 5 types de graphiques différents divisés en deux grandes classes:

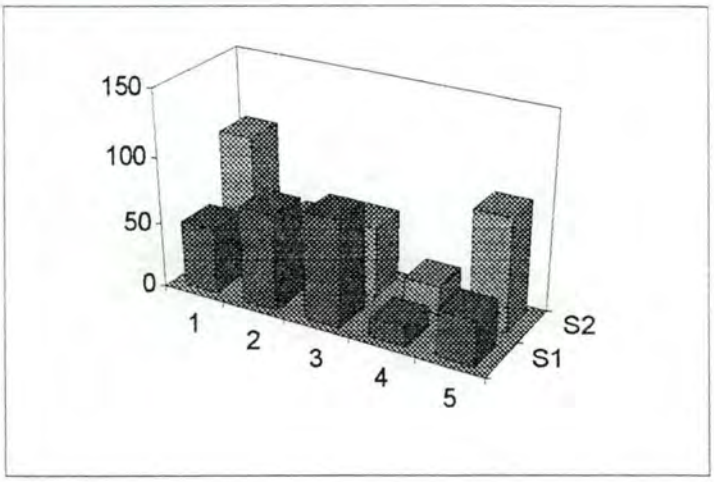
2.4.3.2.1 Les graphiques standards.

Ces graphiques sont des graphiques proposés dans la librairie de graphiques d'Excel. Pour les adapter aux besoins rencontrés dans notre problème, il faut donner des valeurs spécifiques à un certain nombre de leur paramètres:

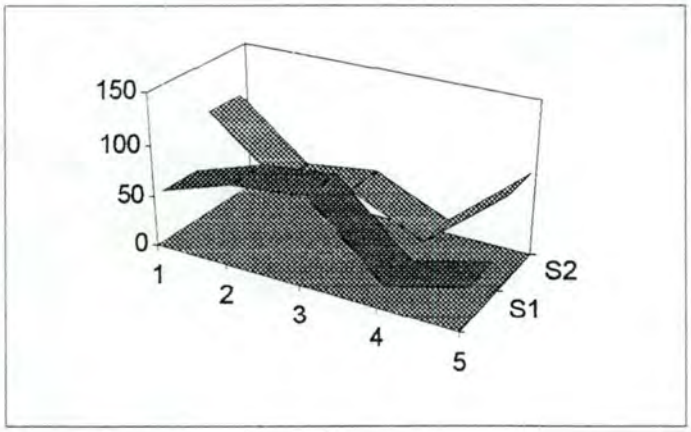
- Un graphique de type histogramme en trois dimensions: Ce type de graphique est courant, facile à interpréter et bien adapté pour faire apparaître des évolutions de valeurs, ou encore pour faire une bonne distinction entre des valeurs discrètes. Il est donc très efficace pour analyser les tendances des séries à partir des résultats de l'analyse qualitative.

  Le choix d'un graphique en trois dimensions permet de voir les graphiques sous différents angles. Pour cela, Excel fournit une panoplie d'outils permettant de redimensionner le graphique, de lui faire faire des rotations... La seule contrainte, est que pour pouvoir retravailler le graphique, l'utilisateur doit avoir une certaine expérience de la manipulation du logiciel.

  Lorsque l'on désire afficher plusieurs séries, ou le résultat d'une analyse de plusieurs séries, celles-ci se superposent (voir la figure suivante) et il est aisé, pour deux séries, de comparer leurs valeurs correspondantes entre elles. Pour plus de deux séries, les dernières séries sont cachées par les premières, c'est pourquoi nous avons délibérément limité le nombre de séries affichées simultanément. Pour l'affichage d'un grand nombre de séries simultanément, le troisième type de graphique est plus intéressant.



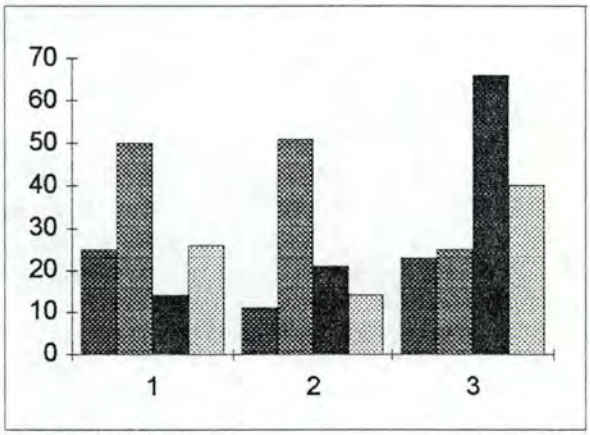
- Un graphique de type linéaire en trois dimensions: Il s'agit également d'un type de graphique "classique" très facile à interpréter, mais cependant plus efficace pour faire ressortir l'évolution de valeurs "continues" que d'autres graphiques plus complexes. Ce type de graphique est également limité à l'affichage maximum de deux séries simultanément.



- Un graphique de type histogramme en deux dimensions: Ce type de graphique est assez proche du premier type, mais la grande différence intervient lors de l'affichage de plusieurs séries simultanément. Les différentes séries ne sont plus superposées l'une devant l'autre, mais sont affichées l'une à la suite de l'autre, en laissant un espacement entre chaque série. L'avantage est qu'il permet l'affichage simultané d'un plus grand nombre de séries sans connaître

des problèmes de lecture, mais le désavantage est qu'il est plus délicat de comparer ensemble les valeurs correspondantes de plusieurs séries.

Nous avons choisi un type de graphique à deux dimensions pour un critère purement esthétique, il s'agit donc d'un choix subjectif. De plus, une vue en trois dimensions n'apporte rien de plus à ce type de graphique.



2.4.3.2.2 Les graphiques "artificiels".

Ces types de graphiques ne sont pas fournis dans la librairie de graphiques proposé par Excel, il a donc été nécessaire de les construire de toute pièces (tracer les axes, calculer les coordonnées, tracer les graduations, tracer les séries...) De ce fait, ces graphiques sont souvent moins performants que les graphiques "standards" et s'adaptent moins facilement à d'autres fins que celles pour lesquelles ils ont été créés. Les choix de ces graphiques ont été faits pour répondre à des besoins plus spécifiques des statisticiens.

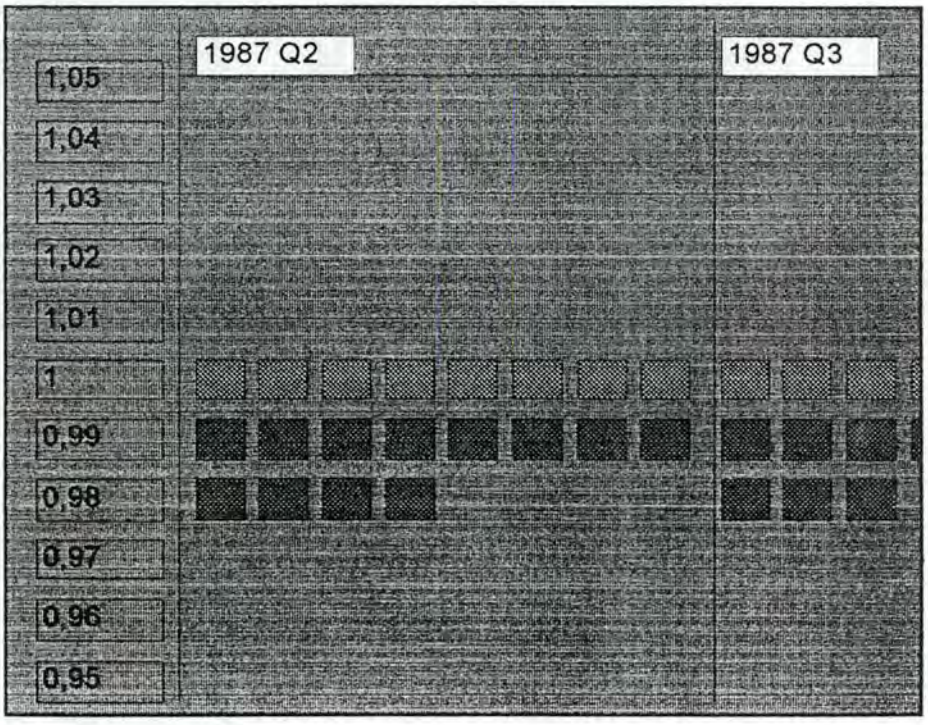
- Un type de graphique permettant de visionner la variabilité des valeurs d'une série:

Ce type de graphique est créé à partir de chaque série d'un tableau reprenant les résultats d'une analyse de variabilité (voir figure suivante).

Pour une série, l'analyse de variabilité alloue pour chaque valeur l'index de la valeur par rapport à la valeur de base de la série (la valeur de base est la première ou la dernière valeur d'une série, selon la façon dont l'analyse a été exécutée). Par exemple si la valeur de base vaut 150 et une seconde valeur vaut 155, l'analyse de variabilité donnera 1 pour la valeur de base et 1.033.. pour la seconde valeur, qui sera arrondi à 1.03. Pour plus de détails sur l'analyse de variabilité, on peut se référer à la section 2.5.

- L'axe horizontal du graphique est placé au niveau de la valeur 1 de l'axe vertical, donc au niveau du résultat de l'analyse pour la valeur de base. Pour chaque valeur d'une série, un carré est tracé sur le graphique, ce carré sera placé, par rapport à l'axe vertical, à une hauteur qui correspondra à la distance entre cette valeur et la valeur 1. Ce type de graphique est intéressant car il permet de donner, pour chaque série, une bonne synthèse de la variabilité des valeurs de la série par rapport à sa valeur de base.

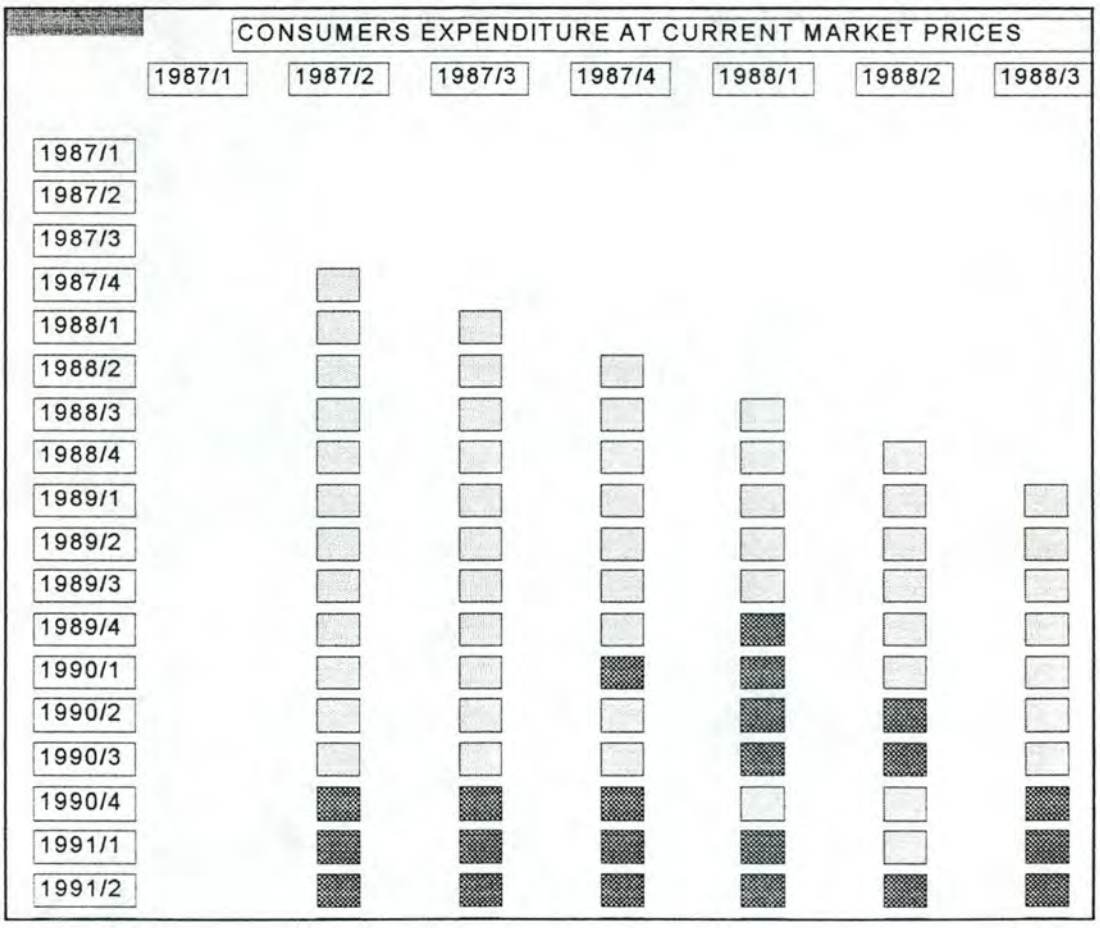
Remarque : Les résultats obtenus par ce type de graphiques n'ayant pas donné satisfaction, il sera abandonné au profit du type de graphique exposé dans le point ci-dessous.



- Un type de graphique permettant l'affichage de "ratios".

Ce graphique est également créé sur base des résultats de l'analyse de variabilité, mais peut être utilisé avec d'autres données. Il s'agit d'une vision "graphique" d'un tableau de données, où chaque donnée est positionnée à la place qu'elle occupe dans le tableau, mais est représentée par un carré coloré avec une couleur qui dépend de la valeur qu'il représente (voir figure suivante).

Ce graphique est intéressant car il est facilement interprétable par des statisticiens habitués par la lecture de tableaux de valeurs. L'avantage par rapport à un tableau classique est qu'il permet une lecture rapide en fonction des concentrations de couleurs dans telle ou telle région du tableau. Il fournit rapidement une bonne synthèse de la variablilité des valeurs par rapport à la valeur de base.



2.5. Implémentation de la solution

Dans ce point, nous allons donner un aperçu de l'implémentation que nous avons envisagé pour résoudre le problème explicité auparavant dans ce chapitre. Nous n'exposerons ici qu'une *vue générale des fonctions principales* (fonctions de calcul et d'analyse) implémentées. En effet, pour éviter toute surcharge en détails, nous n'avons pas développé entièrement les fonctions "clé". *Les fonctions "utilitaires" ainsi que les fonctions graphiques sont, quant à elles, totalement développées en annexe.* Attention, les fonctions graphiques, bien que situées en annexe, ont nécessité la plus grande partie de notre temps lors de l'implémentation. En effet, leurs caractéristiques techniques ont relativement compliqué la solution et c'est d'ailleurs pour cette raison que nous les développons en annexe (ceci évitant une surchage de détails purement techniques). Nous invitons donc le lecteur à consulter les annexes s'il veut un aperçu complet de l'implémentation du problème.

2.5.1. Les fonctions de calcul et d'analyse

2.5.1.1. "Perform-qualitative-analysis"

a) Buts de la fonction

Les statisticiens responsables des "statistiques officielles" ont remarqué qu'un certain nombre de séries (du moins certaines sous-séries) sont régulièrement sur ou sous-estimées. Le but de cette fonctionnalité est donc d'analyser les séries du tableau et de signaler leurs tendances : "les estimations ont-elles tendance à augmenter, à diminuer ou, plus rarement, à se maintenir ?". Cette analyse s'effectue grâce à l'application de certains concepts tirés de la "Qualitative Physics" (cette théorie sera brêvement explicitée dans le point suivant). Dans un souci d'ergonomie, ces tendances sont coloriées à l'écran afin de bien insister sur les variations du tableau (bleu pour les sous-séries avec des tendances à la hausse, rouge pour les sous-séries à tendance à la baisse et vert pour les sous-séries constantes) .

Outre ce rôle de visualisation des tendances, cette fonction permet aussi de tirer quelques conclusions : grâce à l'utilisation de coloris, on peut très vite remarquer certaines périodes de "biaisage" intensif (ex.: on remarque que "x" séries du tableau sont surestimées durant les trimestres t_1, t_2, t_3, t_4 ; pourquoi?, etc...) et donc, en retirer que celles-ci ont été mal estimées et ceci à cause, éventuellement, d'événements conjoncturels relatifs à ces trimestres[1].

Un problème important émanant de cette fonction est le choix du nombre minimum nécessaire de valeurs pour *pouvoir* parler de "tendance". Si l'on parle de séries, le minimum possible est donc de deux valeurs (= estimations de deux trimestres), cependant une telle série peut-être due à une bonne part de hasard. Nous pensons donc que la fiabilité minimum d'une tendance s'apparente à une "longueur minimum" de trois valeurs d'orientation équivalente (baisse, hausse, maintien). Evidemment ce choix est subjectif, mais étant donné les sommes que représentent chaque valeur du tableau, il est important de ne rien négliger. De plus, il en va de la responsabilité du statisticien de relativiser chaque résultat significatif apparaissant dans le tableau (nous vous renvoyons d'ailleurs à la note de bas de page précédente).

Remarque : cette fonction est considérée comme la fonction de base, c'est d'elle que découleront toutes les fonctions d'analyse et de calcul suivantes, elle constitue le point de départ à toute analyse

b) Brève introduction à la "Qualitative Physics"[2]

La physique qualitative est une théorie qui, comme son nom l'indique, est principalement appliquée aux phénomènes physiques. C'est une branche de l'intelligence

[1] On peut ici se rattacher à l'étude du contexte développée dans la thèse de doctorat développée par E. Epprecht (voir bibliographie) et qui insiste sur l'importance de ce composant dans un système de connaissance dédicacé aux applications de statistiques officielles afin de bien comprendre les critères et la méthodologie utilisés et par ce fait, les résultats émanants d'une analyse.

[2] Cette théorie est développée plus en détails dans [IWASAKI]

artificielle qui vise à produire des systèmes experts de deuxième génération c'est-à-dire des systèmes plus précis, plus faciles à développer et plus universels.

Le principe de la physique qualitative consiste à *ne pas s'attacher à des mesures quantitatives exactes mais plutôt à des ordres de grandeur*. Le raisonnement se fait à partir d'indications rudimentaires, de tendances. Comme on ne travaille plus sur des données numériques, on peut résoudre des problèmes dont on ignore certaines valeurs.

Les systèmes se basant sur cette théorie se divisent en deux grandes étapes :

1. L'identification des objets, variables et paramètres relatifs au domaine étudié
2. La description des comportements du système en termes de caractéristiques qualitatives ("augmente", "baisse", "chauffe", ...).

Ensuite, grâce à ces étapes, on peut essayer de tirer quelques conclusions grâce à des raisonnements généraux. Par "raisonnement général", nous entendons un raisonnement non mathématique, basé sur le "bons sens", sans approche théorique du problème (ex: on chauffe de l'eau, donc elle va bouillir et quand la température aura atteint 100°, elle s'évaporera. Pourquoi, on ne le sait pas précisément mais c'est notre connaissance générale, notre bons sens qui nous permet de raisonner).

c) Description de la méthode utilisée

Comme nous l'avons expliqué auparavant, le but de la fonction n'est pas de fournir des renseignements quantitatifs quant aux estimations des valeurs des séries temporelles mais bien de mettre en relief certaines tendances en résultant.

Soit une série S de longueur 1.....n

Soit la valeur val_i (S) la $i^{ème}$ valeur de la série S

Soit $trend_i$ (S) la tendance de la période i pour S

$\forall$ i : i $\in$ (2..n)

Si $val_{i-1}(S) < val_i(S)$ alors $trend_i(S) = 1$ (= hausse)

Si $val_{i-1}(S) > val_i(S)$ alors $trend_i(S) = -1$ (= baisse)

Si $val_{i-1}(S) = val_i(S)$ alors $trend_i(S) = 0$ (= maintien)

Si i = 1 alors $trend_i(S) = 0$ car "i-1" $\nexists$

!!! Cette analyse peut-être effectuée dans les deux sens c'est-à-dire de la première estimation publiée jusqu'à la plus récente ou inversement. Dans ce dernier cas, l'analyse se déroule entre les couples de valeurs "$val_{i+1}(S)$ et $val_i(S)$" $\forall$ i : i $\in$ (n-1...1) et avec $trend_i(S) = 0$ si i = n.

2.5.1.2. "Perform-proportional-analysis"

a) Buts de la fonction

Cette fonction correspond à une des traductions quantitatives de l'analyse qualitative. En effet, elle permet de chiffrer l'évolution de l'estimation d'une valeur dans une série périodique. Elle détermine l'évolution en pourcentage de la valeur estimée à la période "i" par rapport à une valeur de base (c'est-à-dire une valeur avec i=1 ou n pour une série S de longueur n).

Cette fonction, comme celles qui suivent d'ailleurs, doit être considérée comme un outil de compréhension et de mesure du tableau des résultats qualitatifs. Elle donne la possibilité de voir s'il existe un *biais constant* (hausse constante du pourcentage) dans l'estimation d'une valeur étudiée (PDB, GNP, ...). Elle permet également de tester la fiabilité des sous-séries: l'étude des proportions peut montrer que la tendance d'une sous-série de trois valeurs est en fait non significative si les écarts sont très minimes.

Une autre finalité de cette fonction est de tester la validité de chaque estimation. En effet, selon le pourcentage d'évolution calculé, on peut "juger" la cohérence de la valeur éditée: par exemple, si une valeur a évolué de 10 pourcents par rapport à l'estimation de base, des doutes seront émis quant à la fiabilité de cette valeur et donc de la méthodologie sous-jacente. Evidemment un tel écart de différence n'apparaît que très rarement au niveau du PDB cependant cette fonction peut très bien s'appliquer à d'autres styles de tableaux moins stratégiques faisant preuves de moins d'attentions de la part des concepteurs. De tels écarts sont encadrés de couleurs : rouge pour les écarts compris entre 3 et 10% et bleu pour des écarts supérieurs à 10%.

b) Description de la méthode utilisée

Soit une série S de longueur 1.....n

Soit la valeur $val_i(S)$ la $i^{ème}$ valeur de la série S

Soit la valeur $val_b(S)$ la valeur de base d'une série S ($1^{ère}$ ou dernière valeur de la série).

Soit $prop_i(S)$ l'évolution (en %) de la valeur de la période i par rapport à la valeur de base pour une série S.

Si "b" = 1 alors $\forall i : i \in (2..n)$

$[(val_i(S) - val_b(S)) / val_b(S)] * 100 = prop_i(S)$

et $prop_i(S) = 0$ si $i = 1$

Si "b" = n alors $\forall i : i \in (n-1...1)$

$[(val_i(S) - val_b(S)) / val_b(S)] * 100 = prop_i(S)$

et $prop_i(S) = 0$ si $i = n$

2.5.1.3. "Perform-quantitative-analysis"

a) Buts de la fonction

Cette fonction est fort semblable à la précédente. En effet, le but de cette fonction est également de visionner l'évolution d'une estimations dans une série. Cependant, dans la fonction précédente, on comparait l'évolution d'une estimation par rapport à sa valeur de base (la première ou dernière estimation) alors qu'ici, on compare l'évolution des estimations à l'intérieur même de la série : on compare une estimation au temps "t" avec l'estimation précédente (au temps "t-1" ou "t+1").

L'avantage d'une telle fonction est qu'elle représente *parfaitement* (quantitativement) l'analyse qualitative. Elle reprend donc tous les avantages cités au point 2.5.1.1. Elle permet d'ajuster les hypothèses prises lors de l'analyse qualitative : des valeurs (-1, +1) lors de l'analyse qualitative peuvent parfois être insignifiante (ex; : la différence entre 110000 et 109997 est sans intêret mais est quand même représentée par un valeur qualitative : -1 ou +1 selon le sens de l'analyse) ; le recours à cette fonction permet, dès lors, d'éviter toutes déviations dans les interprétations possibles de l'analyse qualitative.

b) Description de la méthode utilisée

Soit une série S de longueur 1.....n

Soit la valeur $val_i(S)$ la $i^{ème}$ valeur de la série S

Soit b la valeur de base indiquant le sens de l'analyse

Soit $quanti_i(S)$ l'évolution (en %) de la valeur de la période i par rapport à sa valeur précédente pour une série S.

Si "b" = 1 alors $\forall i : i \in (2..n)$

$[(val_i(S) - val_{i-1}(S)) / val_{i-1}(S)] * 100 = prop_i(S)$

et $prop_i(S) = 0$ si $i = 1$

Si "b" = n alors $\forall$ i : i $\in$ (n-1...1)

$[(val_i (S) - val_{i+1} (S)) / val_{i+1} (S)] * 100 = prop_i (S)$

et $prop_i (S) = 0$ si $i = n$

2.5.1.4. "Perform-difference-analysis"

a) Buts de la fonction

Cette fonction est implémentée en tant que pure fonction de documentation quantitative. Elle permet d'évaluer l'évolution *réelle* c'est-à-dire exprimée en unité de base (francs, dollars, ...) de l'estimation d'une valeur (comparaison de l'estimation à une période "i" par rapport à l'estimation de base).

Cette fonction sert en fait à nous "rattacher" à la réalité en mettant bien en évidence les enjeux économiques (exprimés parfois en *millions* de francs, livres, dollars, ...) de telles analyses. Elle est équivalente à la fonction précédente du point de vue interprétation mais d'un point de vue "impact", elles est plus marquante car elle n'est pas neutre: "on ne parle pas de pourcentage de quelque chose mais bien de sommes d'argent!". Toutes les différences positives sont coloriées en bleu et toutes les différences négatives sont coloriées en rouge (les différences nulles ne sont pas coloriées).

b) Description de la méthode utilisée

Soit une série S de longueur 1.....n

Soit la valeur $val_i (S)$ la $i^{ème}$ valeur de la série S

Soit la valeur $val_b (S)$ la valeur de base d'une série S ($1^{ère}$ ou dernière valeur de la série).

Soit $diff_i$ (S) la différence réelle entre la valeur de la période i et la valeur de base pour une série S.

Si "b" = 1 alors $\forall$ i : i $\in$ (2..n)

val_i (S) - val_b (S) = $diff_i$ (S)

et $diff_i$ (S) = 0 si i = 1

Si "b" = n alors $\forall$ i : i $\in$ (n-1...1)

val_i (S) - val_b (S) = $diff_i$ (S)

et $diff_i$ (S) = 0 si i = n

2.5.1.5. "Perform-variability-analysis"

a) Buts de la fonction

Cette fonction présente en fait les mêmes caractéristiques que la fonction relative au calcul de proportion c'est-à-dire qu'elle représente l'évolution d'une valeur à une période "i" par rapport à une valeur de base pour une série S. Cependant c'est suite à la nécessité de créer un graphique spécifique (voir le point relatif aux graphiques) représentant des fourchettes d'évolution qu'elle est créée.

La seule différence réside dans le fait qu'ici, on n'exprime plus le résultat sous forme de pourcentage d'évolution par rapport à la base mais plutôt en "coéfficient" de variation par rapport à la base. Généralement, ces coéfficients ont une valeur approximative de 1 ce qui signifie que la valeur "val_i (S)" est 1,... (ou 0,9..) fois plus grande que la valeur de base. On pourrait également interpréter cela comme l'évolution d'un index, celui-ci étant de 1 pour la valeur de base.

b) Description de la méthode utilisée

Soit une série S de longueur 1.....n

Soit la valeur $val_i(S)$ la $i^{ème}$ valeur de la série S

Soit la valeur $val_b(S)$ la valeur de base d'une série S ($1^{ère}$ ou dernière valeur de la série).

Soit $vari_i(S)$ la variation du coéfficient (ou index) à la période i par rapport à la période de base pour une série S.

Si "b" = 1 alors $\forall i : i \in (2..n)$

$1 + [(val_i(S) - val_b(S)) / val_b(S)] = vari_i(S)$

et $vari_i(S) = 0$ si $i = 1$

Si "b" = n alors $\forall i : i \in (n-1...1)$

$1 + [(val_i(S) - val_b(S)) / val_b(S)] = vari_i(S)$

et $vari_i(S) = 0$ si $i = n$

CHAPITRE 3

"Présentation du systéme E.I.S.I"

3.1. Introduction.

Ce chapitre est consacré à une présentation générale du système EISI (Expert Interface for Statistical Information). La présence de cette présentation à ce stade du mémoire peut être discutable, mais la suite du mémoire sera le développement d'un système de résolution automatique de problèmes dans le domaine des statistique officielles, et c'est grâce à l'étude et à la compréhension de tous les concepts utilisés dans EISI que nous avons pu élaborer une structure analogue pour un style de problème bien particulier. C'est pourquoi il nous a semblé important de présenter EISI, la base de notre étude, avant la présentation de notre propre système. De plus, cette présentation de EISI nous permettra d'en faire une comparaison au chapitre 6 et d'essayer d'étudier comment on pourra intégrer notre système avec le système EISI.

3.2. "E.I.S.I."[1]

3.2.1. Philosophie d'EISI.

Le projet EISI a été lancé dans le but d'étudier et de résoudre les aspects de l'acquisition, de la représentation et de la manipulation de "méta-données" statistiques ainsi que des informations conceptuelles dans des domaines d'applications spécifiques. Ce système donne à l'utilisateur l'accès à cet ensemble d'informations et lui fournit également une assistance pour la stratégie à adopter face à ces informations (formulation d'un problème, recherche des données adéquates, sélection d'une technique d'analyse, etc...).

Ce système repose sur deux concepts principaux : le contexte et la méta-information. L'un étant lié à l'autre car la méta-information n'est, en fait, que la représentation "formelle" du contexte. On peut donc déjà mettre en évidence la caractéristique majeure de ce système : il raisonne sur des informations contextuelles et non sur des données concrètes (chiffres et nombres).

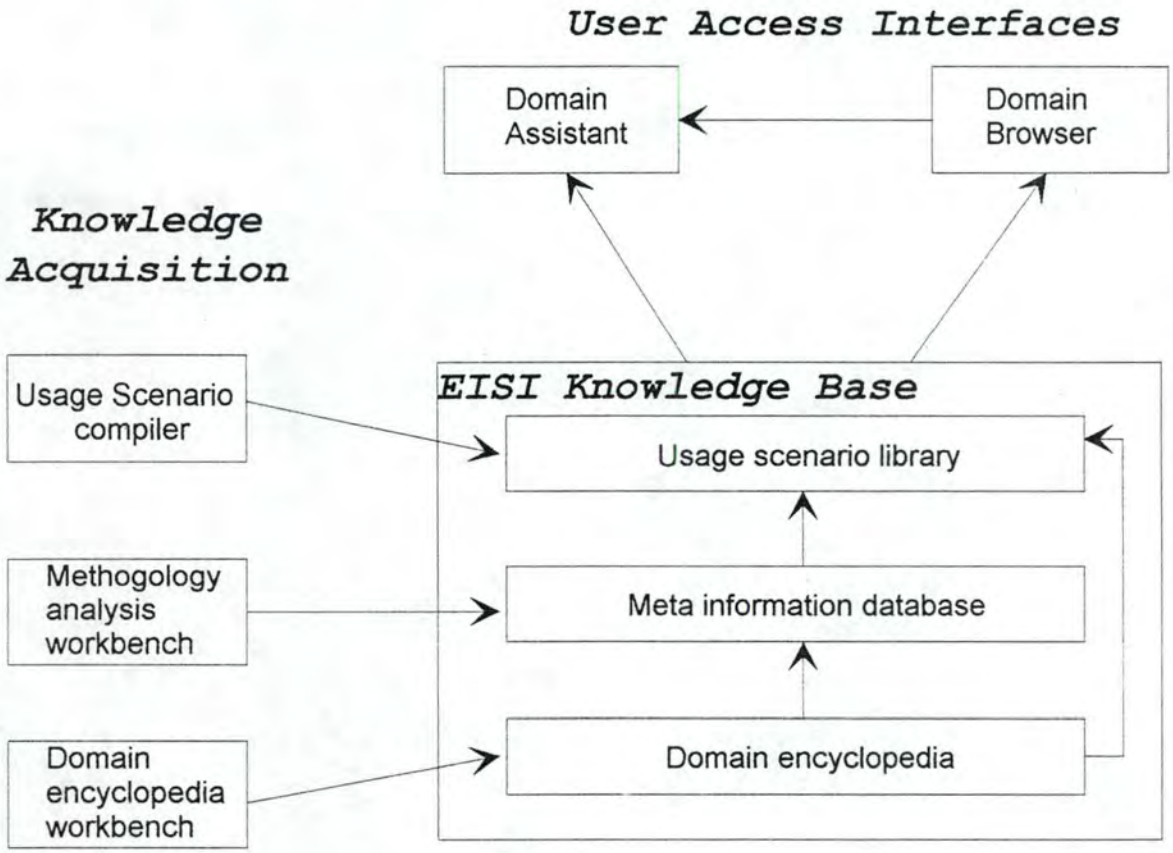
De plus, ce système est basé sur la théorie du "Case Based reasoning"[2] c'est-àdire que le système fonctionne en utilisant des schémas squelettiques (voir point 4.1.3.2.) et des cas déjà résolus. Il n'a donc pas une marge de liberté énorme : il ne bénéficie pas d'une certaine prise de décisions qui pourrait lui permettre de résoudre certains problèmes pour lesquels il ne dispose pas d'indications suffisantes (dans le sens "cas proches résolus" ou "squelettes" équivalents).

---

[1] Les points 5.2.1. et 5.2.2. sont basés sur [VANEY90] et [VANEY92]. Le point 5.2.3. est, quant à lui, basé sur [EPPRECHT] (pp 62-103)

[2] Pour plus de détails sur cette théorie, nous conseillons au lecteur de se référer à [RIESBECK].

3.2.2. Architecture du système.



3.2.2.1. Les "User Access Interfaces".

a. Domain Browser.

Le Domain Browser est un interface utilisateur et un "hypertexte". Il est implémenté pour supporter une navigation interactive et l'affichage du contenu de la base de connaissance et permet à l'utilisateur de naviguer à travers cette information disponible soit grâce à un système d'affichage graphique du contenu de la base (sous forme d'arbre), soit par la sélection de mots clés dans le corps du texte.

b. Domain Assistant.[3]

Ce composant peut être qualifié d'interface "intelligent" alors que le "Domain Browser" serait plutôt qualifié de "passif". Son rôle se décompose en trois étapes :

- Assister l'utilisateur pour la formulation de son problème.
- Donner à l'utilisateur l'identification des données, sources et méta-informations relatives à son problème.
- Donner à l'utilisateur un plan de solution pour l'interprétation et l'analyse de ces données (et des résultats).

a) Définition précise du problème

Il est nécessaire que la communication entre le système et l'utilisateur soit correcte c'est-à-dire qu'ils aient tous deux la même vue et compréhension de la question (complétude, précision , raffinement des questions vagues, vocabulaire compris par les deux correspondants).

Un problème dans le "Domain Assistant" peut-être défini en un nombre connu d'éléments :

- le genre de question (comparaison? , relation ? , calcul? ...)
- les compléments de la question

Le type et le nombre de compléments varient selon le type de question (une comparaison nécessite un minimum de deux éléments ...)

Il existe un nombre limité de questions typiques (et de combinaisons de compléments en résultant) qui permet donc d'envisager une "définition" formelle

[3] [Epprecht]

assez complète.

Tout ceci nécessite une interactivité poussée entre le "Domain Assistant" et les utilisateurs afin de faciliter la formulation correcte d'une question : affichage de boîtes de dialogue, d'aide, ...

b) Identification des données, concepts et sources nécessaires.

Il s'agit, ici, d'une recherche effectuée dans la "méta-information database".

c) Sélection de "méthode"

Chaque méthode, ou plutôt "stratégie", est associée à la définition et au contexte du problème dans lequel elle est impliquée. Elle est composée de :

- informations factuelles et conceptuelles relatives aux sujets, régions d'intérêts, lieux et temps relatifs au problème.
- méta-information sur les caractéristiques formelles et sémantiques des données disponibles.
- le but final (résultat).

La sélection d'une stratégie est simplement effectuée en recherchant une stratégie pour laquelle la définition et le contexte du problème sont équivalents à ceux du problème de l'utilisateur (trois possibilités : une seule stratégie, deux ou plusieurs stratégies, aucune stratégie).

Pour le "Domain Assistant", deux problèmes sont égaux si la stratégie pour les résoudre est la même. Pour différencier ces stratégies, on cherchera une différence dans les stratégies de niveau inférieur appelées par la stratégie de base.

d) Analyse et interprétation

Les stratégies sont représentées dans la "Usage Scénario Library" avec un certain degré d'abstraction. Pour cela, il est nécessaire d'appliquer "l'expansion" de ces stratégies c'est-à-dire le remplacement des références par les variables ou objets correspondants. Une fois cette opération effectuée, on obtient le plan de la solution. L'application de ce plan avec des données propres au problème demandé doit être faite par l'utilisateur.

3.2.2.2. EISI Knowledge Base.

a. Usage Scenario Library

Avant de développer ce point, il est nécessaire de bien définir le vocabulaire utilisé dans ce système.

- Stratégie : méthode prescrite pour résoudre un problème ou un sous-problème.
- Plan : la forme dans laquelle une stratégie est représentée dans le "Domain Assistant". Comme les stratégies, les plans peuvent se rapporter à différents niveaux d'abstraction.
- Scénario : une situation d'utilisation du système ; un scénario est un objet conceptuel qui regroupe une définition d'un problème, l'identification des informations et méta-informations requises pour le résoudre et une stratégie pour l'utilisation des informations. Il s'agit également
- Schéma : la structure logique dans laquelle les plans sont représentés.

Cette librairie garde une collection de stratégies pour la résolution de problèmes (y compris l'identification des données et sources s'y rapportant ainsi que leur interprétation) sous forme de plans abstraits. Chaque scénario comprend le plan,

le contexte associé au problème et les caractéristiques des données. Ce composant constitue les structures de connaissances fondamentales utilisées dans le Domain Assistant.

b. Méta-information Database

Cette base de données contient toute la méta-information concernant la collection de données disponibles relatives à un domaine (ici, les statistiques) ainsi que la méta-information concernant les sources et dérivés.

c. Domain Encyclopedia

La "Domain Encyclopedia" contient la connaissance (théorique) conceptuelle à propos d'un domaine d'application statistique particulier. Ceci inclut toute la connaissance qui n'est pas directement relative à la collecte de données: définitions de termes et concepts, classification et taxonomie, processus et algorithmes, etc...

Cette "encyclopédie" consiste en un ensemble de définitions de concepts organisés selon une structure hiérarchique faible et où chaque concept peut avoir un nombre de structures détaillées qui lui sont associées. Ceci dans le but de décrire des caractéristiques techniques nécessitant l'appel à d'autres concepts (="définitions interdépendantes").

3.2.2.3. Knowledge Acquisition.

a. Usage Scenario Compiler.

Il s'agit en quelques sortes, de l'interface spécifique à la "Usage Scenario Library" (USL). Le contenu de cette USL est généré à partir de scénarios exprimés dans un langage de spécification. Ensuite, ces scénarios sont compilés pour aboutir à un format intermédiaire orienté objet. Cette représentation intermédiaire est alors utilisée pour générer les indices requis. Le tout est finalement stocké dans la structure de la USL.

b. Methodology Analysis Workbench.

Pour le projet EISI, les sources de méta-informations les plus utiles se présentent sous la forme d'enquêtes statistiques et de publications de résultats. Dans le but d'extraire cette méta-information de ces documents sources, il était nécessaire de développer une architecture logique pour la décomposition structurée de ces sources. De la sorte, il était plus aisé d'identifier l'information nécessaire.

Basée sur cette architecture, une interface d'acquisition de connaissance associé à la base de méta-information EISI a été développé. Il s'agit plus précisément, d'un outil interactif de maintenance et de reporting. Cet outil permet d'entrer dans la base de données (méta-information) toute la méta-information relative à un sondage, par exemple, et ceci en se référant à l'architecture logique de décomposition proposée. Attention, ce processus de décomposition est, cependant, une activité manuelle devant être exécutée par des experts du domaine d'application.

c. Domain Encyclopedia Workbench.

Les sources d'informations dans la "Domain Encyclopedia" peuvent être diverses. Le contenu typique, cependant, de cette encyclopédie provient généralement de dictionnaires techniques, de thésauri et d'autres ouvrages techniques qui supportent le domaine d'application.

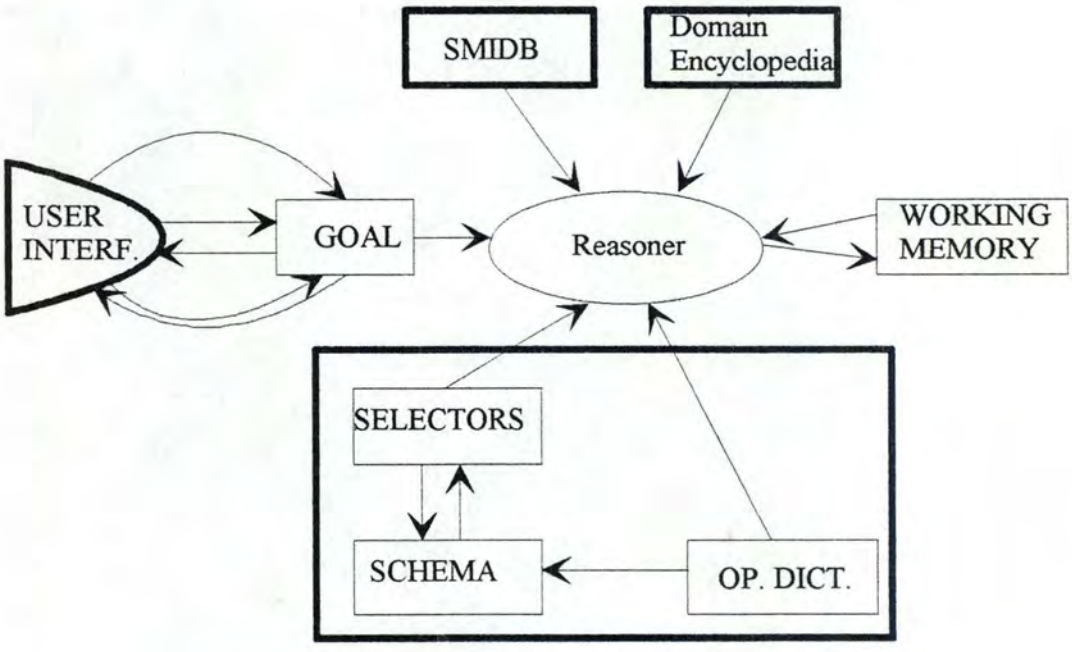
Pour engendrer la construction de cette "Domain Encyclopedia" dans le système EISI, un ensemble d'outils ont été conçus pour faciliter la définition, la maintenance et la documentation du contenu de l'encyclopédie. Cette "boîte à outils" est appelé Domain Encylopedia Workbench".

3.2.3. Description fonctionnelle

Le fonctionnement du système EISI s'articule en fait autour du "Domain Assistant"[4] qui est l'interface intelligent du système. En réalité, c'est plus qu'une simple interface, c'est également un système de "raisonnement" qui permet de répondre à la requête d'un utilisateur en parcourant et en interagissant avec l'ensemble du système EISI.

Cette description aura donc comme but d'expliquer le fonctionnement du Domain Assistant en restant cependant à un niveau assez général.

3.2.3.1. Architecture.



[4]Epprecht a écrit sa thèse de doctorat sur le fonctionnement du Domain Assistant. On peut donc y trouver des renseignements bien plus spécifiques et pointus. (Voir [Epprecht])

3.2.3.2. Processus logique

Séquence générale du processus :

1. obtention de la définition du problème de la part de l'utilisateur
2. identification des données et sources concernées
3. sélection d'une stratégie
4. expansion de la stratégie
5. output du plan

L'interface utilisateur reçoit la définition du problème de la part de l'utilisateur. On la stocke dans le "Goal" et on la retravaille interactivement pour qu'elle soit correcte et complète. Ensuite, le processus de raisonnement débute en mode "batch". Le "Goal" est comparé avec les sélecteurs. Dès qu'un sélecteur correspond, le schéma correspondant est stocké dans la "Working Memory". Le "Reasoner" recherche alors dans la SMIDB (Statistical Meta Information DataBase) et dans la "Domain Encyclopedia" les (méta)informations relatives au problème et les relie au schéma. Ensuite, on passe à l'expansion du schéma (il se peut que ce schéma global ait besoin de schémas partiaux). Après son expansion, le schéma est évalué : le "Reasoner" exécute les opérations de validité (décrites dans "l'Operations Dictionary" pour chaque opération primitive). De cette manière, le "Reasoner" ne laisse à l'utilisateur que les opérations relatives à l'examen des données statistiques actuelles au problème. Finalement, le plan actuel est mis en "forme" et présenté à l'utilisateur.

3.2.3.3. Goal

Il s'agit du lien primaire entre le "Reasoner" et le "User Interface" . Il consiste en un ensemble de qualificateurs, chacun correspondant à un élément de la définition du problème (ex: catégorie de but, objets principaux, période de temps,

lieu, ...). Chaque qualificateur a une structure en 3-uplet : <label> <classe> <valeur>

3.2.3.4. User Interface

Par "User Interface", on entend un composant qui est totalement indépendant du mécanisme de raisonnement. Il s'agit donc d'un simple outil de manipulation permettant à l'utilisateur d'entrer en contact avec le système. C'est en quelques sortes un "simple" gestionnaire d'entrée-sortie.

3.2.3.5. Selectors

Ils contiennent des connaissances sur la relation entre la définition, le contexte du problème et la stratégie de résolution du problème correspondant. Ils contiennent également la spécification de la méta-information requise par les schémas auxquels ils correspondent.

La définition et le contexte du problème sont représentés dans un sélecteur par :

- une description générale du but (ex : la catégorie de questions et les domaines des valeurs des éléments complémentaires de la question).
- des préconditions

!!!! Goal Object = label - classe - valeur

ET sélecteur = label - classe - liste de valeurs

Le sélecteur correspond donc à une description générique du problème (il identifie une catégorie de problème).

3.2.3.6. Schema

Un "schema" est la représentation interne d'une stratégie (ou plan de solution) (stocké dans le corps d'un scénario en tant que modèle (template) pour la résolution de problèmes) . Il contient également l'identification de la méta-information nécessaire pour la résolution du problème auquel il est associé (utilisation de références à ces informations). Pour arriver au plan final, on exécute "l'expansion" du schéma.

3.2.3.7. Operations dictionnary

Il contient les descriptions des opérations pré-définies qui représentent les actions élémentaires d'un statisticien. Il contient également les définitions de macros qui sont des séquences fixes d'opérations élémentaires.

3.2.3.8. Working Memory

Cette mémoire de travail stocke le contexte actif (des processus lancés) durant tout le mécanisme de raisonnement. Elle se compose de différents sous-éléments :

- Operations Stack : utilisé pour activer chaque opération dans le schéma et ceci de façon ordonnée (but de la pile) et pour contrôler la récursivité.
- Agenda : utilisé pour garder une trace des résultats de toutes les opérations de raisonnement
- Variable Table : contient la description et l'état des variables et qualificateurs utilisés dans le système.
- Scratchpad : contient les résultats intermédiaires (y compris les erreurs possibles) durant l'exécution.

3.2.3.9. "Case" library.

Ce module n'est pas représenté dans l'architecture du début de cette section mais fait partie intégrante de la version actuelle du système EISI et est développé dans [VANEY92].

La "Case library" est une mémoire extensible dans laquelle sont stockés les résultats intermédiaires et finaux de tous les problèmes que le "reasoner" a déjà résolu. Quand un problème est résolu par le "reasoner", celui-ci enregistre dans la "Case library" tous les résultats des étapes de la résolution. Chaque résolution de problème, qu'elle se termine sur un échec ou une réussite, est un "cas" et est donc enregistrée dans la "Case library".

L'intérêt de ce module est de permettre au "reasoner", avant la résolution d'un problème, d'aller consulter cette librairie afin de savoir s'il a déjà résolu le même problème avec les mêmes paramètres dans le passé. S'il l'a déjà résolu, il ne va pas le résoudre une seconde fois mais recherche les résultats dans la "Case library" pour les communiquer à l'utilisateur.

La séquence logique du processus donné à la section 3.2.3.2 sera modifiée avec ce module. La seconde étape sera la recherche dans la "Case library" afin de savoir si le problème a déjà été résolu, auquel cas on communique les résultats à l'utilisateur sans exécuter les étapes suivantes du processus.

CHAPITRE 4

"Planning et statistiques officielles"

4.1. Théorie sur le planning

4.1.1. Introduction

Un des concepts de base de l'informatique est la division du travail. Lorsque l'on est confronté à un problème de taille assez imposante, il est préférable de travailler séparément sur des petites parties de ce problème et ensuite, de combiner les solutions "partielles" en une solution globale.

Il y a principalement deux raisons pour effectuer une telle décomposition :

1. Il est important et plus facile, quand on passe d'un état du problème à un autre, de ne pas devoir recalculer entièrement le nouvel état. Il est plus aisé de ne considérer que la partie de l'état qui a changé (ex.: si je passe d'une pièce à une autre, cela n'affectera pas les autres objets se situant dans les pièces).

2. Il est plus aisé de résoudre des sous-parties d'un problème (moins importantes donc moins difficiles) plutôt que le problème en un bloc : on réduit la difficulté générale en des composants de difficulté moindre.

Pour résoudre de tels problèmes "décomposables", nous voudrions une méthode qui puisse nous permettre de travailler sur chaque sous-problème séparément et également d'enregistrer les éventuelles interactions parmi eux. Une des méthodes répondant à ces besoins se nomme "planning".

Dans cette première partie de chapitre, nous analyserons également cette approche "planning" (dans le sens même de l'utilisation que nous en ferons au chapitre 5) en relation avec le concept de "scénario" tel qu'on le retrouve dans le système EISI (qui est une des bases de notre étude pour ce mémoire) et dans [EPPRECHT]). Cette partie nous permettra dans le chapitre 6 d'aborder une étude comparative avec ce système EISI.

4.1.2. Approche générale

"Le planning est une méthode centrée sur le fait de décomposer un problème initial en sous-modules appropriés et sur la manière d'enregistrer et de traiter les interactions entre ces sous-modules quand elles sont détectées dans le processus de résolution du problème."[1]

Plus généralement et plus grossièrement, le planning réfère au fait de :

[1] [RICH]

"Décider sur une suite d'actions avant de les exécuter."[2]

Cette suite d'actions est généralement appelée "plan". Un plan est un processus hiérarchique qui peut contrôler l'ordre dans lequel une séquence d'actions doit être exécutée.

Beaucoup de plans contiennent des étapes vagues, trop générales et qui demandent une spécification plus approfondie. Chacune de ces étapes peut alors être remplacée par un sous-plan un peu plus spécifique et donc plus détaillé. Ce raisonnement s'applique ainsi récursivement afin d'aboutir à une spécification telle qu'il n'y ait plus de doute ou d'incompréhension quant à l'action à effectuer.

Evidemment on peut se demander si un tel raisonnement "récursif" ne risque pas d'être trop lourd et trop important (voir figure 1). Par exemple, un planning trop détaillé de l'action "dîner" donnerait: -aller à table -allumer les bougies de table -s'asseoir -prendre sa serviette -se verser un verre d'eau -se servir -goûter si c'est chaud -etc..... . Alors, faut-il prévoir un plan tenant compte de toutes les réactions du système et du monde extérieur ? Peut-on "simplement" tout prévoir ? Une réponse positive à ces questions nous conduirait à l'élaboration de plans gigantesques dans lesquels on noterait l'intervention d'un nombre d'étapes tout-à-fait décentrées du problème initial.

Il faut cependant constater que la majorité de ces éventuelles étapes sont souvent improbables, c'est pourquoi lors de la création d'un plan, il est préférable de se maintenir à un niveau de vraisemblance et de probabilité assez élevé : envisageons uniquement les étapes réellement possibles. De nouveau, il est nécessaire de se référencer au contexte[3] du problème qui nous permet de différencier l'accessoire du critique.

---

[2] [COHEN]

3 Epprecht dans son ouvrage [EPPRECHT] insiste fortement sur l'importance de ce terme dans la technique de planification et pour la compréhension du problème étudié, il en donne d'ailleurs une définition précise : "*Le contexte est toute information qui n'est pas explicitée dans la définition du problème mais qui est cependant nécessaire pour la compréhension de celui-ci*".

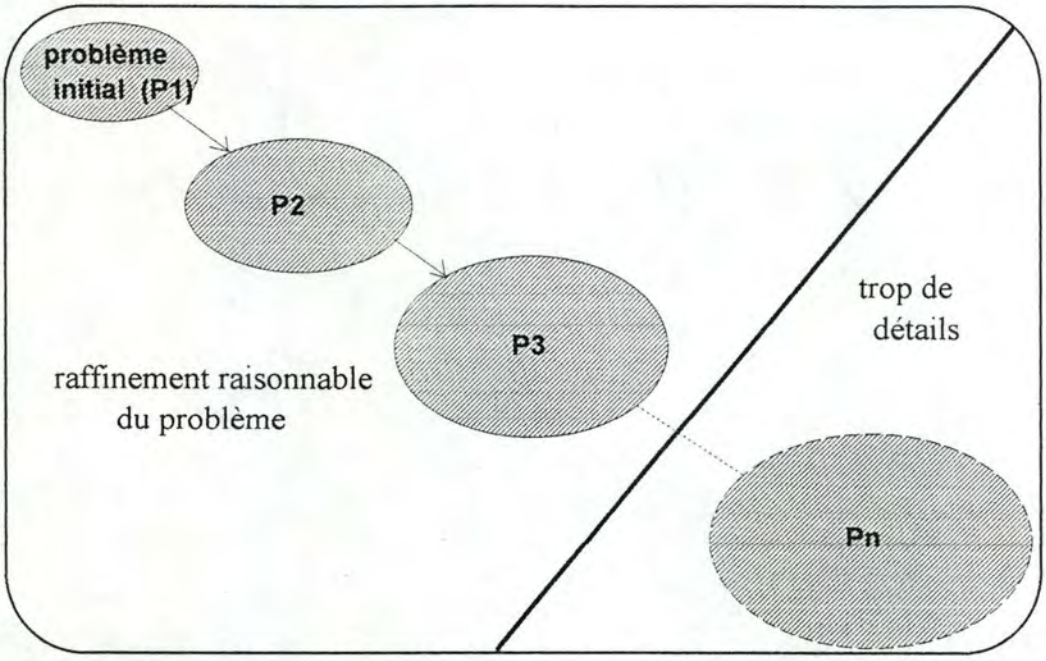


Figure 1

Ces étapes doivent alors se dérouler dans un ordre précis afin d'éviter tout effet de bord. Par exemple, si lors de la restauration d'une pièce d'une maison, on met en couleur le plafond juste avant de frotter les murs, la poussière en résultant annulera la mise à neuf du plafond. Le plan doit donc être attentivement étudié afin de bien agencer toutes les étapes le composant et pour que toutes les informations et conditions nécessaires à l'application de l'une d'elles (= préconditions) soit complètes et correctes sous peine de rétroactions et de bouclages constants (voir fig.2).

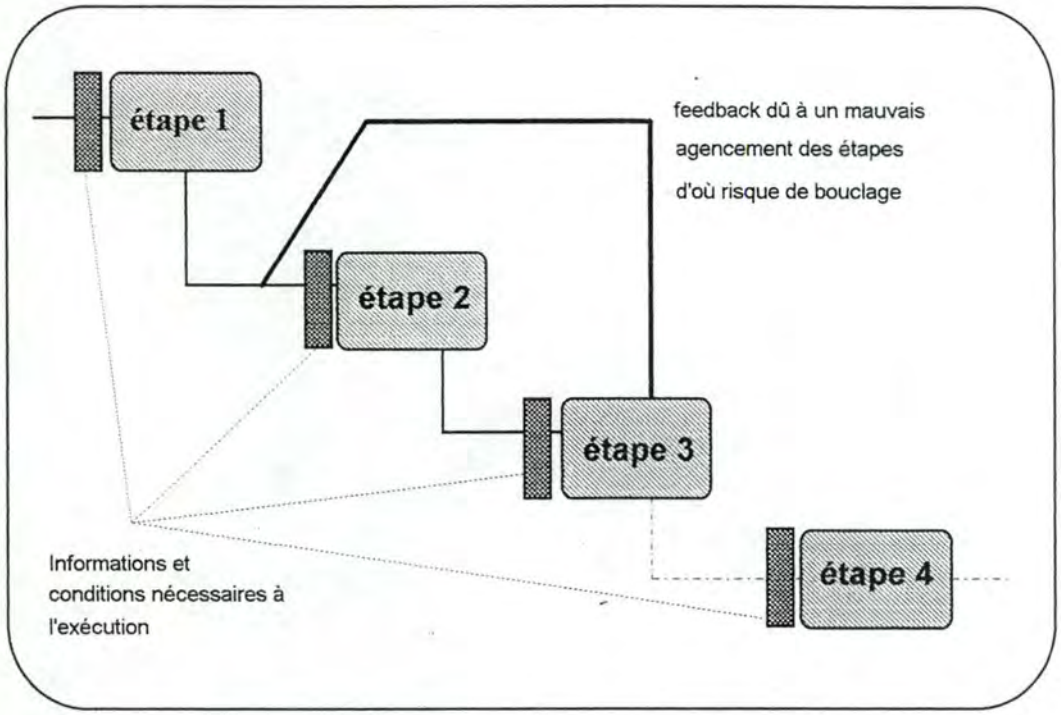


Figure 2

Dans le schéma ci-dessus, on peut envisager quatre cas possibles :

1. Toutes les étapes ont été bien agencées et les "préconditions" sont bien remplies donc tout se passe bien.

2. Les étapes ont été mal agencées et la "faiblesse" des préconditions entraîne un bouclage (ex. : dans le cas de la restauration d'une pièce, si les préconditions sont trop faibles, alors on va recommencer le plafond puis on "re-frottera" les murs, puis le plafond, puis les murs ...).

3. Les étapes ont été mal agencées mais les préconditions sont complètes et *correctes* donc il va y avoir une halte dans l'exécution.

4. Les étapes sont bien agencées et les préconditions sont complètes et correctes mais l'intervention d'un événement inattendu entraîne un blocage à une précondition (ex.: dans le cas *correct* de la restauration de la pièce, s'il n'y a pas plus de peinture pour le plafond).

Dans ce dernier cas, on voit la nécessité d'instaurer la notion de *feed-back* (voir fig.3). Grâce à cette technique, il est possible d'abandonner un chemin bloqué et de

remonter à un niveau permettant la reprise d'une autre direction afin de poursuivre l'exécution du problème. Ce processus de "branchement" est appelé, dans la méthode de planning, le *backtracking*. Le déclenchement de ce processus est dû à l'apparition de *deadends* c'est-à-dire de situations telles que les préconditions exigées ne peuvent être remplies.

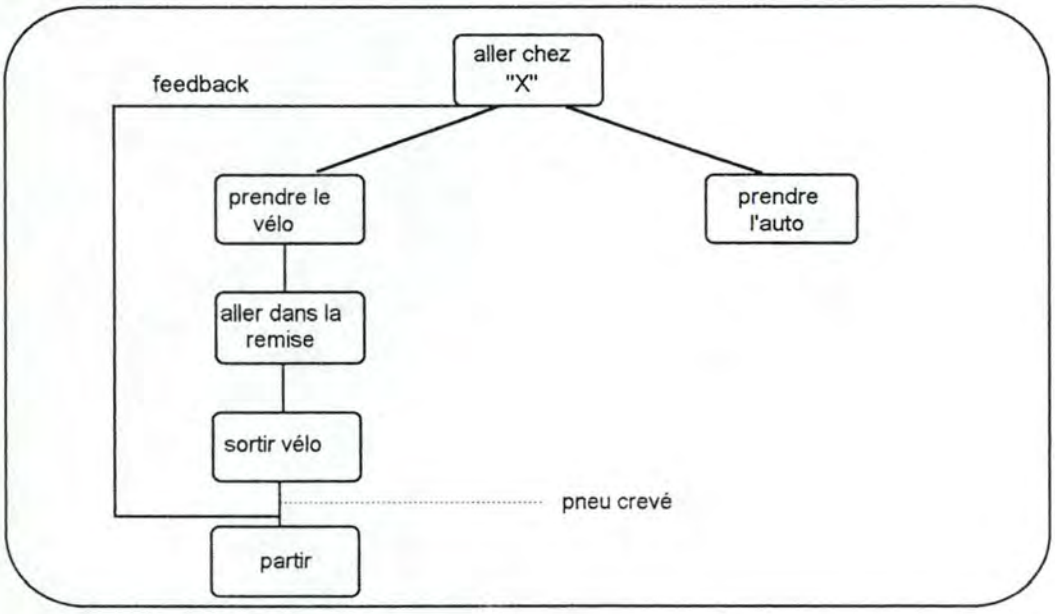


Figure 3

Rôle des préconditions (et postconditions)

L'une des clés de voûte de cette méthode de planning est la "précondition". En effet, ce composant s'avère indispensable pour le bon fonctionnement du plan.

Les préconditions regroupent toutes les conditions nécessaires à l'exécution de chaque étape à laquelle elles se rattachent. Elles stipulent les caractéristiques obligatoires de tous les paramètres utilisés dans l'étape ainsi que l'état demandé du "monde extérieur" avant l'exécution de celle-ci.

Comme on l'a vu auparavant, une négligence au niveau des préconditions peut entraîner des effets néfastes dans le déroulement du plan[4]. Une écriture correcte et complète de ces préconditions demande une connaissance parfaite du but et du déroulement de l'étape du plan : aucun détail n'est à négliger. De plus, par le système de décomposition utilisé dans la méthode de planning, il est nécessaire de ne pas "oublier" l'effet de désagrégation : une étape exécutée au niveau d'abstraction "n" signifie que toutes les préconditions de niveau "1.....n-1" ont été acceptées. Une précondition reprend donc une hiérarchie de préconditions antérieures qui parfois (selon la syntaxe utilisée), à la lecture de la spécification, semblent avoir été oubliées .

4.1.3. Typologie des plannings

Dans la littérature consultée, nous avons répertorié trois grandes catégories de planning. A ceux-ci, nous avons rajouté un quatrième type, moins important, qui envisage une approche assez différente des trois autres. Nous reprendrons les intitulés utilisés par Cohen et Feigenbaum[5]. Il s'agit des plannings :

- Hiérarchiques
- Non-hiérarchiques
- Basés sur des scripts / Squelettes
- Opportunistes

La plus grande majorité des plannings ont une *structure hiérarchique de sous-buts* (voir fig.4). Cependant, il y a une autre interprétation de ce mot "hiérarchie" qui constitue la base pour la comparaison entre les deux premiers types de plannings cités ci-dessus. Cette interprétation est expliquée dans le point suivant

---

[4] voir les quatres interprétations possibles de la figure 2

[5] [COHEN]

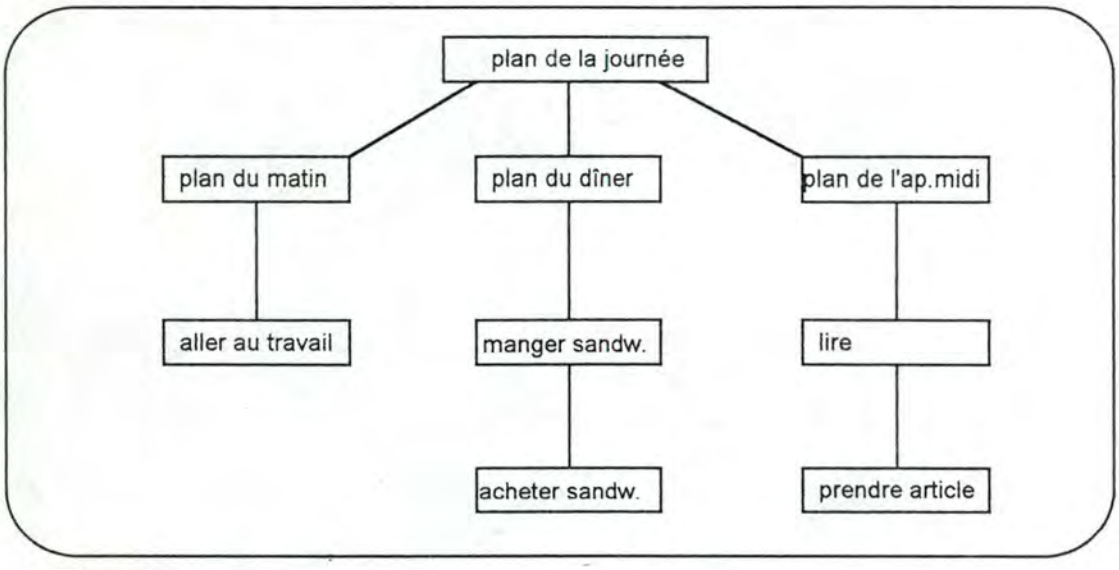


Figure 4

4.1.3.1. Les plannings hiérarchiques et non-hiérarchiques

Un planning hiérarchique génère une *hiérarchie de représentations* d'un plan dans laquelle la première est une simplification du plan et la dernière est un plan détaillé *suffisant* pour résoudre le problème. Par contre, un planning non-hiérarchique dispose seulement d'*une* représentation de plan.

Les plannings non-hiérarchiques correspondent grossièrement à la définition "plate" du mot "planning" : ils développent une séquence de résolutions de problèmes pour aboutir à chacun des buts. Ils peuvent réduire les buts à d'autres plus simples ou ils peuvent utiliser la technique d'analyse "means-ends"[6].

[6] Cette technique consiste à regarder la différence entre l'état courant du problème et l'état désiré (correspondant à l'état après obtention du résultat escompté) et à essayer de trouver une opération de résolution de problème qui réduira cette différence. Ceci continuant récursivement jusqu'à ce que le résultat désiré soit atteint. Les composants principaux de cette technique sont en fait les préconditions et les postconditions. En effet, pour réduire cet écart entre l'état courant et l'état désiré, on recherche des étapes telles que la postcondition se rapproche du but recherché et telles que la précondition tient compte des étapes déjà exécutées.

Le principal désavantage du planning non-hiérarchique est qu'il ne fait aucune distinction entre les étapes de résolution qui sont critiques pour la résolution finale du problème et celles qui ne sont simplement que des détails. En conséquence, Les plans résultants tombent dans un niveau de détails à outrance et nécessitent une dépense d'efforts inutiles.

Sur ce point, les plannings hiérarchiques sont plus performants. La méthode qui y est utilisée consiste à se baser sur un plan qui est complet mais trop vague et à redéfinir les parties "floues" dans des sous-plans plus détaillés jusqu'à ce que le problème initial soit résolu (on détaille toujours vis-à-vis du problème posé ce qui évite les détails inutiles). On retrouve donc bien cette hiérarchie de représentations où chaque niveau correspond à un nouveau pas dans la précision des spécifications utiles à la résolution du problème initial. Chacun des composants (= sous-buts) de chaque niveau a un niveau de détail équivalent. C'est surtout ici que l'on voit la différence avec les plannings non-hiérarchiques: on ne s'y préoccupe pas de ces niveaux de représentation, on développe les sous-buts afin que ceux-ci soient complets sans se soucier du caractère critique ou non de ces multiples raffinements.

4.1.3.2. Les plannings dits "squelettes" ou basés sur des scripts[7]

Cette catégorie de planning utilise des plans "squelettes". Contrairement aux plannings hiérarchiques, ces plans sont issus d'un *stock* de plans à la place d'être générés. Ceux-ci contiennent des lignes générales pour résoudre un grand nombre de problèmes différents.

Le processus de planning procède en deux approches :

1. Un plan squelette qui est applicable au problème donné est trouvé

[7] [STEFIK]

2. Ensuite, les points abstraits ou généraux de ce plan sont remplacés par les opérateurs de résolution de problème émanant du contexte spécifique du problème.

Ce processus d'instantiation nécessite une grande quantité de connaissance spécifique au domaine étudié.

L'idée émanant de ce type de planning est que l'on invente rarement une expérience, un processus sur une base inexistante. On se base souvent sur des expériences passées analogues. On essaie donc de trouver de telles expériences que l'on instantie avec ses propres paramètres. Les plans "squelettes" trouvé lors du processus de résolution peuvent donc être très spécifiques si le problème posé se situe à proximité d'un autre déjà traité auparavant (il n'y a donc qu'à modifier un peu le plan trouvé) mais ils peuvent également être très généraux si on ne dispose d'aucune expérience approximative déjà développée.

4.1.3.3. Les plannings opportunistes

Ce quatrième type de planning a été développé par Hayes-Roth. Il se caractérise par une stratégie de contrôle plus flexible que les trois autres types de planning développés ici auparavant. En effet, on y retrouve une espèce de tableau de bord sur lequel interviennent des "spécialistes-planning" qui émettent des suggestions sur les étapes du plan. Chaque "spécialiste" est implémenté dans le but de générer des décisions spécifiques. Ces "spécialistes" n'agissent pas dans un ordre bien établi, ils interviennent quand c'est nécessaire d'où le nom d'*opportunistes.*

4.1.4. "Planning" versus "Scénario".

Après la présentation dans le chapitre 3 du modèle EISI, nous pouvons remarquer la proximité de ce concept de "planning" avec le concept de "scénario". Le but de ce point est donc de bien situer la limite entre ces deux termes qui concernent, l'un et l'autre, la planification de tâche.

Dans [VFLMS], nous retrouvons une définition précise du concept de "scénario" tel qu'il est utilisé dans le cadre du projet EISI :

" (...) un scénario décrit un problème se déroulant dans le domaine d'application étudié et spécifie une ou plusieurs solutions partielles dans le contexte d'utilisation de méta-information. (...) Une solution partielle décrit premièrement, les caractéristiques de la méta-information requise pour le problème dans un contexte particulier et, deuxièmement, décrit la séquence des opérations qui devraient être appliquées à cette méta-information dans le but de fournir un support de décision dans l'espace du problème."[8]

Reprenons à présent la définition du "planning" citée dans [RICH] et que nous avons exposé au point 4.1.2. :

"Le planning est une méthode centrée sur le fait de décomposer un problème initial en sous-modules appropriés et sur la manière d'enregistrer et de traiter les interactions entre ces sous-modules quand elles sont détectées dans le processus de résolution du problème."[9]

Dans la définition de "scénario", on retrouve trois grandes étapes :

1. Recherche de la méta-information requise.

---

[8] [VFLMS] page 4

[9] [RICH] page 249

2. Recherche des caractéristiques de cette méta-information.

3. Etablissement d'une séquence d'opérations à appliquer à cette méta-information.

Un scénario joue donc deux rôles distincts : un rôle de moyen d'*acquisition de connaissance* et un rôle de *modèle pour représenter et organiser la connaissance pour une utilisation dans un mécanisme basé sur le raisonnement.*[10]

Le planning (dans le sens où nous l'utilisons), quant à lui, joue un rôle différent. Il nous permet de modéliser purement et simplement un processus analytique relatif aux statistiques officielles. Nous entendons par processus analytique, une séquence d'étapes permettant de travailler sur des données statistiques concrètes. Nous ne disposons ici que d'une connaissance "pratique" (en terme de tableaux et de chiffres) du problème. Tout le côté connaissance contextuelle est oublié d'un point de vue implémentation (ce qui n'est pas totalement vrai d'un point de vue interprétation des résultats).

Bien que ces termes désignent tous deux une technique de division des tâches (planification), ils se différencient par la "matière première" qu'ils utilisent. Un plan peut alors prendre deux définitions différentes selon le sujet que l'on étudie (les données ou les méta-données). Dans un premier domaine, celui des données concrètes, le plan s'exprimera comme étant une séquence d'actions à exécuter dans le but d'obtenir des résultats concrets relatifs à un problème sur des données statistiques (dans notre cas bien précis du GDP). Dans un deuxième domaine, celui des méta-données, le plan s'exprimera comme étant une séquence d'actions à appliquer à une méta-information dans le but de fournir un support de décision dans l'espace d'un problème donné. On distingue ici la différence de résultats obtenus selon le domaine d'étude où l'on se trouve : soit on reçoit un résultat unique et concret, soit on reçoit une aide à la décision (qui peut ne pas être univoque) qui permettra *par la suite* de décider quant aux actions concrètes à prendre.

Le point commun entre ces deux concepts est le script ou squelette. En effet, ces deux concepts utilisent des plans (attention à la définition de ce terme selon le

[10] [VFLMS] page 4

domaine étudié) qui sont stockés quelque part en mémoire et qui sont réutilisés pour des cas assez semblables (ce mécanisme est développé dans le chapitre 5).

4.2. Le domaine des statistiques officielles

4.2.1. Le travail du statisticien en statistiques officielles

Le travail du statisticien se décompose en quatre grandes étapes :

1. Dans la première étape, le statisticien reçoit un problème formulé vaguement qu'il doit reformuler explicitement dans des termes propres au domaine d'application (économie intérieure, économie extérieure, ...). Cette phase de reformulation s'avère indispensable pour le choix de la solution adéquate : "*Une grosse partie de l'expertise d'un statisticien consiste à aider le chercheur à reformuler sa première question. (...) il est nécessaire de, d'abord, élucider exactement quel est l'objectif du chercheur avant que l'on puisse décider quelle méthode pourrait être appropriée pour explorer cet objectif*"[11].

2. La deuxième phase concerne plus particulièrement l'identification des données impliquées dans le problème. En effet, il existe des relations entre les données, la définition du problème et les méthodes (solutions) retenues. En d'autres mots, l'ensemble des méthodes appropriées pour la résolution du problème subissent des contraintes des données disponibles et de leurs caractéristiques (ainsi que des résultats escomptés) (certaines méthodes ne peuvent s'appliquer à

[11] [HAND]

certains types de données). C'est pourquoi il est nécessaire de situer cette étape en deuxième position.

3. La troisième partie concerne spécifiquement l'exécution d'une méthode statistique. Par "méthode", nous entendons une procédure (parmi un ensemble) qui stipule ce qu'il y a à faire et comment le faire dans le but de résoudre des types donnés de problème. En effet, dans la statistique officielle, il existe des catégories de problèmes que l'on retrouve quasiment toujours (comparaison, ...) et qui donc, réduisent les solutions à envisager : il "suffit" de puiser dans un ensemble de méthodes "préconçues", d'en retirer la bonne et de l'appliquer.

4. Finalement, le dernier point concerne l'analyse et l'interprétation des résultats émanant de la méthode utilisée. Ce point est assez délicat car les statistiques officielles se caractérisent par le fait que les producteurs des données à étudier ne sont pas leurs utilisateurs et que de plus, ces données proviennent d'horizons multiples assez différents. Il en résulte donc une difficulté pour une bonne interprétation des résultats d'analyse (voir le chapitre relatif à l'exemple du PDB).

Cette division du travail du statisticien vis-à-vis des statistiques officielles est résumée dans la figure 5 suivante :

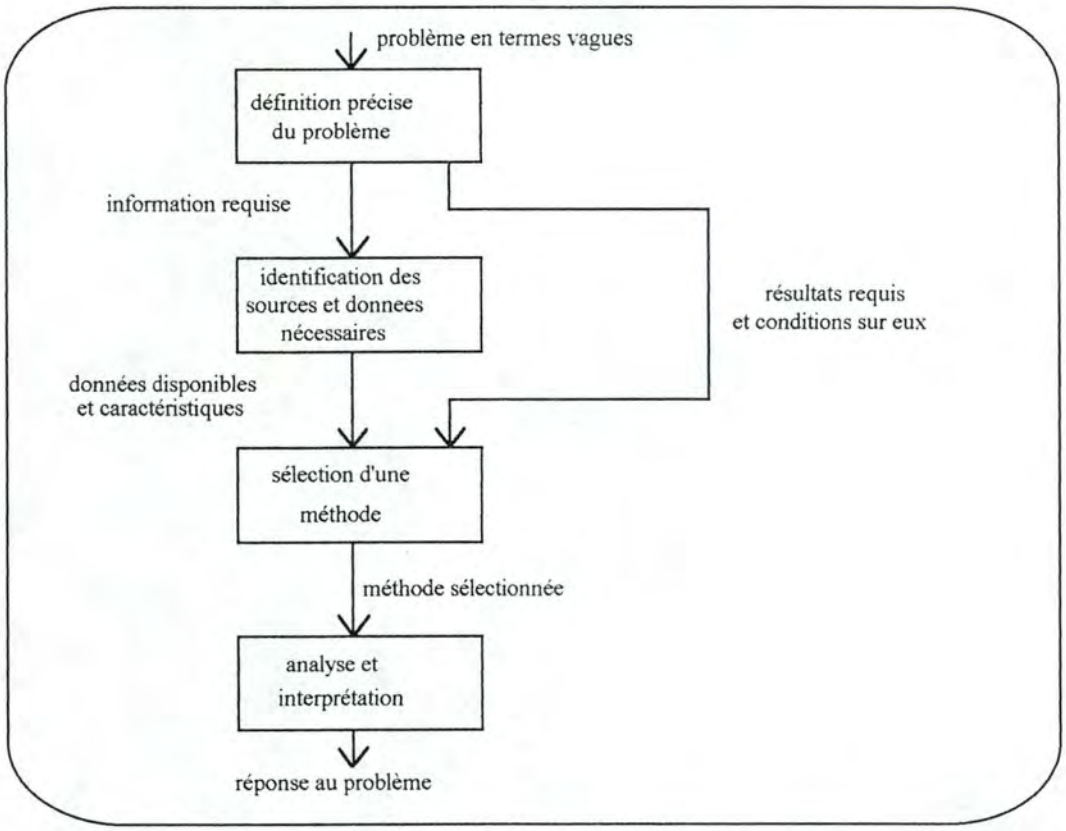


Figure 5

4.2.2. Une implémentation possible : approche globale

Ce schéma peut très bien s'appliquer pour une méthode de planning. En effet, on dispose d'un état de départ (le voeu de l'utilisateur, du chercheur), d'un état final escompté (une réponse à ce voeu) (résultant du raffinement de l'état de départ) et d'un ensemble de méthodes possibles. Une solution serait donc de proposer un planning qui permettrait, suite à la formulation d'un problème, de visionner et d'appliquer une suite d'actions (plan) afin d'obtenir un résultat possible.

Les étapes relatives à la "définition du problème" et à la "sélection d'une méthode" pourrait intervenir dans un interface (dialogue avec l'utilisateur) tandis que l'application de la méthode sélectionnée et la production d'analyse (ainsi qu'une petite partie des interprétations possibles) seraient plutôt reprises dans une sorte de "moteur" ou "d'analyseur".

Pour l'interprétation (l'analyse) correcte des résultats, il est nécessaire de disposer de méta-informations car, comme nous l'avons vu dans le point précédent, les données "officielles" proviennent d'horizons divers ce qui nécessitent le plus d'informations possible afin de donner une analyse correcte aux résultats. Dans le planning que nous développons, nous n'envisageons pas ce domaine "méta-informations" (plus tard, une intégration dans le système EISI peut être possible).

4.2.2.1. Implémentation possible versus théorie des plannings

Le planning que nous allons détailler dans la suite de ce mémoire ne résulte pas de l'application pure d'une des quatre grandes catégories développées auparavant (hiérarchique, non-hiérarchique, squelette, opportuniste). Cependant, il reprend principalement les caractéristiques des plannings "hiérarchiques" et "squelettes":

- On dispose d'une hiérarchie de représentation c'est à dire que l'on part d'un problème vague ("*y-a-t-il des biais d'estimations dans les séries d'estimations du GDP ?*") et on le raffine par vagues pour obtenir des étapes exécutables. Comme tout planning hiérarchique, on ne verse pas dans un déluge de détails, on se tient au principal.

- Les plans ne sont pas totalement générés. On dispose en fait de plans généraux que l'on doit instantier pour qu'ils soient réellement envisageables pour le problème posé. Cette particularité nous rapproche des plannings squelettes. Ce type de planning est particulièrement bien adapté au domaine des statistiques officielles, car comme nous l'avons fait remarquer dans la section 4.2.1, dans le domaine des statistiques officielles, il existe des catégories de problèmes que l'on retrouve quasiment toujours et on peut donc créer des plan "squelettes" spécifiques à la plupart des catégories de problèmes

Nous ne rentrerons pas ici davantage dans la spécification détaillée de l'implémentation envisagée, ceci faisant l'objet du chapitre suivant à part entière.

CHAPITRE 5

" Présentation du système"

5.1. Introduction.

Ce chapitre est consacré à la présentation du système que nous avons développé dans le cadre de notre mémoire. L'objectif principal de celui-ci est la résolution automatique de problèmes dans le domaine des statistiques portant sur des données réelles. Dans le chapitre 2, nous avons présenté et résolu un type de problème complexe que les statisticiens peuvent être amenés à résoudre. Pour la résolution de ce problème, l'utilisateur a à sa disposition un certain nombre de fonctions, chacune d'elles permettant de résoudre une partie du problème. Mais même si l'utilisateur dispose de toutes les fonctions nécessaires, celles-ci ne sont pas suffisantes à elles seules. En effet, seul un spécialiste de la résolution d'un tel problème connaît l'ordre dans lequel ces fonctions doivent être exécutées. Le rôle de notre système sera donc l'ordonnancement des différentes fonctions nécessaires à la résolution d'un problème ainsi que l'exécution de celles-ci, afin de permettre à une

personne non spécialisée dans ce type de problème de le résoudre.

Pour pouvoir remplir ses rôles, le système dispose d'un ensemble de plans, chacun contenant les lignes générales à suivre pour résoudre un type de problème appartenant au domaine des statistiques officielles. Lorsque le système reçoit une demande de résolution d'un problème, il exécute les différentes étapes d'un plan permettant la résolution du type de problème correspondant au problème posé, en adaptant le plan à celui-ci. Le système utilise un système de planning de type "hiérarchique" et "squelette", comme nous l'avons écrit dans la section 4.2.2.1.

La première partie de ce chapitre reprend une présentation de l'architecture du système ainsi qu'une description de ses modules principaux . Le système ayant été développé à partir de l'étude du système EISI, notre architecture s'y rapproche quelque peu, ce rapprochement étant d'ailleurs très intéressant pour l'évolution future de notre système. En effet, l'étape suivante de son développement sera l'intégration de celui-ci avec le système EISI. On disposera ainsi d'un système qui permettra à un utilisateur de connaître, non seulement les informations sur les données et sur la stratégie suivie pour résoudre un problème donné, mais également d'avoir la solution à ce problème. L'extension future du système sera discutée au chapitre 6.

Il importe cependant de prendre ces similitudes avec prudence. Plusieurs modules de notre système se rapportent à des modules du système EISI, mais les objectifs précis de ces deux systèmes étant différents, il s'avère souvent que les rôles de deux modules correspondants sont différents en partie ou entièrement.

Les sections suivantes sont consacrées au développement plus en détails de chaque module du système, en nous intéressant essentiellement à la représentation des plans dans le système et à la présentation de la résolution d'un problème par le système. Ces deux notions représentent les différences principales entre notre système et EISI. Dans la dernière partie, nous développerons un mini-planificateur, qui a été mis au point comme prototype de démonstration du système.

5.2. L'architecture du système.

Le système est constitué de 5 modules:

•La base des données opérationnelles qui contient toutes les données directement nécessaires à l'exécution des processus du système.

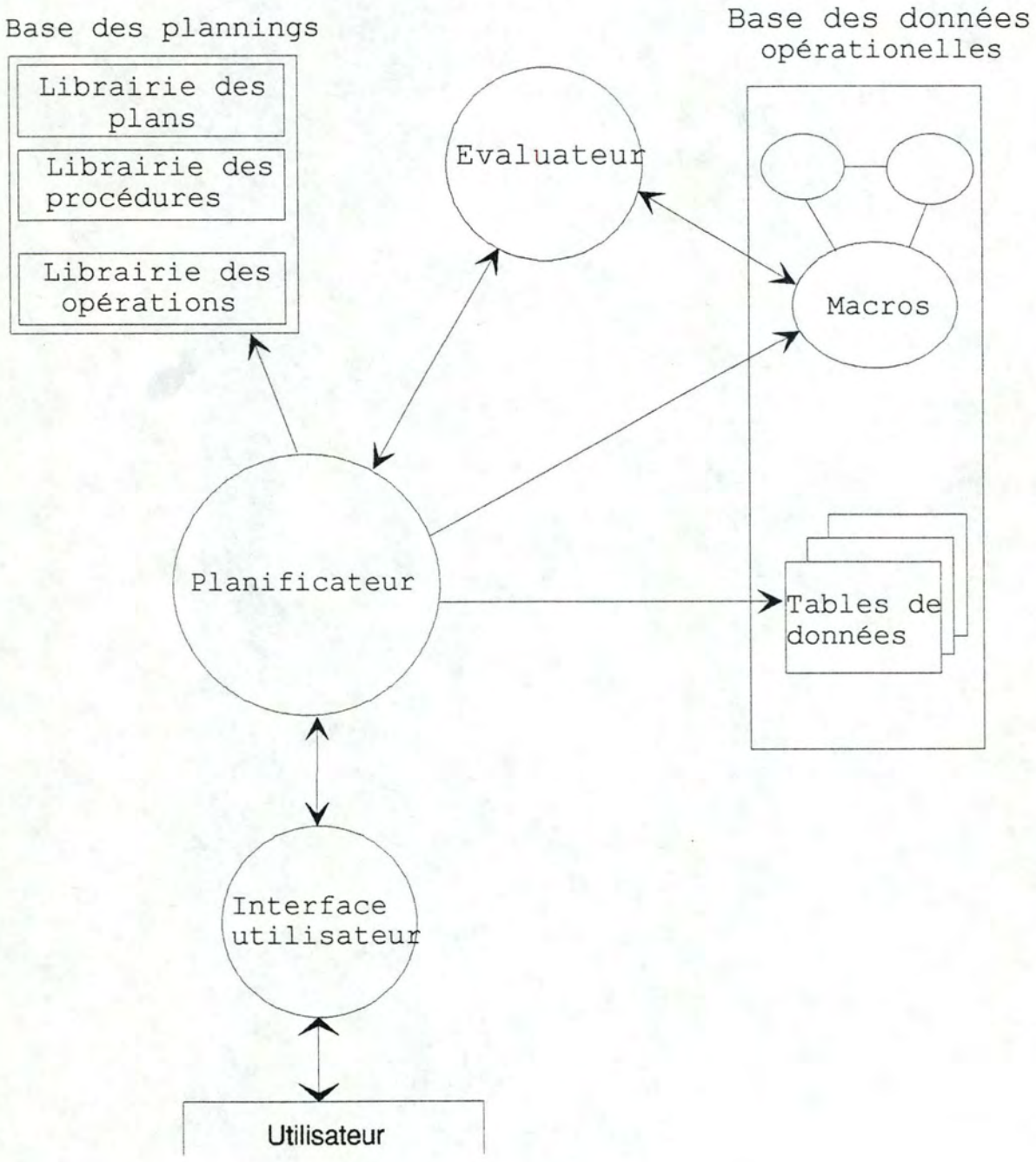
•La base des plans qui contient la description des plans utilisés pour la résolution de problèmes par le système.

•L'évaluateur des macros qui permet d'interpréter et d'exécuter des macros.

•L'interface utilisateur qui permet à un utilisateur de communiquer avec le système.

•Le planificateur qui s'occupe de la résolution des problèmes posés par un utilisateur grâce à un système de planning.

Schéma :


Base des plannings
Librairie des plans
Librairie des procédures
Librairie des opérations
Base des données opérationelles
Evaluateur
Macros
Planificateur
Tables de données
Interface utilisateur
Utilisateur

Chacun de ces modules est développé de manière autonome et peut communiquer avec d'autres modules par l'échange d'informations. L'indépendance de ces modules est nécessaire afin de permettre une réutilisation maximale de ceux-ci dans d'autres systèmes et de permettre au système de s'intégrer avec d'autres systèmes et d'évoluer. Outre ces modules, il est nécessaire de disposer de modules d'acquisition de connaissance, comme l'acquisition de données opérationnelles ou de plans. Nous ne développerons cependant pas ces modules, la présentation se concentrant sur la manière dont les problèmes sont résolus, de plus la description de ces modules dépend des logiciels utilisés. Les sections suivantes vont reprendre chaque module, en donner une explication détaillée des rôles principaux et en développer le fonctionnement. Cependant, avant de passer à ces sections, il peut être intéressant d'avoir une vue schématisée de la résolution d'un problème dans le domaine des statistiques à partir de la formulation de ce problème par l'utilisateur jusqu'à ce que celui-ci reçoive les résultats. Nous essayerons de bien mettre en évidence les rôles de chaque module.

5.2.1. La résolution d'un problème.

Un utilisateur peut communiquer avec le système et lui soumettre des requêtes par l'intermédiaire de *l'interface utilisateur*. Ce module aide l'utilisateur à formuler un problème de manière correcte et précise, et ensuite transmet la requête au planificateur.

Le module central du système est le *planificateur*, c'est lui qui, à l'aide d'un système de planning, lance l'exécution des différentes étapes menant à la solution du problème. Pour cela, il s'aide de plans hiérarchiques prédéfinis donnant les lignes d'actions générales à suivre. Ces plans sont stockés dans *la base des plans*. Le planificateur recherche dans cette base un plan lui permettant de résoudre le problème auquel il est confronté.

Dès que ce plan a été trouvé, le planificateur peut en développer les différentes étapes. Généralement, la requête est une demande de recherche ou de traitement sur des données statistiques, ces données sont contenues dans *la base des*

Chacun de ces modules est développé de manière autonome et peut communiquer avec d'autres modules par l'échange d'informations. L'indépendance de ces modules est nécessaire afin de permettre une réutilisation maximale de ceux-ci dans d'autres systèmes et de permettre au système de s'intégrer avec d'autres systèmes et d'évoluer. Outre ces modules, il est nécessaire de disposer de modules d'acquisition de connaissance, comme l'acquisition de données opérationnelles ou de plans. Nous ne développerons cependant pas ces modules, la présentation se concentrant sur la manière dont les problèmes sont résolus, de plus la description de ces modules dépend des logiciels utilisés. Les sections suivantes vont reprendre chaque module, en donner une explication détaillée des rôles principaux et en développer le fonctionnement. Cependant, avant de passer à ces sections, il peut être intéressant d'avoir une vue schématisée de la résolution d'un problème dans le domaine des statistiques à partir de la formulation de ce problème par l'utilisateur jusqu'à ce que celui-ci reçoive les résultats. Nous essayerons de bien mettre en évidence les rôles de chaque module.

5.2.1. La résolution d'un problème.

Un utilisateur peut communiquer avec le système et lui soumettre des requêtes par l'intermédiaire de *l'interface utilisateur*. Ce module aide l'utilisateur à formuler un problème de manière correcte et précise, et ensuite transmet la requête au planificateur.

Le module central du système est le *planificateur*, c'est lui qui, à l'aide d'un système de planning, lance l'exécution des différentes étapes menant à la solution du problème. Pour cela, il s'aide de plans hiérarchiques prédéfinis donnant les lignes d'actions générales à suivre. Ces plans sont stockés dans *la base des plans*. Le planificateur recherche dans cette base un plan lui permettant de résoudre le problème auquel il est confronté.

Dès que ce plan a été trouvé, le planificateur peut en développer les différentes étapes. Généralement, la requête est une demande de recherche ou de traitement sur des données statistiques, ces données sont contenues dans *la base des*

données opérationnelles. Outre ces données, pour suivre le plan trouvé, le planificateur doit lancer l'exécution d'un certains nombre d'opérations de base également conservées dans la base des données opérationnelles. Si ces opérations ne sont pas stockées sous forme de code immédiatement exécutable, le planificateur fait appel à un *évaluateur* qui sera chargé d'exécuter l'opération désirée en la déchiffrant dans le langage dans lequel elle est écrite. Dans notre exemple, il s'agit de l'interpréteur de macros écrites en Excel.

Quand l'évaluation des étapes du plan a été réalisée, le planificateur communique les résultats de l'évaluation à l'interface qui se charge de les communiquer à l'utilisateur.

5.2.2. Rapprochement avec le travail du statisticien.

Nous avons vu dans la section 4.2. que le travail du statisticien en statistiques officielles pouvait être découpé en 4 parties: la définition précise du problème, l'identification des sources et données nécessaires, la sélection d'une méthode et enfin l'analyse et l'interprétation. Nous pouvons rapprocher ces quatre parties avec la résolution d'un problème par le système, dont le rôle est la résolution de problèmes dans le domaine des statistiques officielles.

La définition précise du problème est mise au point par le module "interface_utilisateur" en interaction avec l'utilisateur. La seconde étape, "l'identification des sources et données nécessaires" est réalisée par le planificateur qui recherche dans le module "base des données opérationnelles" les données auxquelles le problème se rapporte. Par exemple les données concernant le produit domestique brut du Royaume-Uni pour la période allant de 1985 à 1993.

La sélection d'une méthode est réalisée par le planificateur qui recherche, s'il existe dans la base des plans, un plan qui lui permette de résoudre le problème posé. Enfin, l'analyse et l'interprétation se feront par le planificateur qui réalisera les différentes étapes du plan éventuellement en collaboration avec l'évaluateur des macros. Cette étape se fera en s'aidant de la description logique des différents niveaux de plans, contenue dans la base des plans et des données

statistiques contenues dans la base des données opérationnelles.

5.3. La base des données opérationnelles.

Par données "opérationnelles", on sous-entend que ce sont des données directement nécessaires à l'exécution d'une tâche. Sont comprises dans ces données, principalement les codes des programmes qui réalisent ces tâches, par la suite, nous les appellerons les "macros", ainsi que les données réelles, c'est-à-dire les données statistiques, sur lesquelles l'exécution d'une macro se fera.

- Les codes des différents processus: Ces codes seront appelés les "macros" et pourront avoir été écrits dans différents langages, dans notre exemple, nous les avons écrits dans le langage défini par Excel. L'ensemble des codes contenus dans la base des données opérationnelles représente l'ensemble des tâches élémentaires que le système peut exécuter.

  Exemple: Les macros qui exécutent les analyses des séries comptables nationales périodiques.

- Les données sur lesquelles les opérations seront exécutées. Le système permettant à un utilisateur non expérimenté dans la manipulation de données statistiques de consulter et de traiter de telles données, le système doit pouvoir y accéder rapidement. Elles seront donc stockées dans la base des données opérationnelles. Ces données seront généralement sous forme de tables de données, les statistiques officielles se présentant souvent sous forme de séries de nombres contenues dans des tables.

Cette base sera accessible par le planificateur ainsi que par l'évaluateur des macros. Le développement de la façon dont les données sont organisées dans la

base des données opérationnelles sort du cadre de notre mémoire, et de plus dépend du hardware et du software utilisé. Pour être complet, on pourrait cependant proposer une organisation par indexation sur le type et le nom des données.

5.4. L'évaluateur des macros.

Pour exécuter une opération, il est nécessaire de passer par un évaluateur permettant de déchiffrer chaque étape de la macro et d'exécuter cette étape. Lorsque le planificateur doit lancer l'exécution d'une opération, il fait appel à l'évaluateur, lui demandant d'exécuter la macro dont il lui fournit la référence. L'évaluateur va rechercher la macro correspondant à l'opération à exécuter et lance son exécution en renvoyant les résultats trouvés.

Exemple: Dans le cas développé au chapitre 2, l'évaluateur est l'interpréteur du langage Excel.

Remarque: Si le langage utilisé pour écrire les macros est un langage compilé, il est possible de pré-compiler les macros et de stocker le code exécutable de ces macros dans la base des données opérationnelles. Dans ce cas, il n'est pas nécessaire que le système passe par l'intermédiaire d'un évaluateur pour lancer l'exécution d'une macro.

5.5. La base des plans.

Dans le chapitre 4, nous avons introduit la notion de "planning" et de "plan". A la fin de ce chapitre, parmi les différentes classes de plannings définies, nous avons classé le planning utilisé dans ce système dans les catégories "hiérarchique" et "squelettique". Rappelons que pour permettre au système de gérer des plannings squelettiques, celui-ci doit disposer d'un stock de plans contenant les lignes générales

à suivre pour résoudre un problème appartenant à une classe donnée. Rappelons également qu'un planning hiérarchique est un planning dont les plans sont décomposés en différents niveaux de description. Les stocks de plans contenant les différents niveaux de description des plans sont contenus dans la base des plans.

La base des plans contient toutes les descriptions logiques des plans permettant l'exécution des tâches. La base des plans que nous proposons est divisée en trois librairies distinctes: la librairie des plans, la librairie des procédures et la librairie des opérations. Ces trois parties correspondent à la description logique de trois niveaux hiérarchiques des plans. La librairie des plans contenant la description des plans permettant la résolution complète d'un problème posé par l'utilisateur, la librairie des procédures comprenant la description des procédures nécessaires à la réalisation d'un sous-but du problème posé et la librairie des opérations comprenant la description logique des tâches élémentaires, qui sont en fait des opérations de bas niveau, les macros contenues dans la base des données opérationnelles.

5.5.1 Les plans dans la base de plans.

Dans la première partie du chapitre 4 relatif à la théorie sur le planning, nous avons défini un plan comme étant *"Un processus hiérarchique qui peut contrôler l'ordre dans lequel une suite d'actions doit être exécutée"*. Dans notre système, un plan *"représente et décrit des groupes de processus statistiques qui sont exécutés afin d'atteindre un but spécifique de l'organisation ou pour résoudre un type de problème global posé par l'utilisateur"*. Pour chaque type de problème que le système est capable de résoudre, il existe donc dans la base de plans la description logique d'un plan reprenant les grandes lignes à suivre pour résoudre ce type de problème.

A ce stade, il est important d'indiquer ce que l'on entend par type de problème, et quelles sont les frontières d'un type de problème. On regroupe dans un même type de problème, tous les problèmes qui peuvent être résolus en réalisant les mêmes opérations. A partir de cette classification, on peut se demander ce qui différencie deux problèmes différents appartenant à un même type de problème. La principale différence portera essentiellement sur les différentes données auxquelles le

problème se rapportera ou encore sur la différence entre certains paramètres définissant le problème.

Exemple: Voici deux problèmes différents appartenant au même type de problème: - Effectuer une analyse proportionnelle sur les séries du PDB du Royaume_Uni avec la première période comme période de base.

- Effectuer une analyse proportionnelle sur les séries des revenus des ménages en Belgique avec la dernière période comme période de base.

Comme nous sommes dans un système de planning hiérarchique, un plan sera découpé en plusieurs niveaux de représentation, chaque niveau constituant un différent niveau de la description du plan. Les raisons de cette découpe en niveaux d'un plan sont diverses :

- Permettre de diviser un problème général en sous-problèmes et donc de traiter chaque sous-objectif séparément.

- Permettre de ne pas trop alourdir un plan et de le rendre trop difficile à gérer ou de rendre toute modification trop délicate.

- Permettre une réutilisation de certains plans de niveau inférieur. Il arrive fréquemment que deux plans différents aient des sous-objectifs communs; on peut donc utiliser la description des plans nécessaires à la résolution de ces sous-objectifs pour les deux plans sans devoir les recréer.

La découpe d'un plan sera limitée à trois niveaux. Nous avons nommé ces niveaux de manière différente dans un soucis de différenciation:

1. Le plan: Il constitue le niveau supérieur de la description. Ce niveau supérieur est une description générale des grandes étapes du plan, c'est pourquoi nous lui laissons le nom de "plan"

2. Les procédures: les procédures sont des ensembles d'actions qui amènent à la réalisation de certains objectifs identifiables ou de sous-buts pour la réalisation de la solution à un problème posé par l'utilisateur. L'exécution des procédures se fera à partir d'un appel venant d'un plan et celles-ci pourront, pour atteindre leurs fins, faire appel à des opérations, qui sont au niveau le plus bas de la description.

3. Les opérations: Il s'agit de la description de tâches élémentaires au niveau le plus bas; ces descriptions décrivent des tâches élémentaires, c'est-à-dire des fonctions directement exécutables par le système.

Nous décrirons plus loin dans ce chapitre le mode de représentation adopté pour chacun de ces niveaux. Cette représentation devra utiliser une notation standard afin de permettre une indépendance de la base des plans vis-à-vis du reste du système, permettre une réutilisation de cette base par plusieurs planificateurs différents. Cependant, chaque niveau pourra être décrit avec un supplément d'information caractéristique de ce niveau; par exemple, la description d'une opération, en plus des informations standards, contiendra la référence de l'opération dans le système.

5.5.2. La représentation des différents niveaux de description d'un plan.[1]

Les plans seront utilisés directement ou indirectement par trois types d'acteurs. Il est important de les mentionner dans cette partie, car la représentation devra tenir compte des profils cognitifs de chacun de ceux-ci afin de leur permettre de déchiffrer ces plans. Ces acteurs sont:

[1] La représentation se base sur [MCDERMOTT], cependant celle-ci a été adaptée en fonction de nos besoins, et est très différents de l'originale.

a. L'utilisateur.

Pour l'utilisateur du système, il est nécessaire d'avoir des descriptions textuelles des différents niveaux de plans.

b. Le planificateur.

Le planificateur a besoin d'une représentation symbolique qui lui permettra d'exécuter les plans. Il devra connaître essentiellement les entrées, les contraintes, les étapes...

c. L'évaluateur.

Pour pouvoir lancer une exécution, il est nécessaire d'avoir une représentation en commandes des opérations pour permettre à l'évaluateur de connaître l'adresse où la macro est stockée. Cet acteur n'intervient qu'au niveau inférieur d'un plan, c'est-à-dire au niveau des opérations.

Maintenant que nous connaissons les exigences de chacun des acteurs, nous pouvons proposer une représentation pour chaque niveau de description.

5.5.2.1 La représentation d'un plan.

La description des objectifs d'un plan devra être fournie en guise d'explication à l'utilisateur. Dans ce but, la description logique d'un plan pourra comprendre:

- Une description textuelle des objectifs poursuivis par le plan décrit.

- Une description textuelle des paramètres nécessaires à l'exécution du plan.

Pour pouvoir exécuter le plan décrit, il est intéressant que le planificateur connaisse:

- Le nom des paramètres nécessaires à l'exécution du plan.

- Le corps du plan. Il s'agit de l'ensemble des étapes du plan. Une étape d'un plan pourra être un appel à un autre plan, un appel à une procédure, un appel conditionnel à un autre plan ou encore un appel conditionnel à une procédure.

- La séquence de réalisation des étapes d'un plan. Cette séquence est nécessaire lorsque l'on veut imposer un ordre autre que l'ordre séquentiel dans la réalisation des étapes d'un plan.

5.5.2.2 La représentation d'une procédure.

La description des objectifs d'une procédure devra être fournie en guise d'explication à l'utilisateur. Dans ce but, la description logique d'une procédure pourra comprendre:

- Une description textuelle des objectifs poursuivis par la procédure décrite.

- Une description textuelle des paramètres nécessaires à l'exécution de la procédure.

- Une description textuelle des résultats de l'exécution de la procédure.

Pour pouvoir exécuter la procédure décrite, il est intéressant que le planificateur connaisse:

- Le nom des paramètres nécessaires à l'exécution de cette procédure ainsi que les préconditions sur ces paramètres.

- Le corps de la procédure. Il s'agit de l'ensemble des étapes de la procédure. Une étape d'une procédure pourra être un appel d'une opération ou un appel conditionnel d'une opération.

- Le nom des résultats de l'exécution de cette procédure ainsi que les postconditions sur ces paramètres.

- La séquence de réalisation des étapes d'un plan. Cette séquence est nécessaire lorsque l'on veut imposer un ordre autre que l'ordre séquentiel dans la réalisation d'un plan.

5.5.2.3 La représentation d'une opération.

La description des objectifs d'une opération devra être fournie en guise d'explication à l'utilisateur. Dans ce but, la description logique d'une opération pourra comprendre:

- Une description textuelle des objectifs poursuivis par l'opération décrite.

- Une description textuelle des paramètres nécessaires à l'exécution de l'opération.

- Eventuellement une description textuelle de l'algorithme dans laquelle on y remplace le nom des variables par la valeur de ces variables.

- Une description textuelle des résultats de l'exécution de l'opération.

Pour pouvoir exécuter l'opération décrite, il est intéressant que le planificateur connaisse:

- Le nom des paramètres nécessaires à l'exécution de cette opération ainsi que les préconditions sur ces paramètres.

- Le nom des résultats de l'exécution de cette procédure ainsi que les postconditions sur ces paramètres.

Lors de l'exécution d'une opération, intervient un troisième acteur, "l'évaluateur". C'est lui qui exécutera la macro qui correspond à la description de l'opération. Pour pouvoir exécuter cette macro, la description de cette opération doit comprendre la référence de la macro concernée dans le système.

5.5.3 Implémentation de la représentation d'un plan.[2]

Dans la section précédente, nous avons eu une proposition de découpe en niveau d'un plan ainsi qu'une idée de représentation de différents niveaux d'un plan. Dans cette section, nous allons présenter une implémentation d'un plan à partir d'un exemple se basant sur le cas développé dans le chapitre 2. A partir du problème de l'exemple, nous allons décrire un plan capable de le résoudre. Ensuite, nous allons décrire une procédure capable de résoudre une étape du plan décrit, et enfin nous allons décrire une opération appelée par une étape de la procédure.

Nous prendrons comme exemple le cas suivant: On désire découvrir l'existence de biais dans les estimations des séries du produit domestique brut du Royaume-Uni. On désire connaître l'existence de biais dans ces séries par rapport à la première valeur de chaque série. En plus de la connaissance des biais, on voudrait connaître un

[2] Un exemple d'implémentation proche du nôtre est développé dans [VANEY93] avec l'exemple présenté au chapitre 2.

ordre de grandeur pour ceux-ci afin de pouvoir déceler s'ils sont significatifs ou non.

5.5.3.1. Description du plan Ratio_analysis

Le plan Ratio_analysis permet de résoudre un type de problème comme celui posé ci-dessus.

```
Define plan Ratio_analysis

Surface: Ce plan réalise une analyse des séries statistiques données en
         entrée. Ces analyses permettront de repérer l'existence de biais
         d'estimation dans ces séries.

Inputs: $input_sheet
        /* C'est la référence aux données qui contiennent les séries à
        traiter */
        $base_year
        /* Il s'agit d'un indicateur qui indique si la comparaison des
        valeurs des séries s'effectuera sur la première période ou sur la
        dernière période */

Body : Qual Achieve-proc qualitative_analysis $Base_year $Input_sheet
       Prop Achieve-proc proportional_analysis $Base_year
       $Input_sheet
       Vari Achieve-proc variability_analysis $Base_year $Input_sheet


Sequence: Qual Seq Prop

End define plan.
```

Explications:

- La description d'un plan est marquée par la clause "DEFINE

PLAN" suivie par le nom du plan et la fin de la description est marquée par la clause "END DEFINE PLAN".

• La clause "SURFACE": Il s'agit de la description textuelle des objectifs poursuivis par le plan.

• Les "INPUTS" du plan: Il s'agit des paramètres du plan. Pour éviter de confondre les paramètres, ou variables, avec d'autres éléments de la description, les variables seront précédées d'un signe particulier (par exemple le signe $). De plus, après la nomination des paramètres, on pourra ajouter un commentaire explicatif qui sera placé entre les clauses /* et */. Lors de l'exécution, la variable $input_sheet sera la référence des séries du PDB du Royaume_Uni et la variable $base_year prendra la valeur "FIRST" qui signifiera que les analyses s'effectuent par rapport à la première valeur d'une série.

• Le corps du plan (Body): Il s'agit de l'ensemble des étapes du plan. Chaque étape sera décrite sous forme d'une notation générale.

Un appel à un autre plan sera noté de la sorte:

Name1 Achieve_plan plan1(parameter) où

Name1 est un identifiant de l'étape, Achieve_plan indique que l'on doit atteindre la réalisation d'un autre plan, plan1 est le nom du plan à exécuter et parameter est l'ensemble des paramètres passés au plan.

Un appel à une procédure sera noté comme suit:

Name2 Achieve_proc proc1(parameter) où

Name2 est un identifiant de l'étape, Achieve_proc indique que l'on doit atteindre la réalisation d'une procédure, proc1 est le nom de la procédure à exécuter et parameter est l'ensemble des

paramètres passés à la procédure.

Un appel conditionnel sera noté:

If (Cond1) Achieve_plan plan1(paramater)

ou

If (Cond2) Achieve_proc proc1(parameter).

Dans notre exemple, il y a trois étapes à réaliser dans le plan.

- La clause "SEQUENCE": on ecrira

Name1 SEQ Name2, ce qui signifie que l'instruction identifiée par Name1 doit être exécutée avant l'instruction identifiée par Name2.

5.5.3.2. Description de la procédure proportional analysis.

Dans cette section, nous allons décrire la procédure proportional_analysis. Il s'agit de la deuxième étape du plan Ratio_analysis.

Define procedure proportional_analysis
Surface: Exécute une analyse proportionnelle des séries données, affiche le tableau des résultats, permet de créer et d'afficher des graphiques avec les résultats obtenus.
Input: $input_sheet /* C'est la référence aux données qui contiennent les données à traiter */ $base_year /* Il s'agit d'un indicateur qui indique si la comparaison des valeurs des séries s'effectuera sur la première période ou sur la dern,ière période */
Preconds : performed (qualitative_analysis $Input_sheet $Base_year)

```
Body:
   prop  If (performed (proportional_analysis  $input_sheet $Base_year)
            Achieve proportional_analysis  $Input_sheet $Base_year
   disp   Achieve display_table $Result_sheet
   crea   Achieve create_graphic $Result_sheet
   prin   Achieve print_result $Graphic


Expected_results:
   $result_sheet  /* Il s'agit du tableau résultant du traitement des
données */
   $aplot /* Il s'agit des graphiques créés sur base des résultats */

Postconds:

Sequence:
   prop seq disp
   prop seq crea
   prop seq prin

End_define_procedure.
```

Explications:

La description d'une procédure ressemble autant que possible à la description d'un plan dans un souci d'homogénéite et de facilité de gestion par le planificateur. Certaines explications supplémentaires sont cependant nécessaires:

- La description de la procédure débute par la clause "DEFINE PROC" suivie par le nom de la procédure et se termine par la clause "END DEFINE PROC".

- Les préconditions: les préconditions seront exprimées par des prédicats qui devront être vérifiés. La manière dont la valeur des prédicats seront connus par le planificateur sera abordé dans la section 5.7.

• Le corps de la procédure (Body): La représentation des étapes est équivalente à la représentation dans la plan sauf qu'il s'agira d'appels à des opérations.

• La description des résultats (Expected_result) et des postconditions est identique à celle des paramètres et des préconditions

5.5.3.3. La représentation de l'opération "perform proportional analysis".

Dans cette section, nous allons décrire l'opération perform_proportional_analysis, opération appelée par la procédure décrite précédemment.

```
Define operation perform_proportional_analysis

Surface: Exécute une analyse proportionnelle des séries concernées et
place le résultat de son analyse dans un fichier de résultat..

Input: $input_sheet
        /* C'est la référence aux données qui contiennent les données à
traiter            */

       $base_year
       /* Il s'agit d'un indicateur qui indique si la comparaison des valeurs
       des séries s'effectuera sur la première période ou sur la dern,ière
       période */

Preconds : Exist($input_sheet)
           Numeric(Data($input_sheet))

Outputs: $result_sheet
         /* Table qui contient les résultat de l'analyse appliquée à
         $input_sheet */

Resource_link:  to DM : macro "PropAn1.mac" with $input_sheet
$base_year
```

```
Algorithm : none /* Pas défini */
End define operation.
```

• La description de l'opération sera marquée par la clause "DEFINE OPERATOR" suivie par le nom de l'opération et la fin de la description par la clause " END DEFINE OPERATOR".

• La clause "ALGORITHM": La description de l'algorithme se fait de façon textuelle.

• La clause "RESSOURCE-LINK": Il s'agit de la référence ou l'adresse, dans le systéme, de la macro permettant de réaliser l'opération. Cette référence est à destination de l'évaluateur.

5.6. L'interface avec l'utilisateur.

Elle s'occupe de la gestion de l'échange des informations entre un utilisateur du système et le planificateur. Ses rôles sont:

• Permettre à un utilisateur de soumettre au système un problème complet et précis, compréhensible par le système et en particulier par le planificateur. Il aide et guide l'utilisateur dans la formulation d'un problème en lui demandant des informations supplémentaires si la formulation est incomplète, en lui demandant des précisions si la formulation est ambiguë, ou encore en lui proposant certains choix, par exemple à l'aide de menus, si l'utilisateur hésite sur certaines options.

• Transmettre l'énoncé complet d'un problème au planificateur sous forme compréhensible pour celui-ci.

• Présenter sous forme adéquate à l'utilisateur les informations et les résultats que le planificateur lui passe. La communication se fera par l'intermédiaire d'un périphérique.

5.6.1. La définition d'un problème.

L'utilisateur peut communiquer avec le système par l'intermédiaire de l'interface utilisateur. Celui-ci aide l'utilisateur à formuler un problème précis, compréhensible par le planificateur. Il doit être conçu pour s'adapter aux compétences de l'utilisateur dans le domaine des statistiques officielles ainsi que dans la manipulation du système. Pour cela, on peut imaginer deux types d'interfaces: une interface passive ou une interface active.

Une interface passive est une interface qui laisse les initiatives de la conversation à l'utilisateur. Dans ce type d'interface, l'utilisateur sera une personne ayant déjà une certaine expérience avec la manipulation du système. L'exemple d'interface qui vient directement à l'esprit est une interface dans lequel l'utilisateur entre la formulation de son problème par une phrase dans un langage proche du langage naturel. L'interface interprète la sémantique de la phrase en reconnaissant les mots significatifs et en supprimant les mots qui ne sont pas significatifs tels que les articles, les pronoms,... Ce type d'interface fait l'objet d'études depuis de nombreuses années et soulève un grand nombre d'interrogations, c'est pourquoi nous allons directement l'abandonner.

Un second type d'interface passive serait une interface dans laquelle l'utilisateur entre la formulation du problème suivant une syntaxe stricte, prédéfinie, interprétable par l'interface utilisateur. La formulation suit la syntaxe suivante:

< type problème>, {<paramètres>}

Prenons un exemple en supposant que l'utilisateur désire demander au système d'effectuer une analyse des biais existant dans les séries du produit domestique brut du Royaume-Uni pour les périodes de 1985 à 1990 (Pour bien comprendre cet exemple, il est nécessaire d'avoir lu le second chapitre de ce mémoire). La syntaxe standard pour un problème de ce type sera la suivante:

<nom_analyse>, <Données>, <Pays/Région>,<Période>.

L'utilisateur entrera donc:

Analyse des biais, Séries du PDB, Royaume_Uni, 1985-1990.

Ce type d'interface est évidemment plus aisé à réaliser qu'un analyseur sémantique, mais il est également beaucoup moins puissant et offre moins de liberté à l'utilisateur forcé de suivre une syntaxe très stricte. Ce type d'interface exige que l'utilisateur connaisse la syntaxe pour un type de problème donné. En effet, les paramètres à entrer pour une analyse de séries comme dans l'exemple pris dans cette section ne sera pas les mêmes que pour une comparaison de données. Il y aura donc presque autant de syntaxes différentes que de types de problèmes différents. Cela posera non seulement des problèmes de mémorisation à l'utilisateur, mais également pour la gestion de l'interface utilisateur qui devra repérer un type de problème en fonction de la syntaxe et éventuellement effectuer une correction de la formulation d'un problème. Cela devient vite trop complexe quand le nombre de types de problèmes différents que le système peut résoudre grandit.

5.6.1.1. La communication avec l'interface en mode actif.

Les difficultés posées par un mode d'interface passif étant trop importantes, nous abandonnons les interfaces passives pour nous concentrer sur les interfaces en mode actif. Une interface qui travail en mode actif est une interface qui prend les initiatives, qui aide et guide l'utilisateur pendant toute la durée d'utilisation du système. Dans ce mode, l'utlisateur possède une marge de liberté moins importante que dans un mode passif puisque ses actions sont entièrement sous le

contrôle de l'interface. Par contre, le risque d'erreur est beaucoup moins élevé. Enfin, ce mode évite à un utilisateur de devoir étudier une syntaxe particulière, ce qui est surtout très important lorsque le nombre de types de problèmes différents est important.

L'interface active que nous proposons est la suivante: L'utilisateur est mis en présence, par l'intermédiaire de menus, avec les types de problèmes que le système peut résoudre. A partir de là, il peut choisir le type de problème qu'il désire formuler. L'interface demande ensuite à l'utilisateur d'autres renseignements afin de formuler complètement à partir du type de problème choisi, le problème choisi. L'utilisateur entrera chaque renseignement nécessaire soit en choisissant parmi un menu, soit en le proposant lui-même.

Pour bien comprendre comment l'interface définit un problème en collaboration avec l'utilisateur, on reprendra comme exemple le cas d'un utilisateur qui désire effectuer une analyse des biais pour les séries du produit domestique brut pour le Royaume-Uni pour la période allant de 1985 à 1990. L'interface propose à l'utilisateur un ensemble de problèmes possibles via un système de menus. Parmi ceux-ci figure l'ANALYSE DES BIAIS que l'utilisateur choisira. L'interface lui propose ensuite un menu reprenant les données pouvant être soumises à une analyse proportionnelle, parmi celles là, l'utilisateur choisira les "séries du PDB", ensuite, on lui proposera un ensemble de pays pour lesquels les données peuvent se rapporter, et enfin, on lui demandera les périodes pour lesquelles l'analyse proportionnelle devra être réalisée. On permettra à l'utilisateur de nommer les périodes désirées ou de donner un intervalle de périodes.

Remarques:

- Ce type d'interface est relativement facile à gérer, à l'exception peut-être de la gestion des différentes options à demander à l'utilisateur d'après le type de problème choisi. Pour la gestion de ces différents types de problèmes, l'interface est composée d'une mémoire contenant chaque type de problème avec les paramètres à entrer pour ramener ce type de problème à un problème particulier.

- Si le nombre de problèmes proposés par le système est trop

important, il est possible de grouper les problèmes par types génériques, et ceci dans le but de ne pas trop allourdir les menus par des listes trop importantes. On peut, par exemple, regrouper toutes les analyses de séries entres elles.

- Il se peut que certaines données entrées comme paramètres ne soient pas dans la base des données opérationnelles, mais cela sera découvert par le système au début de la résolution de problème. Pour éviter cela, il faudrait que l'interface ait une connaissance précise des données contenues dans la base des données opérationnelles. La résolution à ce problème sera évoquée dans le chapitre V.

5.6.2. Communication du problème au planificateur.

Lorsque le problème posé par l'utilisateur a bien été formulé, l'interface utilisateur communique la requête au planificateur. Pour ce faire, il est nécessaire que les deux modules aient une syntaxe commune leur permettant de communiquer l'un avec l'autre. Cette syntaxe sera :

<type_problème>, {<paramètres>} où type_problème sera le nom du type de problème à résoudre et la liste de paramètres sera utilisée pour définir le problème avec précision. Pour faciliter la tâche du planificateur, tous les paramètres nécessaires à la recherche d'une même donnée dans la base des données opérationnelles seront regroupés dans un même ensemble de paramètres. De plus, pour chaque ensemble de paramètres (éventuellement constitué d'un seul paramètre), un indicateur signalera si cet ensemble sert à retrouver des données ou s'il s'agit juste de paramètres nécessaires à la résolution du plan.

5.6.3. Communication des résultats à l'utilisateur.

Le planificateur communique les résultats de l'évaluation d'un problème, que l'évaluation ait réussi ou non, à l'interface utilisateur. La dernière fonction de l'interface sera donc de présenter ces informations sous forme adéquate à l'utilisateur.

5.7. Le planificateur.

5.7.1. Définition et rôles.

Le planificateur est un "moteur" de base dont le rôle est de supporter la résolution de problèmes statistiques par l'ordonnancement et l'exécution des tâches nécessaires. Ses rôles sont:

- Recevoir des requêtes relatives à la demande de résolution d'un problème de la part de l'interface utilisateur.

- Valider les données nécessaires à la résolution de ce problème.

- Rechercher dans la librairie des plans un plan qui lui permette de résoudre le problème posé.

- Réaliser les étapes du plan trouvé, appliquées au problème précis .

- Communiquer à l'interface utilisateur les résultats de la réalisation.

5.7.2. Structure d'un planificateur.

La structure d'un planificateur est composée de trois modules principaux: le moteur, la mémoire de travail et le journal de bord.

1. Un moteur : le moteur est en quelque sorte le cerveau du planificateur, c'est lui qui s'occupe de lancer l'exécution d'un plan, qui va rechercher les données nécessaires à la résolution d'un problème, qui va rechercher un plan spécifique dans la base des plans...

2. Une mémoire de travail: la mémoire de travail est un espace mémoire résident, qui est utilisé pour gérer les variables intermédiaires, les valeurs de ces variables et les liens entre celles-ci lors de l'exécution. Il s'agit en quelque sorte de la mémoire du planificateur. Il peut y stocker:

- les requêtes de l'utilisateur.

- les variables de travail avec leur valeur.

- les résultats intermédiaires de certaines étapes ou sous-étapes d'un plan..

- la description logique d'un plan.

- des indications reprenant les étapes ou sous-étapes du plan qui ont déjà été réalisées.

3. Un journal de bord: Le journal de bord est une structure de mémoire résidente permettant de stocker les résultats d'une exécution ou d'une évaluation d'un plan par le moteur. Le moteur peut y stocker n'importe quel résultat intermédiaire, quel que soit son format, y envoyer le contenu vers l'interface utilisateur en envoyant chaque élément dans l'ordre chronologique de son stockage, et vider le journal de bord de son contenu.

5.7.3. Validation des données nécessaires à la réalisation.

Lorsque le planificateur reçoit la formulation d'un problème par l'interface utilisateur, la première tâche du planificateur est de rechercher les données nécessaires à la bonne résolution du problème posé. Le planificateur recherche ces données dans la base des données opérationnelles; si celles-ci ne s'y trouvent pas, il

signalera leur absence à l'interface utilisateur et terminera son exécution. Si les données sont présentes, il enregistrera leur référence dans sa mémoire de travail. Dans la section 5.3, relative à la "base des données opérationnelles", nous n'avons pas détaillé la façon dont les données étaient organisées. Ici, nous n'allons donc pas développer la manière dont cette fonction sera implémentée, cependant comme dans la section 5.3 nous avons proposé une organisation par index sur les types et les noms des données, nous pouvons facilement imaginer que le planificateur recherchera les données d'après l'index.

Exemple: Pour la validation des données identifiées par :

Données = (Séries PDB, Royaume_Uni, 1985..1990), la recherche s'effectuera d'abord sur le nom de la donnée, ici les séries du PDB; ensuite, le second index portera sur la région ou le pays concerné, ici le Royaume_Uni; enfin, on recherchera les données sur l'intervalle de temps donné.

5.7.4. Recherche d'un plan exécutable.

La seconde tâche du planificateur sera la recherche d'un plan qui lui permette de résoudre le problème posé. Les plans sont enregistrés dans la base des plans et le rôle du planificateur est de rechercher parmi ces plans celui qui convient le mieux.

Pour éviter une recherche séquentielle sur l'ensemble des plans, le planificateur accède à un index classant les types de plans disponibles, et pour chaque type de plans les références des plans de ce type dans la librairie.Si un plan adéquat est trouvé, celui-ci est chargé dans la mémoire de travail du planificateur, sinon le planificateur signale à l'interface utilisateur l'impossibilité de résoudre le problème et arrête son exécution.

A ce niveau se situe une des limitations les plus importantes de notre système. En effet, il est très difficile de créer un planificateur, qui puisse rechercher

dans la base des plans, à partir de la formulation d'un problème, le "meilleur" plan qui soit capable de résoudre ce problème. Il est peut-être possible de créer un tel planificateur, mais uniquement pour un petit nombre de problèmes différents. Sur un grand nombre de problèmes différents, cela devient de plus en plus complexe.

Dans notre système, à chaque type de problème correspond un plan dans la base des plans. Il s'agit donc d'une relation un à un entre un plan et un type de problème. La recherche d'un "meilleur" plan se résume à la recherche dans la base des plans du plan qui corresponde au type de problème à résoudre et pas véritablement au choix d'un "meilleur" plan parmi le stock de plans. Un plan sera trouvé par une correspondance entre le plan et le nom du type de problème à résoudre.

5.7.5. Réalisation des étapes d'un plan.

Quand les données sont validées et qu'un plan de résolution pour le type de problème a été trouvé, l'étape suivante est la réalisation des étapes du plan. La réalisation consiste à exécuter chaque étape du plan avec les données spécifiques au problème. Le système permet la réalisation des étapes d'un plan dans trois modes différents: le mode "perform", le mode "explain" et le mode "perform/explain". L'utlisateur est mis en présence d'un menu lui demandant de choisir un des trois modes suivants:

- Le mode explain: dans le mode explain, le système fournit à l'utilisateur une explication du développement des différentes étapes du plan menant à la solution du problème, mais sans exécuter ces étapes. Ce mode permet à l'utilisateur de connaître la façon dont un expert résoud un certain type de problème, même s'il n'a pas des données précises à appliquer au problème posé. Le rôle du système dans ce cas-ci est un peu un rôle pédagogique.

- Le mode perform: dans le mode perform, le système fournit à l'utilisateur les résultats de l'exécution des différentes étapes du planning menant à la solution du problème. Ce mode permet de faire résoudre de manière automatique par le système un problème

précis avec des données bien définies.

• Le mode perform/explain: dans ce mode, le système travaille en même temps en mode perform et en mode explain. Ce mode permet à un utilisateur d'obtenir de manière automatique les résultats à un problème précis sur des données définies, tout en obtenant des informations sur les différentes étapes suivies par le système.

Quand la description logique d'un plan est chargée dans la mémoire de travail du planificateur, celui-ci peut commencer à évaluer ce plan dans un des trois modes définis plus haut. Cette évaluation s'effectue grâce à trois fonctions principales: le développement d'un plan, le développement d'une procédure et l'exécution d'une opération.

5.7.5.1. Développement d'un plan.

Cette fonction évalue un plan en le développant dans son contexte, c'est-à-dire en le particularisant au problème précis avec les données, et en retournant les résultats de l'évaluation enregistrés dans le carnet de bord. Les différentes étapes de cette fonction sont les suivantes:

• Si l'exécution s'effectue en mode explain ou perform/explain, écrire dans le journal de bord les renseignements sur le plan évalué, par exemple le nom du plan, la description des objectifs du plan, les variables utilisées dans le plan... Ce sont tous les renseignements sous forme textuelle à destination de l'utilisateur.

• Initialiser les variables du plan dans la mémoire de travail. Cela se fait en assignant à certaines variables les valeurs des données enregistrées dans la mémoire de travail lors de la phase de validation des données, ou encore en assignant à certaines variables les valeurs des paramètres fournis par l'interface utilisateur lors de la formulation du problème.

• Exécuter le corps du plan: Pour chaque élément du corps du plan, le planificateur construit un environnement d'évaluation, c'est-à-dire que l'on développe cette étape avec les variables correspondant au problème posé. Un élément du corps est soit un appel à un autre plan, soit un appel à une procédure, soit un appel conditionnel à un plan ou à une procédure.

Si c'est un appel à un autre plan, rechercher le plan dans la base des plans, le charger dans la mémoire de travail puis appeler récursivement la fonction "Développement d'un plan" appliquée au plan nommé.

Si c'est un appel à une procédure, lancer la fonction "Développement d'une procédure" appliquée à la procédure concernée avec les paramètres nécessaires à l'exécution de celle-ci.

Enfin, s'il s'agit d'un appel conditionnel, on évalue en premier lieu la condition et si celle-ci est vérifiée, on appelle soit la fonction "Développement d'un plan", soit la fonction "Développement d'une procédure". Une condition peut porter sur les variables de travail, sur les résultats intermédiaires de l'exécution, ou encore sur les étapes du plan qui ont déjà été réalisées. Pour cela, le moteur dispose d'un ensemble de prédicats qu'il peut appliquer sur les données concernées par la condition.

• La dernière étape du développement d'un plan consiste à stocker, dans la mémoire de travail, que la réalisation des étapes de ce plan a été réalisée. Cette étape peut être utile pour la réalisation de certaines étapes conditionnelles dont la condition est la réalisation préalable d'une autre étape.

5.7.5.2. Développement d'une procédure.

Cette fonction évalue une procédure en la développant dans son contexte et en retournant les résultats de l'évaluation dans le journal de bord. Les différentes étapes de cette fonction sont les suivantes:

- Rechercher la procédure dans la librairie des procédures. Si la procédure est contenue dans la librairie des procédures, la charger dans la mémoire de travail, sinon arrêter l'exécution en signalant l'erreur dans le journal de bord.

- Si l'exécution s'effectue en mode Explain ou Perform/Explain, écrire dans le journal de bord tous les renseignements sous forme textuelle sur la procédure, à destination de l'utilisateur.

- Initialiser les variables de la procédure dans la mémoire de travail.

- Evaluer les préconditions de la procédure. Pour chaque précondition, le moteur s'assure qu'elle est vérifiée. Si une des préconditions n'est pas vérifiée, le moteur le signale à l'interface utilisateur et arrête son exécution. Les préconditions peuvent porter sur des paramètres de la procédure ou sur des variables intermédiaires stockées dans la mémoire de travail. Pour vérifier les préconditions, le moteur dispose d'un ensemble de prédicats qu'il peut appliquer aux données concernées.

- Exécuter le corps de la procédure. Pour chaque élément du corps de la procédure, le planificateur construit un environnement d'évaluation. Un élément du corps est un appel, conditionnel ou non, à une opération.

S'il s'agit d'un appel non conditionnel, on lance la fonction "Exécuter une opération" avec les paramètres nécessaires à

l'exécution de cette opération.

S'il s'agit d'un appel conditionnel, on évalue préalablement la condition avant de lancer l'appel à une opération si la condition est vérifiée.

• Initialiser les variables contenant les résultats attendus dans la mémoire de travail. Ces variables sont des variables intermédiaires définies lors de la réalisation des étapes de cette procédure.

• Vérifier que les prédicats constituant les postconditions sont vérifiés. Si certains prédicats ne le sont pas, on l'indique dans le journal de bord et on arrête l'exécution. Les postconditions peuvent porter sur les résultats de la procédure ou sur des variables intermédiaires stockées dans la mémoire de travail.

• La dernière étape du développement d'une procédure consiste à stocker, dans la mémoire de travail, que la réalisation des étapes de cette procédure a été réalisée. Cette étape peut être utile pour la réalisation de certaines étapes conditionnelles dont la condition est la réalisation préalable d'une autre étape.

5.7.5.3. Exécution d'une opération.

Cette fonction évalue et développe une opération dans le contexte d'un environnement d'agrégation de variables donné. La fonction s'exécute en plusieurs étapes:

• Rechercher la description logique de l'opération dans la librairie des opérations et charger cette description dans la mémoire de travail. Si la description de cette opération ne s'y trouve pas, le signaler dans la journal de bord et arrêter l'exécution.

• Initialiser les variables de l'opération et enregistrer les paires (variable, valeur) dans la mémoire de travail.

• Vérifier les préconditions de l'opération. Cette étape est réalisée de la même façon que dans la fonction "Développement d'une procédure".

• Si l'on travaille en mode explain ou en mode explain/perform, écrire dans le journal de bord les renseignements concernant l'opération développée, par exemple le nom de l'opération, les variables de l'opération, une description des rôles de l'opération, la description de l'algorithme de l'opération, en y intégrant les valeurs des variables nécessaires...

• Si l'on travaille en mode perform ou en mode explain/perform, lancer l'exécution de l'opération et enregistrer les résultats dans le journal de bord. Le lancement de l'exécution s'effectue en appelant l'évaluateur chargé d'exécuter les "macros" décrites par les opérations. L'évaluateur connaît la référence de la macro à exécuter par la référence contenue dans la description de l'opération et que le planificateur lui communique.

• Initialiser les variables contenant les résultats attendus dans la mémoire de travail.

• Vérifier les postconditions de l'opération. Chaque postcondition est testée, et si l'une d'elles n'est pas vérifiée, on le signale dans le journal de bord et on arrête l'exécution.

• La dernière étape de l'exécution d'une opération consiste à stocker, dans la mémoire de travail, que l'exécution de cette opération a été réalisée. Cette étape peut être utile pour la réalisation de certaines étapes conditionnelles dont la condition est la réalisation préalable d'une autre étape.

5.7.6. Communication des résultats à l'interface.

La réalisation des étapes du plan étant achevée, la dernière opération du planificateur est la communication des résultats à l'interface utilisateur. Les résultats du développement d'un plan sont enregistrés dans le journal de bord, il suffit de communiquer chaque élément de ce journal de bord dans l'ordre chronologique de leur écriture à l'interface utilisateur.

5.8. Présentation d'un prototype du système.

Pour illustrer les notions développées dans la section 5.7, nous allons présenter dans cette section un prototype simplifié du système. Ce système a été programmé dans un langage d'intelligence artificielle: le LISP. Il s'agit d'une utilisation peu courante d'un langage d'intelligence artificielle. Généralement, ils sont utilisés pour amener un problème vers le but recherché, mais ici, le LISP est utilisé pour gérer le planificateur, la recherche et l'exécution de plans squelettes.

Ce planificateur permet d'évaluer un plan et les opérations qui s'y rapportent en mode Explain, c'est-à-dire que l'on fournit à l'utilisateur une explication de chaque étape du plan permettant de résoudre un problème posé. Dans la section 5.4., nous avons découpé un plan en trois niveaux hiérarchiques, les plans, les procédures et les opérations. Ce prototype ne supporte que deux niveaux de hiérarchie: les plans et les opérations. Il s'agit d'un choix essentiellement pratique, mais le principe est le même que pour la découpe en trois niveaux. Un plan est découpé en étapes, chacune étant un appel à un autre plan ou à une opération.

5.8.1. La structure du système.

Le système est composé de quatre composants principaux.

5.8.1.1. La librairie des plans.

Dans le mini-planificateur, la librairie des plans stocke des objets "plans" définis pour la solution de problèmes dans le domaine, indexé sur les noms des plans.

Le planificateur peut appliquer 3 fonctions sur la librairie des plans:

- FETCH_PLAN: Etant donné le nom d'un plan, cette fonction retrouvera le plan dans la librairie, ou retournera la valeur FALSE si le plan n'existe pas.

- DEFPLAN: Prend la spécification d'un plan et la stocke dans la table représentant la librairie des plans si la spécification est valide.

- DEFPLANS: Exécute la fonction defplan sur une liste de spécifications de plans et les stocke dans la librairie de plans.

Les attributs des plans sont:

- Name: Nom symbolique d'un plan.

- Variables: Les variables formant l'environnement global du plan lorsqu'il est exécuté.

- Préconds: Enoncé logique des préconditions dépendant du

contexte qui doivent être vérifiées pour l'exécution du plan.

• Body: Séquence d'opérations pour décrire le processus, présenté comme une liste de noms d'opérations avec l'agrégat de variables approprié. La syntaxe d'une étape d'un plan est la suivante:

- S'il s'agit d'un appel à un autre plan, le premier mot est "Evaluate", suivi du nom du plan à évaluer.

- S'il s'agit d'un appel à une opération, le premier mot est "Perform", suivi du nom de l'opération à exécuter.

5.8.1.2. La librairie des opérations.

Dans le mini-planificateur, la librairie des opérations est une table indexée qui permet de stocker la description logique des opérations définies. La table est indexée sur le nom des opérations. Le mini-planificateur permet 3 types d'opérations sur cette table:

• FETCH_OP : recevant le nom d'une opération cette fonction tentera de retrouver la description de l'opération, ou retournera FALSE si la description de l'opération n'est pas stockée dans la table.

• DEFOP: permet de définir une structure d'opération et la stocke dans la table si la spécification est valide.

• DEFOPS: exécute la fonction defop sur une liste de spécifications d'opérations, et les stocke dans la table.

Les attributs d'une opération sont:

- Name: Nom symbolique de l'opération.

- Synopsis: la description sous forme simple de l'opération

- Inputs: Une liste de spécifications de paramètres pour l'opération.

- Préconds: Enoncé logique des préconditions dépendant du contexte qui doivent être vérifiées pour pouvoir lancer l'application de l'opération.

- Results: Une liste de spécifications pour les résultats de l'évaluation de l'opération.

- Surface: Présentation sous forme textuelle des rôles de l'opération.

- Link: Référence pour l'exécution d'une fonction externe (ex: par DDE dans le contexte d'un programme sous Windows 3.1.)

- Postconds: Enoncé logique des postconditions à être vérifiées par les résultats de l'évaluation de l'opération.

Le planificateur peut appliquer 4 types d'applications sur les opérations.

- CHECK-VAR-IN-PATTERN (operation, pattern-list)

  Vérifie que chaque élément de la liste de variables fournie comme paramètre a une entrée correspondante dans les entrées des opérations, en d'autres termes vérifie qu'il a bien une variable qui lui corresponde dans la liste des entrées de la fonction.

- EXPAND_SURFACE_FORM (operation,pattern-list)

Exécute une substitution de variables dans la description textuelle de l'opération.

- EXPAND_LINK_FORM (operation, pattern_list)

Exécute une substitution de variables dans la spécification de liaison d'une opération; c'est-à-dire dans la spécification opérationnelle de l'opération.

- PRINT_OPERATION

Imprime le tout d'un objet opération.

5.8.1.3. La mémoire de travail (Working Memory).

Dans le mini-planificateur, la mémoire est une table indexée comprenant comme éléments des paires (nom_variables, valeur_de_variable). Les noms des variables sont utilisés pour indexer la table. Le planificateur peut exécuter 6 fonctions sur cette mémoire de travail:

- STORE_VALUE (a_key, a_value): Stocke une valeur pour une variable dans la table où la clé est le nom de la variable.

- FETCH_VALUE (a_key): recherche la valeur associée avec le nom de variable "a_key". Renvoie FALSE si la variable n'existe pas dans la mémoire de travail.

- SPLICE_VALUE (list_of_variable_names): Pour chaque nom de variable dans le liste, retrouve la valeur correspondante dans la mémoire de travail et retourne une paire correspondant à (nom_variable, valeur_variable).

• BIND_VALUE_IN_CALL (paire): Soit une paire consistant en (nom_variable, valeur_variable), substitue la partie valeur dans le contenu correspondant à "nom_variable" dans la mémoire de travail.

• BIND_ALL_VALUES_IN_CALL (liste de paramètres): Soit une liste associative où chaque paire est de la forme (nom_variable, nom_paramètre), remplace chaque nom_paramètre par la valeur courante de le variable indexée sous nom_variable.

• CLEAR_MEMORY: Nettoie la mémoire de travail de tout son contenu courant.

5.8.1.4. Le journal de bord (log)

Structure de mémoire résidente accessible de manière séquentielle. Le planificateur peut exécuter 3 fonctions sur le journal de bord:

• LOGIT (anythings): Stocke n'importe quoi dans la structure de log.

• NORMALISE_LOG: Exécute la structure log pour la sortie, c'est-à-dire qu'elle retourne la liste de manière à accéder aux éléments dans l'ordre chronologique de leur entrée.

• CLEAR_LOG: Remet à blanc toutes les entrées dans le journal de bord.

5.8.2.Le Mini_planificateur.

Le mini_planificateur a trois fonctions principales lui permettant

d'évaluer des plans pour un utilisateur. Pour demander au planificateur la résolution d'une requête, on lance l'exécution de la fonction START avec comme paramètres l'identifiant du plan et les paramètres d'exécution du plan.

La fonction Start (Les paramètres du plan).

Cette fonction est le processus de contrôle pour les opérations de planification. Dans sa forme la plus simple, il met la structure du journal de bord à blanc et appelle la fonction d'évaluation du plan. Les résultats sont ensuite présentés à l'utilisateur.

La fonction Evaluate_plan (nom_plan, paramètres)

Cette fonction évalue un plan en le développant dans son contexte, c'est-à-dire avec les paramètres spécifiques au problème à résoudre, et en retournant les résultats de l'évaluation qui sont stockés dans le journal de bord. La fonction s'exécute en:

- retrouvant la spécification du plan dans la librairie des plans.
- instantiant les variables de plan dans la mémoire de travail.
- évaluant les pré-conditions du plan.
- éxécutant le corps du plan, en construisant séquentiellement un environnement d'évaluation pour chaque opération nommée dans le corps du plan, et en évaluant chaque opération en utilisant la fonction "EXECUTER_OPERATION". Il faut noter que l'opération peut être un appel pour évaluer un autre plan, dans ce cas, la fonction "EVALUER_PLAN" est appelée récursivement.
- normaliser et retourner les résultats stockés dans le log.

La fonction Process_operation.

Cette fonction évalue et étend une opération dans son contexte. La fonction s'exécute en:

- retrouvant la description logique de l'opération dans la librairie d'opérations.

- développant l'agrégat de variables avec la valeur courante dans la mémoire de travail.

- développant la forme en surface de l'opération, pour refléter l'environnement à l'utilisateur.

- développant le forme opérationnelle de l'opération.

- retournant le corps entier des opérations développées sous forme de liste de formes en surface et de formes opérationnelles.

On peut trouver dans les annexes le listing du mini_planificateur ainsi qu'un exemple d'opération, de plan, et d'exécution de fonction.

CHAPITRE 6

"Etude comparative des systèmes de planning"

6.1. Introduction.

Le système, tel que nous l'avons présenté au chapitre précédent, est un prototype. En tant que tel, il doit donc subir un ensemble de critiques débouchant sur certaines améliorations possibles. Une solution pour arriver à ces améliorations serait la possibilité d'incorporer notre système à d'autres plus "consistants" et ayant fait leurs preuves.

Parmi ceux-ci, on retrouve bien entendu EISI (Expert Interface for Statistical Information) puisque celui-ci a servi de base à notre étude. Nous essayerons également d'approcher superficiellement (car son domaine d'application étant tout autre, nous préferons ne pas trop le détailler) un autre type de système de planning intitulé "MolGen" et de voir quels sont les points communs et les points

divergeants entre ce système et le nôtre.

La première partie sera consacrée à une critique de notre système ainsi que la recherche des limites de celui-ci. Dans la seconde partie, nous présenterons brièvement le système MolGen afin de pouvoir le comparer avec le nôtre par la suite. La troisième partie de ce chapitre sera consacrée à l'étude d'une intégration possible de notre système avec le système EISI et enfin avec le système MolGen. L'intention de coupler notre système avec d'autres systèmes a pour but d'en élargir la portée.

6.2. Critiques du système.

Le système développé au chapitre 5 est un système de résolution automatique de problèmes dans le domaine des statistiques officielles. Pour résoudre les problèmes, il dispose de plans génériques, chacun lui permettant de résoudre un type de problème précis. Les problèmes qu'il peut résoudre appartiennent à des types de problèmes pour lesquels on lui a donné un plan de résolution. Ce type de système convient bien au domaine des statistiques officielles où, comme nous l'avons signalé dans la section 4.2.1, il existe des catégories de problèmes que l'on retrouve presque toujours et qui donc réduisent le nombre de plans différents à créer.

L'utilisation de ce système peut être étendue à d'autres domaines que le domaine des statistiques officielles, principalement à des domaines où, comme pour les statistiques officielles, il existe des catégories de problèmes que l'on rencontre presque toujours. Nous pensons par exemple à la planification des tâches d'un robot dont les tâches sont routinières et bien définies.

Par contre, lorsque l'on désire faire résoudre par le système un nouveau problème pour lequel il ne dispose pas encore de plans génériques, celui-ci sera incapable de le résoudre. Le système ne convient donc pas à des domaines où les problèmes sont variés ou nouveaux; il n'est pas capable d'effectuer un auto-apprentissage d'informations à partir d'autres informations, ni de s'engager vers

l'inconnu lors de la résolution d'un problème en espérant arriver au but recherché.

Une seconde limitation du système se situe lors de la recherche d'un plan, développée à la section 5.7.4. . En effet, un des rôles du planificateur est de rechercher, parmi les plans stockés dans la base des plans, le plan qui soit le "meilleur" pour la résolution d'un problème donné. En fait, dans notre système, chaque problème posé est classé dans un type de problème précis et pour chaque type de problème il existe un plan générique. On ne peut donc pas vraiment considérer que le planificateur recherche le "meilleur" plan puisque il va chercher l'unique plan qui correspond au type de problème qu'il doit résoudre.

Une dernière limitation est la connaissance du contexte et des données traitées. Le système résoud des problèmes sur des données réelles, mais l'utilisateur ne dispose que d'une connaissance limitée des données et du contexte d'exécution. Il faudrait pouvoir intégrer de la méta-information. Celle-ci serait très utile pour aider l'utilisateur à analyser les résultats, même si le système peut fournir une description des différentes étapes menant à la réalisation du problème, il faudrait intégrer des informations supplémentaires sur les données traitées.

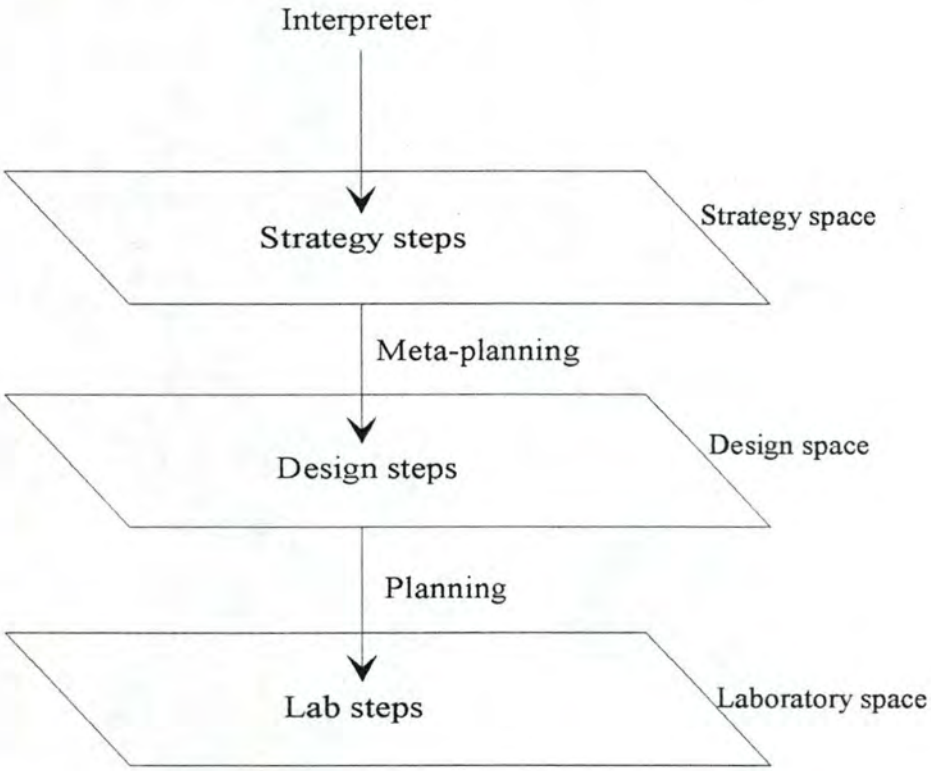
6.3. "MolGen"[1]

Ce système est un "knowledge-based program" conçu pour assister les chercheurs dans la génétique moléculaire : il essaie de modéliser les expériences effectuées dans un laboratoire de génétique moléculaire. L'étude de ce système nous a semblé intéressante en ce sens qu'il possède des capacités d'apprentissage de

[1] Pour plus de détails sur ce système, nous conseillons au lecteur de consulter [COHEN] et [STEFIK]

nouvelles informations et de résolution de nouveaux problèmes.

6.3.1. Architecture du système.



6.3.1.1. Strategy space.

Cet espace contient la connaissance à propos de la stratégie. Il dispose de deux approches pour la résolution d'un problème : l'approche "heuristique" ou l'approche du "moindre engagement".

La technique du "moindre engagement" consiste à reporter une décision lorsqu'elle est subordonnée à trop peu de contraintes. Par exemple, si un problème est subdivisé en plusieurs sous-problèmes et que pour ceux-ci, on dispose de trop peu de réponses satisfaisantes, alors on décidera de postposer la résolution de ce problème. Par contre, l'approche "heuristique", quant à elle, utilise des

raisonnements plausibles dans le but d'apporter une solution à un problème pour lequel toutes les informations requises ne sont pas disponibles.

6.3.1.2. Design space.

Cet espace est le premier espace de contrôle de MolGen. Il contient les opérateurs qui créent et arrangent les étapes du "Lab Space". Cet espace modélise les actions d'un "concepteur d'expérience". A ce niveau MolGen prend des décisions concernant la façon de développer un plan.

L'idée principale du "Design Space" est qu'un planning peut être vu comme un ensemble d'opérations sur des contraintes. Trois de ces opérations sont importantes pour ce système : la formulation de contraintes, la propagation de contraintes et la satisfaction pour une contrainte.

- La formulation est la création dynamique de contraintes qui fixent des limites sur d'éventuelles solutions acceptables.
- La propagation permet la formulation de nouvelles contraintes à partir d'autres.
- La satisfaction raffine des entités abstraites pour ne plus en faire qu'une plus spécifique.

En plus de ces opérations, cet espace possède également des opérateurs de "design" qui permettent réellement la gestion de plans. On y retrouve trois grands types d'opérateurs :

- Les opérateurs de comparaison qui comparent des buts et calculent les différences.
- Les opérateurs d'extension qui étendent un plan "vers l'avant" ou "vers l'arrière" (en d'autres mots : extension ou rétrécissement).
- Les opérateurs de spécialisation qui font d'un plan abstrait un plan plus spécifique.

6.3.1.3. Lab Space (ou Domain Space).

Cet espace définit un ensemble d'expériences possibles en décrivant les objets et opérateurs que l'on "retrouve" dans un laboratoire pour un certain domaine d'application. Les objets représentent les objets physiques qui peuvent être manipulés dans un laboratoire de génétique c'est-à-dire les antibiotiques, les enzymes, les gênes, etc... Les opérateurs représentent les processus physiques qui peuvent être réalisés dans un tel laboratoire : combiner des objets, changer les propriétés d'un objet, éclater un objet en ses composants, etc...

Attention, le "Lab space" ne contient pas la connaissance qui est nécessaire pour savoir comment réaliser une expérience avec tous ces composants décrits ci-dessus. Cette connaissance est stockée dans les autres "espaces". En d'autres mots, cet espace , qui se situe au niveau le plus bas, ne contient que les outils de base pour toute expérimentation à effectuer.

6.3.2. Description fonctionnelle.

La technique de planning dans MolGen est dirigée par une technique "means-ends" qui consiste en fait, à rechercher des opérateurs qui, lors de chaque étape du planning, réduisent la différence entre l'état courant et l'état désiré (le but).

On retrouve la technique du planning hiérarchique qui consiste à raffiner son but à chaque niveau d'abstraction. On choisit son but ainsi que sa stratégie d'analyse (voir "Strategy Space"), ensuite, on découpe ce but en petites étapes (dessin du plan = "Design Space") grâce aux différents opérateurs disponibles à ce niveau et à la connaissance du domaine d'application (ensemble de contraintes ou de "règles d'inférence" pour utiliser le vocabulaire de l'intelligence artificielle) et finalement on génère des étapes qui travaillent sur les entités que l'on retrouve au niveau "Lab Space". On retrouve donc bien ici la description à l'état pur des plannings hiérarchiques.

Remarque : Le but de ce mémoire n'étant pas l'étude de MolGen, nous

n'irons pas plus en profondeur dans sa description, notre but étant de comparer le schéma de base de ce système avec le système développé tout au long de cet ouvrage.

6.4. Comparaison entre "EISI" et "MolGen".

Ces deux systèmes, bien que différents à propos de leur domaine d'application (les statistiques et la génétique) et leur mécanisme de résolution (l'un ne résout que les problèmes contextuels de son domaine d'application tandis que l'autre utilise la connaissance contextuelle relative à son domaine d'application pour résoudre des problèmes concrets), utilisent, tous deux, la méta-information.

Cette vue assez récente de la résolution de problèmes peut avoir différents avantages : d'une part, on peut appliquer un mécanisme d'inférence procurant une certaine intelligence au système (sachant les règles relatives au domaine d'application étudié, le système peut en déduire certaines conclusions) et d'autre part, cela permet d'avoir un certain recul sur le domaine d'application c'est-à-dire que l'on ne se borne pas à résoudre un type de problème mais bien à essayer de l'expliquer (on procure à l'utilisateur un outil d'aide à la décision qui peut lui permettre d'orienter ses choix à propos de la résolution d'un cas spécifique).

D'un point de vue mécanisme interne, on retrouve deux tendances : d'une part le mécanisme hiérarchique strict de MolGen et d'autre part le mécanisme semi-hiérarchisé et semi-éclaté de EISI. En d'autres mots, dans MolGen, la résolution se fait de façon interne dans les différents niveaux du système de planning. Par contre, dans EISI, la résolution d'un problème respecte une certaine hiérarchie mais on ne retrouve pas cette découpe stricte en niveaux dans la description de l'exécution. On retrouve un composant "animateur" (le Domain Assistant") et d'autres composants auxiliaires qui interagissent entre eux afin d'aboutir à un résultat. De plus, la découpe en niveaux dans EISI concerne particulièrement la méta-connaissance (Domain Encyclopedia, Meta Information Database et Usage Scenario Library) tandis que

Molgen applique cette découpe en niveaux de façon générale sur le système.

6.5. Les améliorations possible à apporter.

6.5.1 Intégration avec "EISI".

Si l'on compare ces deux systèmes avec le nôtre développé dans ce mémoire, on constate immédiatement un certain parallélisme entre EISI et notre système de planning. En effet, comme nous l'avons répété souvent dans ce mémoire, nous nous sommes fréquemment reportés à l'étude d'EISI afin de voir quels pouvaient être les apports de ce système pour la généralisation de notre problème et l'élaboration d'un système prototype de planification.

Dans ce point, nous allons donc essayer de voir dans quelle direction nous pourrions faire évoluer notre système afin de l'améliorer et de le généraliser davantage. La majeure partie de cette étude sera évidemment consacrée à une possible intégration dans EISI et ceci pour les raisons citées dans le paragraphe précédent. Une intégration de notre système avec le système EISI peut-être bénéfique à deux points de vues importants: la possibilité d'utiliser la "Case library" (voir la section 3.2.3.9) et la possibilité d'utiliser la méta-information du système EISI.

6.5.1.1. L'utilisation de la "Case library"[2].

La "Case library" dans EISI est utilisée par le "reasoner" pour éviter de

[2] Cette intégration vient d'être réalisée par Monsieur de Vaney, et un prototype de démonstration sera très bientôt disponible.

résoudre un même problème plusieurs fois. Deux problèmes étant exactement les mêmes lorsqu'ils sont exécutés avec les mêmes paramètres.

L'intégration de notre système avec EISI permettra au moteur du planificateur de rechercher dans la librairie si le problème qu'il doit résoudre a déjà été résolu dans le passé avec les mêmes paramètres. Dans l'affirmative, les résultats de la précédente exécution, qui sont stockés dans la "Case library", sont fournis à l'utilisateur. Dans la négative, le problème est résolu puis, les résultats, en plus d'être fournis à l'utilisateur, sont enregistrés dans la "Case library".

La "Case library" devra être étendue pour pouvoir contenir, en plus des résultats de la résolution des problèmes utilisant de la méta-information, les résultats sur ces mêmes problèmes utilisés concrètement (données réelles).

Dans notre système, le journal de bord, dans lequel on inscrit les résultats intermédiaires et finaux de l'évaluation d'un problème, sera "coulé" dans la "Case library" et son contenu ne sera plus effacé après chaque exécution mais sera enregistré de manière permanente dans la "Case library" (la structure "Opres", qui contient les résultats générés lors d'une évaluation d'un problème).

Cette intégration permet de gagner un temps d'exécution important lors de la résolution d'un problème déjà résolu auparavant.

6.5.1.2. Utilisation de la méta-information.

Tout d'abord, la première question, qui nous vient à l'esprit lorsque l'on parle d'amélioration de notre système par rapport aux deux cas étudiés auparavant dans ce chapitre, peut s'exprimer de la sorte : "*est-il possible d'intégrer la méta-connaissance dans notre système de planning ?*".

En fait, la solution la plus simple, la plus rapide et la plus performante consite à contourner cette question. En effet, afin d'éviter une répétition de travail inutile (pourquoi élaborer un système de méta-connaissance spécifique à notre

architecture alors qu'il serait peut-être possible d'intégrer cette même architecture dans un système existant), posons la question de la façon suivante : "*est-il possible d'intégrer notre système de planning dans un système dédié à la méta-connaissance?*".

Cette intégration de notre système dans EISI nécessiterait quelques étapes supplémentaires dans l'élaboration de notre architecture. En effet, le concept même de méta-données définit à lui seul, les modifications nécessaires à apporter. En d'autres mots, l'intégration dans EISI nécessite une étude contextuelle de nos données concrètes afin de mettre à jour les composants de EISI. Il est nécessaire de fournir toute l'information relative aux analyses effectuées, à la méthodologie suivie et aux résultats fournis ainsi qu'à leur signification et leurs causes. Grossièrement, il s'agirait en quelques sortes d'implémenter le chapitre 2 de ce mémoire dans le système EISI. Ceci nous permettrait de créer une nouvelle "Meta-Database" pour les données sources relatives à notre système ainsi que de créer une nouvelle "Domain Encyclopedia" pour les concepts relatifs à notre système (méthodologie des estimations, méthodologie des publications,...).

Intéressons-nous maintenant plus particulièrement au mécanisme qui serait nécessaire pour "couler" notre système dans ce moule "EISI". Comme nous l'avons vu en détails au point 4.1.4., nous avons deux concepts de base ("scénario" et "planning") en "opposition". Dans le cadre d'une intégration dans EISI, nous devons donc modeler ce concept de "planning" pour qu'il soit utilisable par les scénarios. Voici donc les différentes étapes nécessaires à ce "modelage" :

- Transformer la structure logique de notre système de planning en une structure logique spécifique à EISI. Cette structure s'appelle "schéma".

- Intégrer des références ainsi que liens entre les données concrètes et les différents composants d'EISI contenant la méta-connaissance nécessaire à une bonne interprétation de ces données (signification des tables, de leurs composants, etc...).[3]

[3] Actuellement, le système EISI permet de référencer des données concrètes via les

- Bien définir explicitement la stratégie utilisée pour les différentes analyses (c'est-à-dire ne pas se limiter à l'explication pure et simple de l'algorithme mais aussi développer tous ses aspects cachés : but de l'analyse, choix effectués, interprétation des résultats...).

N'oublions pas que cette possible intégration signifie le mélange de deux domaines d'applications différents (les données et les méta-données). L'intégration de notre système devrait donc nécessiter l'ajout de deux nouveaux composants spécifiques à ces données "concrètes" à l'architecture EISI : la base des données opérationnelles ainsi que l'évaluateur de macros. Ces composants seront alors appelés à partir des nouveaux scénarios créés. Le module Interface sera plus ou moins "fusionné" avec le module "User Access Interfaces" .

Mais selon nous, l'apport le plus bénéfique de la fusion de notre système avec EISI est sa capacité de résoudre des problèmes en utilisant des données concrètes. Le système pourrait fournir à l'utilisateur le résultat de la résolution d'un problème tout en fournissant à celui-ci toutes les informations concernant le problème et les données. Pour cela, il faudrait intégrer le "planificateur" de notre système dans le "Reasoner" d'EISI (Voir l'Architecture à la section 3.2.3.1). Le "planificateur" et le "Reasoner" du modèle EISI pourront être fusionnés, leurs rôles étant proches, cependant "le planificateur" résoud des problèmes sur des données concrètes et le "Reasoner" travail sur la méta-information. Ils sont donc complémentaires.

6.5.2 Apports possibles de "MolGen".

Une deuxième amélioration possible pourrait porter sur l'"intelligence" du système de planning. En effet, comme on le voit dans MolGen, la résolution de certains problèmes grâce à certaines déductions faites via les contraintes et la connaissance du contexte du problème permet une plus grande liberté, une plus

méta-données (un concept est relié aux tables s'y rapportant).

grande indépendance du système. "MolGen" permet entre autres la résolution, grâce à son moteur d'inférence de problèmes dont les grandes lignes de la résolution ne sont pas définies auparavant.

Si on considère notre système couplé avec EISI, qui lui apporte justement la connaissance contextuelle relative au problème traité, il serait éventuellement possible d'envisager une orientation plus "intelligente" de notre système. Pour cela, un nouveau module dédié à ce travail de déduction (moteur d'inférence) devrait être mis au point. Cependant, une nouvelle analyse devrait être faite afin de bien étudier justement ce couplage entre EISI et notre système pour bien situer le rôle des contraintes (que l'on retrouve dans les plans et scénarios) et de la méta-information : on pourrait aboutir alors à la création d'un ensemble de règles permettant de gérer ce travail de déduction. En tout cas, il s'agit ici d'un travail considérable car les bases d'EISI et celles de notre système ne sont pas réellement orientées "Expert System" mais plus "Case Based reasoning". Un telle approche nécessiterait une autre approche au point de vue "architecture du système" ainsi qu'une autre approche du point de vue "étude de l'existant" (étude non seulement des données contextuelles relatives à un domaine d'application mais aussi des règles régissant ce domaine).

CONCLUSION

Ce mémoire n'est qu'une goutte dans ces océans que l'on appelle "systèmes experts", "intelligence artificielle", "problem solving", ... En effet, notre mémoire bien que débouchant sur une résolution générale de problèmes, ne fait qu'ajouter une petite pierre à ces édifices. Grâce à sa possible intégration dans un système plus complexe (EISI), il permet d'avancer dans un domaine bien particulier qu'est celui des statistiques officielles. Pour des domaines plus généraux, il doit encore faire ses preuves : il s'agit d'une proposition d'architecture qui ne demande qu'à être essayée et critiquée.

Ce mémoire nous a permis de mieux percevoir quels sont les souhaits actuels que l'on formulent dans le monde informatique : que veut-on maintenant et quelles sont les réponses à ces souhaits. Ils nous a également permis de voir qu'il y a encore pas mal d'efforts à faire pour arriver à des solutions informatiques "intelligentes" et très complètes. En un an, nous avons essayé d'approcher ce monde de l'informatique qui essaie de plus en plus d'imiter l'Homme mais notre conclusion est qu'il y a encore un long chemin à parcourir pour mettre en oeuvre ces applications tant désirées.

D'un point de vue plus pratique, ce mémoire laisse des portes ouvertes (intégration d'une notion "d'intelligence artificielle") mais sutout nous pensons qu'il met en valeur un type de résolution de problèmes qui est très intéressant : la résolution pratique et contextuelle (grâce à EISI) d'un problème qui laisse à l'Homme encore pas mal de libertés.

Des efforts sont encore à fournir à propos de la gestion des interfaces c'est-à-dire l'évolution de ceux-ci vers un mode d'utilisation "naturel" (approche du langage naturel : utopique en ce moment ?) ainsi que la gestion de l'acquisition de connaissances nécessaires à l'élaboration de plans exécutables et à l'extension de plans pré-existants mais quelque peu différents de ceux voulus. Ces extensions futures font appel à des théories complexes et pas encore totalement explorées de nos jours.

BIBLIOGRAPHIE

[BODART] F. Bodart & Y. Pigneur, "*Conception Assistée des Systèmes d'Information : méthode, modèle, outils*", Masson, seconde édition, Paris,1989

[COHEN] Paul R Cohen and Edward A. Feigenbaum, "*Planning and Problem Solving*" in *The Handbook of Artificial Intelligence,* Pitman, Volume 3, Chapitre 15, pp 514-561, Stanford, 1983.

[EPPRECHT] Eugénio K. Epprecht, "*An Approach for Knowledge Representation and Reasoning with Contextual Strategies in Knowledge-Based Statistical Consultancy Systems*", Thèse de doctorat en informatique, Institut d'Informatique, FUNDP Namur, Novembre 1992.

[EUROSTAT] Eurostat, "*Development of Statistical Expert Sytems*", Eurostat News Special Edition , Luxembourg, 1992.

[HAND] D. J. Hand, "*Artificial Intelligence in Statistics*" in *New Techniques and Technologies for Statistics, Proceeding*, Office for Official Publications of the European Communities, Luxembourg, 1992.

[HARDAN] Anatole d'Hardancourt, "*Microsoft Excel 4 pour Windows par la pratique*", Sybex, Paris, 1992.

[IWASAKI] Yumi Iwasaki, "*Qualitative Physics*" in *The Handbook of Artificial Intelligence*, Addison-Wesley Publishing Company, Volume 4, chapitre 21, pp 323-413, Stanford, 1989.

[KENNY85] P.B. Kenny, "*Revisions to Quartely Estimates of GDP*" in *Economic Trends n°381*, Central Statistics Office, Londres, Juillet 1985.

[KENNY87] P.B. Kenny, "*Revisions to Quartely Estimates of GDP*" in *Economic Trends n°406*, Central Statistics Office, Londres, Août 1987.

[KENNY92] P.B. Kenny and U.M. Rizki, "*Testing for Bias in Initial Estimates of Economic Indicators*" in *Economic Trends n°463*, Central Statistics Office, Londres, Mai 1992.

[MCDERMOTT] D. Mc Dermott and E. Chafniak, "*Introduction to Artificial Intelligence*", Addison-Westley Publishing Company, Massachusetts, 1985.

[MINSKY] M. Minsky, "*A framework for Representing Knowledge*" in *Mind Design*, the MIT Press, pp 95-128, Cambridge, 1981.

[MOSER] C.A. Moser & G. Kalton, "*Survey Method in Social Investigation*", Editions Gower, Michigan, 1979.

[PRODIGY] J. Carbonell, O. Etzioni, Y. Gil, R. Joseph, C. Knoblock, S. Minton and M. Veloso, "*PRODIGY : An Integrated Architecture for Planning and Learning*" in Sigart Bulletin, Volume 2, N°4, pp 51-55, 1989.

[RICH] Elaine Rich, "*Advanced Problem-Solving Systems*" in *Artificial Intelligence*, Mc Graw-Hill Book Company, pp 247-277, Austin, 1983.

[SART] Didier Sart, "*Des systèmes experts plus précis, grâce à la physique qualitative*" in *Science & Vie Micro*, N°54, Octobre 1988, pp 59-62.

[SATO] Hidela Sato, "*A Data Model, Knowledge Base, and Natural Language Processing for Sharing a Large Statistical Database*" in *Statistical and Scientific Database Management*, Springer, pp 205-225, Berlin, 1989.

[STEFIK] Mark Stefik, "*Planning and Meta-Planning*" in *Readings in Artificial Intelligence*, Tioga Publishing Company, pp 272-285, Palo Alto, 1989.

[TOWNSEND] Carl Townsend, "*Mastering Excel 4 for Windows*", Sybex, San Francisco, 1992.

[UKNA85] CSO United Kingdom National Accounts, "*Sources and Methods, a publication of the Government Statistical Service*", Central Statistics Office, Troisième édition, London, 1985.

[UKNA91] CSO United Kingdom National Accounts, "*The CSO Blue Book 1991 edition*", Central Statistics Office, London, 1991.

[VANEY90] Christopher J. de Vaney, "*Expert Interface to statistical Information (EISI) - Overview of the Conceptual Architecture*", EISI Project Working Paper, World Systems Europe, Luxembourg, 1990

[VANEY92] Christopher J. de Vaney, "*An Overview of Reasoning in the EISI Sytem*", World Systems Europe, Luxembourg, 1992.

[VANEY93] Christopher J. de Vaney, "*Integrating Statistical Meta-Information and Methodologies*", World Systems Europe, Luxembourg, 1993.

[VFLMS] C. de Vaney, M. Fleming, J. Lamb, G. Mersch, J. Sanchez, "*The Expert Interface to Statistical Information Rationale, Techniques and Experiences*" in *New Techniques and Technologies for Statistics, Proceeding*, Bonn, Office for Official Publications of the European Communities, Luxembourg, 1992.

[WINOGRAD] T. Winograd & D.G. Bobrow, "*An overview of KRL, a Knowledge Representation Language*" in *Cognitive Science 1*, pp 3-46, Houston, 1977.

ANNEXE A

"fonctions auxiliaires, sous-fonctions et algorithmes"

A.1. Les fonctions de création de graphiques.

A.1.1. "Create-graphic".

a) Buts de la fonction.

Cette fonction permet la création d'un graphique qui sera construit à partir des valeurs d'une ou deux séries statistiques périodiques. On peut créer 5 types de graphiques standard. Le premier type est un graphique histogramme permettant d'afficher une série, le second type est un graphique histogramme permettant de comparer deux séries entre elles, le troisième type est un graphique histogramme permettant de comparer plusieurs séries (maximum de cinq à six séries pour la clarté du graphismes), le quatrième type est un graphique linéaire permettant d'afficher une série statistique, et enfin le cinquième type est un graphique linéaire permettant de comparer deux séries entre elles.

Les périodes concernant les séries à afficher sont données comme paramètres à la fonction.

b) Algorithme :

Arguments :
Sheet: nom du tableau (de la feuille) qui contient les données à afficher..
Per: La ou les période(s) dont on veut afficher la série.
Type: Type du graphique désiré, il s'agit d'un chiffre de 1 à 6.
Result: Nom du tableau (de la feuille) qui contiendra le graphique créé si on désire conserver le graphique en vue d'une utilisation ultérieure.

Preconditions: Number(Data(Sheet)) * Les données doivent être numériques.

Exist(Sheet)	* La feuille de données doit exister.
Standard(Sheet)	* La feuille de données doit avoir un format standard.
In_period(period)	* La période doit faire partie des périodes reprises dans la feuille de donnée.
In_type(type)	* Le type du graphique est un type autorisé.

appel fonction "ouverture_fichier()"
si erreur alors fin d'exécution
appel fonction "rech.ref(sheet,per)"
appel fonction "axe.x(type)"
appel fonction "création.graphique(la référence aux données,type,per,booleen qui indique si on affiche une ou deux séries,le titre du graphique)"
Pause
Quand fin pause
appel fonction "fermeture_fichier()"
Fin

A.1.2. Graphic_variability.

a) Buts de la fonction

Cette fonction trace un graphique du 4° type définit dans la section sur les choix des graphiques (point 2.4.3.2.2.). A partir des résultats de l'analyse de variabilité, on peut construire un graphique représentant la variabilité des données de toute les séries de la feuille de données. Pour chaque élément d'une série, on trace un carré coloré à la hauteur correspondant à sa valeur sur l'axe des ordonnées. Il y a trois couleurs différentes: Une couleur pour les carrés représentant des valeurs inférieures à 1, une couleur pour les carrés représentant des valeurs égales à 1 et une couleur pour les carrés représentant des valeurs supérieures à 1.

b) Algorithme:

Arguments :
Sheet: nom du tableau (de la feuille) qui contient les données à afficher..
Result: Nom du tableau (de la feuille) qui contiendra le graphique créé si on désire conserver le graphique en vue d'une utilisation ultérieure.
Preconditions:
Number(Data(Sheet)) * Les données doivent être numériques.
Exist(Sheet) * La feuille de données doit exister.
Standard(Sheet) * La feuille de données doit avoir un format standard.

appel fonction "ouverture_fichier()"
si erreur alors fin d'exécution
appel fonction "chercher_borne(sheet)"
appel fonction "tracer_borne(sheet)"
appel fonction "Rech.nb.series(sheet)"
appel fonction "tracer(indications sur les graduations,indications sur le nombre de série, sheet)
Affichage du graphique à l'écran
Pause
Quand fin pause
appel fonction "fermeture_fichier()"
Fin

A.1.3. Graphic-ratio.

a) buts de la fonction

Cette fonction trace un graphique du 5° type définit dans la section sur les choix des graphiques (point 2.4.3.2.2.). A partir des résultats de l'analyse de variabilité, on peut construire un graphique représentant chaque élément du tableau par un carré coloré. Une couleur représente un intervale de valeurs pour les éléments. Cette vue du graphique permettra de voir les données classées par groupe de valeurs et plus uniquement comme valeur individuelle

b) Algorithme:

Arguments :
Sheet: nom du tableau (de la feuille) qui contient les données à afficher..
Result: Nom du tableau (de la feuille) qui contiendra le graphique créé si on désire conserver le graphique en vue d'une utilisation ultérieure.
Preconditions:
Number(Data(Sheet)) * Les données doivent être numériques.
Exist(Sheet) * La feuille de données doit exister.
Standard(Sheet) * La feuille de données doit avoir un format standard.

```
appel fonction "ouverture_fichier()"
        si erreur alors fin d'exécution
appel fonction "creation.axes.y(sheet)"
appel fonction  "creation.axes.x(sheet)"
Affichage du graphique à l'écran
Pause
Quand fin pause
    appel fonction "fermeture_fichier()"
Fin
```

A.2. Les fonctions "utilitaires"

A.2.1. "Display-table"

a) Buts de la fonction

Les quatres fonctions décrites auparavant dans ce document s'occupent exclusivement d'effectuer des calculs et de créer des tableaux de résultats. La visualisation de ces derniers n'est possible que grâce à cette fonction : elle affiche les tableaux à l'écran et permet également grâce à l'interface d'Excel de les modifier si nécessaire. Il s'agit donc d'une fonction très simple.

b) Algorithme

```
Argument: file name : nom du fichier (tableau) à afficher à l'écran

appel fonction "ouverture_fichier()"
si erreur
    appel fonction "erreur_fichier()"
    fin d'exécution
activation de la feuille de calcul (ou tableau)
réduction (rétrécissement) pour bien visualiser le tableau à l'écran
diviser la feuille en 2 sous-fenêtres afin de bien avoir les intitulés des lignes
affichage du bouton Pause et mise en pause
    quand bouton activé mettre feuille active sous forme d'icône
fin d'exécution
```

A.3. Les fonctions d'impression

Afin de pouvoir imprimer correctement tous les documents utiles à une analyse, il est nécessaire de créer quatre fonctions différentes. Celles-ci se différencient sur le plan de l'implémentation car elles concernent chacune un type de support "logique" différent (feuille de calcul pour les tableaux, feuille de graphisme pour les graphiques standard, feuille de calcul pour les graphiques "variabilité" et "ratio") et des entrées de tailles différentes (un graphique standard tient sur une page ce qui n'est pas toujours le cas des tableaux et des autres graphiques). Ces fonctions sont créées dans le but de simplifier le travail de l'utilisateur c'est-à-dire en lui fournissant une impression de qualité du point de vue de la mise en page : choix automatique des options d'impression, réduction des tableaux afin de les imprimer sur une seule page (si c'est possible), etc...

A.3.1. "Print-table"

Cette fonction permet l'impression d'un tableau de données. Si nécessaire, elle réduit le tableau afin de le présenter entièrement sur une page.

A.3.2. "Print-chart"

Cette fonction permet l'impression des graphiques standards créés et désirés par l'utilisateur.

A.3.3. "Print-variab"

Cette fonction déclenche l'impression d'un graphique de "variabilité". Cette impression se fera généralement sur plusieurs pages (selon la longueur des séries temporelles).

A.3.4. "Print-ratio"

Cette fonction déclenche l'impression d'un graphique des "ratios".

Remarque : toutes ces impressions nécessitent une imprimante couleur.

A.3.5. Algorithme

Argument : file name = nom de la "feuille" à imprimer

appel fonction "ouverture_fichier()"
si erreur
    appel fonction "erreur_fichier()"
    fin d'exécution
****
(**) établissement de la zone à imprimer (car un tableau n'est qu'une partie d'une feuille Excel) et rétrécissemnt si nécessaire
(**) sélection du garphique de la feuille
appel fonction "iconiser()"
activation de la feuille à imprimer
mise en page
impression
appel fonction "fermer_fichier()"
fin d'exécution
(**) : Indique qu'à cet endroit il faut faire un choix selon le type de la fonction d'impression (choix 1 pour les feuilles de calcul c'est-à-dire les impressions de tableaux et de graphiques non-standards et choix 2 pour les graphiques standards)

A.4. Algorithmes des fonctions principales

L'algorithme suivant s'applique aux fonctions "perform-proportional-analysis", "perform-difference-analysis" et "perform-variability-analysis" car lors du changement de base (dernière ou première valeur éditée) la formule de calcul ne change quasi pas. Par contre, pour les fonctions "perform-qualitative-analysis" et "perform-quantitative-analysis", lors du changement de cette base, on doit changer la valeur des indices intervenant dans la formule de calcul utilisée ce qui provoque quelques changements dans l'algorithme. Ceux-ci seront montrés dans le deuxième algorithme.

```
Arguments :     base = "First" ou "Last" (pour première ou dernière valeur)
                input file name = nom du fichier contenant les données à
                analyser output file name = nom du fichier recevant les
                résultats d'analyse

appel fonction "ouverture_fichier()"
   si erreur alors fin d'exécution
appel fonction "conversion_arguments()"
indice_colonne = 0
appel fonction "format_sheet(input file name ; output file name)"
tant que nombre de séries du tableau > 0
   appel fonction "affiche_pourcent()"
   indice_cellule_active = OK
   indice_ligne = 0
   pour chaque cellule d'une colonne du tableau
         si cellule_analysée n'est pas vide
               si indice_cellule_active = OK (=cellule de base)
                     si base= "first"
                           première_cellule = cellule_analysée
                     sinon
                           appel fonction "rech_ref(intervalle)"
```

```
                                    première_cellule = dernière cellule non-vide
                                    de la colonne du tableau
                            ****
                    ****
                    application de la formule de calcul
                    coloration de la cellule contenant le résultat d'analyse selon
                    le type de résultat
                    indice_cellule_active = valeur de la cellule_analysée
            ****
            indice_ligne + 1
    ****
    indice_colonne +1
    nombre de séries du tableau - 1
****
appel fonction "fermer_fichier()"
fin
```

Pour la fonction "perform_qualitative_analysis", seule l'intérieur de la boucle "pour chaque" change. Nous ne développerons donc que cette partie de l'algorithme. Les arguments ne changent pas.

```
Si cellule_analysée n'est pas vide
    si indice_cellule_active = OK
            i = indice_ligne
            si base = "first"
                    début_série = indice_ligne
                    mettre 0 dans la première cellule de la série du tableau
                    résultat
            sinon
```

```
                début_série = indice_ligne
                mettre " " dans la première cellule de la série du tableau
                résultat
        ****
        tendance = 0
sinon
        si base = "first"
                tendance = résultat de la formule de calcul dans le cas ou
                l'indice de la base = 1
                écrire le résultat dans la cellule du tableau résultat
                        correspondante
        sinon
                tendance = résultat de la formule de calcul dans le cas ou
                l'indice de la base = n
                écrire le résultat dans la cellule du tableau résultat
                        correspondante
        ****
        i = i+1
        appel fonctiion "série()"
****
indice_cellule_active = cellule_analysée
```

A.5. Lexique des sous-fonctions

Fonction affiche-pourcent

Entrées :

- le nombre total de séries à analyser dans un tableau stipulé en input

Définition :

Cette fonction affiche dans le bas de l'écran le pourcentage d'analyse déjà effectué lors de l'exécution d'une fonction d'analyse. Elle permet de voir l'evolution en temps de l'analyse en cours.

Sorties :

- affichage du pourcentage de travail déjà effectué

Fonction axe.x

Entrée:

- Nom de la feuille de données.

Définition:

Recherche le titre d'une feuille de donnée fin de pouvoir l'intégrer sur le graphique. Ce titre est placé à la première ligne de la feuille de données.

Sorties:

- Le titre de la feuile de données.

Fonction chercher borne

Entrée:

Le nom de la feuille de donnée à traiter.

Définition:

Fonction qui recherche la plus grande et la plus petite valeur de la feuille de données, et qui calcule les graduations à placer sur un axe des ordonnées en fonction de ces valeurs.

Sortie:

- Le nombre de graduations à placer
- La différence de valeurs entre deux graduations.

Fonction colorier.

Entrée:

La valeur de l'élément à traiter.

Définition:

Fonction qui colorie le dernier carré crée. La couleur choisie dépend de la valeur de l'élément représenté par le carré.

sortie:

Le carré est mis en couleur.

Fonction conversion-arguments

Entrées :

- nom du tableau contenant les données à analyser

- nom du tableau pour les résultats

Définition :

Cette fonction retourne la référence de la première cellule du tableau stipulé en input susceptible de contenir une donnée à analyser. Elle retourne également la référence de la première cellule du tableau-résultat susceptible de recevoir un résultat d'analyse.

Sorties :

- intervalle = référence de la première cellule du tableau stipulé en input susceptible de contenir une donnée à analyser

- sresult = référence de la première cellule du tableau-résultat susceptible de recevoir un résultat d'analyse.

Fonction création.axes.y

Entrée:

le nom de la feuille de données.

Définition:

Fonction qui trace sur le graphique les graduations sur l'axe de ordonnées.

Sortie:

L'axe y et ses graduations sont ajoutées au graphique.

Fonction création.axes.x

Entrée:

La feuille de données et le nombre de séries a tracer.

Définition:

Fonction qui trace sur le graphique les données des séries, série par série.

Sortie:

Les données de chaque séries sont tracées sur le graphique.

Fonction creation graphique.

Entrées:

- Référence (adresses) aux données.
- Le type du graphique souhaité.
- Les titres (Ensembles des labels à afficher comme titres,légende,nom des axes ..)

Définition:

Fonction qui crée un objet de type graphique. Le type du graphique ainsi que les références aux données à faire apparaître sur ce graphique sont fournis en entrée. En plus de la création du graphique, cette fonction affiche également un grand nombre de labels nécessaire pour l'interprétation de ce graphique.

Sorties:

L'affichage du graphique à l'écran.

Fonction creer.carre

Entrée:

- La valeur d'un élément à traiter
- Des renseignements sur les graduations de l'axe des ordonnées.

Définition:

Fonction qui ajoute un carré au graphique. Les coordonnées de ce carré dépendent de la valeur passé comme paramètre.

Sortie:

Un carré est ajouté au graphique.

Fonction creer-fichier

Entrées :

- nom du tableau pour les résultats

Définitions :

Cette fonction crée un tableau destiné à recevoir les résultats d'une analyse. Ce nouveau tableau reçoit comme nom celui stipulé en entrée.

Sorties : /

Fonction créer légende.

Entrée:

Les coordonnées où la légende sera placée sut le graphique.

Définition:

Fonction qui trace la légende en bas du graphique.

Sortie:

La légende est ajoutée au graphique.

Fonction effacer-contenu

Entrées :

- nom du tableau pour les résultats

Définitions :

Cette fonction efface le contenu d'un tableau stipulé en output

Sorties : /

Fonction erreur-fichier

Entrées : /

Définitions :

Cette fonction affiche à l'écran un message d'alerte prévenant de la non-validité du nom de fichier en input encodé.

Sorties :

un message prévenant de l'erreur : "fichier non-valide"

Fonction fermer-fichier

Entrées : /

Définitions :

Cette fonction ferme (en sauvant) ***tous*** les fichiers utilisés lors de l'exécution d'une analyse. Les fichiers de données (tableau, graphiques) sont fermés en premier, ensuite ce sont les fichiers contenant les programmes (donc les modules).

Sorties : /

Fonction format-sheet

Entrées :

- nom du tableau contenant les données à analyser
- nom du tableau pour les résultats

Définitions :

Cette fonction permet de mettre d'écrire des intitulés de colonnes et de lignes dans le tableau-résultats. Ces intitulés seront identiques à ceux employés dans le tableau contenant les données qui sont à la base de l'analyse.

Sorties :

- un tableau-résultats vide et formatté

Fonction iconiser

Entrées : /

Définitions :

Cette fonction permet de réduire en icône tous les fichiers qui sont en cours d'utilisation lors de son appel.

Sorties : /

Fonction initialiser.tab

Entrée:

- Le nombre de valeurs possibles sur l'axe de Y.

Définition:

Fonction qui initialise un tableau qui reprend les coordonnées en X du prochain carré à tracer sur le graphique, et ce pour chaque valeur possible des Y.

Sortie:

Un tableau dont chaque élément vaut 60 qui est la coordonée sur l'axe des X du premier carré à afficher.

Fonction ouverture-fichier

Entrées :

- nom du tableau contenant les données à analyser

- nom du tableau pour les résultats

Définitions :

Cette fonction permet d'ouvrir les tableaux stipulés en entrée. Si le tableau à analyser n'existe pas alors on déclenche la fonction "erreur-fichier". Si le tableau-résultats n'existe pas encore, alors on fait appel à la fonction "créer-fichier".

Sorties : /

Fonction premier.element

Entrée:

La nom de la feuille de donnée et un numéro de colonne i de cette feuille.

Définition:

Fonction qui recherche la référence, sur une feuille de donnée, du premier élément d'une série donnée.

Sortie:

Le numéro de la ligne qui contient le premier élément.

Fonction rech.nb.séries

Entrée:

Le nom de la feuille des données.

Définition:

Fonction qui recherche le nombre de séries apparaissant sur une feuille de données.

Sortie:

Le nombre de séries.

Fonction rech.period

Entrée:

Le numéro de la ligne de la feuille de donnée concernée.

Définition:

Fonction qui recherche le nom de la période de la série située à la i° colonne de la feuille de données.

Sortie :

Le nom de la période.

Fonction rech-refer

Entrées :

-intervalle (provient de la fonction "conversion-arguments") : contient la référence de la première cellule du tableau susceptible de contenir une donnée à analyser.

Définition :

Cette fonction recherche la référence de la dernière cellule non-vide d'une série

Sortie :

référence de la dernière cellule non-vide

Fonction rech.ref

Entrées:

- La feuille de données.
- La période de la série concernée.
- Le nombre de séries dont on recherche les références (= 1 ou 2)

Définition:

Fonction qui calcul une référence absolue des valeurs d'une ou deux séries. Le calcul de ces références se fait à partir du nom de la feuille de données, ainsi que de la période de la ou des série(s) concernées sur la feuille.

Sorties :

Les références aux données concernées.

Fonction série

Entrées : /

Définition :

Fonction qui, lors de l'analyse d'une série d'un tableau, met progressivement en couleur les sous-séries de résultats ayant des interprétations équivalentes ayant une longueur >= 3.

Sorties :

- sous-séries de longueur >= 3 colorées selon leur tendance.

Fonction tracer.borne

Entrees:

-Le nombre de graduations désirées.

- La différence de valeur entre deux graduations.

Définition:

Fonction qui trace sur une feuille de données un axe vertical avec le nombre de graduations désirées et le label des graduations.

Sortie:

-L'axe vertical est tracé.

Fonction tracer.

Entrée:

-Le nombre de graduations désirées.

- La différence de valeur entre deux graduations.

- La référence à la fauille qui contient les données.

- Le nombre de séries que contient la feuille de données.

Définition:

Fonction qui trace le graphique série par série.

Sorties:

-Les valeurs de chaque série sont représentées sur le graphique.

Fonction tracer.ligne

Entrée: Le nombre de lignes.

Définition:

Fonction qui sépare le traçage du graphique entre deux séries successives.

Sortie:

Une séparation entre deux série est traçée et le tableau d'initialisation est mis à jour.

Fonction traiter.serie

Entrée:

- La feuille de données.
- Une indication sur les graduations de l'axe Y.

Définition:

Fonction qui trace le graphique pour une série donnée.

Sortie:

Les données de la série traitée sont présentes sur le graphique.

Fonction tracer.serie

Entrée:

Le numéro de la série et la feuille de données.

Définition:

Fonction qui représente sur le graphique les données pour une série.

Sortie:

Les données de la série sont représentés sur le graphique.

ANNEXE B

Listings des macros "Excel"

	1	2
1	=format_sheet	*Fonction qui met une feuille de travail en page sous un format standard.
2	=ARGUMENT("feuille",2)	*feuille: feuille de référence.
3	=ARGUMENT("fresult",2)	*fresult: nom de la feuille à formater.
4	=SET.NAME("inter";SUBSTITUTE("nomIR8C3";"nom";feuille))	
5	=SET.NAME("inter";TEXTREF(inter;))	*inter: référence la 8° ligne et la 3° colonne de la feuille de référence.
6	=SET.NAME("sres";SUBSTITUTE("nomIR8C1";"nom";fresult))	
7	=SET.NAME("sres";TEXTREF(sres;))	*sres: référence la 8° ligne et la 1° colonne de la feuille de résultat.
8	=FORMULA.GOTO(OFFSET(sres;-5;0):OFFSET(sres;-5;col+1);FALSE)	*mise en page (tracer en gras) des lignes et colonnes pour la feuille de résultat
9	=BORDER(0;0;0;0;2;0;0;0;0;0;0)	*
10	=FORMULA.GOTO(OFFSET(sres;-1;2):OFFSET(sres;-1;col+1);FALSE)	*
11	=BORDER(0;0;0;0;6;0;0;0;0;0;0)	*
12	=FORMULA.GOTO(OFFSET(sres;1;2):OFFSET(sres;1;col+1);FALSE)	*
13	=BORDER(0;0;0;0;2;0;0;0;0;0;0)	*
14	=FORMULA.GOTO(OFFSET(sres;-4;1):OFFSET(sres;lig+1;1);FALSE)	*
15	=BORDER(0;0;2;0;0;0;0;0;0;0;0)	*
16	=FORMULA.GOTO(OFFSET(sres;lig+1;0):OFFSET(sres;lig+1;col+1);FALSE)	*
17	=BORDER(0;0;0;0;2;0;0;0;0;0;0)	*
18	=FORMULA.GOTO(OFFSET(sres;-7;col+1):OFFSET(sres;lig+1;col+1);FALSE)	*
19	=BORDER(0;0;2;0;0;0;0;0;0;0;0)	*
20	=FORMULA.GOTO(OFFSET(sres;-1;col+1);FALSE)	*
21	=BORDER(0;0;2;0;6;0;0;0;0;0;0)	*
22	=FORMULA.GOTO(OFFSET(sres;1;col+1);FALSE)	*
23	=BORDER(0;0;2;0;2;0;0;0;0;0;0)	*
24	=FORMULA.GOTO(OFFSET(sres;lig+1;1);FALSE)	*
25	=BORDER(0;0;2;0;2;0;0;0;0;0;0)	*
26	=FORMULA.GOTO(OFFSET(sres;lig+1;col+1);FALSE)	*
27	=BORDER(0;0;2;0;2;0;0;0;0;0;0)	*
28	=FORMULA.GOTO(OFFSET(sres;-5;col+1);FALSE)	*
29	=BORDER(0;0;2;0;2;0;0;0;0;0;0)	*
30	=FORMULA.GOTO(OFFSET(sres;-7;0):OFFSET(sres;1;col+1);FALSE)	*mise en page des cellules composants les lignes et les colonnes de titres
31	=PATTERNS(1;2;0)	*c'est-à-dire effacement du quadrillage.
32	=FORMULA.GOTO(OFFSET(sres;1;0):OFFSET(sres;lig+1;1);FALSE)	*
33	=PATTERNS(1;2;0)	*
34	=FORMULA.GOTO(OFFSET(sres;lig+2;0):OFFSET(sres;248;col+2);FALSE)	*
35	=PATTERNS(1;2;0)	*
36	=FORMULA("Issue of";OFFSET(sres;-4;0))	*Ecriture des titres généraux
37	=FORMULA("Economic Trends";OFFSET(sres;-3;0))	*
38	=FORMULA("in which figures";OFFSET(sres;-2;0))	*
39	=FORMULA("were published";OFFSET(sres;-1;0))	*
40	=FORMULA("Source : Table 1";OFFSET(sres;lig+3;col-2))	*
41	=FORMULA.GOTO(OFFSET(sres;lig+3;col-2);FALSE)	*
42	=FORMAT.FONT("Times New Roman";14;FALSE;FALSE;FALSE;FALSE;0)	*
43	=SET.NAME("titre";SUM(-3;PRODUCT(col;1/2)))	*
44	=SET.NAME("entete";OFFSET(inter;-6;4))	*
45	=FORMULA(UPPER(entete);OFFSET(sresult;-5;titre))	*
46	=FORMULA.GOTO(OFFSET(sresult;-5;titre);FALSE)	*
47	=FORMAT.FONT("Times New Roman";12;TRUE;TRUE;TRUE;FALSE;0)	*
48	=SET.NAME("cln";0)	
49	=SET.NAME("lgn";0)	
50	=FOR.CELL("refer";OFFSET(inter;0;0):OFFSET(inter;0;col-1);TRUE)	*Ecriture des intitulés des années dans chaque colonne et ligne de la feuille de
51	= FORMULA(refer;OFFSET(sres;0;cln+2;1;1))	* résultat.
52	= FORMULA(refer;OFFSET(sres;lgn+2;0;1;1))	*
53	= SET.NAME("cln";cln+4)	*
54	= SET.NAME("lgn";lgn+4)	*

	1	2
55	=NEXT()	*
56	=SET.NAME("cln";0)	
57	=SET.NAME("refer";inter)	
58	=FOR.CELL("refer";OFFSET(inter;1;0):OFFSET(inter;1;col-1);)	* Ecriture des intitulés des trimestres dans chaque colonne et ligne de la feuille
59	= FORMULA(refer;OFFSET(sres;1;cln+2;1;1))	* de résultat
60	= SET.NAME("cln";cln+1)	*
61	=NEXT()	*
62	=SET.NAME("lgn";0)	*
63	=FOR.CELL("refer";OFFSET(inter;2;-1):OFFSET(inter;lig+1;-1);)	*
64	= FORMULA(refer;OFFSET(sres;lgn+2;1;1;1))	*
65	= SET.NAME("lgn";lgn+1)	*
66	=NEXT()	*
67	=RETURN()	
68		
69		
70		*Fonction qui effecture une analyse proportionnelle des séries contenues
71	perform_proportional_analysis	*dans une feuille de travail donnée.
72	=ARGUMENT("base";2)	*base: indique quelle donnée d'une série on choisit comme base de comparaison.
73	=ARGUMENT("feuille";2)	*feuille: nom de la feuille de donnée.
74	=ARGUMENT("fresult";2)	*frésult : nom de la feuille de résultat.
75	=MESSAGE(TRUE;"WORKING !!!!!! WAIT A FEW SECONDS, PLEASE !!")	
76	=IF(ouverture_fichier()=0)	*ouverture des fichiers.
77	= GOTO(R137C1)	
78	=END.IF()	
79	=effacer_contenu()	*On vide la contenu de la feuille de résultat.
80	=ECHO(FALSE)	
81	=conversion_arguments()	* On convertit les arguments en référence sur lesquelles on pourra travailler.
82	=SET.NAME("n";0)	
83	=format_sheet(feuille;fresult)	* Appel de la fonction de mise en page d'une feuille de résultat
84	=FORMULA("percentage difference from figure first published";OFFSET(sresult;-4;titre+1))	* Ecriture des titres généraux
85	=FORMULA("percentage %";OFFSET(sresult;-5;col-2))	*
86	=FORMULA("Table 2";OFFSET(sresult;-8;-1))	*
87	=FORMULA.GOTO(OFFSET(sresult;-8;-1);FALSE)	*
88	=FORMAT.FONT("Courier New";14;TRUE;FALSE;FALSE;FALSE;0)	*
89	=SET.NAME("pourcent";col)	* pourcent: nombre de séries à traiter.
90	=WHILE(col > 0)	*On boucle sur chaque série à analyser.
91	= affiche_pourcent()	* appel de la fonction qui permet de visualiser l'état d'avancement de l'analyse
92	= SET.NAME("indice";0)	
93	= SET.NAME("m";0)	*m:déplacement (exprimé en nombre de lignes) par rapport à la première ligne
94	= WHILE(AND(ISBLANK(OFFSET(intervale;m;n)));m<lig+1))	*On boucle sur chaque donnée de la série traitée (une série = une colonne).
95	= SET.NAME("m";m+1)	
96	= NEXT()	
97	= WHILE(AND(NOT(ISBLANK(OFFSET(intervale;m;n)))));m<lig+1))	
98	= SET.NAME("cel";OFFSET(intervale;m;n))	
99	= IF(indice=0)	*Choix de la valeur de base en fonction du paramètre "base".
100	= IF(base="first")	*choix de la première valeur de la série.
101	= SET.NAME("premier";cel)	*
102	= ELSE()	*choix de la dernière valeur de la série.
103	= SET.NAME("p";rech.ref(intervale))	*on recherche la référence de la dernière donnée de la série.
104	= SET.NAME("premier";DEREF(OFFSET(intervale;p-1;n)))	*
105	= END.IF()	
106	= END.IF()	
107	= PRODUCT((SUM(cel;(-premier)));(1/premier))	*application de la formule d'analyse
108	= SET.NAME("resultat";PRODUCT(R[-1]C;100))	*résultat: résultat de l'application de la formule d'analyse sur la données courante.

	1	2
109	= IF(ABS(resultat)<0,000000001)	
110	= SET.NAME("resultat";0)	*On arrondit une valeur proche de 0.
111	= END.IF()	
112	= IF(ABS(resultat)>=10)	*On fait ressortir le résultat > 10.
113	= FORMULA.GOTO(OFFSET(sresult;m;n;1;1);FALSE)	*
114	= BORDER(2;0;0;0;0;0;5;0;0;0;0)	*
115	= FORMULA(resultat;OFFSET(sresult;m;n;1;1))	*
116	= ELSE()	
117	= IF(AND(ABS(resultat)>3;ABS(resultat)<=10))	*On fait ressortir les valeurs comprises entre 3 et 10.
118	= FORMULA.GOTO(OFFSET(sresult;m;n;1;1);FALSE)	*
119	= BORDER(2;0;0;0;0;0;3;0;0;0;0)	*
120	= FORMULA(resultat;OFFSET(sresult;m;n;1;1))	*
121	= ELSE()	*On transcrit les autres résultats tels quels.
122	= FORMULA.GOTO(OFFSET(sresult;m;n;1;1);FALSE)	*
123	= BORDER(0;0;0;0;0)	*
124	= FORMULA(resultat;OFFSET(sresult;m;n))	*
125	= END.IF()	
126	= END.IF()	
127	= SET.NAME("indice";cel)	
128	= SET.NAME("m";m+1)	*On passe à la donnée(= ligne) suivante
129	= FORMAT.NUMBER("0.00")	
130	= NEXT()	*On passe à la série suivante(=colonne suivante).
131	= SET.NAME("n";n+1)	*incrémentation du compteur de positionnement dans une colonne
132	= SET.NAME("col";col-1)	* on décrémente le nombre de colonnes à analyser.
133	=NEXT()	
134	=FORMULA.GOTO(sresult)	
135	=MESSAGE(FALSE)	
136	=fermer_fichier()	*fermeture des fichiers dont on ne se sert plus dans un avenir immédiat.
137	=RETURN()	
138		
139		
140		*Fonction qui effecture une analyse différentielle des séries contenues
141	perform_difference_analysis	*dans une feuille de travail donnée.
142	=ARGUMENT("base";2)	*base: indique quelle donnée d'une série on choisit comme base de comparaison.
143	=ARGUMENT("feuille";2)	*feuille: nom de la feuille de donnée.
144	=ARGUMENT("fresult";2)	*frésult : nom de la feuille de résultat.
145	=MESSAGE(TRUE;"WORKING !!!!!! WAIT A FEW SECONDS, PLEASE !!")	
146	=IF(ouverture_fichier()=0)	*ouverture des fichiers.
147	= GOTO(R206C1)	
148	=END.IF()	
149	=effacer_contenu()	*On vide la contenu de la feuille de résultat.
150	=ECHO(FALSE)	
151	=conversion_arguments()	* On convertit les arguments en référence sur lesquelles on pourra travailler.
152	=SET.NAME("n";0)	
153	=format_sheet(feuille;fresult)	* Appel de la fonction de mise en page d'une feuille de résultat
154	=FORMULA("£millions";OFFSET(sresult;-5;col-2))	* Ecriture des titres généraux
155	=FORMULA("difference between the current figure and the figure first published";OFFSET(sresult;-4;titre))	*
156	=FORMULA("Table 4";OFFSET(sresult;-8;-1))	*
157	=FORMULA.GOTO(OFFSET(sresult;-8;-1);FALSE)	*
158	=FORMAT.FONT("Courier New";14;TRUE;FALSE;FALSE;FALSE;0)	*
159	=SET.NAME("pourcent";col)	* pourcent: nombre de séries à traiter.
160	=WHILE(col > 0)	*On boucle sur chaque série (=colonne) à analyser.
161	= affiche_pourcent()	* appel de la fonction qui permet de visualiser l'état d'avancement de l'analyse
162	= SET.NAME("indice";0)	

	1	2
163	= SET.NAME("m";0)	*m:déplacement (exprimé en nombre de lignes) par rapport à la première ligne
164	= WHILE(AND((ISBLANK(OFFSET(intervale;m;n)));m<lig+1))	*on boucle tant que l'on se trouve sur des cellules vides
165	= SET.NAME("m";m+1)	
166	= NEXT()	
167	= WHILE(AND((NOT(ISBLANK(OFFSET(intervale;m;n))));m<lig+1))	*On boucle sur chaque donnée de la série traitée.
168	= SET.NAME("cel";OFFSET(intervale;m;n))	
169	= IF(indice=0)	*Choix de la valeur de base en fonction du paramètre "base".
170	= IF(base="first")	*choix de la première valeur de la série.
171	= SET.NAME("premier";cel)	*
172	= ELSE()	*choix de la dernière valeur de la série.
173	= SET.NAME("p";rech.ref(intervale))	*
174	= SET.NAME("premier";DEREF(OFFSET(intervale;p-1;n)))	*
175	= END.IF()	
176	= END.IF()	
177	= SUM(cel;-premier)	
178	= SET.NAME("resultat";R[-1]C)	*résultat: résultat de l'application de la formule d'analyse sur la donnée courante.
179	= IF(resultat>0)	*On fait ressortir le résultat positifs.
180	= FORMULA.GOTO(OFFSET(sresult;m;n;1;1);FALSE)	*
181	= BORDER(0;2;2;2;2;;;1;1;1;1)	*
182	= PATTERNS(1;8;8)	*
183	= FORMULA(resultat;OFFSET(sresult;m;n;1;1))	*
184	= ELSE()	
185	= IF(resultat<0)	*On fait ressortir les résultats négatifs.
186	= FORMULA.GOTO(OFFSET(sresult;m;n;1;1);FALSE)	*
187	= BORDER(0;2;2;2;2;;;1;1;1;1)	*
188	= PATTERNS(1;3;1)	*
189	= FORMULA(resultat;OFFSET(sresult;m;n;1;1))	*
190	= ELSE()	*On transcrit les autres résultats tels quels.
191	= FORMULA.GOTO(OFFSET(sresult;m;n;1;1);FALSE)	*
192	= BORDER(0;2;2;2;2;;;1;1;1;1)	*
193	= PATTERNS(0;0;1)	*
194	= FORMULA(resultat;OFFSET(sresult;m;n))	*
195	= END.IF()	
196	= END.IF()	
197	= SET.NAME("indice";cel)	*On passe à la donnée suivante
198	= SET.NAME("m";m+1)	
199	= NEXT()	*On passe à la série suivante.
200	= SET.NAME("n";n+1)	*incrémentation du compteur de positionnement dans une colonne
201	= SET.NAME("col";col-1)	* on décrémente le nombre de colonnes à analyser.
202	=NEXT()	
203	=FORMULA.GOTO(sresult)	
204	=MESSAGE(FALSE)	
205	=fermer_fichier()	*fermeture des fichiers dont on ne se sert plus dans un avenir immédiat.
206	=RETURN()	
207		
208		
209		
210	rech.ref	*Fonction qui recherche la référence de la dernière donnée d'une série.
211	=RESULT(1)	
212	=ARGUMENT("intervale";8)	
213	=SET.NAME("lign";0)	
214	=WHILE(lign<256)	*On boucle jusqu'à ce qu'on ait retrouvé la première donnée significative d'une série.
215	= IF(NOT(ISBLANK(DEREF(OFFSET(intervale;lign;n)))))	*
216	= SET.NAME("pile";lign)	*

	1	2
217	= SET.NAME("lign";300)	*
218	= END.IF()	*
219	= SET.NAME("lign";lign+1)	*
220	=NEXT()	*
221	=SET.NAME("lign";pile)	
222	=WHILE(lign<256)	*On boucle jusqu'à ce qu'on ait retrouvé la dernière donnée d'une série.
223	= IF(ISBLANK(DEREF(OFFSET(intervale;lign;n))))	*
224	= SET.NAME("p";lign)	*
225	= SET.NAME("lign";300)	*
226	= END.IF()	*
227	= SET.NAME("lign";lign+1)	*
228	=NEXT()	
229	=RETURN(p)	*p:position de la dernière donnée d'une série.
230		
231		
232		*Fonction qui effecture une analyse des séries contenues dans une feuille de
233	perform_variability_analysis	*données afin de calculer la variabilité des données.
234	=ARGUMENT("base";2)	*base: indique quelle donnée d'une série on choisit comme base de comparaison.
235	=ARGUMENT("feuille";2)	*feuille: nom de la feuille de donnée.
236	=ARGUMENT("fresult";2)	*frésult : nom de la feuille de résultat.
237	=MESSAGE(TRUE;"WORKING !!!!!! WAIT A FEW SECONDS, PLEASE !!")	
238	=IF(ouverture_fichier()=0)	*ouverture des fichiers.
239	= GOTO(R294C1)	
240	=END.IF()	
241	=effacer_contenu()	*On vide la contenu de la feuille de résultat.
242	=ECHO(FALSE)	
243	=conversion_arguments()	* On convertit les arguments en référence sur lesquelles on pourra travailler.
244	=SET.NAME("n";0)	
245	=format_sheet(feuille;fresult)	* mise en page de la feuille de résultat
246	=FORMULA("index variation between the current figure and the figure first published";OFFSET(sresult;-4;titre))	* Ecriture des titres généraux
247	=FORMULA("percentage %";OFFSET(sresult;-5;col-2))	*
248	=FORMULA("Table 5";OFFSET(sresult;-8;-1))	*
249	=FORMULA.GOTO(OFFSET(sresult;-8;-1);FALSE)	*
250	=FORMAT.FONT("Courier New";14;TRUE;FALSE;FALSE;FALSE;0)	*
251	=SET.NAME("max";1)	*max,min: valeurs minimales et maximales de la feuille des données.
252	=SET.NAME("pourcent";col)	* pourcent: nombre de séries à traiter.
253	=SET.NAME("min";1)	
254	=WHILE(col > 0)	*On boucle sur chaque série à analyser.
255	= affiche_pourcent()	
256	= SET.NAME("m";0)	*m:déplacement (exprimé en nombre de lignes) par rapport à la première ligne
257	= SET.NAME("indice";0)	
258	= WHILE(AND((ISBLANK(OFFSET(intervale;m;n)));m<lig+1))	*On boucle tant que l'on se trouve sur des cellules vides
259	= SET.NAME("m";m+1)	
260	= NEXT()	
261	= WHILE(AND((NOT(ISBLANK(OFFSET(intervale;m;n))));m<lig+1))	*On boucle sur chaque donnée de la série traitée.
262	= SET.NAME("cel";OFFSET(intervale;m;n))	
263	= IF(indice=0)	*Choix de la valeur de base en fonction du paramètre "base".
264	= IF(base="first")	*choix de la première valeur de la série.
265	= SET.NAME("premier";cel)	*
266	= ELSE()	*choix de la dernière valeur de la série.
267	= SET.NAME("p";rech.ref(intervale))	*
268	= SET.NAME("premier";DEREF(OFFSET(intervale;p-1;n)))	*
269	= END.IF()	
270	= END.IF()	

	1	2
271	= PRODUCT((SUM(cel;-(premier)));(1/premier))	
272	= SET.NAME("resultat";SUM(R[-1]C;1))	*résultat: résultat de l'application de la formule d'analyse sur la donnée courante.
273	= FORMULA(resultat;OFFSET(sresult;m;n))	
274	= IF(resultat>max)	*on regarde si la valeur traitée est une valeur minimale ou maximale
275	= SET.NAME("max";resultat)	*de la feuille de calcul et on change les valeurs de min et max si nécessaire.
276	= ELSE()	*
277	= IF(resultat<min)	*
278	= SET.NAME("min";resultat)	*
279	= END.IF()	*
280	= END.IF()	*
281	= SELECT(OFFSET(sresult;m;n))	
282	= FORMAT.NUMBER("0.00")	*On place le résultat sous forme d'un réel à deux décimales après le virgule.
283	= SET.NAME("indice";cel)	*
284	= SET.NAME("m";m+1)	*
285	= NEXT()	*On passe à la donnée suivante.
286	= SET.NAME("n";n+1)	*on passe à la série suivante.
287	= SET.NAME("col";col-1)	*on décrémente le nombre de séries (=colonnes) à analyser
288	=NEXT()	
289	=FORMULA(min;OFFSET(sresult;200;0))	*On renvoit les valeurs maximales et minimales de la feuille.
290	=FORMULA(max;OFFSET(sresult;200;1))	*
291	=FORMULA.GOTO(sresult)	
292	=MESSAGE(FALSE)	
293	=fermer_fichier()	*On ferme les fichiers.
294	=RETURN()	
295		
296		
297		
298	display_table	*Fonction qui affiche une table à l'écran.
299	=ERROR(FALSE)	
300	=ARGUMENT("feuille",2)	*feuille: nom de la table à afficher.
301	=SET.NAME("res";1)	
302	=IF(ERROR.TYPE(OPEN(feuille,3))=3;erreur_fichier())	*Ouverture de la feuille concernée.
303	=IF(res=0)	*
304	= GOTO(R316C1)	*
305	=END.IF()	*
306	=ACTIVATE(feuille)	*Activation de la feuille
307	=ZOOM(65)	*préparation de l'affichage visuel à l'échelle 65%
308	=SET.NAME("fresult";SUBSTITUTE("nomlR1C1";"nom";feuille))	*
309	=SET.NAME("fresult";TEXTREF(fresult;))	*fresult: référence la table concernée.
310	=WINDOW.MAXIMIZE()	*affichage de la feuille à l'écran
311	=FORMULA.GOTO(fresult)	
312	=SPLIT(1,97619047619048)	*On sépare l'écran en deux parties.
313	=PAUSE()	
314	=WINDOW.MINIMIZE(feuille)	*après activation de la pause, on réduit la feuille en un icône.
315	=iconiser()	*
316	=RETURN()	
317		
318		
319		*Fonction qui effecture une analyse qualitative des séries contenues dans une
320	perform_qualitative_analyse	*feuille de données.
321	=ARGUMENT("base",2)	*base: indique quelle donnée d'une série on choisit comme base de comparaison.
322	=ARGUMENT("feuille",2)	*feuille: nom de la feuille de donnée.
323	=ARGUMENT("fresult",2)	*frésult : nom de la feuille de résultat.
324	=RESULT(1)	

	1	2
325	=MESSAGE(TRUE;"WORKING !!!!!! WAIT A FEW SECONDS, PLEASE !!")	*ouverture des fichiers.
326	=IF(ouverture_fichier()=0)	
327	= GOTO(R382C1)	
328	=END.IF()	
329	=effacer_contenu()	*On vide la contenu de la feuille de résultat.
330	=ECHO(FALSE)	
331	=conversion_arguments()	* On convertit les arguments en référence sur lesquelles on pourra travailler.
332	=SET.NAME("n";0)	
333	=format_sheet(feuille;fresult)	* Mise en page de la feuille de résultat
334	=FORMULA("quality";OFFSET(sresult;-5;col-2))	*Ecriture des titres généraux
335	=FORMULA("upwards and downwards variations of the published estimations ";OFFSET(sresult;-4;titre))	*
336	=FORMULA("Table 3";OFFSET(sresult;-8;-1))	*
337	=FORMULA.GOTO(OFFSET(sresult;-8;-1);FALSE)	*
338	=FORMAT.FONT("Courier New";14;TRUE;FALSE;FALSE;FALSE;0)	*
339	=SET.NAME("pourcent";col)	* pourcent: nombre de séries à traiter.
340	=WHILE(col > 0)	*On boucle sur chaque série à analyser.
341	= affiche_pourcent()	* Fonction permettant de visualiser l'état d'avancement de l'analyse
342	= SET.NAME("cpt",1)	* cpt= compteur de longueur pour une série à tendance identique
343	= SET.NAME("premier",0)	
344	= SET.NAME("i",0)	
345	= SET.NAME("m",0)	*m:déplacement (exprimé en nombre de lignes) par rapport à la première ligne
346	= WHILE(AND((ISBLANK(OFFSET(intervale;m;n)));m<lig+1))	* on boucle tant que l'on se trouve sur des cellules vides
347	= SET.NAME("m";m+1)	*
348	= NEXT()	*
349	= WHILE(AND((NOT(ISBLANK(OFFSET(intervale;m;n))));m<lig+1))	*On boucle sur chaque donnée de la série traitée.
350	= SET.NAME("cel";OFFSET(intervale;m;n))	
351	= IF(premier=0)	*si on se trouve la première donnée (ou dernière) de la série, on sauvegarde la
352	= SET.NAME("i";m)	*la référence et on inscrit un 0 ou un " " dans la cellule de la feuille de résultat.
353	= IF(base="first")	*
354	= SET.NAME("deb_ser";m)	*
355	= FORMULA("0";OFFSET(sresult;m;n;1;1))	*
356	= ELSE()	*
357	= SET.NAME("deb_ser";m-1)	*
358	= FORMULA(" ";OFFSET(sresult;m;n;1;1))	*
359	= END.IF()	*
360	= SET.NAME("trends";0)	*
361	= ELSE()	*Si on est sur la deuxième (ou plus) cellule de la ligne, on effectue le calcul.
362	= IF(base="first")	*
363	= SET.NAME("trends";SIGN(cel-premier))	*(trends = variable indiquant la tendance de la cellule analysée)
364	= FORMULA(trends;OFFSET(sresult;m;n;1;1))	*
365	= ELSE()	*
366	= SET.NAME("trends";SIGN(premier-cel))	*
367	= FORMULA(trends;OFFSET(sresult;m-1;n;1;1))	*
368	= END.IF()	*
369	= SET.NAME("i";i+1)	*appel de la fonction qui analyse la longueur de la série à tendance identique
370	= serie()	
371	= END.IF()	
372	= SET.NAME("premier";cel)	
373	= SET.NAME("anc_trends";trends)	*anc_trends = sauvegarde de tendance de la cellule qui vient d'être analysée
374	= SET.NAME("m";m+1)	*on passe à la donnée suivante.
375	= NEXT()	
376	= SET.NAME("n";n+1)	
377	= SET.NAME("col";col-1)	*On passe à la série suivante.
378	=NEXT()	*On décrémente le nombre de séries qu'il reste à analyser

	1	2
379	=FORMULA.GOTO(sresult)	
380	=MESSAGE(FALSE)	
381	=fermer_fichier()	*Fermeture des fichiers qui ne serviront plus dans un avenir immédiat.
382	=RETURN()	
383		
384		
385		
386	serie()	*Fonction qui met en evidence les séries trouvées (selon leur longueur).
387	=IF(cpt=1)	
388	= IF(trends=0)	*définition des couleurs selon les trois tendances possibles
389	= SET.NAME("couleur";10)	*
390	= ELSE()	*
391	= IF(trends=1)	*
392	= SET.NAME("couleur";5)	*
393	= ELSE()	*
394	= IF(trends=-1)	*
395	= SET.NAME("couleur";3)	*
396	= END.IF()	*
397	= END.IF()	*
398	= END.IF()	*
399	=END.IF()	
400	=IF(trends=anc_trends)	*Si la cellule précédemment analysée est de même tendance que celle en court
401	= SET.NAME("cpt";cpt+1)	*d'analyse, alors on incrémente la longueur de la série à tendance identique
402	=ELSE()	*Dans le cas où la cellule analysée est de tendance différente de la cellule précédente
403	= IF(cpt>=3)	*alors on regarde si on a créé une série de longueur au moins égale à 3
404	= FORMULA.GOTO(OFFSET(sresult;deb_ser;n):OFFSET(sresult;deb_ser+cpt-1;n))	*si oui, on la met en evidence en couleur.
405	= PATTERNS(1;couleur;1)	*
406	= SET.NAME("deb_ser";deb_ser+cpt)	*
407	= SET.NAME("cpt";1)	*
408	= ELSE()	*sinon, on recommence le mécanisme en réinitialisant les variables
409	= SET.NAME("deb_ser";deb_ser+cpt)	*
410	= SET.NAME("cpt";1)	*
411	= END.IF()	
412	=END.IF()	
413	=RETURN()	
414		
415		
416		*Fonction qui effecture une analyse quantitative des séries contenues dans une
417	perform_quantitative_analyse	*feuille de données.
418	=ARGUMENT("base";2)	*base: indique quelle donnée d'une série on choisit comme base de comparaison.
419	=ARGUMENT("feuille";2)	*feuille: nom de la feuille de donnée.
420	=ARGUMENT("fresult";2)	*frésult : nom de la feuille de résultat.
421	=RESULT(1)	
422	=MESSAGE(TRUE;"WORKING !!!!!! WAIT A FEW SECONDS, PLEASE !!")	
423	=IF(ouverture_fichier()=0)	*ouverture des fichiers.
424	= GOTO(R481C1)	
425	=END.IF()	
426	=effacer_contenu()	*On vide la contenu de la feuille de résultat.
427	=ECHO(FALSE)	
428	=conversion_arguments()	* On convertit les arguments en référence sur lesquelles on pourra travailler.
429	=SET.NAME("n";0)	
430	=format_sheet(feuille;fresult)	* Mise en page de la feuille de résultat
431	=FORMULA("percentage %";OFFSET(sresult;-5;col-2))	*Ecriture des titres généraux
432	=FORMULA("variations between a figure and its precedent";OFFSET(sresult;-4;titre+1))	*

	1	2
433	=FORMULA("Table 6";OFFSET(sresult;-8;-1))	*
434	=FORMULA.GOTO(OFFSET(sresult;-8;-1);FALSE)	*
435	=FORMAT.FONT("Courier New";14;TRUE;FALSE;FALSE;FALSE;0)	*
436	=SET.NAME("pourcent";col)	* pourcent: nombre de séries à traiter.
437	=WHILE(col > 0)	*On boucle sur chaque série à analyser.
438	= affiche_pourcent()	* Fonction permettant de visualiser l'état d'avancement de l'analyse
439	= SET.NAME("cpt";1)	* cpt= compteur de longueur pour une série à tendance identique
440	= SET.NAME("premier";0)	
441	= SET.NAME("i";0)	
442	= SET.NAME("m";0)	*m:déplacement (exprimé en nombre de lignes) par rapport à la première ligne
443	= WHILE(AND((ISBLANK(OFFSET(intervale;m;n)));m<lig+1))	* on boucle tant que l'on se trouve sur des cellules vides
444	= SET.NAME("m";m+1)	*
445	= NEXT()	*
446	= WHILE(AND((NOT(ISBLANK(OFFSET(intervale;m;n))));m<lig+1))	*On boucle sur chaque donnée de la série traitée.
447	= SET.NAME("cel";OFFSET(intervale;m;n))	
448	= IF(premier=0)	*si on se trouve la première donnée (ou dernière) de la série, on sauvegarde la
449	= SET.NAME("i";m)	*la référence et on inscrit un 0 ou un " " dans la cellule de la feuille de résultat.
450	= IF(base="first")	*
451	= SET.NAME("deb_ser";m)	*
452	= FORMULA("0";OFFSET(sresult;m;n;1;1))	*
453	= ELSE()	*
454	= SET.NAME("deb_ser";m-1)	*
455	= FORMULA(" ";OFFSET(sresult;m;n;1;1))	*
456	= END.IF()	*
457	= SET.NAME("trends";0)	*
458	= ELSE()	*Si on est sur la deuxième (ou plus) cellule de la ligne, on effectue le calcul.
459	= IF(base="first")	*
460	= SET.NAME("trends";SIGN(cel-premier))	*(trends = variable indiquant la tendance de la cellule analysée)
461	= SET.NAME("trends2";((cel-premier)/premier)*100)	*trends2 = variable contenant le résultat du calcul quantitatif
462	= FORMULA(trends2;OFFSET(sresult;m;n;1;1))	*
463	= ELSE()	*
464	= SET.NAME("trends";SIGN(premier-cel))	*
465	= SET.NAME("trends2";((premier-cel)/premier)*100)	*
466	= FORMULA(trends2;OFFSET(sresult;m-1;n;1;1))	*
467	= END.IF()	*
468	= SET.NAME("i";i+1)	
469	= serie()	
470	= END.IF()	
471	= SET.NAME("premier";cel)	
472	= SET.NAME("anc_trends";trends)	*anc_trends = sauvegarde de tendance de la cellule qui vient d'être analysée
473	= SET.NAME("m";m+1)	
474	= NEXT()	*on passe à la donnée suivante.
475	= SET.NAME("n";n+1)	
476	= SET.NAME("col";col-1)	
477	=NEXT()	*On passe à la série suivante.
478	=FORMULA.GOTO(sresult)	*On décrémente le nombre de séries qu'il reste à analyser
479	=MESSAGE(FALSE)	
480	=fermer_fichier()	*Fermeture des fichiers qui ne serviront plus dans un avenir immédiat.
481	=RETURN()	
482		
483		
484	affiche_pourcent	*fonction qui permet l'état d'avancement d'un processus : affichage dans la barre d'état
485	=SUM(100;-(PRODUCT(100;PRODUCT(col;1/pourcent))))	*du pourcentage de travail effectué.
486	=ROUND(R[-1]C;0)	

	1	2
487	=MESSAGE(TRUE;SUBSTITUTE("percent %";"percent";R[-1]C))	
488	=RETURN()	
489		
490		
491		
492	conversion_arguments	*Fonction qui convertit les noms des feuilles données comme paramètres
493	=SET.NAME("intervale";SUBSTITUTE("nomlR9C3:nomlR9C256","nom";feuille))	* aux fonctions en référence vers ces feuilles.
494	=SET.NAME("intervale";TEXTREF(intervale;))	
495	=SET.NAME("col";COUNTA(intervale))	*col : nombre de périodes d'éditions à traiter.
496	=SET.NAME("intervale";SUBSTITUTE("nomlR10C2:nomlR256C2","nom";feuille))	
497	=SET.NAME("intervale";TEXTREF(intervale;))	
498	=SET.NAME("lig";COUNTA(intervale))	*lig: nombre de séries à traiter.
499	=SET.NAME("intervale";SUBSTITUTE("nomlR10C3";"nom";feuille))	
500	=SET.NAME("intervale";TEXTREF(intervale;))	*intervale: référence la première cellule pouvant contenir une donnée réelles
501	=SET.NAME("sresult";SUBSTITUTE("nomlR10C3";"nom";fresult))	*sur la feuille de données.
502	=SET.NAME("sresult";TEXTREF(sresult;))	*sresult: référence la première cellule de la feuille de résultats pouvant
503	=RETURN()	*contenir un résultat réel.
504		
505		
506		
507	ouverture_fichier()	*Fonction qui effectue l'ouverture des feuilles.
508	=RESULT(1)	
509	=ERROR(FALSE)	
510	=SET.NAME("res";1)	
511	=IF(ERROR.TYPE(OPEN(feuille;3))=3;erreur_fichier())	Si le fichier de sortie n'existe pas, on le crée
512	=ERROR(FALSE)	*
513	=IF(ERROR.TYPE(OPEN(fresult;3))=3;creer_fichier())	*
514	=ERROR(TRUE)	*
515	=iconiser()	
516	=RETURN(res)	
517		
518		
519		
520	iconiser()	*Fonction qui met les feuilles ouvertes sous forme d'icones.
521	=SET.NAME("docs";DOCUMENTS())	
522	=FOR("counter";1;COLUMNS(docs))	
523	= SET.NAME("fenetre";INDEX(docs;counter))	
524	= ACTIVATE(fenetre)	
525	= WINDOW.MINIMIZE(fenetre)	
526	=NEXT()	
527	=RETURN()	
528		
529		
530		
531	fermer_fichier()	*Fonction qui ferme les feuilles qui ne seront plus utilisées dans un avenir immédiat
532	=SET.NAME("docs";DOCUMENTS())	
533	=FOR("counter";1;COLUMNS(docs))	
534	= SET.NAME("fenetre";INDEX(docs;counter))	
535	= ACTIVATE(fenetre)	
536	= IF(OR(fenetre="printing.xlm";fenetre="analyse.xlm";fenetre="graphe2.xlm";fenetre="main_pr.xlm";fenetre="graphe1.xlm";fenetre="graphe3.xlm";fenetre=	
537	= GOTO(R542C1)	
538	= ELSE()	
539	= SAVE()	

	1	2
540	= END.IF()	
541	= CLOSE()	
542	=NEXT()	
543	=RETURN()	
544		
545		
546		
547	effacer_contenu()	*Fonction qui effece le contenu d'une feuille.
548	=ACTIVATE(fresult)	
549	=SELECT("R1:R16384","R1C3")	
550	=EDIT.DELETE(2)	
551	=RETURN()	
552		
553		
554	erreur_fichier()	*Fonction qui prévient l'utilisateur que le fichier d'entrée qu'il a fourni n'existe pas
555	=ALERT("the 'input-sheet' does not exist !!!!")	*dans le répertoire courant.
556	=ERROR(TRUE)	
557	=SET.NAME("res",0)	
558	=fermer_fichier()	
559	=RETURN()	
560		
561		
562		
563	creer_fichier()	*Fonction qui crée une nouvelle table.
564	=NEW(1)	
565	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
566	=RETURN()	
567		
568		

	1	2	3	4	5	6	7
1				840	400		
2	5	330	15			***ANALYSE OF TIME SERIE***	
3	5	15	380				
4	14	20	35	700	155	AVAILABLE FUNCTIONS	
5	115	120	60	250	100	R31C3:R43C3	6
6	215	460	73	232	88	R45C3:R51C3	5
7	14	20	200	600	50	SHEETS	
8	6	140	220	160			a2.xls
9	14	20	310	600	80	ENTER THE PERIODS	
10	14	20	250	600	50	BASE VALUE	
11	1	660	220	125	30	OK	
12	2	660	275	125	30	Cancel	
13	24	660	330	125	30	Help	
14	6	430	220	160			
15	5	460	58			TYPES OF GRAPHICS	
16	5	40	220			Input sheet:	
17	5	320	220			Output sheet:	
18	5	80	330			YEAR1:	
19	5	35	350			QUARTER1:	
20	5	370	330			YEAR2:	
21	5	325	350			QUARTER2:	
22	206	140	330	150			1987
23	206	140	350	150			2
24	206	430	330	150			1988
25	206	430	350	150			4
26	11	80	320				2
27	12	120	270			First value	
28	12	260	270			Last value	
29							
30	=DIALOG.BOX(dialogue)						
31	=RETURN()		Qualitative analyse				
32			Quantitative analyse				
33			Proportional analyse				
34			Difference analyse				
35			Variability analyse				
36			Display a table				
37			Create a standard graphic				
38			Create variability graphic				
39			Create ratio graphic				
40			Print a sheet				
41			Print a standard chart				
42			Print a variability chart				
43			Print a ratio chart				

	1	2	3
44	programme.principal		
45	=OPEN("graphe1.xlm")		Two series histogram
46	=OPEN("graphe2.xlm")		One serie histogram
47	=OPEN("graphe3.xlm")		One serie line
48	=OPEN("graphe4.xlm")		Two series lines
49	=OPEN("analyse.xlm")		One serie column
50	=OPEN("printing.xlm")		Two series column
51	='C:\WINDOWS\EXCEL\ANALYSE.XLM'!iconiser()		Several series column
52	=demonstration()		
53	=fermer_fichier()		
54	=RETURN()		
55			
56			
57			
58			
59	demonstration		
60	*initialisation de la boîte de dialogue.	* initialisation de la boître de dialogue.	
61	= FORMULA(1;R5C7)		
62	= FORMULA("";R8C7)		
63	= FORMULA("";R14C7)		
64	= SET.NAME("cont";1)	* cont = 1 tant que l'utilisateur n'a pas choisi la fonction OK	
65		* ou CANCEL dans la boîte de dialogue.	
66	= WHILE(cont = 1)		
67	= IF(OR(R5C7=1;R5C7=2;R5C7=3;R5C7=4;R5C7=5))	*	
68	= FORMULA(215;R6C1)	*	
69	= FORMULA(106;R14C1)	*	
70	= FORMULA(105;R17C1)	*	
71	= FORMULA(206;R22C1)	*	
72	= FORMULA(206;R23C1)	*	
73	= FORMULA(206;R24C1)	*	
74	= FORMULA(206;R25C1)	*	
75	= ELSE()	*	
76	= IF(OR(R5C7=6;R5C7=8;R5C7=9))	*	
77	= FORMULA(215;R6C1)	*	
78	= FORMULA(6;R14C1)	*	
79	= FORMULA(5;R17C1)	*	
80	= FORMULA(206;R22C1)	*	
81	= FORMULA(206;R23C1)	*	
82	= FORMULA(206;R24C1)	*	
83	= FORMULA("";R14C7)	*	
84	= FORMULA(206;R25C1)	*	
85	= ELSE()	*	
86	= IF(R5C7=7)	*	
87	= FORMULA(115;R6C1)	*	
88	= FORMULA(6;R14C1)	*	
89	= FORMULA(5;R17C1)	*	
90	= FORMULA(106;R22C1)	*	
91	= FORMULA("";R14C7)	*	
92	= FORMULA(106;R23C1)	*	
93	= IF(OR(R6C7=1;R6C7=4;R6C7=6;R6C7=7))	*	
94	= FORMULA(6;R24C1)	*	
95	= FORMULA(6;R25C1)	*	
96	= ELSE()	*	
97	= FORMULA(206;R24C1)	*	

	1	2	3
98	= FORMULA(206;R25C1)	*	
99	= END.IF()	* GESTION DE LA BOÎTE DE DIALOGUE.	
100	= IF(R6C7=7)	*	
101	= FORMULA("ENTER THE RANGE OF PERIODS:";R9C6)	*	
102	=ELSE()	*	
103	=FORMULA("ENTER THE PERIODS";R9C6)	*	
104	=END.IF()	*	
105	= END.IF()	*	
106	= END.IF()	*	
107	= END.IF()	*	
108	= FORMULA(DIALOG.BOX(dialogue);R109C1)	*	
109	FALSE	*	
110	= IF(OR(R109C1 = 10;R109C1 = FALSE))	*	
111	= SET.NAME("cont";0)	*	
112	= END.IF()	*	
113	= NEXT()	*	
114			
115	= IF(R109C1=10)	* R109c1 = 10 lorsque l'utilisateur a choisi la fonction OK.	
116	= IF(R26C7=1)		
117	= SET.NAME("base","first")	* base: permet de choisir la période de base pour les comparaison	
118	= ELSE()	* de séries.	
119	= SET.NAME("base","last")	* base = "first" : la première période est la période de base.	
120	= END.IF()	* base = "last": la dernière période est la période de base.	
121	= IF(verifier.fichier.entree()=1)		
122	= GOTO(R64C1)		
123	= END.IF()		
124	= IF(OR(R5C7=1;R5C7=2;R5C7=3;R5C7=4;R5C7=5))	* R5C7 = 1 à 5: la fonction choisie est une analyse.	
125	= IF(verifier.fichier.sortie()=1)	* dans ce cas, on vérifie qu'un fichier de résultat a été entré.	
126	= GOTO(R64C1)		
127	= END.IF()		
128	= END.IF()		
129	=IF(OR(R5C7=7;R5C7=8;R5C7=9))	*R5C7 = 7 à 9: signifie que la fonction choisie est un graphique.	
130	=SET.NAME("res";traiter.resultat())		
131	=END.IF()		
132	= IF(R5C7=1)	* suivant la valeur de R5C7, on lance l'exécution de la fonction	
133	= 'C:\WINDOWS\EXCEL\ANALYSE.XLM'!perform_qualitative_analyse(base;DEREF(R8C7);DEREF(R14C7))	* correspondante.	
134	= ELSE()		
135	= IF(R5C7=2)		
136	= 'C:\WINDOWS\EXCEL\ANALYSE.XLM'!perform_quantitative_analyse(base;DEREF(R8C7);DEREF(R14C7))		
137	= ELSE()		
138	= IF(R5C7=3)		
139	= 'C:\WINDOWS\EXCEL\ANALYSE.XLM'!perform_proportional_analysis(base;DEREF(R8C7);DEREF(R14C7))		
140	= ELSE()		
141	= IF(R5C7=4)		
142	= 'C:\WINDOWS\EXCEL\ANALYSE.XLM'!perform_difference_analysis(base;DEREF(R8C7);DEREF(R14C7))		
143	= ELSE()		
144	= IF(R5C7=5)		
145	= 'C:\WINDOWS\EXCEL\ANALYSE.XLM'!perform_variability_analysis(base;DEREF(R8C7);DEREF(R14C7))		
146	= ELSE()		
147	= IF(R5C7=6)		
148	= 'C:\WINDOWS\EXCEL\ANALYSE.XLM'!display_table(DEREF(R8C7))		
149	= ELSE()		
150	= IF(R5C7 = 7)		
151	= lancer_graphe(res)		

	1	2	3
152	= ELSE()		
153	= IF(R5C7=8)		
154	= 'C:\WINDOWS\EXCEL\GRAPHE2.XLM'!graphe.ratio(DEREF(R8C7);res)		
155	= ELSE()		
156	= IF(R5C7=9)		
157	= 'C:\WINDOWS\EXCEL\GRAPHE3.XLM'!graphe.ratio(DEREF(R8C7);res)		
158	= ELSE()		
159	= IF(R5C7=10)		
160	= 'C:\WINDOWS\EXCEL\PRINTING.XLM'!print_table(DEREF(R8C7))		
161	= ELSE()		
162	= IF(R5C7=11)		
163	= 'C:\WINDOWS\EXCEL\PRINTING.XLM'!print_chart(DEREF(R8C7))		
164	= ELSE()		
165	= IF(R5C7=12)		
166	= 'C:\WINDOWS\EXCEL\PRINTING.XLM'!print_vari(DEREF(R8C7))		
167	= ELSE()		
168	= IF(R5C7=13)		
169	= 'C:\WINDOWS\EXCEL\PRINTING.XLM'!print_ratio(DEREF(R8C7))		
170	= END.IF()		
171	= END.IF()		
172	= END.IF()		
173	= END.IF()		
174	= END.IF()		
175	= END.IF()		
176	= END.IF()		
177	= END.IF()		
178	= END.IF()		
179	= END.IF()		
180	= END.IF()		
181	= END.IF()		
182	= END.IF()		
183	= GOTO(R64C1)		
184	= END.IF()		
185	= IF(R109C1=FALSE)	* R109C1 = False: l'utilisateur a choisi la fonction CANCEL.	
186	= GOTO(R189C1)		
187	= END.IF()		
188			
189	= RETURN()		
190			
191			
192			
193			
194			
195	lancer_graphe	* Fonction qui gère la création d'un graphique.	
196	=ARGUMENT("fresult";2)	* Fresult: feuille qui contiendra le graphique à créer.	
197	= SET.NAME("type";DEREF(R6C7))	* type: numéro de fonction choisie par l'utilisateur.	
198	= SET.NAME("an1";DEREF(R22C7))	* an1: Année de la 1° série choisie.	
199	= SET.NAME("quart1";DEREF(R23C7))	* quart1: trimestre de la 1° série choisie.	
200	= SET.NAME("an2";DEREF(R24C7))	* an2: Année de la deuxième série.	
201	= SET.NAME("quart2";DEREF(R25C7))	* quart2: Quart de la seconde série choisie.	
202	= IF(OR(OR(ISBLANK(R22C7);ISBLANK(R23C7));AND(OR(type=1;type=4);(OR(ISBLANK(R24C7);ISBLANK(R25C7)))))	* Vérification que la (ou les) série(s) choisie(s)	
203	= ALERT("A Criteria is missing.";3)	* ont bien été entrée par l'utilisateur.	
204	= GOTO(R221C1)		
205	= END.IF()		

	1	2	3
206	= IF(OR(type=1;type=4;type=6))		
207	= IF(OR(an1>an2;AND(an1=an2;quart1>quart2)))	* Lorsque l'on traite deux séries, ou un intervale de séries,	
208	= SET.NAME("arg";an1)	* si la première série entrée est postérieure à la seconde,	
209	= SET.NAME("an1";an2)	* on intervertit les deux séries.	
210	= SET.NAME("an2";arg)		
211	= SET.NAME("arg";quart1)		
212	= SET.NAME("quart1";quart2)		
213	= SET.NAME("quart2";arg)		
214	= END.IF()		
215	= END.IF()		
216	= IF(OR(type=5;type=6;type=7))	* On lance l'exécution de la fonction correspondant au	
217	= 'C:\WINDOWS\EXCEL\GRAPHE4.XLM'!gestion.graphique(DEREF(R8C7);type;an1;quart1;an2;quart2;fresult)	* graphique choisi.	
218	=ELSE()		
219	= 'C:\WINDOWS\EXCEL\GRAPHE1.XLM'!gestion.graphique(DEREF(R8C7);type;an1;quart1;an2;quart2;fresult)		
220	=END.IF()		
221	= RETURN()		
222			
223			
224			
225	1		
226			
227			
228	fermer_fichier()	* Fonction qui effectue la fermeture de toute les feuilles	
229	= SET.NAME("docs",DOCUMENTS())	* ouvertes à l'exception de la feuille main_pr.xlm	
230	= FOR("counter";1;COLUMNS(docs))		
231	= SET.NAME("fenetre",INDEX(docs;counter))		
232	= ACTIVATE(fenetre)		
233	= IF(fenetre="main_pr.xlm")		
234	= GOTO(R239C1)		
235	= ELSE()		
236	= SAVE()		
237	= END.IF()		
238	= CLOSE()		
239	= NEXT()		
240	= RETURN()		
241			
242			
243			
244			
245	verifier.fichier.entree	* Fonction qui vérifie qu'un nom de fichier d'entrée	
246	= RESULT(1)	* a bien été donné par l'utilisateur.	
247	= SET.NAME("res";0)		
248	= IF(!ISBLANK(R8C7))		
249	= ALERT("file does not exist";3)		
250	= SET.NAME("res";1)		
251	= END.IF()		
252	= RETURN(res)		
253			
254			
255			
256	verifier.fichier.sortie	* Fonction qui vérifie qu'un nom de fichier de résultat a bien	
257	= RESULT(1)	* été donné par l'utilisateur.	
258	= SET.NAME("res";0)		
259	= IF(ISBLANK(R14C7))		

	1	2	3
260	= ALERT("file does not exist";3)		
261	= SET.NAME("res";1)		
262	= END.IF()		
263	= RETURN(res)		
264			
265			
266			
267	traiter.resultat	* Fonction qui vérifie qu'un fichier de résultat a bien été	
268	=RESULT(2)	* donné par l'utilisateur, sinon la fonction en donne un	
269	=IF(ISBLANK(R14C7))	* par défaut.	
270	= SET.NAME("res";"chart1.xlc")		
271	=ELSE()		
272	= SET.NAME("res";DEREF(R14C7))		
273	=END.IF()		
274	=RETURN(res)		
275			
276			

	1	2
1	print_table	*Fonction qui effectue l'impression d'une feuille donnée.
2	=ERROR(FALSE)	
3	=ARGUMENT("feuille",2)	*Feuille: nom de la feuille à imprimer.
4	=SET.NAME("res",1)	*res: indicateur qui indique si l'ouverture de la feuille s'est bien déroulée.
5	=IF(ERROR.TYPE(OPEN(feuille))=3;erreur_fichier())	*ouverture de la feuille à imprimer
6	=IF(res=0)	
7	= GOTO(R32C1)	
8	=END.IF()	
9	=ACTIVATE(feuille)	*activation de la feuille
10	=SET.NAME("intervale";SUBSTITUTE("nom!R9C3:nom!R9C256";"nom";feuille))	
11	=SET.NAME("intervale";TEXTREF(intervale;))	
12	=SET.NAME("col";COUNTA(intervale))	*col:Nombre de colonne à imprimer.
13	=SET.NAME("intervale";SUBSTITUTE("nom!R10C2:nom!R256C2";"nom";feuille))	
14	=SET.NAME("intervale";TEXTREF(intervale;))	
15	=SET.NAME("lig",COUNTA(intervale))	*lig:nombre de colonnes à imprimer.
16	=SET.NAME("taille",MIN(INT(1400/(col+2));INT(3600/(lig+14))))	*taille:taille en pourcent de la taille maximale de la feuille à imprimer.
17	=IF(taille>100)	*
18	= SET.NAME("taille",100)	*
19	=END.IF()	*
20	=IF(taille<50)	*
21	= SET.NAME("taille",50)	*
22	=END.IF()	*
23	=SET.NAME("sel";SUBSTITUTE("R1C1:R&C$";"&";lig+14))	
24	=SET.NAME("sel";SUBSTITUTE(sel;"$";col+2))	*sel:représente les cellules à imprimer.
25	=SELECT(sel)	*On selectionne les cellules à imprimer.
26	=SET.PRINT.AREA()	
27	=iconiser()	
28	=ACTIVATE(feuille)	
29	=PAGE.SETUP("&F","Page &P";0,748031496062992;0,748031496062992;0,984251968503937;0,984251968503937;FALSE;TRUE;FALSE;FALSE;2;9;taille;	*On effectue une mise en page
30	=PRINT(1)	*On lance l'impression.
31	=fermer_fichier()	
32	=RETURN()	
33		
34		
35	iconiser()	*Fonction qui met les fichiers ouverts sous forme d'icones.
36	=SET.NAME("docs";DOCUMENTS())	
37	=FOR("counter";1;COLUMNS(docs))	
38	= SET.NAME("fenetre";INDEX(docs;counter))	
39	= ACTIVATE(fenetre)	
40	= WINDOW.MINIMIZE(fenetre)	
41	=NEXT()	
42	=RETURN()	
43		
44		
45	fermer_fichier()	*Fonction qui effectue la fermeture de tous les fichiers que l'on utilisera
46	=SET.NAME("docs";DOCUMENTS())	*plus dans un avenir immédiat.
47	=FOR("counter";1;COLUMNS(docs))	
48	= SET.NAME("fenetre";INDEX(docs;counter))	
49	= ACTIVATE(fenetre)	
50	= IF(OR(fenetre="printing.xlm";fenetre="analyse.xlm";fenetre="graphe2.xlm";fenetre="main_pr.xlm";fenetre="graphe1.xlm";fenetre="graphe3.xlm";fenetre='	
51	= GOTO(R56C1)	
52	= ELSE()	
53	= SAVE()	
54	= END.IF()	

	1	2
55	= CLOSE()	
56	=NEXT()	
57	=RETURN()	
58		
59		
60		
61	effacer_contenu()	*Fonction qui efface le contenu d'un fichier existant
62	=ACTIVATE(fresult)	
63	=SELECT("R1:R16384","R1C3")	
64	=EDIT.DELETE(2)	
65	=RETURN()	
66		
67		
68	erreur_fichier()	*Fonction qui indique à l'uilisateur que le fichier qu'il a entré comme paramètre
69	=ALERT("the 'input-sheet' does not exist !!!!")	*n'existe pas dans le répertoire courant.
70	=ERROR(TRUE)	
71	=SET.NAME("res";0)	
72	=fermer_fichier()	
73	=RETURN()	
74		
75		
76		
77	ouverture_fichier()	*Fonction qui effectue l'ouverture d'un fichier donné.
78	=RESULT(1)	
79	=ERROR(FALSE)	
80	=SET.NAME("res";1)	
81	=IF(ERROR.TYPE(OPEN(feuille))=3;erreur_fichier())	
82	=ERROR(FALSE)	
83	=iconiser()	
84	=RETURN(res)	
85		
86		
87	print_chart	*Fonction qui effectue l'impression d'un graphique.
88	=ARGUMENT("feuille";2)	*Feuille: nom de la feuille qui contient le graphique.
89	=SET.NAME("res";1)	
90	=IF(ERROR.TYPE(OPEN(feuille;3))=3;erreur_fichier())	*Ouverture de la feuille.
91	=IF(res=0)	
92	= GOTO(R102C1)	
93	=END.IF()	
94	=ERROR(FALSE)	
95	=ACTIVATE(feuille)	*activation de la feuille
96	=PAGE.SETUP("&F","Page &P";0,748031496062992;0,748031496062992;0,984251968503937;0,984251968503937;2;FALSE;FALSE;2;9;100;1)	*mise en page.
97	=SELECT("Chart")	*sélection du graphique.
98	=PATTERNS(2;1;1;1;FALSE;0;1;2;5;FALSE)	*élaboration de la présentation du graphique
99	=PRINT(1)	*impression
100	=iconiser()	
101	=fermer_fichier()	
102	=RETURN()	
103		
104		
105	print_vari	*Fonction qui effectue l'impression d'un graphique de "variabilité" donné.
106	=ERROR(FALSE)	
107	=ARGUMENT("feuille";2)	*feuille:nom de la feuille à imprimer.

	1	2
108	=SET.NAME("res";1)	*rés: indique si l'ouverture de la feuille s'est bien déroulée ou non.
109	=IF(ERROR.TYPE(OPEN(feuille))=3;erreur_fichier())	
110	=IF(res=0)	
111	= GOTO(R126C1)	
112	=END.IF()	
113	=SET.NAME("intervale";SUBSTITUTE("nomIR2C1";"nom";feuille))	*en nomIR2C1, on trouve la coordonnée maximale du graphique sur
114	=SET.NAME("intervale";TEXTREF(intervale;))	*l'axe des X.
115	=SET.NAME("long";DEREF(intervale))	
116	=SET.NAME("long";INT(long/47,6)+1)	*long: nombre de cellules à imprimer.
117	=SET.NAME("sel";SUBSTITUTE("r1c1:r1c$";"$";long))	*sel: référence les cellules à imprimer.
118	=ACTIVATE(feuille)	
119	=SELECT(sel)	*sélection des cellules à imprimer
120	=SET.PRINT.AREA()	
121	=iconiser()	
122	=ACTIVATE(feuille)	*activation de la feuille
123	=PAGE.SETUP("&F","Page &P";0,748031496062992;0,748031496062992;0,984251968503937;0,984251968503937;FALSE;TRUE;FALSE;FALSE;2;9;100;1	*mise en page générale (numéro de page,...)
124	=PRINT(1)	
125	=fermer_fichier()	
126	=RETURN()	
127		
128		
129	print_ratio	*Fonction qui effectue l'impression d'un graphique de "variabilité" donné.
130	=ERROR(FALSE)	
131	=ARGUMENT("feuille";2)	*feuille:nom de la feuille à imprimer.
132	=SET.NAME("res";1)	*rés: indique si l'ouverture de la feuille s'est bien déroulée ou non.
133	=IF(ERROR.TYPE(OPEN(feuille))=3;erreur_fichier())	
134	=IF(res=0)	
135	= GOTO(R155C1)	
136	=END.IF()	
137	=SET.NAME("intervale";SUBSTITUTE("nomIR200C1";"nom";feuille))	*en nomIR200C1, on trouve la coordonnée maximale du graphique sur
138	=SET.NAME("intervale";TEXTREF(intervale;))	*l'axe des X.
139	=SET.NAME("haut";DEREF(intervale))	
140	=SET.NAME("haut";INT(haut/12)+1)	
141	=SET.NAME("intervale";SUBSTITUTE("nomIR200C2";"nom";feuille))	
142	=SET.NAME("intervale";TEXTREF(intervale;))	
143	=SET.NAME("long";DEREF(intervale))	
144	=SET.NAME("long";INT(long/47,6)+1)	*long: nombre de cellules à imprimer.
145	=SET.NAME("sel";SUBSTITUTE("r1c1:rùc$";"$";long))	*sel: référence les cellules à imprimer.
146	=SET.NAME("sel";SUBSTITUTE(sel;"ù";haut))	
147	=ACTIVATE(feuille)	
148	=SELECT(sel)	
149	=SET.PRINT.AREA()	
150	=iconiser()	
151	=ACTIVATE(feuille)	
152	=PAGE.SETUP("&F","Page &P";0,748031496062992;0,748031496062992;0,984251968503937;0,984251968503937;FALSE;TRUE;FALSE;FALSE;2;9;60;1;	
153	=PRINT.PREVIEW()	
154	=fermer_fichier()	
155	=RETURN()	
156		

	1	2
1	=gestion.graphique("a5.xls";1;1988;2;1989;2)	
2	gestion.graphique	*Fonction qui crée un graphique à partir de séries données.
3	=ARGUMENT("feuille";2)	*feuille: nom de la feuille qui contient les données.
4	=ARGUMENT("type";1)	*Type: type du graphique désiré (7 types différents).
5	=ARGUMENT("an1";1)	*an1:année de la première série.
6	=ARGUMENT("quart1";1)	*an2: année de la seconde série.
7	=ARGUMENT("an2";1)	*quart1: trimestre de la première série.
8	=ARGUMENT("quart2";1)	*quart2:trimestre de la seconde série.
9	=ARGUMENT("fresult";2)	*fresult: nom de la feuille qui contiendra le graphique.
10	=ouverture_fichier()	*
11	=IF(OR(type=1;type=5;type =4;type = 8))	* double = 1 indique que l'on affiche une seule série.
12	= SET.NAME("double";1)	*
13	= SET.NAME("ref1";rech.ref(feuille;an1;quart1;type;2;an2;quart2))	*
14	=ELSE()	*
15	= SET.NAME("ref1";rech.ref(feuille;an1;quart1;type;1;an2;quart2))	
16	= SET.NAME("double";0)	* double = 2 indique que l'on va afficher 2 séries.
17	=END.IF()	
18	=SET.NAME("titre.x";axe.x(type))	* Titre.x: référence les labels à placer sur l'axe des ordonnées.
19	=IF(OR(type=1;type=2;type=5;type=6))	
20	= SET.NAME("type";11)	* type = 11 indique que l'on crée un graphique de type batonnet
21	=ELSE()	* en trois dimensions.
22	= SET.NAME("type";12)	
23	=END.IF()	* type = 12 indique que l'on crée un graphique de type linéaire
24	=creation.graphique(ref1;type;an1;quart1;an2;quart2;double;titre.x)	* en trois dimensions.
25	=PAUSE()	
26	=WINDOW.MINIMIZE()	
27	=fermer_fichier()	
28	=RETURN()	
29		
30		
31		
32	creation.graphique	* Fonction qui gère le tracé du graphique.
33	=ECHO(FALSE)	
34	=ARGUMENT("gr";8)	* gr: référence les valeurs des séries.
35	=ARGUMENT("type";1)	
36	=ARGUMENT("an1";1)	
37	=ARGUMENT("quart1";1)	
38	=ARGUMENT("an2";1)	
39	=ARGUMENT("quart2";1)	
40	=ARGUMENT("double";1)	* double: indique le nombre de séries à afficher sur le graphique.
41	=ARGUMENT("titre.x";8)	* titre.x: référence les labels à afficher sur l'axe des abcisses.
42	=ACTIVATE(fresult)	
43	=CREATE.OBJECT(5;"R1C4";12;0,75;"R16C11";0;0,75;1;TRUE)	* création d'un objet de type graphique.
44	=SET.NAME("label1";titrage(1))	* label1: Titre à afficher sur le graphique.
45	=SET.NAME("label2";titrage(2))	* label2: Titre de l'axe des abcisses.
46	=CHART.WIZARD(TRUE;gr;type;5;2;2;2;1;label1;"QUARTERS";label2;"")	* Fonction de création de graphique.
47	=SELECT("Axis 1")	
48	=SCALE(TRUE;TRUE;TRUE;TRUE;TRUE;FALSE;FALSE;FALSE)	* Fonction de calcul des échelles des axes.
49	=SELECT("Title")	
50	=FORMAT.FONT(0;1;FALSE;"MS Sans Serif";14;FALSE;FALSE;TRUE;FALSE)	* Fonction de mise en page du titre.
51	=SELECT("Plot")	
52	=VIEW.3D(22;30;33;FALSE;50;TRUE)	* Fonction de mise au point de la vue en trois dimensions.
53	=GRIDLINES(TRUE;FALSE;TRUE;FALSE;TRUE;FALSE)	* Fonction de gestion de grillages sur le graphique.
54	=SELECT("Text Axis 1")	

	1	2
55	=FORMAT.TEXT(2;2;0;FALSE;TRUE)	*Fonction de traitement du texte des labels de l'axe x
56	=SELECT("Chart")	
57	=PATTERNS(2;1;1;1;FALSE;0;3;2;6;FALSE)	* Fonction de gestion de la couleur du graphique.
58	=SELECT("Axis 2")	*
59	=AXES(TRUE;FALSE;TRUE)	*
60	=FORMAT.TEXT(;;0)	*
61	=SET.NAME("legende1";"0000 Qs")	*
62	=SET.NAME("legende2";"0000 Qs")	*
63	=SUBSTITUTE(legende1,"0000";an1)	*
64	=SET.NAME("leg1";R[-1]C)	*
65	=SUBSTITUTE(leg1;"s";quart1)	*
66	=EDIT.SERIES(1;R[-1]C;titre.x)	* GESTION DE LA LEGENDE
67	=IF(double = 1)	*
68	= SUBSTITUTE(legende2,"0000";an2)	*
69	= SET.NAME("leg2";R[-1]C)	*
70	= SUBSTITUTE(leg2;"s";quart2)	*
71	= EDIT.SERIES(2;R[-1]C)	*
72	=END.IF()	*
73	=SELECT("Text Axis 1")	
74	=FORMAT.TEXT(2;2;3;FALSE;TRUE)	* Fonction de gestion du texte sous laxe des X.
75	=WINDOW.MAXIMIZE()	
76	=RETURN()	
77		
78		
79	rech.ref	* Fonction de recherche des références des valeurs des séries.
80	=RESULT(8)	
81	=ARGUMENT("feuille";2)	
82	=ARGUMENT("an";1)	
83	=ARGUMENT("quart";1)	
84	=ARGUMENT("type";1)	
85	=ARGUMENT("nr";1)	
86	=conversion_arguments()	
87	=SET.NAME("nr";nr)	* nr: nombre de séries à afficher.
88	=SET.NAME("n";(an-depart))	
89	=SET.NAME("n";(n*4)+(quart-1))	* n: position de la séries dans la feuille de données
90	=FOR("i";1;nr)	
91	=SET.NAME("lig";0)	
92	=WHILE(lig<256)	
93	=IF(NOT(ISBLANK(DEREF(OFFSET(debut_chiffre;lig;n)))))	* Recherche de la référence de la première valeur d'une série.
94	=SET.NAME("m";lig)	
95	=SET.NAME("lig";300)	
96	=END.IF()	
97	=SET.NAME("lig";lig+1)	
98	=NEXT()	
99	=SET.NAME("lig";m)	
100	=WHILE(lig<256)	
101	=IF(ISBLANK(DEREF(OFFSET(debut_chiffre;lig;n))))	* Recherche de la référence de la dernière valeur d'une série.
102	=SET.NAME("p";lig)	
103	=SET.NAME("lig";300)	
104	=END.IF()	
105	=SET.NAME("lig";lig+1)	
106	=NEXT()	
107	=IF(AND(nr=2;i=1))	* si le nombre de séries est égal à deux, on change
108	=SET.NAME("n1";n)	* la valeur des paramètres de travail, pour travailler

	1	2
109	=ARGUMENT("an";1)	* sur la seconde série.
110	=ARGUMENT("quart";1)	
111	=SET.NAME("m1";m)	
112	=SET.NAME("p1";p)	
113	=SET.NAME("n";(an-depart))	
114	=SET.NAME("n";(n*4)+(quart-1))	
115	=END.IF()	
116	=IF(nr>2)	* Si on travail sur plus de deux séries, on concatène les références
117	=IF(i=1)	* des différentes séries à chjaque boucle.
118	=SET.NAME("m1";m)	
119	=SET.NAME("p1";p)	
120	=SET.NAME("ref";OFFSET(debut_chiffre;m;n):OFFSET(debut_chiffre;p;n))	
121	=ELSE()	
122	=SET.NAME("m1";MIN(m1;m))	
123	=SET.NAME("p1";MAX(p;p1))	
124	=SET.NAME("n1";n)	
125	=SET.NAME("ref1";OFFSET(debut_chiffre;m1;n1):OFFSET(debut_chiffre;p1;n1))	
126	=SET.NAME("ref";(ref;ref1))	
127	=END.IF()	
128	=SET.NAME("n";n+1)	
129	=END.IF()	
130		
131	=NEXT()	
132	=IF(nr=2)	* Si on travail sur deux séries, on concatène les références de ces
133	=SET.NAME("m";MIN(m1;m))	* deux séries à la sortie de la boucle.
134	=SET.NAME("p";MAX(p;p1))	
135	=SET.NAME("ref";OFFSET(debut_chiffre;m;n1):OFFSET(debut_chiffre;p;n1))	
136	=SET.NAME("ref1";OFFSET(debut_chiffre;m;n):OFFSET(debut_chiffre;p;n))	
137	=SET.NAME("ref";(ref;ref1))	
138	=ELSE()	
139	=IF(nr=1)	
140	=SET.NAME("ref";OFFSET(debut_chiffre;m;n):OFFSET(debut_chiffre;p;n))	
141	=ELSE()	
142	=SET.NAME("p";p1)	
143	=SET.NAME("m";m1)	
144	=END.IF()	
145	=END.IF()	
146	=FORMULA(m;R149C1)	
147	=FORMULA(p;R150C1)	
148	=RETURN(ref)	
149	3	
150	23	
151		
152		
153	axe.x	* Fonction de recherche des labels à afficher sur l'axe des X.
154	=RESULT(8)	
155	=ARGUMENT("type";1)	
156	=SET.NAME("ref";OFFSET(depart;R149C1;0):OFFSET(depart;R150C1;0))	
157	=SET.NAME("ref";ref)	
158	=RETURN(ref)	
159		
160		
161		
162	conversion_arguments	* Fonction qui convertit des chaines de caractère

	1	2
163	=RESULT(8)	* représentant des cellules d'une feuille en références utilisables
164	=SET.NAME("depart";SUBSTITUTE("nomIR10C1","nom";feuille))	* par le programme.
165	=SET.NAME("depart";TEXTREF(depart;))	* départ: référence la première cellule contenant la 1° label d'une
166	=SET.NAME("debut_chiffre";SUBSTITUTE("nomIR10C3","nom";feuille))	* année
167	=SET.NAME("debut_chiffre";TEXTREF(debut_chiffre;))	* début_chiffre: référence la cellule pouvant contenir la première
168	=SET.NAME("quarter";SUBSTITUTE("nomIR9C3","nom";feuille))	* valeur d'une série.
169	=SET.NAME("quarter";TEXTREF(quarter;))	* quarter: référence la cellule contenant la 1° label d'un trimestre
170	=RETURN()	
171		
172		
173		
174	fermer_fichier()	* Fonction de fermeture des feuilles qui ne seront plus utilisées
175	=ACTIVATE(fresult)	* dans un avenir immédiat
176	=SAVE()	
177	=CLOSE()	
178	=SET.NAME("docs";DOCUMENTS())	
179	=FOR("counter";1;COLUMNS(docs))	
180	= SET.NAME("fenetre";INDEX(docs;counter))	
181	= ACTIVATE(fenetre)	
182	= IF(OR(fenetre="printing.xlm";fenetre="graphe1.xlm";fenetre="main_pr.xlm";fenetre="graphe2.xlm";fenetre="analyse.xlm";fenetre="graphe3.xlm";fenetre='	
183	= GOTO(R188C1)	
184	= ELSE()	
185	= SAVE()	
186	= END.IF()	
187	= CLOSE()	
188	=NEXT()	
189	=RETURN()	
190		
191		
192		
193		
194	erreur_fich()	* Fonction qui avertit l'utilisateur que le fichier qu'il a fournit
195	=ALERT("the 'input-sheet' does not exist !!!!")	* n'existe pas dans la répertoire du système.
196	=ERROR(TRUE)	
197	=fermer_fichier()	
198	=RETURN()	
199		
200		
201		
202	iconiser()	* Fonction qui met les feuilles ouvertes sous forme d'Icones.
203	=SET.NAME("docs";DOCUMENTS())	
204	=FOR("counter";1;COLUMNS(docs))	
205	= SET.NAME("fenetre";INDEX(docs;counter))	
206	= ACTIVATE(fenetre)	
207	= WINDOW.MINIMIZE(fenetre)	
208	=NEXT()	
209	=RETURN()	
210		
211		
212		
213	ouverture_fichier()	* Fonction qui ouvre les feuilles nécessaires.
214	=RESULT(1)	
215	=ERROR(FALSE)	

	1	2
216	=SET.NAME("res";1)	
217	=IF(ERROR.TYPE(OPEN(feuille;3))=3;erreur_fichier())	
218	=ERROR(FALSE)	
219	=NEW(2;1)	
220	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
221	=RETURN()	
222	=ERROR(TRUE)	
223	=iconiser()	
224	=RETURN(res)	
225		
226		
227	creer_fichier()	* Fonction qui crée un nouveau fichier (Feuille).
228	=NEW(2;1)	
229	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
230	=RETURN()	
231		
232		
233	titre	* Fonction qui recherche le titre sur la feuille de données.
234	=RESULT(2)	*
235	=ARGUMENT("ntitre";1)	* Si ntitre = 1 , cette fonction recherche le titre principal.
236	=SET.NAME("titre";" ")	* Si ntitre = 2, cette fonction recherche le nom de l'unité des données.
237	=SET.NAME("n";0)	
238	=IF(ntitre=1)	
239	=SET.NAME("lig";-4)	
240	=ELSE()	
241	=SET.NAME("lig";-5)	
242	=END.IF()	
243	=WHILE(n<256)	* Bouclage sur la ligne contenant le titre (ou le nom de l'unité utilisée).
244	= IF(NOT(ISBLANK(DEREF(OFFSET(debut_chiffre;lig;n)))))	
245	= SET.NAME("titre";OFFSET(debut_chiffre;lig;n))	
246	= IF(ntitre=1)	
247	= SET.NAME("n";300)	
248	= ELSE()	
249	= SET.NAME("ntitre";1)	
250	= END.IF()	
251	= END.IF()	
252	= SET.NAME("n";n+1)	
253	=NEXT()	
254	=RETURN(titre)	
255		

	1	2
1	graphe.ratio	
2	=ARGUMENT("fichier";2)	* fichier : nom de la feuile qui contient les données.
3	=ARGUMENT("fresult";2)	
4	=ECHO(FALSE)	*ouverture des fichiers et création de la feuille
5	=IF(ouverture_fichier(fichier)=0)	*qui contiendra le graphique.
6	=GOTO(R24C1)	
7	=END.IF()	
8	=SET.NAME("sresult";SUBSTITUTE("nom!R1C1","nom";fichier))	*sresult : fait référence à la première case de la feuille qui contient les données.
9	=SET.NAME("sresult";TEXTREF(sresult))	
10	=chercher_borne(sresult)	
11	=tracer.borne(R255C1;R256C1)	
12	=SET.NAME("nb_serie";rech.nb.series(fichier))	
13	=tracer(R255C1;R256C1;sresult;nb_serie)	* mise en page de la feuille qui contient le graphique.
14	=ACTIVATE(fresult)	*
15	=SELECT("R1:R16384")	*
16	=PATTERNS(2;15;2)	*
17	=WINDOW.MAXIMIZE()	
18	=SPLIT(1;0)	
19	=PAUSE()	
20	=SAVE()	* fermeture des feuilles et suppression de la feuille qui contient
21	=CLOSE()	*le graphique.
22	=WINDOW.MINIMIZE()	*
23	=fermer_fichier()	*
24	=RETURN()	*
25		
26		
27		
28	tracer.borne	* fonction qui trace les graduations sur l'axe Y.
29	=ARGUMENT("nb";1)	*nb: nombre de graduations à ecrire de chaque côté de la valeur 1.
30	=ARGUMENT("espace";1)	*espace: différence de valeur entre chaque graduation
31	=ACTIVATE(fresult)	
32	=ROW.HEIGHT(260,1,"R1")	
33		
34	=CREATE.OBJECT(1,"R1C1";55;255,"R1C1";55;5;;FALSE)	*création de l'axe des Y.
35	=SET.NAME("axey";255)	
36	=SET.NAME("borne_inf";1-nb*espace)	*borne_inf : graduation minimale à afficher.
37	=SET.NAME("borne_sup";1+nb*espace)	*borne_sup : graduation maximale à afficher.
38	=FOR("i";borne_inf;borne_sup;espace)	
39	= SET.NAME("taille";axey-(200/((nb*2))))	*taille : taille qu'aura la boîte dans laquelle on ecrira les graduations.
40	= IF(taille<axey-15)	*
41	= SET.NAME("taille";axey-15)	*
42	= END.IF()	
43	= CREATE.OBJECT(6,"R1C1";10;axey,"R1C1";50;taille;;TRUE)	*création de la boîte.
44	= IF(ABS(i-1)<0,001)	
45	= SET.NAME("i",1)	
46	= END.IF()	
47	= TEXT.BOX(i)	
48	= IF(i=1)	* Mise en couleur de la boîte contenant une graduation.
49	= PATTERNS(1;1;1;1;FALSE;0;2;4;2;FALSE)	*
50	= ELSE()	*
51	= IF(i>1)	*
52	= PATTERNS(1;1;1;1;FALSE;0;2;2;8;FALSE)	*
53	= ELSE()	*
54	= PATTERNS(1;1;1;1;FALSE;0;2;3;2;FALSE)	*

	1	2
55	= END.IF()	
56	= END.IF()	
57	= SET.NAME("axey";axey-220/(((nb*2))))	*mise à jour de la hauteur pour placer la boîte suivante.
58	=NEXT()	
59	=RETURN()	
60		
61		
62		
63		
64	tracer	*fonction qui trace le graphique pour chaque série.
65	=ARGUMENT("nb";1)	*nb: nombre de graduations situées à chaque côté de la valeur 1.
66	=ARGUMENT("espace";1)	*espace: différence de valeur entre chaque graduation.
67	=ARGUMENT("sresult";8)	*sresult: référence la première case de la feuille qui contient les données.
68	=ARGUMENT("nb_serie";1)	*nb_série: nombre de séries que contient la feuille des données.
69	=initialiser.tab(1;(nb*2)+2)	
70	=SET.NAME("j";DEREF(R135C1))	*j: première ligne de données non vide.
71	=FOR("z";j;nb_serie+j-1;1)	* boucle qui traite la création du graphique pour chaque série.
72	= MESSAGE(TRUE;z)	
73	= traiter.serie(z;nb;espace;sresult)	
74	= tracer.ligne((nb*2)+1)	
75	=NEXT()	
76	=RETURN()	
77		
78		
79		
80	initialiser.tab	*fonction qui initialise un tableau qui reprend les coordonnées en X du
81	=ARGUMENT("min";1)	*prochain carré à afficher, et ce pour chaque valeur possible.
82	=ARGUMENT("max";1)	* max : nombre de valeur possible sur l'axe des Y.
83	=SET.NAME("j";max-min)	
84	=FOR("i";1;j)	* Pour chaque valeur possible, on initialise les coordonnées à 60 à partir
85	= FORMULA(60;OFFSET(R1C3;i-1;0))	* de la gauche de l'écran.
86	=NEXT()	*
87	=RETURN()	
88		
89		
90		
91	tracer.ligne	*Fonction qui trace une ligne verticale entre deux séries sur le graphique
92	=ARGUMENT("nb_ligne";1)	*et met le tableau reprenant les coordonnées en X à jour.
93	=SET.NAME("max";R1C3)	*nb_ligne : le nombre de graduations sur l'axe de Y.
94	=FOR("i";1;nb_ligne-1)	*Recherche de la coordonée la plus à droite sur le graphique
95	= IF(max < DEREF(OFFSET(R1C3;i;0)))	
96	= SET.NAME("max";DEREF(OFFSET(R1C3;i;0)))	
97	= END.IF()	
98	=NEXT()	
99	=FOR("i";0;nb_ligne-1)	*Mise à niveau des coordonnées pour chaque valeur possible.
100	= FORMULA(max+5;OFFSET(R1C3;i;0))	
101	=NEXT()	
102	=ACTIVATE(fresult)	
103	=SET.NAME("refer";SUBSTITUTE("nomlR1C1";"nom";fresult))	
104	=SET.NAME("refer";TEXTREF(refer))	
105	=CREATE.OBJECT(1;refer;max+3;255;refer;max+3;5;;FALSE)	*création de la ligne verticale
106	=CREATE.OBJECT(1;refer;55;25;refer;max+3;25;;FALSE)	*création d'une ligne horizontale.
107	=FORMULA(max+3;OFFSET(refer;1;0))	
108	=RETURN()	

	1	2
109		
110		
111		
112	rech.nb.series	*Fonction qui recherche le nombre de séries contenues dans une feuille donnée.
113	=RESULT(1)	
114	=ARGUMENT("fichier";2)	*fichier : nom de la feuille concernée.
115	=SET.NAME("slig";SUBSTITUTE("nomlR8C3:nomlr8c256","nom",fichier))	
116	=SET.NAME("slig";TEXTREF(slig))	*slig : référence la première colonne de la feuille de données
117	=SET.NAME("lig";COUNTA(slig))	*lig: nombre d'années comprises dans la feuille de données.
118	=SET.NAME("lig";lig*4)	
119	=SET.NAME("scol";SUBSTITUTE("nomlC1","nom";fichier))	
120	=SET.NAME("scol";TEXTREF(scol))	*scol: référence la première ligne de la feuille de données.
121	=FOR("i";0;3)	
122	= IF(COUNTA(OFFSET(scol;0;i+2))>3)	*recherche du premier quart de la première année contenant des données.
123	= SET.NAME("i";i)	*
124	= GOTO(R127C1)	*
125	= END.IF()	*
126	=NEXT()	*
127	=FORMULA(i+1;R135C1)	
128	=FOR("j";lig+1;lig+4)	*
129	= IF(COUNTA(OFFSET(scol;0;j))<2)	*recherche du dernier quart de la dernière année contenant des données.
130	= SET.NAME("fin";lig+4-j)	*
131	= GOTO(R134C1)	*
132	= END.IF()	*
133	=NEXT()	*
134	=SET.NAME("lig";lig-i-fin)	*lig : nombre de séries contenues dans la feuille de données.
135	2	
136	=RETURN(lig)	
137		
138		
139		
140		
141	traiter.serie	*Fonction qui crée le graphique pour une série donnée.
142	=ARGUMENT("i";1)	*numéro de la ligne contenant la série à traiter.
143	=ARGUMENT("nb";1)	*nb: nombre de valeurs possibles soit inférieures à 1,soit supérieures à 1.
144	=ARGUMENT("espace";1)	*espace : différence de valeurs entre deux graduations successives.
145	=ARGUMENT("sresult";8)	*sresult: référence la première ligne et la première colonne de la feuille de données.
146	=SET.NAME("cel";Premier.element(i;sresult))	*cel : référence la première cellule contenant une donnée dans la série concernée.
147	=SET.NAME("period";rech.period(i;sresult))	*period: période correspondant à la série traitée.
148	=CREATE.OBJECT(6,"R1C1";DEREF(R1C3);10,"r1c1";DEREF(R1C3)+50;25;;TRUE)	*Affichage de la période.
149	=TEXT.BOX(period)	*
150	=WHILE(NOT(ISBLANK(DEREF(OFFSET(sresult;cel+8;i+1)))))	*Pour chaque valeur de la série, on construit le graphique.
151	= SET.NAME("val";DEREF(OFFSET(sresult;cel+8;i+1)))	*
152	= creer.carre(val;nb;espace)	*
153	= SET.NAME("cel";cel+1)	*
154	=NEXT()	*
155	=RETURN()	
156		
157		
158		
159		
160	rech.period	*fonction qui recherche la période concernée par une série donnée.
161	=RESULT(7)	
162	=ARGUMENT("i";1)	* i : numéro de la ligne contenant la série traitée.

	1	2
163	=ARGUMENT("sresult";8)	*sresult: référence la première cellule (haut à gauche) de la feuille de données.
164	=SET.NAME("period";DEREF(OFFSET(sresult;8;i+1)))	*period: trimestre concerné
165	=SET.NAME("an";DEREF(OFFSET(sresult;7;(i-MOD((i+3);4)+1))))	*an : année concernée.
166	=SET.NAME("legende1";"aaaa bb")	*concaténation de l'année et du trimestre.
167	=SUBSTITUTE(legende1;"bb";period)	*
168	=SET.NAME("leg";R[-1]C)	*
169	=SUBSTITUTE(leg;"aaaa";an)	*
170	=SET.NAME("period";R[-1]C)	*
171	=RETURN(period)	
172		
173		
174		
175	Premier.element	*Fonction qui recherche la colonne contenant la première donnée d'une série
176	=ARGUMENT("i";1)	*donnée.
177	=ARGUMENT("sresult";8)	*i: numéro de la ligne contenant la série traitée.
178	=RESULT(1)	*sresult: référence la première case de la feuille contenant les données.
179	=SET.NAME("col";1)	
180	=WHILE(col<256)	*recherche de la première colonne non blanche.
181	= IF(NOT(ISBLANK(DEREF(OFFSET(sresult;col+8;i+1)))))	*
182	= SET.NAME("m";col)	*
183	= SET.NAME("col";300)	*
184	= END.IF()	*
185	= SET.NAME("col";col+1)	
186	=NEXT()	
187	=SET.NAME("col";m)	
188	=RETURN(col)	
189		
190		
191		
192		
193	creer.carre	*Fonction qui ajoute un carré sur le graphique.
194	=ARGUMENT("val";1)	*val: valeur de la donnée traitée.
195	=ARGUMENT("nb";1)	
196	=ARGUMENT("espace";1)	
197		
198	=IF(val>1)	*Mise au point de la valeur en fonction des graduations de
199	= SET.NAME("val";INT(((1-val)+(espace/2))/espace))	*l'axe des Y.
200	=ELSE()	*
201	= SET.NAME("val";INT(((1-val)+(espace/2))/espace))	*
202	=END.IF()	*
203	=SET.NAME("val";1-(val*espace))	*
204	=SET.NAME("min_val";1-(nb*espace))	*min_val: valeur minimale possible.
205	=SET.NAME("haut";255-((val-min_val)/espace)*(110/nb))	*hauteur: hauteur à laquelle on place le carré, en fonction de la valeur de la donnée.
206	=SET.NAME("taille";100/nb)	*taille: taille du carré.
207	=IF(taille>15)	
208	= SET.NAME("taille";15)	
209	=END.IF()	
210	=SET.NAME("long";DEREF(OFFSET(R1C3;((val-min_val)/espace);0)))	*long: coordonnée horizontale à laquelle on place le carré.
211	=ACTIVATE(fresult)	
212	=CREATE.OBJECT(2;"r1c1";long;haut;"r1c1";long+taille;haut-taille;;FALSE)	*création du carré.
213	=IF(val = 1)	*mise en couleur du carré.
214	= PATTERNS(;;;;;0;1;4)	*
215	=ELSE()	*
216	= IF(val<1)	*

	1	2
217	= PATTERNS(;;;;;0;1;3)	*
218	= ELSE()	*
219	= PATTERNS(;;;;;0;1;5)	*
220	= END.IF()	*
221	=END.IF()	*
222	=SET.NAME("borne";INT((val-min_val)/espace))	* borne: nombre de graduations existants entre la valeur minimale et la valeur
223	=FORMULA(OFFSET(R1C3;borne;0)+(taille+5);OFFSET(R1C3;borne;0))	*courante
224	=RETURN()	*mise à jour du tableau de travail.
225		
226		
227		
228		
229		
230	chercher_borne	
231	=ARGUMENT("sresult";8)	*Fonction qui recherche les graduations sur l'axe des Y.
232	=SET.NAME("bas";DEREF(OFFSET(sresult;209;2)))	*sresult: référence la première cellule du fichier de données.
233	=SET.NAME("haut";DEREF(OFFSET(sresult;209;3)))	*bas: valeur minimale à traiter
234	=SET.NAME("difference";MAX((haut-1);(1-bas)))	*haut: valeur maximale à traiter
235	=IF(difference<0,1)	*différence: écart maximal par rapport à 1.
236	= SET.NAME("rmax";0,01)	
237	= SET.NAME("nb",ROUND(difference*100,0))	*rmax: différence de valeur entre deux graduations.
238	= IF(nb<5)	*nb: nombre de graduations de chaque côté de la valeur 1.
239	= SET.NAME("nb";5)	*on prend au minimum 5 graduations de chaque côté de la valeur 1.
240	= END.IF()	*
241	=ELSE()	*
242	= SET.NAME("difference";difference*100)	
243	= SET.NAME("rmax",10000)	*recherche de rmax et de nb.
244	= FOR("i";5;10)	*
245	= SET.NAME("inter";INT((difference/i)+1)/100)	*
246	= IF(inter<rmax)	* on choisit l'espace minimal possible avec un nombre de graduations
247	= SET.NAME("rmax";inter)	* compris entre 5 et 10.
248	= SET.NAME("nb";i)	*
249	= END.IF()	*
250	= NEXT()	*
251	=END.IF()	*
252	=FORMULA(nb;R255C1)	*
253	=FORMULA(rmax;R256C1)	
254	=RETURN()	
255	5	
256	0,01	
257		
258		
259		
260		
261	fermer_fichier()	*Fonction qui effectue la fermeture de toutes les feuilles ouvertes.
262	=SET.NAME("docs",DOCUMENTS())	
263	=FOR("counter",1;COLUMNS(docs))	
264	= SET.NAME("fenetre",INDEX(docs;counter))	
265	= ACTIVATE(fenetre)	
266	= IF(OR(fenetre="printing.xlm";fenetre="graphe2.xlm";fenetre="main_pr.xlm";fenetre="analyse.xlm",fenetre="graphe1.xlm",fenetre="graphe3.xlm",fenetre=	*On effectue la fermeture de ce fichier en dernier lieu.
267	= GOTO(R272C1)	*
268	= ELSE()	
269	= SAVE()	
270	= END.IF()	

	1	2
271	= CLOSE()	
272	=NEXT()	
273	=RETURN()	
274		
275		
276		
277		
278	erreur_fich()	*Fonction qui gère le cas où le fichier de données n'existe pas.
279	=ALERT("the 'input-sheet' does not exist !!!!")	
280	=ERROR(TRUE)	
281	=SET.NAME("res";0)	
282	=fermer_fichier()	
283	=RETURN()	
284		
285		
286		
287	iconiser()	*Fonction qui iconise les feuilles ouvertes.
288	=SET.NAME("docs";DOCUMENTS())	
289	=FOR("counter";1;COLUMNS(docs))	
290	= SET.NAME("fenetre";INDEX(docs;counter))	
291	= ACTIVATE(fenetre)	
292	= WINDOW.MINIMIZE(fenetre)	
293	=NEXT()	
294	=RETURN()	
295		
296		
297		
298	ouverture_fichier()	*Fonction qui effectue l'ouverture de la feuille qui contient les données.
299	=RESULT(1)	
300	=ERROR(FALSE)	
301	=SET.NAME("res";1)	
302	=IF(ERROR.TYPE(OPEN(fichier))=3;erreur_fichier())	
303	=ERROR(FALSE)	
304	=FILE.DELETE(fresult)	
305	=NEW(1)	
306	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
307	=ERROR(TRUE)	
308	=iconiser()	
309	=RETURN(res)	
310		
311		
312	creer_fichier	
313	=NEW(1)	
314	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
315	=RETURN()	
316		

	1	2
1	graphe.ratio	*Fonction qui crée un graphique à partir d'une feuile de données.
2	=ARGUMENT("fichier",2)	* fichier : nom de la feuile qui contient les données.
3	=ARGUMENT("fresult",2)	
4	=ECHO(FALSE)	*ouverture des fichiers et création de la feuille
5	=IF(ouverture_fichier(fichier)=0)	*qui contiendra le graphique.
6	= GOTO(R24C1)	
7	=END.IF()	
8	=SET.NAME("sresult",SUBSTITUTE("nomlR1C1","nom",fichier))	*sresult : fait référence à la première case de la feuille qui contient les données.
9	=SET.NAME("sresult",TEXTREF(sresult))	
10	=creation.axes.y(fichier)	*Création de l'axe des ordonnées.
11	=creation.axes.x(fichier)	*Création de l'axe des coordonnées ainsi que du graphe.
12	=SELECT("R1:R16384")	* mise en page de la feuille qui contient le graphique.
13	=PATTERNS(1,2)	*
14	=SELECT("R1c1")	*
15	=PATTERNS(2;15;2)	*
16	=ZOOM(70)	
17	=WINDOW.MAXIMIZE()	
18	=SPLIT(1;3)	
19	=PAUSE()	
20	=SAVE()	* fermeture des feuilles et suppression de la feuille qui contient
21	=CLOSE()	*le graphique.
22	=WINDOW.MINIMIZE()	*
23	=fermer_fichier()	*
24	=RETURN()	*
25		
26		
27		
28		
29	creation.axes.y	*Fonction qui trace les graduations sur l'axe des ordonnées.
30	=ARGUMENT("fichier",2)	
31	=SET.NAME("nb.serie",SUBSTITUTE("nomlR8C3:nomlr8c256","nom",fichier))	*nb.série:référence la première colonne du fichier de données.
32	=SET.NAME("nb.serie",TEXTREF(nb.serie))	*
33	=SET.NAME("lig",COUNTA(nb.serie))	*lig:nombre d'années dont on désire afficher le graphique.
34	=SET.NAME("premier",DEREF(OFFSET(sresult,7;2)))	*premier:première année à afficher.
35	=ACTIVATE(fresult)	*On active la feuille qui contiendra le graphique.
36	=SET.NAME("hauteur",60)	*hauteur: ordonnée à laquelle on travaille.
37	=CREATE.OBJECT(6,"R1C1",80;5,"R1C1",470;20;;TRUE)	
38	=TEXT.BOX("CONSUMERS EXPENDITURE AT CURRENT MARKET PRICES")	*création du titre.
39	=FOR("nb",0;lig-1)	*On boucle sur le nombre d'année et pour chaque année sur ses 4 trimestres
40	= FOR("nbquart",1;4)	* afin de créer l'axe de ordonnées.
41	= SET.NAME("titre",SUBSTITUTE("y/d","y",premier+nb))	*
42	= SET.NAME("titre",SUBSTITUTE(titre,"d",nbquart))	*
43	= CREATE.OBJECT(6,"R1C1",10;hauteur,"R1C1",45;hauteur+15;;TRUE)	*
44	= TEXT.BOX(titre)	*
45	= SET.NAME("hauteur",hauteur+20)	*
46	= NEXT()	*
47	=NEXT()	*
48	=creer_legende()	
49	=RETURN()	
50		
51		
52		
53		
54	creation.axes.x	* Fonction qui effectue lé création du graphique pour chaque série, période par

	1	2
55	=ARGUMENT("fichier";2)	*période.
56	=SET.NAME("nb.serie";SUBSTITUTE("nomIR8C3:nomIr8c256","nom";fichier))	
57	=SET.NAME("nb.serie";TEXTREF(nb.serie))	
58	=SET.NAME("col";COUNTA(nb.serie))	*Col nombre d'années à traiter.
59	=SET.NAME("premier";DEREF(OFFSET(sresult;7;2)))	*premier : représente la première année.
60	=ACTIVATE(fresult)	
61	=SET.NAME("longueur";1)	* longueur: représente la coordonnnée courante sur le graphique.
62	=FOR("nb";0;col-1)	* On boucle sur chaque année et pour chaque année, on boucle sur chacun
63	= FOR("nbquart";1;4)	*de ses 4 trimestres.
64	= SET.NAME("titre";SUBSTITUTE("y/d";"y";premier+nb))	*
65	= SET.NAME("titre";SUBSTITUTE(titre;"d";nbquart))	*
66	= CREATE.OBJECT(6;"R1C1";(longueur*50);25;"R1C1";(longueur*50)+35;40;;TRUE)	*
67	= TEXT.BOX(titre)	*
68	= affiche_pourcent(longueur;col*4)	*
69	= tracer.serie(fichier;longueur)	*
70	= SET.NAME("longueur";longueur+1)	*
71	= NEXT()	
72	=NEXT()	
73	=SET.NAME("plot";(SUBSTITUTE("nomIR200C2","nom";fresult)))	
74	=SET.NAME("plot";TEXTREF(plot))	
75	=FORMULA((longueur*50)+35;plot)	
76	=RETURN()	
77		
78		
79		
80		
81	tracer.serie	*Fonction qui trace le graphique pour une période donnée.
82	=ARGUMENT("fichier";2)	
83	=ARGUMENT("nb.serie";1)	
84	=SET.NAME("numero";SUBSTITUTE("nomIR10ca:nomIR100Ca","a";nb.serie+2))	
85	=SET.NAME("numero";SUBSTITUTE(numero;"nom";fichier))	
86	=SET.NAME("numero";TEXTREF(numero))	
87	=SET.NAME("nbdate";COUNTA(numero))	*nbdate: nombre de donnée contenue dans la série concernée.
88	=SET.NAME("nbser";1)	*nbser: numéro de la donnée courante recherchée.
89	=SET.NAME("nbcol";0)	*nbcol: nombre de colonne de la série que l'on a déja visité.
90	=WHILE(nbser<=nbdate)	*On boucle tant que l'on n'a pas retrouvé chaque donnée.
91	= IF(NOT(ISBLANK(DEREF(OFFSET(sresult;9+nbcol;1+nb.serie)))))	*création et coloriage d'un carré si la cellule courement visitée
92	= CREATE.OBJECT(2;"R1C1";((nb.serie)*50)+10;((nbcol)*20)+60;"R1C1";((nb.serie)*50)+25;((nbcol)*20)+75;;FALSE)	*n'est pas vide.
93	= colorier(DEREF(OFFSET(sresult;9+nbcol;1+nb.serie)))	*
94	= SET.NAME("nbser";nbser+1)	*
95	= END.IF()	*
96	= SET.NAME("nbcol";nbcol+1)	
97	=NEXT()	
98	=RETURN()	
99		
100		
101		
102	affiche_pourcent	*Fonction qui affiche à l'écran le pourcentage du travail déja effectué.
103	=ARGUMENT("lig";1)	
104	=ARGUMENT("pourcent";1)	
105	=PRODUCT(100;PRODUCT(lig;1/pourcent))	
106	=ROUND(R[-1]C;0)	
107	=MESSAGE(TRUE;SUBSTITUTE("percent %","percent";R[-1]C))	
108	=RETURN()	

	1	2
109		
110		
111		
112	colorier	*Fonction qui colore les carrés tracés en fonction de la valeur
113	=ARGUMENT("valeur",1)	* de la donnée qui lui correspond.
114	=IF(ROUND(valeur;2)<=0,94)	
115	= PATTERNS(1;1;1;1;FALSE;0;1;1;2;FALSE)	*Noir Foncé.
116	=ELSE()	
117	= IF(ROUND(valeur;2)<=0,96)	
118	= PATTERNS(1;1;1;1;FALSE;0;2;1;2;FALSE)	*Noir un peu plus clair.
119	= ELSE()	
120	= IF(ROUND(valeur;2)<=0,99)	
121	= PATTERNS(1;1;1;1;FALSE;0;17;1;2;FALSE)	*Gris-noir.
122	= ELSE()	
123	= IF(ROUND(valeur;2)<=1)	
124	= PATTERNS(1;1;1;1;FALSE;0;1;4;2;FALSE)	*vert.
125	= ELSE()	
126	= IF(ROUND(valeur;2)<=1,03)	
127	= PATTERNS(1;1;1;1;FALSE;0;17;3;2;FALSE)	*Bleu foncé.
128	= ELSE()	
129	= IF(ROUND(valeur;2)<=1,06)	
130	= PATTERNS(1;1;1;1;FALSE;0;2;3;2;FALSE)	*Bleu un peu plus clair.
131	= ELSE()	
132	= PATTERNS(1;1;1;1;FALSE;0;1;3;2;FALSE)	*bleu clair.
133	= END.IF()	
134	= END.IF()	
135	= END.IF()	
136	= END.IF()	
137	= END.IF()	
138	=END.IF()	
139	=RETURN()	
140		
141		
142		
143		
144	fermer_fichier()	*Fonction qui effectue la fermeture des fichiers que l'on n'utilise plus.
145	=SET.NAME("docs",DOCUMENTS())	
146	=FOR("counter",1,COLUMNS(docs))	
147	= SET.NAME("fenetre",INDEX(docs;counter))	
148	= ACTIVATE(fenetre)	
149	= IF(OR(fenetre="printing.xlm",fenetre="graphe2.xlm",fenetre="main_pr.xlm",fenetre="analyse.xlm",fenetre="graphe1.xlm",fenetre="graphe3.xlm",fenetre=	
150	= GOTO(R155C1)	
151	= ELSE()	
152	= SAVE()	
153	= END.IF()	
154	= CLOSE()	
155	=NEXT()	
156	=RETURN()	
157		
158		
159		
160		
161	ouverture_fichier()	*Fonction qui effectue l'ouverture des fichiers utilisés par le programme.
162	=RESULT(1)	

	1	2
163	=SET.NAME("res",1)	
164	=ARGUMENT("feuille",2)	
165	=ERROR(FALSE)	
166	=IF(ERROR.TYPE(OPEN(feuille))=3;erreur_fich())	
167	=FILE.DELETE(fresult)	
168	=NEW(1)	
169	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
170	=ERROR(TRUE)	
171	=iconiser()	
172	=RETURN(res)	
173		
174		
175		
176	erreur_fich()	* Fonction qui alerte l'utilisateur que le fichier d'entrée qu'il a donné n'existe
177	=ALERT("the 'input-sheet' does not exist !!!!")	* pas dans le répertoire courant.
178	=ERROR(TRUE)	
179	=SET.NAME("res",0)	
180	=fermer_fichier()	
181	=RETURN()	
182		
183		
184		
185	iconiser()	* Fonction qui met les fichiers ouverts sous forme d'icônes.
186	=SET.NAME("docs";DOCUMENTS())	
187	=FOR("counter",1,COLUMNS(docs))	
188	= SET.NAME("fenetre",INDEX(docs;counter))	
189	= ACTIVATE(fenetre)	
190	= WINDOW.MINIMIZE(fenetre)	
191	=NEXT()	
192	=RETURN()	
193		
194		
195		
196		
197	creer_legende	*Fonction qui crée une légende en bas du graphique.
198	= CREATE.OBJECT(6,"R1C1";80;hauteur+10;"R1C1";500;hauteur+120;;TRUE)	
199	=TEXT.BOX("KEY TO RATIOS:□ Less than 0.94 More than 1.06□ □ 0.94 - 0.96	
200	=TEXT.BOX("1.00 1.04 - 1.06□ □ 0.97- 0.99 1.01 - 1.03□";;201)	
201	=FORMAT.FONT("MS Sans Serif";12;FALSE;FALSE;FALSE;FALSE;0;;;;1;356)	
202	=CREATE.OBJECT(2,"R1C1";155;hauteur+25;"R1C1";170;hauteur+40;;FALSE)	
203	=PATTERNS(1;1;1;1;FALSE;0;1;1;2;FALSE)	
204	=CREATE.OBJECT(2,"R1C1";155;hauteur+55;"R1C1";170;hauteur+70;;FALSE)	
205	=PATTERNS(1;1;1;1;FALSE;0;2;1;2;FALSE)	
206	=CREATE.OBJECT(2,"R1C1";155;hauteur+85;"R1C1";170;hauteur+100;;FALSE)	
207	=PATTERNS(1;1;1;1;FALSE;0;17;1;2;FALSE)	
208	=CREATE.OBJECT(2,"R1C1";280;hauteur+55;"R1C1";295;hauteur+70;;FALSE)	
209	=PATTERNS(1;1;1;1;FALSE;0;1;4;2;FALSE)	
210	=CREATE.OBJECT(2,"R1C1";390;hauteur+25;"R1C1";405;hauteur+40;;FALSE)	
211	=PATTERNS(1;1;1;1;FALSE;0;1;3;2;FALSE)	
212	=CREATE.OBJECT(2,"R1C1";390;hauteur+55;"R1C1";405;hauteur+70;;FALSE)	
213	=PATTERNS(1;1;1;1;FALSE;0;2;3;2;FALSE)	
214	=CREATE.OBJECT(2,"R1C1";390;hauteur+85;"R1C1";405;hauteur+100;;FALSE)	
215	=PATTERNS(1;1;1;1;FALSE;0;17;3;2;FALSE)	
216	=SET.NAME("plot";(SUBSTITUTE("nomIR200C1";"nom";fresult)))	

	1	2
217	=SET.NAME("plot";TEXTREF(plot))	
218	=FORMULA(hauteur+100,plot)	
219	=RETURN()	

	1	2
1	gestion.graphique	*Fonction qui crée un graphique à partir de séries données.
2	=ARGUMENT("feuille",2)	*feuille: nom de la feuille qui contient les données.
3	=ARGUMENT("type",1)	*Type: type du graphique désiré (7 types différents).
4	=ARGUMENT("an1",1)	*an1:année de la première série.
5	=ARGUMENT("quart1",1)	*an2: année de la seconde série.
6	=ARGUMENT("an2",1)	*quart1: trimestre de la première série.
7	=ARGUMENT("quart2",1)	*quart2:trimestre de la seconde série.
8	=ARGUMENT("fresult",2)	*fresult: nom de la feuille qui contiendra le graphique.
9	=ouverture_fichier()	*
10	=IF(OR(type=1;type=4;type=5))	* double = 1 indique que l'on affiche une seule série.
11	= SET.NAME("double",1)	*
12	= SET.NAME("ref1";rech.ref(feuille;an1;quart1;type;1;an2;quart2))	*
13	=ELSE()	*
14	= IF(OR(type=2;type=3;type=6))	
15	= SET.NAME("ref1";rech.ref(feuille;an1;quart1;type;2;an2;quart2))	
16	= SET.NAME("double",2)	* double = 2 indique que l'on va afficher 2 séries.
17	= ELSE()	
18	= SET.NAME("double";test(an1;quart1;an2;quart2))	* double= le nombre de séries à afficher simultanément.
19	= SET.NAME("ref1";rech.ref(feuille;an1;quart1;type;double;an2;quart2))	* ref1: référence les valeurs des séries à afficher.
20	= END.IF()	
21	=END.IF()	
22	=SET.NAME("ref1";ref1)	
23	=SET.NAME("titre.x";axe.x(feuille;an1;quart1;type;double;an2;quart2))	* Titre.x: référence les labels à placer sur l'axe des ordonnées.
24	=IF(OR(type=1;type=2))	
25	= SET.NAME("type",11)	* type = 11 indique que l'on crée un graphique de type batonnet
26	=ELSE()	* en trois dimensions.
27	= IF(OR(type=3;type=4))	
28	= SET.NAME("type",12)	* type = 12 indique que l'on crée un graphique de type linéaire
29	= ELSE()	* en trois dimensions.
30	= SET.NAME("type",3)	* type = 3 indique que l'on crée un graphique de type batonnet
31	= END.IF()	* en deux dimensions.
32	=END.IF()	
33	=creation.graphique(ref1;type;an1;quart1;an2;quart2;double;titre.x)	
34	=PAUSE()	
35	=delaxe()	
36	=WINDOW.MINIMIZE()	
37	=fermer_fichier()	
38	=RETURN()	
39		
40		
41		
42		
43	creation.graphique	* Fonction qui gère le tracé du graphique.
44	=ECHO(FALSE)	
45	=ARGUMENT("gr",8)	* gr: référence les valeurs des séries.
46	=ARGUMENT("type",1)	
47	=ARGUMENT("an1",1)	
48	=ARGUMENT("quart1",1)	
49	=ARGUMENT("an2",1)	
50	=ARGUMENT("quart2",1)	
51	=ARGUMENT("double",1)	* double: indique le nombre de séries à afficher sur le graphique.
52	=ARGUMENT("titre.x",8)	* titre.x: référence les labels à afficher sur l'axe des abcisses.
53	=ACTIVATE(fresult)	
54	=WINDOW.MAXIMIZE()	

	1	2
55	=CREATE.OBJECT(5;"R1C4";12;0,75;"R16C11";0;0,75;1;TRUE)	* création d'un objet de type graphique.
56	=SET.NAME("label1";titrage(1))	* label1: Titre à afficher sur le graphique.
57	=SET.NAME("label2";titrage(2))	* label2: titre de l'axe des ordonnées.
58	=CHART.WIZARD(TRUE;gr;type;6;1;2;2;1;label1;"QUARTERS";label2;"")	* Fonction de création de graphique.
59	=SELECT("Axis 1")	
60	=SCALE(TRUE;TRUE;TRUE;TRUE;TRUE;FALSE;FALSE;FALSE)	* Fonction de calcul des échelles des axes.
61	=SELECT("Title")	
62	=FORMAT.FONT(0;1;FALSE;"MS Sans Serif";14;FALSE;FALSE;TRUE;FALSE)	* Fonction de mise en page du titre.
63	=SELECT("Plot")	
64	=GRIDLINES(TRUE;FALSE;TRUE;FALSE;TRUE;FALSE)	* Fonction de gestion de grillages sur le graphique.
65	=FORMAT.MAIN(3;1;0;0;FALSE;FALSE;FALSE;;;;FALSE;FALSE;TRUE)	
66	=SELECT("Text Axis 1")	
67	=FORMAT.TEXT(2,2,3,FALSE;TRUE)	*Fonction de traitement du texte des labels de l'axe x.
68	=SELECT("Chart")	
69	=PATTERNS(2,1,1,1;FALSE;0;3;2;6;FALSE)	* Fonction de gestion de la couleur du graphique.
70	=SET.NAME("deb";OFFSET(R167C1;0;0))	*
71	=SET.NAME("fin";OFFSET(R168C1;0;0))	*
72	=FOR("i";deb;fin+1)	*
73	= SELECT("Axis 2")	*
74	= FORMAT.TEXT(;;;0)	*
75	= SET.NAME("legende1";"MMM 0000")	*
76	= SET.NAME("annee";OFFSET(quarter;((INT(i/4)*4)+1);-2))	*
77	= SUBSTITUTE(legende1;"0000";annee)	*
78	= SET.NAME("leg1";R[-1]C)	* GESTION DE LA LEGENDE
79	= SUBSTITUTE(leg1;"MMM";OFFSET(quarter;i+1;-1))	*
80	= EDIT.SERIES(i-deb+1;R[-1]C;titre.x)	*
81	=NEXT()	*
82	=SELECT("legend")	*
83	=FORMAT.FONT(0;1;FALSE;"MS Sans Serif";6;FALSE;FALSE;FALSE;FALSE)	*
84	=FORMAT.LEGEND(1)	*
85	=SELECT("Gridline 1")	
86	=FORMULA("=""Text2""")	*m: numéro de colonne de la première valeur.
87	=FORMAT.MOVE(270;3)	
88	=FORMULA("=""Publication in the Economic Trends:""")	* Ajout d'un texte en bas du graphique.
89	=SELECT("Chart")	
90	=WINDOW.MAXIMIZE()	
91	=RETURN()	
92		
93		
94		
95	rech.ref	* Fonction de recherche des références des valeurs des séries.
96	=RESULT(8)	
97	=ARGUMENT("feuille";2)	
98	=ARGUMENT("an";1)	
99	=ARGUMENT("quart";1)	
100	=ARGUMENT("type";1)	
101	=ARGUMENT("nr";1)	
102	=conversion_arguments()	
103	=SET.NAME("nr";nr)	* nr: nombre de séries à afficher.
104	=SET.NAME("n";(an-depart))	
105	=SET.NAME("n";(n*4)+(quart-1))	* n: position de la séries dans la feuille de données
106	=FOR("i";1;nr)	
107	= SET.NAME("lig";0)	
108	= WHILE(lig<100)	

	1	2
109	= IF(NOT(ISBLANK(DEREF(OFFSET(debut_chiffre;lig;n)))))	* Recherche de la référence de la première valeur d'une série.
110	= SET.NAME("m";lig)	
111	= SET.NAME("lig";300)	
112	= END.IF()	
113	= SET.NAME("lig";lig+1)	
114	= NEXT()	
115	= SET.NAME("lig";m)	
116	= WHILE(lig<100)	
117	= IF(ISBLANK(DEREF(OFFSET(debut_chiffre;lig;n))))	* Recherche de la référence de la dernière valeur d'une série.
118	= SET.NAME("p";lig)	
119	= SET.NAME("lig";300)	
120	= END.IF()	
121	= SET.NAME("lig";lig+1)	
122	= NEXT()	
123	= IF(AND(nr=2;i=1))	* si le nombre de séries est égal à deux, on change
124	= SET.NAME("n1";n)	* la valeur des paramètres de travail, pour travailler
125	= ARGUMENT("an";1)	* sur la seconde série.
126	= ARGUMENT("quart";1)	
127	= SET.NAME("m1";m)	
128	= SET.NAME("p1";p)	
129	= SET.NAME("n";(an-depart))	
130	= SET.NAME("n";(n*4)+(quart-1))	
131	= END.IF()	
132	= IF(nr>2)	* Si on travail sur plus de deux séries, on concatène les références
133	= IF(i=1)	* des différentes séries à chjaque boucle.
134	= SET.NAME("m1";m)	
135	= SET.NAME("p1";p)	
136	= SET.NAME("ref";OFFSET(debut_chiffre;m;n):OFFSET(debut_chiffre;p;n))	
137	= ELSE()	
138	= SET.NAME("m1";MIN(m1;m))	
139	= SET.NAME("p1";MAX(p;p1))	
140	= SET.NAME("n1";n)	
141	= SET.NAME("ref1";OFFSET(debut_chiffre;m1;n1):OFFSET(debut_chiffre;p1;n1))	
142	= SET.NAME("ref";(ref;ref1))	
143	= END.IF()	
144	= SET.NAME("n";n+1)	
145	= END.IF()	
146		
147	=NEXT()	
148	=IF(nr=2)	* Si on travail sur deux séries, on concatène les références de ces
149	= SET.NAME("m";MIN(m1;m))	* deux séries à la sortie de la boucle.
150	= SET.NAME("p";MAX(p;p1))	
151	= SET.NAME("ref";OFFSET(debut_chiffre;m;n1):OFFSET(debut_chiffre;p;n1))	
152	= SET.NAME("ref1";OFFSET(debut_chiffre;m;n):OFFSET(debut_chiffre;p;n))	
153	= SET.NAME("ref";(ref;ref1))	
154	=ELSE()	
155	= IF(nr=1)	
156	= SET.NAME("ref";OFFSET(debut_chiffre;m;n):OFFSET(debut_chiffre;p;n))	
157	= ELSE()	
158	= SET.NAME("p";p1)	
159	= SET.NAME("m";m1)	
160		
161	= END.IF()	
162	=END.IF()	

	1	2
163	=FORMULA(m;R167C1)	
164	=FORMULA(p;R168C1)	
165	=RETURN(ref)	
166		
167	3	
168	23	
169		
170		
171		
172		
173	conversion_arguments	* Fonction qui convertit des chaines de caractère
174	=RESULT(8)	* représentant des cellules d'une feuille en références utilisables
175	=SET.NAME("depart";SUBSTITUTE("nom!R10C1";"nom";feuille))	* par le programme.
176	=SET.NAME("depart";TEXTREF(depart;))	* départ: référence la première cellule contenant la 1° label d'une
177	=SET.NAME("debut_chiffre";SUBSTITUTE("nom!R10C3";"nom";feuille))	* année
178	=SET.NAME("debut_chiffre";TEXTREF(debut_chiffre;))	* début_chiffre: référence la cellule pouvant contenir la première
179	=SET.NAME("quarter";SUBSTITUTE("nom!R9C3";"nom";feuille))	* valeur d'une série.
180	=SET.NAME("quarter";TEXTREF(quarter;))	* quarter: référence la cellule contenant la 1° label d'un trimestre
181	=RETURN()	
182		
183		
184		
185	fermer_fichier()	* Fonction de fermeture des feuilles qui ne seront plus utilisées
186	=ACTIVATE(fresult)	* dans un avenir immédiat
187	=SAVE()	
188	=SET.NAME("docs";DOCUMENTS())	
189	=FOR("counter";1;COLUMNS(docs))	
190	= SET.NAME("fenetre";INDEX(docs;counter))	
191	= ACTIVATE(fenetre)	
192	= IF(OR(fenetre="printing.xlm";fenetre="graphe4.xlm";fenetre="main_pr.xlm";fenetre="graphe2.xlm";fenetre="analyse.xlm";fenetre="graphe3.xlm";fenetre=	
193	= GOTO(R198C1)	
194	= ELSE()	
195	= SAVE()	
196	= END.IF()	
197	= CLOSE()	
198	=NEXT()	
199	=RETURN()	
200		
201		
202		
203		
204	erreur_fich()	* Fonction qui avertit l'utilisateur que le fichier qu'il a fournit
205	=ALERT("the 'input-sheet' does not exist !!!!")	* n'existe pas dans la répertoire du système.
206	=ERROR(TRUE)	
207	=fermer_fichier()	
208	=RETURN()	
209		
210		
211		
212	iconiser()	* Fonction qui met les feuilles ouvertes sous forme d'Icones.
213	=SET.NAME("docs";DOCUMENTS())	
214	=FOR("counter";1;COLUMNS(docs))	
215	= SET.NAME("fenetre";INDEX(docs;counter))	

	1	2
216	= ACTIVATE(fenetre)	
217	= WINDOW.MINIMIZE(fenetre)	
218	=NEXT()	
219	=RETURN()	
220		
221		
222		
223	ouverture_fichier()	* Fonction qui ouvre les feuilles nécessaires.
224	=RESULT(1)	
225	=ERROR(FALSE)	
226	=SET.NAME("res";1)	
227	=IF(ERROR.TYPE(OPEN(feuille;3))=3;erreur_fichier())	
228	=ERROR(FALSE)	
229	=NEW(2;1)	
230	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
231	=RETURN()	
232	=ERROR(TRUE)	
233	=iconiser()	
234	=RETURN(res)	
235		
236		
237	creer_fichier()	* Fonction qui crée un nouveau fichier (Feuille).
238	=NEW(2;1)	
239	=SAVE.AS(fresult;1;"";FALSE;"";FALSE)	
240	=RETURN()	
241		
242		
243		
244	test	* Fonction de calcul du nombre de périodes contenues
245	=RESULT(1)	* dans un intervale donné.
246	=ARGUMENT("an1";1)	
247	=ARGUMENT("quart1";1)	
248	=ARGUMENT("an2";1)	
249	=ARGUMENT("quart2";1)	
250	=SET.NAME("nb";((an2-an1)*4+(quart2-quart1))+1)	
251	=RETURN(nb)	
252		
253		
254	titre	* Fonction qui recherche le titre sur la feuille de données.
255	=RESULT(2)	*
256	=ARGUMENT("ntitre",1)	* Si ntitre = 1 , cette fonction recherche le titre principal.
257	=SET.NAME("titre"," ")	* Si ntitre = 2, cette fonction recherche le nom de l'unité des données.
258	=SET.NAME("n";0)	
259	=IF(ntitre=1)	
260	=SET.NAME("lig";-4)	
261	=ELSE()	
262	=SET.NAME("lig";-5)	
263	=END.IF()	
264	=WHILE(n<256)	* Bouclage sur la ligne contenant le titre (ou le nom de l'unité utilisée).
265	= IF(NOT(ISBLANK(DEREF(OFFSET(debut_chiffre;lig;n)))))	
266	= SET.NAME("titre";OFFSET(debut_chiffre;lig;n))	
267	= IF(ntitre=1)	
268	= SET.NAME("n";300)	

	1	2
269	= ELSE()	
270	= SET.NAME("ntitre";1)	
271	= END.IF()	
272	= END.IF()	
273	= SET.NAME("n";n+1)	
274	=NEXT()	
275	=RETURN(titre)	
276		
277		
278	axe.x	* Fonction qui recherche la référence des labels de l'axe des abcisses.
279	=RESULT(8)	
280	=ARGUMENT("feuille";2)	
281	=ARGUMENT("an";1)	
282	=ARGUMENT("quart";1)	
283	=ARGUMENT("type";1)	
284	=ARGUMENT("nr";1)	
285	=conversion_arguments()	
286	2	
287	=SET.NAME("nr";nr)	
288	=FORMULA(nr;R286C1)	
289	=SET.NAME("n";(an-depart))	
290	=SET.NAME("n";(n*4)+(quart-1))	
291		
292	=FOR("i";1;nr)	* Bouclage sur l'ensemble des séries a afficher pour
293	=SET.NAME("inter";DEREF(OFFSET(quarter;0;n)))	* retrouver les labels de ces séries.
294	=SET.NAME("inter2";DEREF(OFFSET(quarter;-1;(INT(n/4))*4)))	
295	=SET.NAME("inter1";"aa qq")	
296	=SUBSTITUTE(inter1;"aa";inter2)	
297	=SET.NAME("inter3";R[-1]C)	
298	=SET.NAME("inter3";SUBSTITUTE(inter3;"qq";inter))	
299	=FORMULA(inter3;OFFSET(quarter;150;i))	
300	=IF(nr=2)	
301	=ARGUMENT("an";1)	
302	=ARGUMENT("quart";1)	
303	=SET.NAME("n";(an-depart))	
304	=SET.NAME("n";(n*4)+(quart-1))	
305	=ELSE()	
306	=SET.NAME("n";n+1)	
307	=END.IF()	
308	= NEXT()	
309	=SET.NAME("ref";OFFSET(quarter;150;1):OFFSET(quarter;150;nr))	* On stocke les référence de labels pour une utilisation ultérieure.
310	=RETURN(ref)	
311		
312		
313	delaxe	* Fonction qui élimine les références des labels de l'axe des ordonnées
314	=SET.NAME("nb";DEREF(R286C1))	
315	=ERROR(FALSE)	
316	=FOR("i";1;nb)	
317	=CLEAR(OFFSET(quarter;150;i))	
318	=NEXT()	
319	=RETURN()	
320		
321		

ANNEXE C

Graphismes et tableaux

Remarques :

Les tableaux et graphismes présentés dans cette annexe sont en réalité en couleur. Par souci de facilité pour la mise-en-page de ce mémoire, nous les avons imprimés en noir et blanc ce qui peut entraîner parfois des difficultés de lecture. Nous nous excusons auprès du lecteur pour ces éventuels désagréments.

En ce qui concerne les graphismes, nous n'avons tiré qu'un exemplaire de chaque type afin de ne pas surcharger cette partie. Ceci permet toutefois de se faire une bonne idée des résultats générés.

Table 1 **UK gross domestic product at market prices**

Issue of Economic Trends in wich figures were published		Gross Domestic Product at Market prices																						£million
		1987				1988				1989				1990				1991				1992		
		Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
1987	January																							
	April																							
	July																							
	October		101100																					
1988	January		101600	105200																				
	April		102000	105400	107400																			
	July		101900	105500	107800	109100																		
	October		101718	105631	107501	109601	112060																	
1989	January		102021	106066	107614	109890	112855	116471																
	April		101814	105750	107749	110387	113196	116158	118857															
	July		101899	105647	107635	110433	113229	115981	118738	121317														
	October		102448	106184	109228	111685	113621	117756	120871	123805	125717													
1990	January		102640	106333	109503	111483	113763	118084	121120	123738	126533	128193												
	April		102576	106376	109339	111760	114353	118243	121539	123953	126105	127886	130915											
	July		102744	106421	109380	111779	114471	118309	121732	123917	126242	128141	131104	133840										
	October		103521	107041	109880	112440	115778	119173	122726	125218	127545	128720	131759	134994	137127									
1991	January		103657	106930	109554	111568	115532	118635	122173	124487	126664	127582	131075	134161	136489	137363								
	April		103657	106930	109554	111844	115035	118550	122601	124449	126428	127337	131299	133613	135827	136161	138058							
	July		103524	107037	109579	112217	114883	118936	122051	124854	126403	128153	130953	134313	135388	137401	138266	138634						
	October		103524	107037	109579	112217	114883	118936	122051	124854	126403	128153	130953	134853	137573	139033	139138	139309	143080					
1992	January		103524	107037	109579	112217	114883	118936	122051	124775	126469	128801	131368	134904	137481	138831	139289	139541	143599	145135				
	April		103383	107225	109579	112217	114883	118936	122051	124775	126469	128801	131368	134947	137264	138769	139218	140083	143402	145303	146588			
	July		103383	107225	109979	111735	114919	119039	122170	124953	126678	129013	131577	138029	137289	138817	139202	139806	143715	145598	146488	147058		
	October																							

Source: Table 2 of Economic Trends
(series code CAOB)

Table 3

Issue of Economic Trends in which figures were published

UK GROSS DOMESTIC PRODUCT AT MARKET PRICES

upwards and downwards variations of the published estimations

quality

	1987 Q1	1987 Q2	1987 Q3	1987 Q4	1988 Q1	1988 Q2	1988 Q3	1988 Q4	1989 Q1	1989 Q2	1989 Q3	1989 Q4	1990 Q1	1990 Q2	1990 Q3	1990 Q4	1991 Q1	1991 Q2	1991 Q3	1991 Q4	1992 Q1	1992 Q2	1992 Q3	1992 Q4
1987 January																								
April																								
July																								
October			0																					
1988 January			1	0																				
April			1	1	0																			
July			-1	1	1	0																		
October			-1	1	-1	1	0																	
1989 January			1	1	1	1	1	0																
April			-1	-1	1	1	1	-1	0															
July			1	-1	-1	1	1	-1	-1	0														
October			1	1	1	1	1	1	1	1	0													
1990 January			1	1	1	-1	1	1	1	-1	1	0												
April			-1	1	-1	1	1	1	1	1	-1	-1	0											
July			1	1	1	1	1	1	1	-1	1	1	1	0										
October			1	1	1	1	1	1	1	1	1	1	1	1	0									
1991 January			1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0								
April			0	0	0	1	-1	-1	1	-1	-1	-1	1	-1	-1	-1	0							
July			-1	1	1	1	-1	1	-1	1	-1	1	-1	1	-1	1	1	0						
October			0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0					
1992 January			0	0	0	0	0	0	-1	1	1	1	1	-1	-1	1	1	1	0					
April			-1	1	0	0	0	0	0	0	0	0	1	-1	-1	-1	1	-1	1	0				
July			0	0	1	-1	1	1	1	1	1	1	1	1	1	-1	-1	1	1	-1	0			
October																								

Source : Table 1

Table 6

Issue of Economic Trends in which figures were published

UK GROSS DOMESTIC PRODUCT AT MARKET PRICES

variations between a figure and its precedent

percentage %

	1987				1988				1989				1990				1991				1992			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
1987 January																								
April																								
July																								
October			0																					
1988 January			0,49	0,00																				
April			0,39	0,19	0,00																			
July			-0,10	0,09	0,37	0,00																		
October			-0,18	0,12	-0,28	0,46	0,00																	
1989 January			0,30	0,41	0,11	0,26	0,71	0,00																
April			-0,20	-0,30	0,13	0,45	0,30	-0,27	0,00															
July			0,08	-0,10	-0,11	0,04	0,03	-0,15	-0,10	0,00														
October			0,54	0,51	1,48	1,13	0,35	1,53	1,80	2,05	0,00													
1990 January			0,19	0,14	0,25	-0,18	0,12	0,28	0,21	-0,05	0,65	0,00												
April			-0,06	0,04	-0,15	0,25	0,52	0,13	0,35	0,17	-0,34	-0,24	0,00											
July			0,16	0,04	0,04	0,02	0,10	0,06	0,16	-0,03	0,11	0,20	0,14	0,00										
October			0,76	0,58	0,46	0,59	1,14	0,73	0,82	1,05	1,03	0,45	0,50	0,86	0,00									
1991 January			0,13	-0,10	-0,30	-0,78	-0,21	-0,45	-0,45	-0,58	-0,69	-0,88	-0,52	-0,62	-0,47	0,00								
April			0,00	0,00	0,00	0,25	-0,43	-0,07	0,35	-0,03	-0,19	-0,19	0,17	-0,41	-0,49	-0,88	0,00							
July			-0,13	0,10	0,02	0,33	-0,13	0,33	-0,45	0,33	-0,02	0,64	-0,26	0,52	-0,32	0,91	0,15	0,00						
October			0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,40	1,61	1,19	0,63	0,49	0,00					
1992 January			0,00	0,00	0,00	0,00	0,00	0,00	0,00	-0,06	0,05	0,51	0,32	0,04	-0,07	-0,15	0,11	0,17	0,36	0,00				
April			-0,14	0,18	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,03	-0,16	-0,04	-0,05	0,39	-0,14	0,12	0,00			
July			0,00	0,00	0,37	-0,43	0,03	0,09	0,10	0,14	0,17	0,16	0,16	2,28	0,02	0,03	-0,01	-0,20	0,22	0,20	-0,07	0,00		
October																								

Source : Table 1

Table 2

UK GROSS DOMESTIC PRODUCT AT MARKET PRICES

percentage difference from figure first published

percentage %

Issue of Economic Trends in which figures were published	1987 Q1	1987 Q2	1987 Q3	1987 Q4	1988 Q1	1988 Q2	1988 Q3	1988 Q4	1989 Q1	1989 Q2	1989 Q3	1989 Q4	1990 Q1	1990 Q2	1990 Q3	1990 Q4	1991 Q1	1991 Q2	1991 Q3	1991 Q4	1992 Q1	1992 Q2	1992 Q3	1992 Q4
1987 January																								
April																								
July																								
October			0,00																					
1988 January			0,49	0,00																				
April			0,89	0,19	0,00																			
July			0,79	0,29	0,37	0,00																		
October			0,61	0,41	0,09	0,46	0,00																	
1989 January			0,91	0,82	0,20	0,72	0,71	0,00																
April			0,71	0,52	0,32	1,18	1,01	-0,27	0,00															
July			0,79	0,42	0,22	1,22	1,04	-0,42	-0,10	0,00														
October			1,33	0,94	1,70	2,37	1,39	1,10	1,69	2,05	0,00													
1990 January			1,52	1,08	1,96	2,18	1,52	1,38	1,90	2,00	0,65	0,00												
April			1,46	1,12	1,81	2,44	2,05	1,52	2,26	2,17	0,31	-0,24	0,00											
July			1,63	1,16	1,84	2,46	2,15	1,58	2,42	2,14	0,42	-0,04	0,14	0,00										
October			2,39	1,75	2,31	3,06	3,32	2,32	3,26	3,22	1,45	0,41	0,64	0,86	0,00									
1991 January			2,53	1,64	2,01	2,26	3,10	1,86	2,79	2,61	0,75	-0,48	0,12	0,24	-0,47	0,00								
April			2,53	1,64	2,01	2,52	2,65	1,78	3,15	2,58	0,57	-0,67	0,29	-0,17	-0,95	-0,88	0,00							
July			2,40	1,75	2,03	2,86	2,52	2,12	2,69	2,92	0,55	-0,03	0,03	0,35	-1,27	0,03	0,15	0,00						
October			2,40	1,75	2,03	2,86	2,52	2,12	2,69	2,92	0,55	-0,03	0,03	0,76	0,33	1,22	0,78	0,49	0,00					
1992 January			2,40	1,75	2,03	2,86	2,52	2,12	2,69	2,85	0,60	0,47	0,35	0,79	0,26	1,07	0,89	0,65	0,36	0,00				
April			2,26	1,92	2,03	2,86	2,52	2,12	2,69	2,85	0,60	0,47	0,35	0,83	0,10	1,02	0,84	1,05	0,23	0,12	0,00			
July			2,26	1,92	2,40	2,42	2,55	2,20	2,79	3,00	0,76	0,64	0,51	3,13	0,12	1,06	0,83	0,85	0,44	0,32	-0,07	0,00		
October																								

Source : Table 1

Table 4

UK GROSS DOMESTIC PRODUCT AT MARKET PRICES

difference between the current figure and the figure first published

£millions

Issue of Economic Trends in which figures were published	1987 Q1	1987 Q2	1987 Q3	1987 Q4	1988 Q1	1988 Q2	1988 Q3	1988 Q4	1989 Q1	1989 Q2	1989 Q3	1989 Q4	1990 Q1	1990 Q2	1990 Q3	1990 Q4	1991 Q1	1991 Q2	1991 Q3	1991 Q4	1992 Q1	1992 Q2	1992 Q3	1992 Q4
1987 January																								
April																								
July																								
October		0																						
1988 January		500	0																					
April		900	200	0																				
July		800	300	400	0																			
October		618	431	101	501	0																		
1989 January		921	868	214	790	795	0																	
April		714	550	349	1287	1136	-313	0																
July		799	447	235	1333	1169	-490	-119	0															
October		1348	984	1828	2585	1561	1285	2014	2488	0														
1990 January		1540	1133	2103	2383	1703	1613	2263	2421	816	0													
April		1476	1176	1939	2660	2293	1772	2682	2636	388	-307	0												
July		1644	1221	1980	2679	2411	1838	2875	2600	525	-52	189	0											
October		2421	1841	2480	3340	3718	2702	3869	3901	1828	527	844	1154	0										
1991 January		2557	1730	2154	2468	3472	2164	3316	3170	947	-611	160	321	-638	0									
April		2557	1730	2154	2744	2975	2079	3744	3132	711	-856	384	-227	-1300	-1202	0								
July		2424	1837	2179	3117	2823	2465	3194	3537	686	-40	38	473	-1739	38	208	0							
October		2424	1837	2179	3117	2823	2465	3194	3537	686	-40	38	1013	446	1670	1080	675	0						
1992 January		2424	1837	2179	3117	2823	2465	3194	3458	752	608	453	1064	354	1468	1231	907	519	0					
April		2283	2025	2179	3117	2823	2465	3194	3458	752	608	453	1107	137	1406	1160	1449	322	168	0				
July		2283	2025	2579	2635	2859	2568	3313	3636	981	820	662	4189	162	1454	1144	1172	635	463	-100	0			
October																								

Source : Table 1

Table 5

Issue of Economic Trends in which figures were published

UK GROSS DOMESTIC PRODUCT AT MARKET PRICES

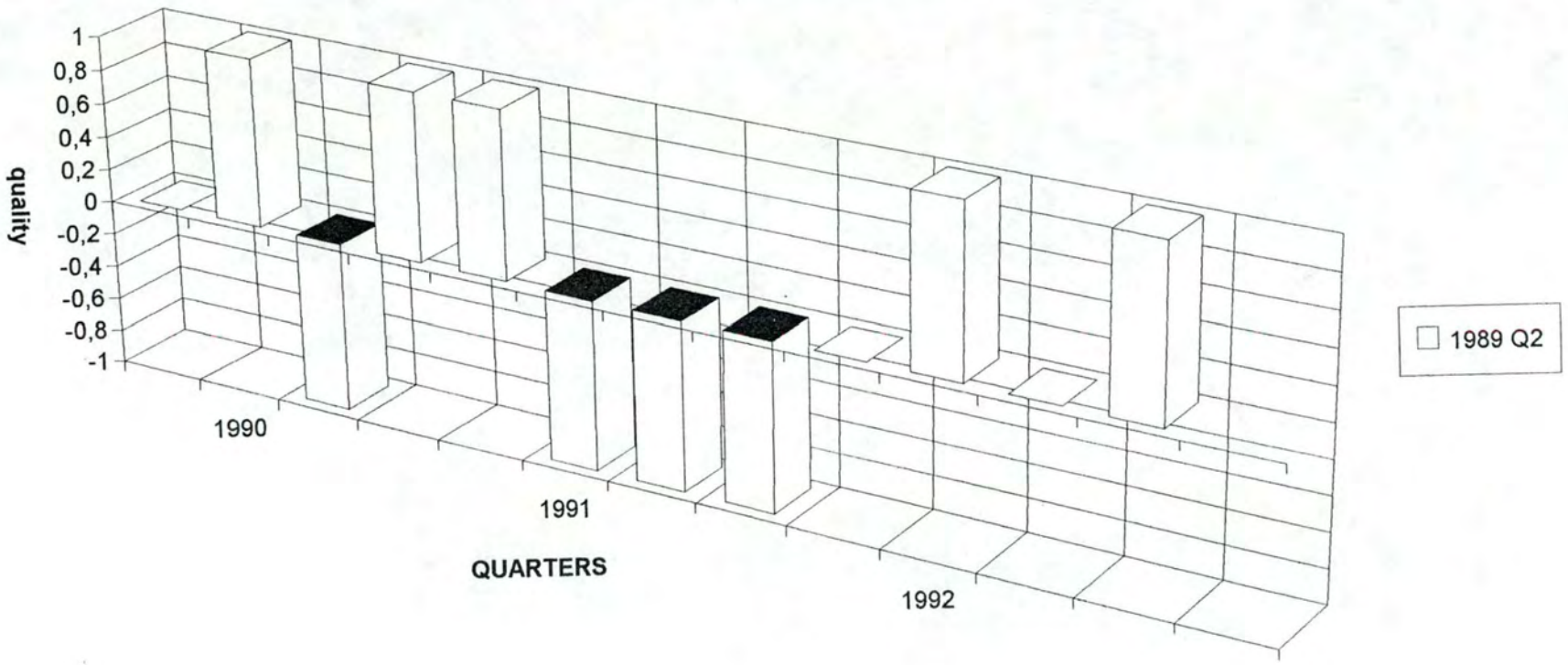
index variation between the current figure and the figure first published

percentage %

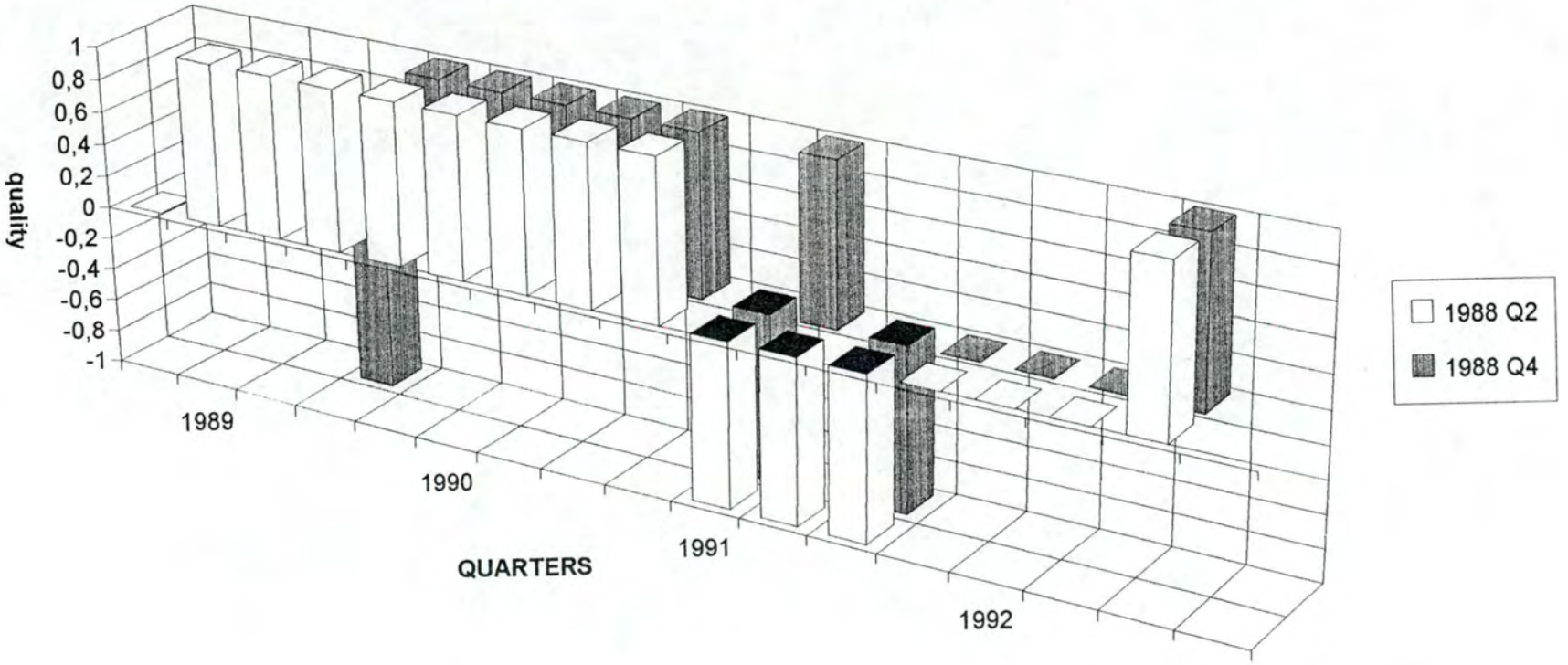
	1987				1988				1989				1990				1991				1992			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
1987 January																								
April																								
July																								
October			1,00																					
1988 January			1,00	1,00																				
April			1,01	1,00	1,00																			
July			1,01	1,00	1,00	1,00																		
October			1,01	1,00	1,00	1,00	1,00																	
1989 January			1,01	1,01	1,00	1,01	1,01	1,00																
April			1,01	1,01	1,00	1,01	1,01	1,00	1,00															
July			1,01	1,00	1,00	1,01	1,01	1,00	1,00	1,00														
October			1,01	1,01	1,02	1,02	1,01	1,01	1,02	1,02	1,00													
1990 January			1,02	1,01	1,02	1,02	1,02	1,01	1,02	1,02	1,01	1,00												
April			1,01	1,01	1,02	1,02	1,02	1,02	1,02	1,02	1,00	1,00	1,00											
July			1,02	1,01	1,02	1,02	1,02	1,02	1,02	1,02	1,00	1,00	1,00	1,00										
October			1,02	1,02	1,02	1,03	1,03	1,02	1,03	1,03	1,01	1,00	1,01	1,01	1,00									
1991 January			1,03	1,02	1,02	1,02	1,03	1,02	1,03	1,03	1,01	1,00	1,00	1,00	1,00	1,00								
April			1,03	1,02	1,02	1,03	1,03	1,02	1,03	1,03	1,01	0,99	1,00	1,00	0,99	0,99	1,00							
July			1,02	1,02	1,02	1,03	1,03	1,02	1,03	1,03	1,01	1,00	1,00	1,00	0,99	1,00	1,00	1,00						
October			1,02	1,02	1,02	1,03	1,03	1,02	1,03	1,03	1,01	1,00	1,00	1,01	1,00	1,01	1,01	1,00	1,00					
1992 January			1,02	1,02	1,02	1,03	1,03	1,02	1,03	1,03	1,01	1,00	1,00	1,01	1,00	1,01	1,01	1,01	1,00	1,00				
April			1,02	1,02	1,02	1,03	1,03	1,02	1,03	1,03	1,01	1,00	1,00	1,01	1,00	1,01	1,01	1,01	1,00	1,00	1,00			
July			1,02	1,02	1,02	1,02	1,03	1,02	1,03	1,03	1,01	1,01	1,01	1,03	1,00	1,01	1,01	1,01	1,00	1,00	1,00	1,00		
October																								

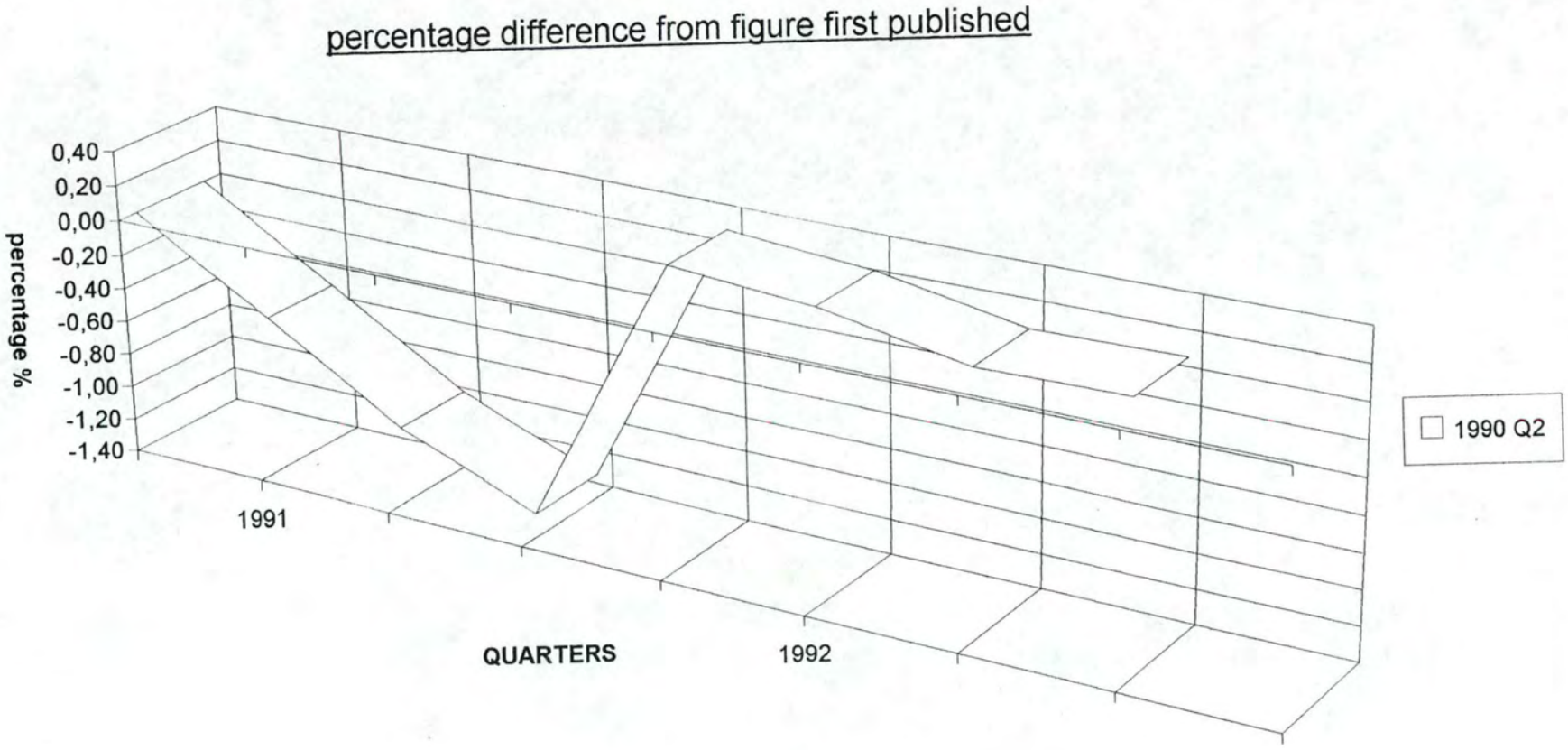
Source : Table 1

upwards and downwards variations of the published estimations

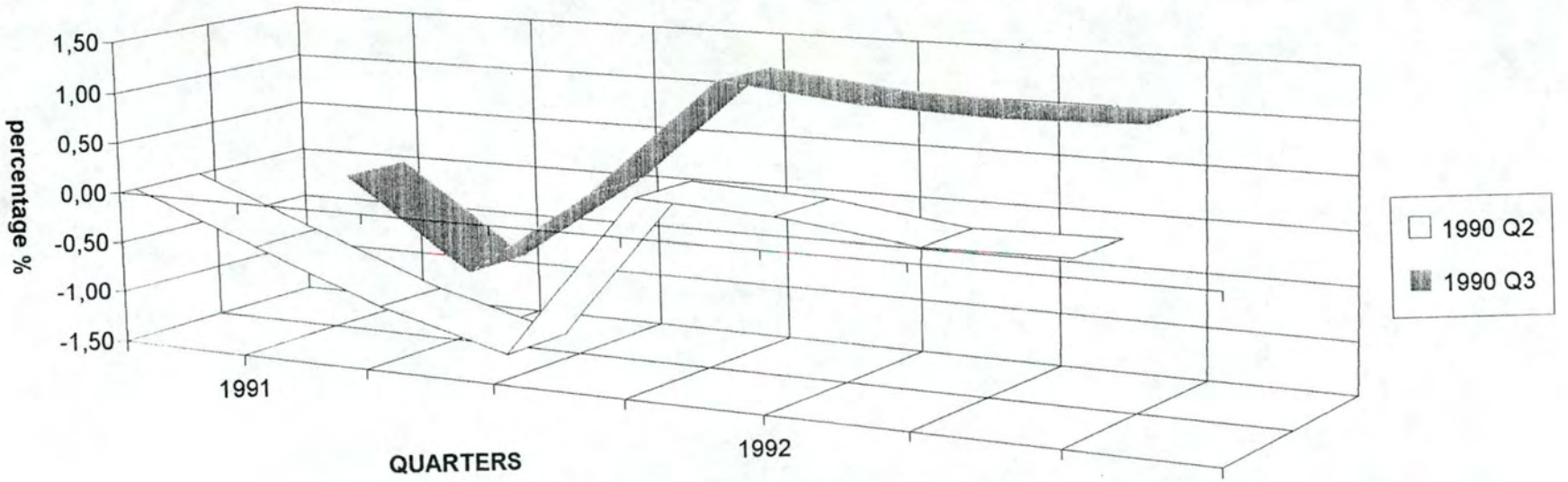


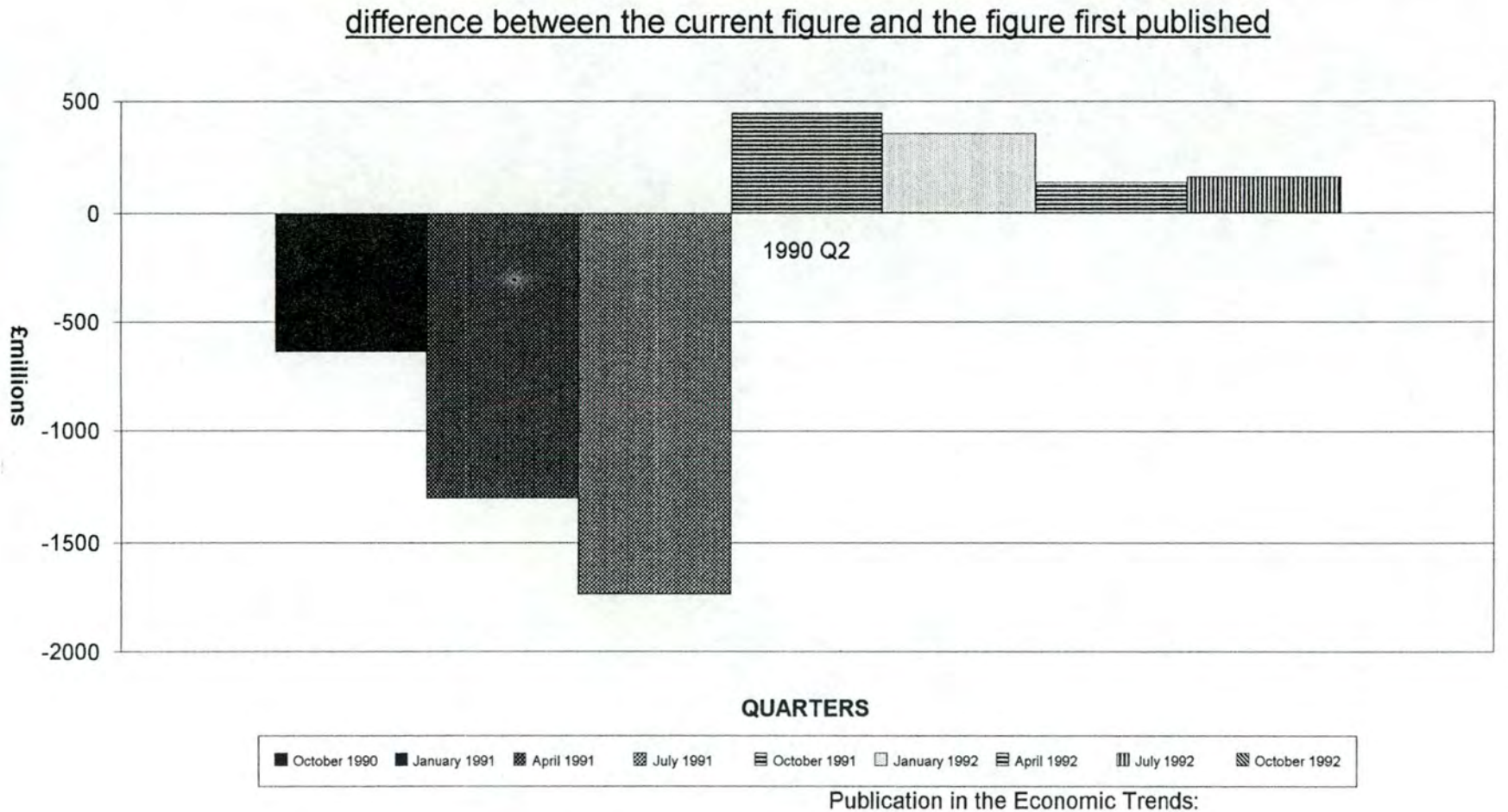
upwards and downwards variations of the published estimations



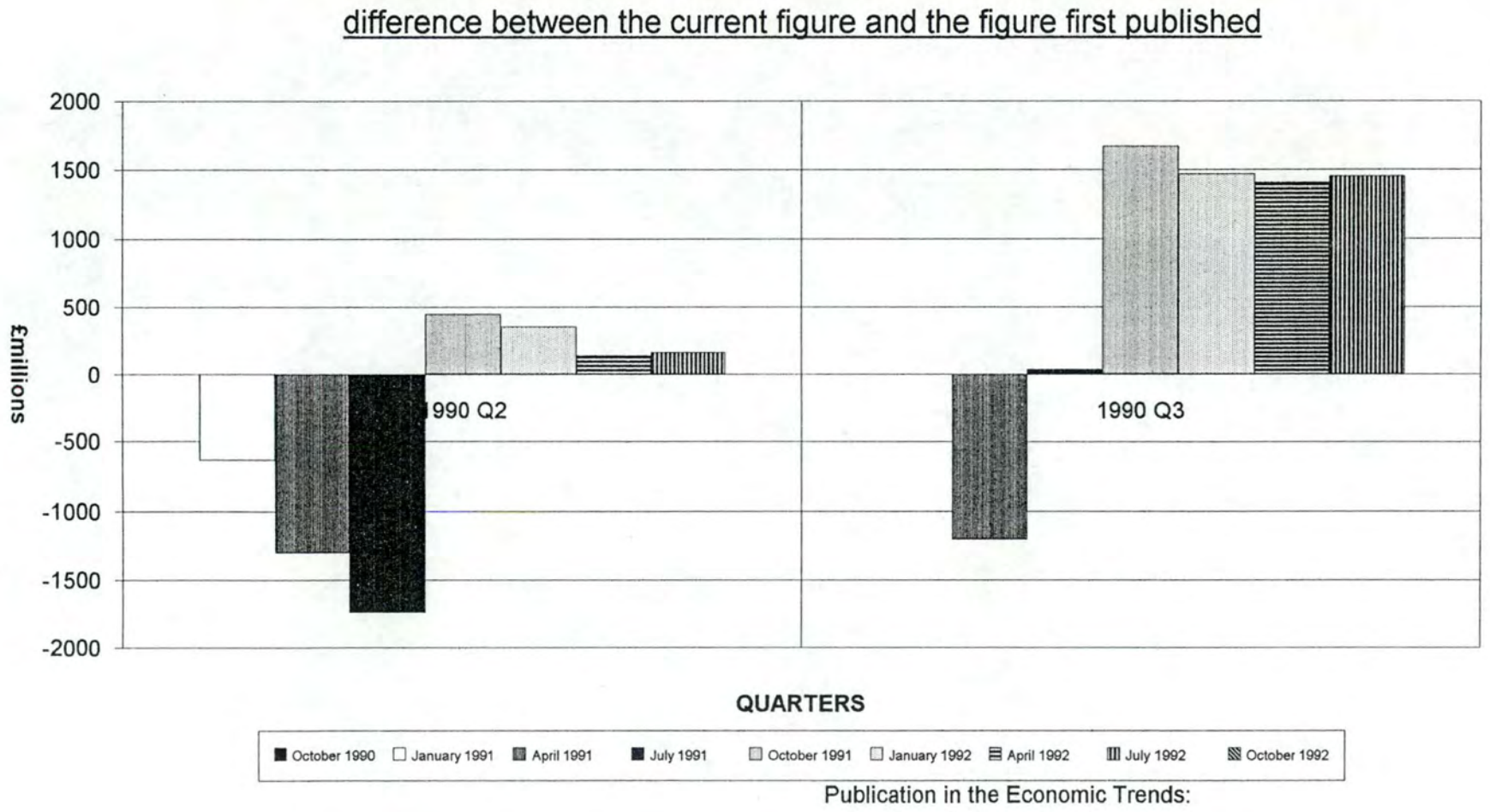

percentage difference from figure first published
0,40
0,20
0,00
-0,20
-0,40
-0,60
-0,80
-1,00
-1,20
-1,40
percentage %
1991
QUARTERS
1992
1990 Q2

percentage difference from figure first published



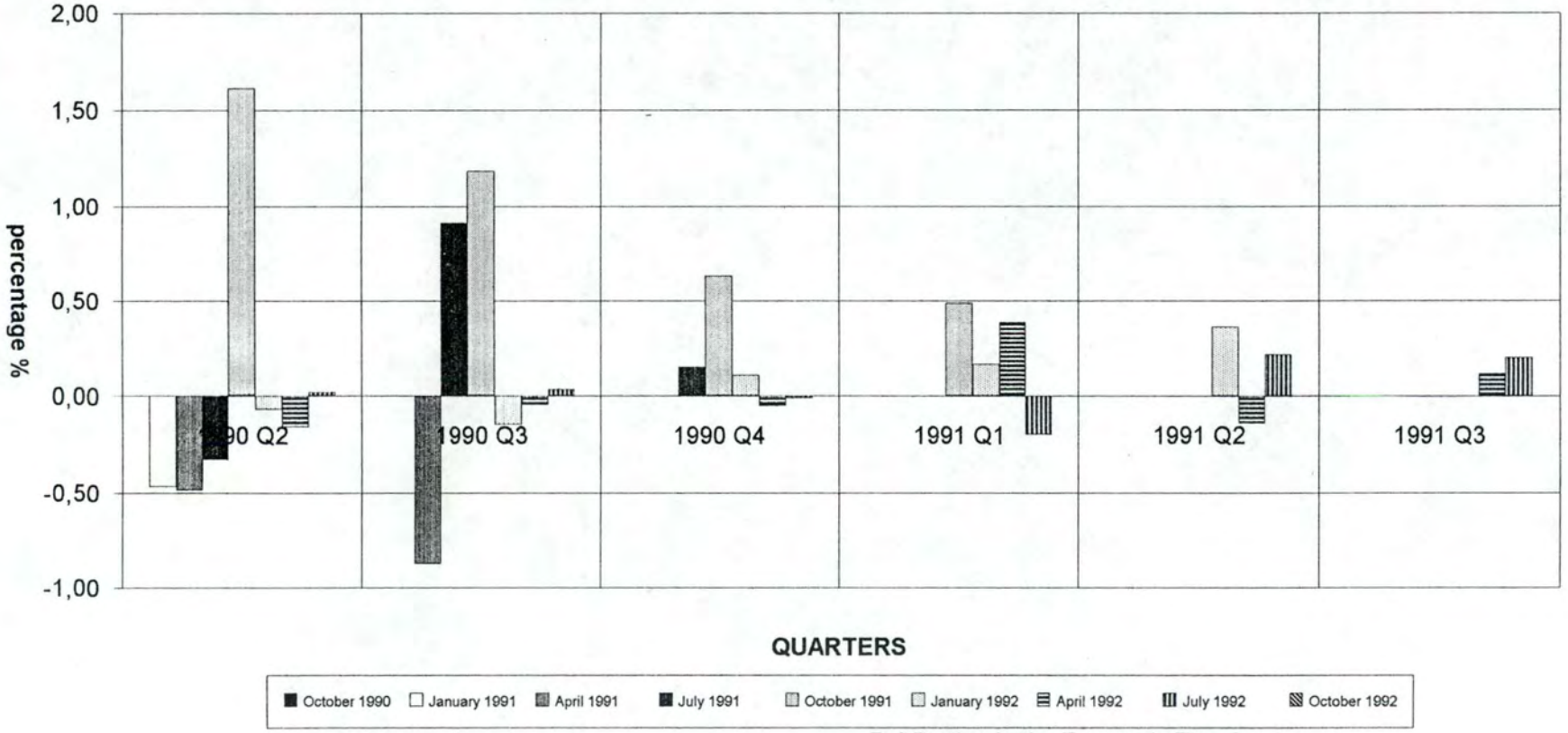


Publication in the Economic Trends:



Publication in the Economic Trends:

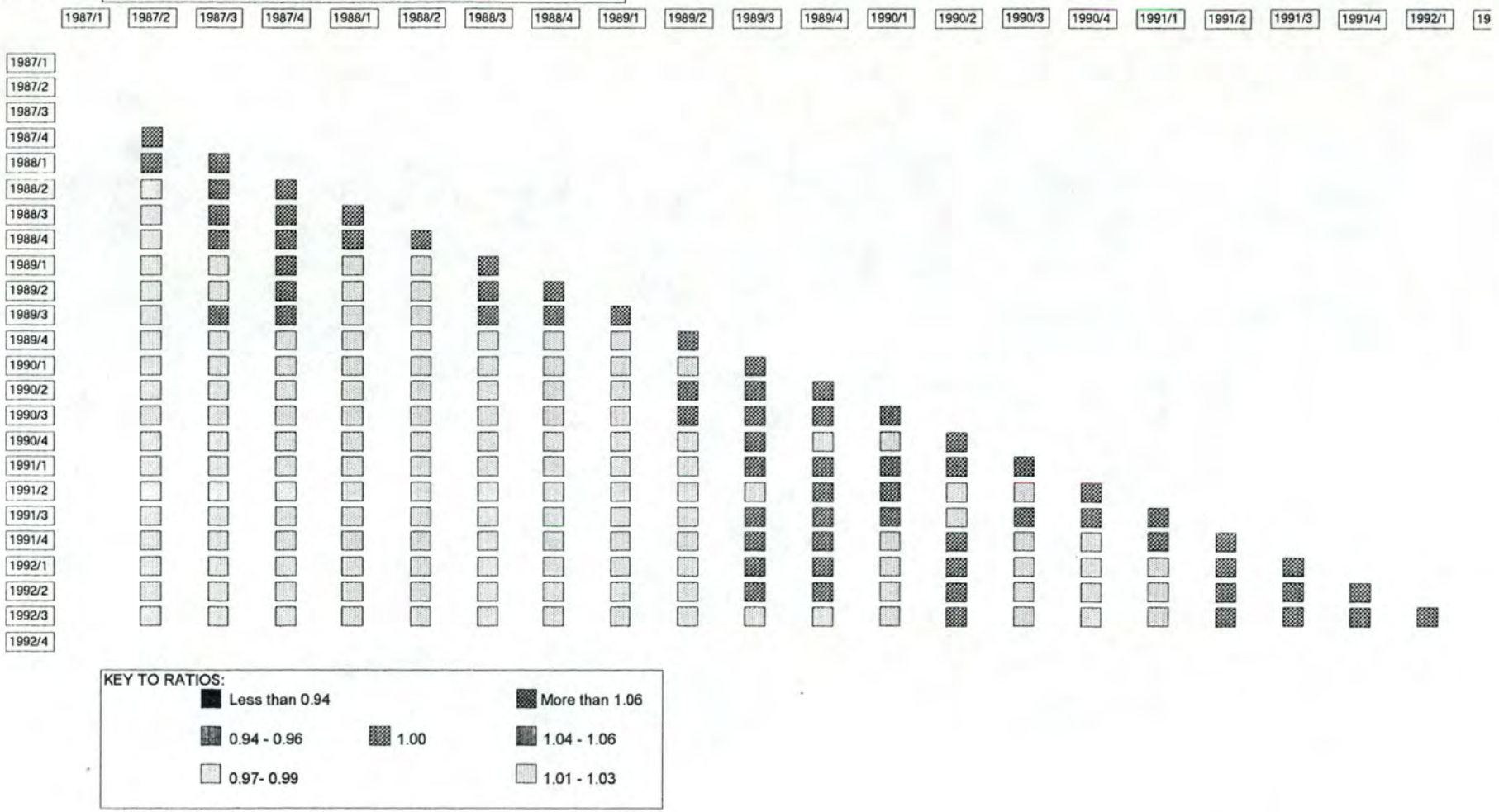
variations between a figure and its precedent



Publication in the Economic Trends:

Table 1

index variation between the curent figure and the figue first published



ANNEXE D

Listings LISP du "Mini-plan"

Remarque :

La version présentée dans ce mémoire du "Mini-plan" est la version datant du début de l'année 1993 (c'est avec cette version que nous avons travaillé). Depuis, un certain nombre de changements y ont été apportés. Une version de démonstration assez complète est en cours d'élaboration dans les bureaux de World Systems.

```
#|

minplan.lsp

Basic planning engine

|#

;;; OPERATION Structure

( defstruct operation
    ( name nil :type symbol )
    ( synopsis "" :type string )
    ( inputs nil :type list )
    ( preconds nil :type list )
    ( results nil :type list )
    ( surface nil :type list )
    ( link nil :type list )
    ( postconds nil :type list ))

;;; Operation Accessors

( defmethod name (( anOp operation ))
    ( operation-name anOp ))

( defmethod synopsis (( anOp operation ))
    ( operation-synopsis anOp ))

( defmethod inputs (( anOp operation ))
    ( operation-inputs anOp ))

( defmethod preconds (( anOp operation ))
    ( operation-preconds anOp ))

( defmethod results (( anOp operation ))
    ( operation-results anOp ))

( defmethod surface (( anOp operation ))
    ( operation-surface anOp ))

( defmethod link (( anOp operation ))
    ( operation-link anOp ))

( defmethod postconds (( anOp operation ))
    ( operation-postconds anOp ))

;;;  Application Methods

( defmethod check-vars-in-pattern (( anOp operation ) ( aPattern list
    "Check that each variable in a pattern is valid for an operation"
    ( for ( aVar aValue ) in-alist aPattern
       all-satisfy ( member aVar ( inputs anOp ))))

( defmethod expand-surface-form (( anOp operation ) ( aPattern list
    "Replace the names of variables in the surface form of an operat
      with the corresponding values in a pattern"
   ( let (( tmp ( surface anOp )))
    ( if ( check-vars-in-pattern anOp aPattern )
        ( for ( aVar aVal ) in-alist aPattern
           do ( setf tmp
```

```
                        ( subst aVal aVar  tmp )))
        ( setf tmp nil ))
      tmp ))

( defmethod expand-link-form (( anOp operation ) ( aPattern list ))
    "Replace the names of variables in the link form of an operation
      with the corresponding values in a pattern"
   ( let (( tmp ( link anOp )))
    ( if ( check-vars-in-pattern anOp aPattern )
        ( for ( aVar aVal ) in-alist aPattern
          do ( setf tmp
                        ( subst  aVal  aVar  tmp )))
        ( setf tmp nil ))
      tmp ))


( defmethod print-operation (( anOp operation ) &optional aStream )
    ( let (( oport ( or aStream t )))
      ( for ( aField aValue ) in-structure anOp
        do ( format oport "~%~:(~a~)~25T~/a-splice/" aField aValue


( defvar *opslib* ( make-hash-table :size 51
                                    :rehash-size 1.3
                                    :rehash-threshold 0.7 )
    "Table of all defined operations for planner" )

( defun fetch-op ( aKey )
    "Retrieve an operation structure from the Operations Library"
    ( gethash aKey *opslib* ))


( defun defop ( &rest args )
    "Define an operator and store it in the Operations Library"
    ( let (( anOp ( apply  #'make-operation args )))
      ( if ( name anOp )
          ( setf ( gethash (name anOp ) *opslib* ) anOp )
          nil )))


( defun defops ( anOpsList )
    "Sequentially define a set of operations"
    ( let (( anOp nil ))
      ( for x in anOpsList
        do ( progn
                ( setq anOp ( apply #'make-operation x ))
                ( if ( and ( operation-p anOp )
                           ( name anOp ))
                   ( setf ( gethash (name anOp) *opslib* ) anOp ))


( defstruct plan
    ( name nil :type symbol )
    ( variables nil :type list )
    ( preconds nil :type list )
    ( body nil :type list ))

;;; Plan Accessors
```

```
( defmethod name (( aPlan plan ))
    ( plan-name aPlan ))


( defmethod variables (( aPlan plan ))
    ( plan-variables aPlan ))

( defmethod preconds (( aPlan plan ))
    ( plan-preconds aPlan ))

( defmethod body (( aPlan plan ))
    ( plan-body aPlan ))

;;;
( defvar *planlib* ( make-hash-table :size 51
                                                    :rehash-size 1.
                                                    :rehash-thresh
    "Table of all defined plans in the planner scope" )

( defun fetch-plan ( aKey )
    "Get a plan by name"
    ( gethash  aKey *planlib*  ))

( defun defplan ( &rest args )
    "Define a plan and store it in the plan library"
    ( let (( aPlan ( apply  #'make-plan args )))
       ( if ( name aPlan )
            ( setf ( gethash (name aPlan ) *planlib* ) aPlan )
           nil )))

( defun defplans ( aPlanList )
    "Sequentially define a set of plans"
    ( let (( aPlan nil ))
       ( for x in aPlanList
          do ( progn
                 ( setq aPlan ( apply #'make-plan x ))
                 ( if ( and ( plan-p aPlan )
                                ( name aPlan ))
                    ( setf ( gethash ( name aPlan ) *planlib* ) aPlan


( defmethod print-plan (( aPlan plan ) &optional aStream)
    "Print a textual form of the plan for the user"
   ( let (( oport ( or aStream t )))
      ( for ( aField aValue ) in-structure aPlan
         do ( if ( not ( listp aValue ))
                 ( format oport "~%~:(~a~)~25T~:(~a~)" aField aValue )
                 ( progn
                   ( format oport "~%~:(~a~)" aField )
                   ( for y in aValue
                      do ( format oport "~%~25T~:(~a~)" y )))))))

( defvar *WM* ( make-hash-table )
    "Working memory for the planner " )

( defun store-value ( aKey aValue )
    "Place a value in working memory indexed by a variable name"
    ( setf ( gethash aKey *WM* ) aValue ))

( defun fetch-value ( aKey )
```

```
    "Fetch a previously stored value from the working memory, under
     variable name"
    ( gethash aKey *WM* ))

( defun splice-values ( aKeyList )
    "Generate an association list consisting of all keys in the key-list
      their associated values"
    ( if ( null aKeyList )
         nil
       ( for x in aKeyList
          collect ( list x ( fetch-value x )))))

( defun clear-memory ()
    "Clear all entries in the working memory"
    ( clrhash *WM* ))

( defun bind-value-in-call ( x )
  "bind the Value part of a ( var-name value-reference ) pair"
  ( let (( tmp ( fetch-value ( second x ))))
     ( if tmp
          ( setf ( second x ) tmp ))
     x ))

( defun bind-all-values-in-call ( parameters )
  "Bind the variable parts of a parameter list"
  ( for x in parameters
     collect ( bind-value-in-call x )))


;;; LOG operation
( defvar *Log* nil
    "Process log for the planner, used as a stack" )

( defun logit  ( aThing )
    "put a thing on the log"
    ( push aThing *Log* ))

( defun normalise-log ()
    "Reverse the log so that it's in the right order for sequential pro
    ( reverse *Log* ))

( defun clear-log ()
    "Clear all entries in the log"
    ( setf *Log* nil ))

( defun process-operation ( anOpName theOpVars )
    "Process an operation by checking the variables and expanding the
      SURFACE and LINK forms of the operation.  Return a list of the
      operation name, the expanded variables and the expanded forms."
    ( let (( tmp ( fetch-op anOpName ))
           ( bindings  theOpVars ))
      ( if ( and tmp bindings )
           ( list
              anOpName
              bindings
              ( expand-surface-form tmp bindings )
              ( expand-link-form tmp bindings )))))

( defun evaluate-plan ( aPlanName &rest vars-and-values )
    "Evaluate a plan and write the evaluation results to the log"
```

```
( let (( thePlan ( fetch-plan aPlanName ))
       ( tvar nil ))
    ( if ( not thePlan )
         (progn
             ( logit ( format nil "Error:: ~a is not a plan" aPlanNam
             nil )
       ( progn
           ;;; 1. Log the plan header
           ( logit ( format nil "Evaluating plan ~a" aPlanName ))
           ( logit ( format nil "Variables: ~a" vars-and-values ))
           ;;; 2. Instantiate the working values into the *WM*
           ( for x in ( variables thePlan )
              do ( progn
                      ( setf tvar ( second ( assoc x vars-and-values
                       ( store-value x tvar )))
           ;;; 3. evaluate the preconditions ( TBD )

           ;;; 4. process the plan body
           ( for x in (body thePlan)
              do ( case ( first x )
                      ( perform ( logit
                                  ( process-operation (second x)
                                    ( bind-all-values-in-call (cddr x)
                      ( evaluate ( evaluate-plan ( second x ) ( cddr
                      ( note ( logit ( rest x )))))
           ;;; Return the results on the log
           ( normalise-log)))))


( defun start ( aPlan SomeVars &optional OutStream )
   "Run the planner with an initial plan, and pretty-print the results'
  ( clear-log )
   ( let (( results ( apply #'evaluate-plan aPlan someVars ))
          ( oport ( or OutStream t )))
     ( if  results
        ( for x in results
           do ( format oport "~%~:(~a~)" x )))))


;;;; setup example

( defops
    '(
      ( :name  perform-qualitative-analysis
        :synopsis "Analyse repeated publications of estimated data for
                           over/under estimation"
       :inputs ( $sheet $base $result )
       :surface ( open $sheet and format $result with titles |.|
                  until no more data in $sheet analyse data
                  and place results in $result |.|
                  Search the series in $result and
                  colour according to scheme |.|
                  close $sheet and $result )
       :link ( $result = XMSJunk.1 $sheet $base ))
      (  :name  perform-proportional--analysis
       :synopsis "Analyse repeated publications of estimated data for
       :inputs ( $sheet $base $result )
       :surface ( open $sheet and format $result with titles |.|
                  until no more data in $sheet analyse data and
                  place results in $result |.|
```

```
                    · Search the series in $result
                      and colour according to scheme |.|
                      close $sheet and $result )
          :link ( $result = XMSJunk.2 $sheet $base· ))
        ))

( defplan
    :name 'test
    :variables '( $aSheet $theBase $theResult )
    :body
     '(( perform perform-qualitative-analysis ( $sheet $aSheet )
                                              ( $base $theBase )
                                              ( $result $theResult ))
        ( perform perform-proportional-analysis ( $sheet $aSheet )
                                                ( $base $theBase )
                                                ( $result $theResult )))


;;; test form for plan -- surface call

( evaluate-plan 'test '( $aSheet "gdpr.xls" ) '( $theBase first )
    '( $theResult "gdprqual.xls" ))

;;; test for expanding the surface form of an operator

( expand-surface-form ( fetch-op 'perform-qualitative-analysis )
    '(( $sheet "grpr.xls" ) ( $base first ) ( $result "gdpr.xls" )))
```