MASTER EN SCIENCES INFORMATIQUES

Measurement and Tuning of Computer Systems

Mambour, Bernard

Award date:
1989

Awarding institution:
Universite de Namur

Facultés Universitaires Notre-Dame de la Paix, Namur

Institut Informatique

Année académique 1988 - 1989

Measurement and Tuning of Computer Systems

Bernard Mambour

Mémoire présenté en vue de l'obtention du grade de Licencié et Maître en Sciences Informatiques

AVANT-PROPOS

Au terme de ce mémoire, je tiens à remercier Monsieur Ramaekers d'en avoir accepté la direction.
Ses nombreux conseils m'ont été d'une aide très précieuse; je lui en suis reconnaissant.

Je remercie également Monsieur Crismer pour ses critiques avisées.

Mes remerciements vont aussi à Monsieur David pour son aide technique et son aimable collaboration.

Je tiens également à remercier Monsieur Vandendaelen.

Enfin, je tiens à exprimer toute ma gratitude envers l'ensemble des professeurs et des assistants qui ont assuré ma formation durant ces années.

ABSTRACT

Lorsque l'on désire améliorer les performances d'un ordinateur, il est nécessaire d'analyser son fonctionnement. Dès lors, une étude d'évaluation des performances doit être entreprise.

Dans la première partie (chapitre 1 à 3), nous décrirons les éléments essentiels à l'élaboration d'une étude d'évaluation des performances. Nous définirons comment caractériser une charge de travail. Ensuite, nous examinerons les principales techniques de mesure ainsi que les outils permettant de quantifier les résultats des diverses observations réalisées.

Dans la seconde partie (chapitre 4), nous explorerons les phases d'une méthodologie d'amélioration des performances tout en les appliquant à un cas pratique.

SOMMAIRE

INTRODUCTION

Outre le fait qu'un système informatique doit réaliser correctement un certain nombre de fonctions, il doit également proposer des performances "adéquates" à un coût "raisonnable".

Les performances varient d'après les fonctions que le système doit accomplir. Un utilisateur d'un traitement de texte souhaitera une réponse instantanée à la majorité de ses commandes tandis qu'un programmeur trouvera parfois acceptable d'attendre quelques minutes pour la compilation d'un programme volumineux.

La complexité des ordinateurs s'est fortement accrue. De nouvelles techniques sont apparues pour évaluer leurs performances car l'intuition seule n'a plus suffit pour déterminer leurs causes d'efficience ou d'inefficience.

Ces performances sont caractérisées par un ensemble d'indices de performance qui sont quantifiables et qui peuvent être évalués de diverses façons.

Ils peuvent être : * mesurés,
* calculés,
* estimés.

Bien que nous ne considérions que les facteurs pouvant être quantifiés, de nombreux facteurs pris en compte lors de la

sélection d'un système sont de nature qualitative et donc, difficilement quantifiables.

Nous étudierons les principales techniques utilisées pour l'évaluation quantitative des performances.

Le type de charge de travail et les modes d'utilisation du système ont une influence directe sur les performances du système. Les principaux modes d'utilisation d'un système seront définis au point 2 du chapitre 1. L'analyse de la charge de travail est réalisée au chapitre 2.

Il est souvent possible de maintenir ces performances à un niveau acceptable en effectuant des interventions relativement asimples. Cependant, l'évaluation des performances et la recherche des solutions pouvant améliorer ces performances peuvent parfois s'avérer très ardues.

Si les valeurs des performances sont inférieures à celles requises, ou s'il semble possible d'améliorer les performances du système, une analyse des causes d'inefficience et des moyens d'y remédier doit être effectuée. Cette analyse utilisera des outils d'évaluation des performances qui seront étudiés au chapitre 3.

CHAPITRE 1 : Les performances d'un système

I. Le besoin d'évaluation des performances.

L'évaluation des performances d'un système est réalisée afin de vérifier que tout au long de son évolution il répond bien aux exigences d'efficacité et aux buts fixés.

La vérification de la satisfaction des exigences est d'autant plus importante que le système est onéreux. Souvent, ces types de systèmes ont des exigences de performance plus critiques.

L'évaluation des performances doit non seulement s'effectuer lorsque le système est actif mais aussi pendant la phase de conception, ou pendant la phase de sélection par le client, ou encore pendant sa phase d'installation.

Suivant l'objectif de l'évaluation, nous appliquerons une technique d'évaluation spécifique. On distingue quatre catégories principales :

1. L'acquisition : elle reprend tous les problèmes d'évaluation relatifs au choix d'un système ou des composants d'un système parmi les diverses alternatives possibles.

2. Les améliorations : cette catégorie comprend tous les problèmes de performances survenant dans les systèmes en activité.

3. La planification de capacité : Elle doit permettre d'évaluer avec précision l'évolution de l'activité du système afin de déterminer la capacité nécessaire de ce système.

   Il est souvent préférable de choisir un système ayant une capacité supérieure par rapport au niveau d'activité désiré et ce afin de pouvoir répondre à une augmentation de l'activité sans pour autant changer de système.

4. La conception : Elle contient tous les problèmes que les concepteurs doivent affronter pendant la création d'un nouveau système.

Nous étudierons essentiellement la deuxième classe, c'est-à-dire les améliorations. Ces améliorations provoquent généralement des effets importants mais difficiles à mesurer.

Le but réellement poursuivi par les études d'évaluation est souvent l'amélioration du ratio coût-performances.

Une procédure itérative, contenant une phase de diagnostic et une phase de thérapie, est habituellement utilisée pour aborder le problème d'amélioration de ce ratio (Fig 1.1).

La phase de diagnostic doit permettre d'établir les causes d'un ratio coût-performances insatisfaisant. Celles-ci trouvent souvent leur origine dans des goulots d'étranglement. Par conséquent, une thérapie efficace doit donc éliminer ou réduire les goulots d'étranglement ou autres causes qui limitent les performances du système.

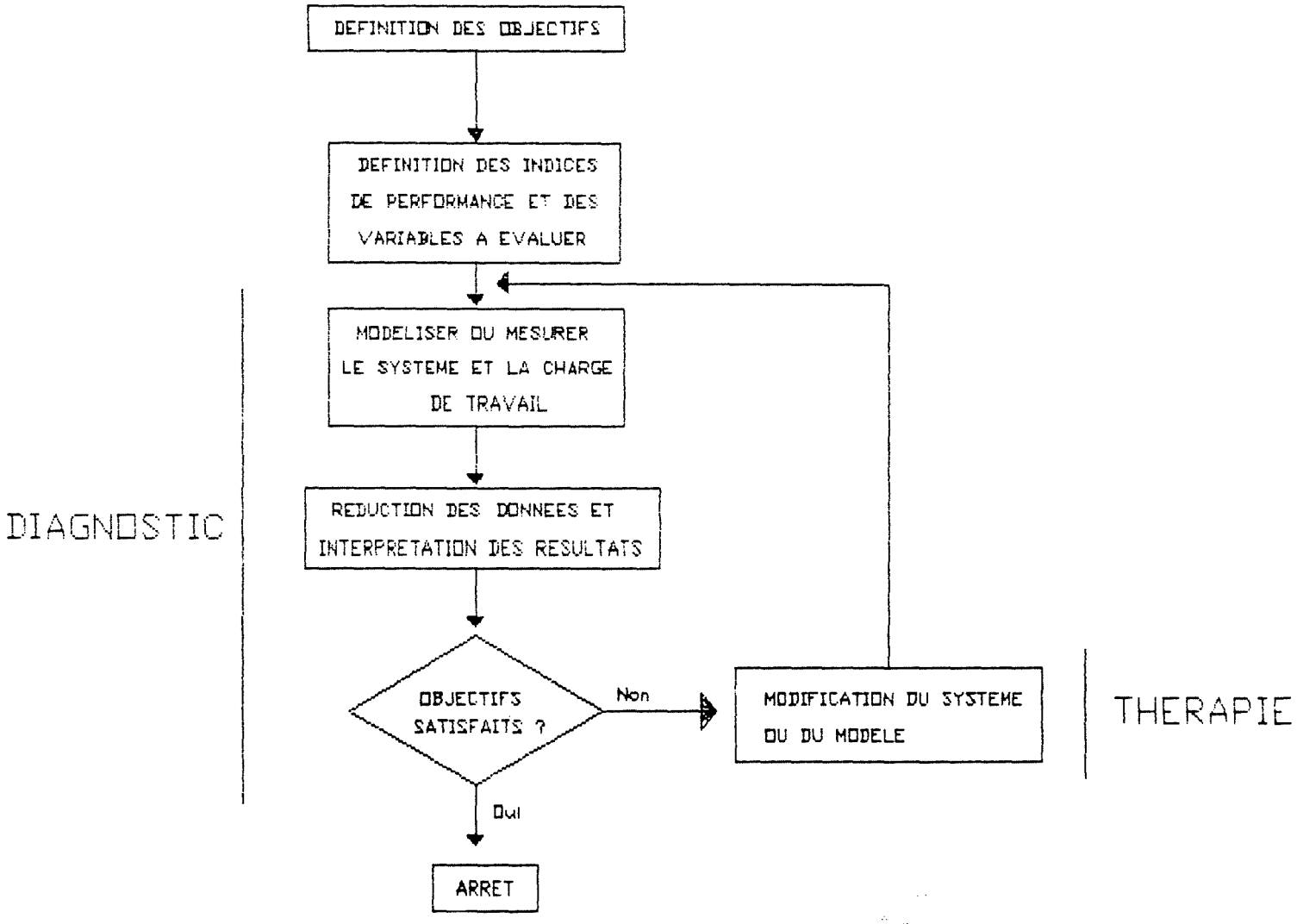


Figure 1.1 : Méthode générale d'évaluation des performances.

Lorsque plusieurs thérapies seront applicables, nous sélectionnerons généralement celle qui présentera un accroissement de performance maximal pour un coût non pas toujours minimal mais acceptable. Certaines thérapies sont onéreuses et il est donc préférable d'essayer de prédire les effets de celles-ci sur le ratio coût-performances avant de les appliquer.

Il est parfois nécessaire de répéter le cycle diagnostic-thérapie car l'élimination de certains goulots d'étranglement peut en faire apparaître de nouveaux.

II. Les différents types de système.

Les types de système influencent fortement l'étude d'évaluation des performances.

Il existe différents systèmes de référence. Les caractéristiques les distinguant sont principalement les modes d'utilisation et les types de gestion des ressources utilisés par le système d'exploitation.

On distingue généralement les systèmes du type :

* UBRS : Uniprogrammed Batch-processing Reference System. Comme son nom l'indique, ce système travaille en mode de traitement batch et les ressources sont gérées en mode "uniprogramme".

* MBRS : Multiprogrammed Batch-processing Reference System.
Dans ce type de système, des programmes et données correspondant à des "job steps" de différents utilisateurs peuvent être présents simultanément dans le système.

* MIRS : Multiprogrammed Interactive Reference System.
Le mode d'utilisation de ce système est mixte : mode batch et mode interactif.
Le système travaille en time-sharing (une requête monopolise pendant un certain "quantum" de temps les ressources du système).

* MIVRS : Multiprogrammed Interactive Virtual-memory Reference System.
Pour permettre son utilisation effective, la plupart des machines disposant d'un espace virtuel très grand ont actuellement recours à une technique de mémoire virtuelle qui permet de gérer une mémoire avec la relation "espace physique" < "espace virtuel". Le principal intérêt de cette méthode réside dans le fait que l'on peut exécuter un programme dont la taille est supérieure à la taille de la mémoire physique disponible.

Puisque tout le programme n'entre pas en mémoire, il faut disposer d'un mécanisme capable de charger, à un moment donné, les parties du programme nécessaires à son exécution. Le mécanisme de pagination permet le chargement automatique de ces parties.

Nous étudierons un système du type MIVRS. Les mécanismes de ce système seront décrits à l'annexe B.

III. Classification des performances.

Nous commencerons par introduire une classification des personnes pouvant être confrontées à des problèmes d'évaluation des performances.

Nous distinguerons trois catégories de personnes qui recherchent l'efficacité du système mais qui considèrent le problème de façon différente :

- Les concepteurs hardware et software doivent considérer l'ensemble des applications possibles pour lequel le système sera utilisé.

- Les "installation managers" doivent satisfaire les exigences d'un environnement spécifique.

- Les utilisateurs considèrent le traitement de leurs programmes.

Ces trois catégories (Fig 1.2) utilisent des variables spécifiques afin d'exprimer leur point de vue quant aux performances. Le poids attribué à chaque variable peut varier selon la catégorie.

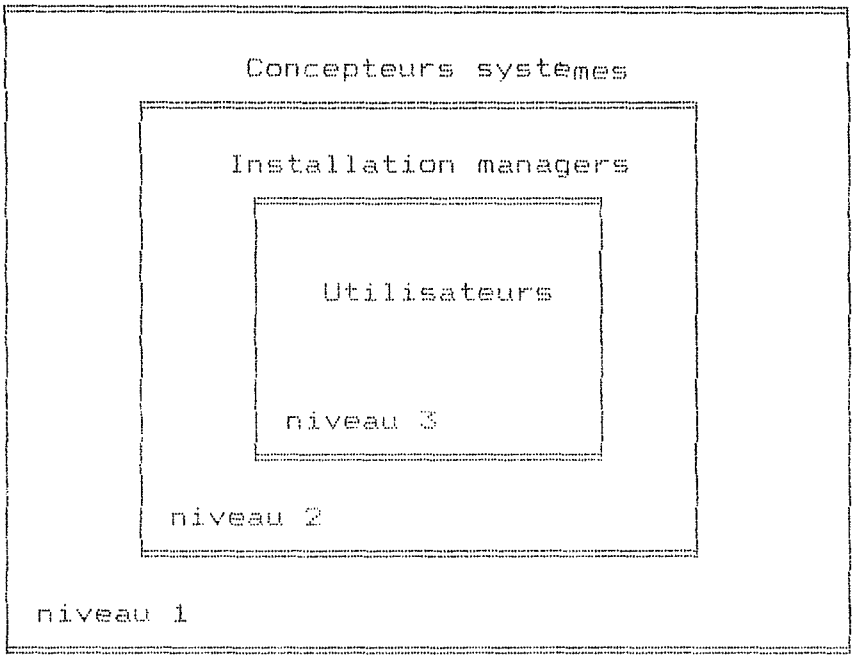


Figure 1.2 : Diverses catégories de personnes considérant des problèmes d'évaluation des performances.

Remarquons encore que toute variation de la charge de travail sur les divers composants d'un système affecte les performances globales. On ne peut donc étudier un système sans étudier les caractéristiques de sa charge de travail.

La charge de travail est constituée par l'ensemble des programmes, données et commandes que les utilisateurs entrent dans le système.

Nous utiliserons des variables qui décrivent objectivement certains aspects spécifiques des performances globales d'un système. La subjectivité sera toutefois encore présente dans le choix des variables et le poids que nous leur accorderons.

Les valeurs des variables suivantes sont nécessaires lors d'une étude d'évaluation des performances :

1. Les caractéristiques du système à évaluer.
   Cette classe comprend des données relatives à la configuration hardware et software du système et des données sur les opérations des différents composants.

2. Les conditions d'exploitation du système au moment de l'évaluation.
   Cette classe reprend la description de la charge présente lors de l'évaluation.

3. Les indices de performances du système.

Ces indices sont généralement classifiés en :

- Indices internes : ils quantifient l'efficacité d'utilisation de chaque composant du système.

  Indices internes :

  - Utilisation du CPU.
  - Recouvrement des activités.
  - Niveau de multiprogrammation
  - Taux de pagination.
  - Temps de réaction.

- Indices externes : ils sont orientés vers une évaluation externe de l'efficacité de traitement du système.

  Indices externes :

  - Turnaround time.
  - Temps de réponse.
  - Throughput.
  - Capacité.
  - Disponibilité.
  - Sécurité.

Une définiton plus détaillée de ces indices est donnée en annexe A.

Les techniques permettant d'évaluer ces quantités doivent être analysées.

Elles se subdivisent en deux catégories :

1. Les techniques de mesure.
2. Les techniques de modélisation.

Les techniques de mesure feront l'objet d'une étude approfondie puisqu'elles sont basées sur le mesurage direct du système. Celui-ci doit donc exister et être disponible.

Les techniques de modélisation se différencient des premières par le fait qu'elles n'ont pas besoin que le système existe et par le fait qu'elles constituent une représentation approximative de la réalité. Les résultats sont basés sur un modèle du système.

Ce dernier type de techniques ne fera pas l'objet d'une étude approfondie.

Les techniques de mesure utilisent des outils qui seront décrits au chapitre 3.

Auparavant, nous définirons et étudierons la charge de travail qui détermine le taux d'activité d'un système et sous laquelle ces mesures seront effectuées.

CHAPITRE 2. LA CHARGE DE TRAVAIL

I. Définition de la charge de travail.

Par charge de travail, nous entendons l'ensemble des programmes, données et commandes soumis par les utilisateurs au système, à chaque instant.

Les performances de toute étude d'évaluation sont exprimées par des quantités qui dépendent de la charge de travail. Pour évaluer avec précision les performances d'un système, il est donc nécessaire de spécifier ou du moins connaître la charge de travail.

La comparaison entre différents systèmes nécessite de pouvoir disposer d'une charge de travail identique car toute variation de la charge de travail provoque inévitablement une altération dans l'utilisation des ressources.

Les relations entre les performances, le système et la charge de travail sont relativement difficiles à exprimer puisque cette dernière varie sans cesse.

Les causes principales de cette variation sont :

* des causes intrinsèques aux programmes de la charge de travail. Ainsi, le temps d'exécution d'un programme peut varier avec les données introduites, l'allocation de ses fichiers sur disque, la position du bras du disque lors de l'arrivée d'une demande de lecture ou d'écriture, le degré d'occupation des composants hardware et software que le programme désire acquérir,...

* des causes relatives à l'organisation et aux caractéristiques des applications du système à évaluer.

* certains phénomènes de "feedback". Généralement, deux niveaux de feedback sont définis (Fig 2.1).

  Le niveau 1 de feedback est dû aux performances du système et aux combinaisons de la charge sur le comportement des utilisateurs et donc sur la charge de travail elle-même.

  Le niveau 2 de feedback est celui implémenté par les algorithmes de contrôle de l'OS. On y trouve par exemple la modification dynamique des priorités, l'allocation de l'espace mémoire variable en fonction de la classe de certains programmes.

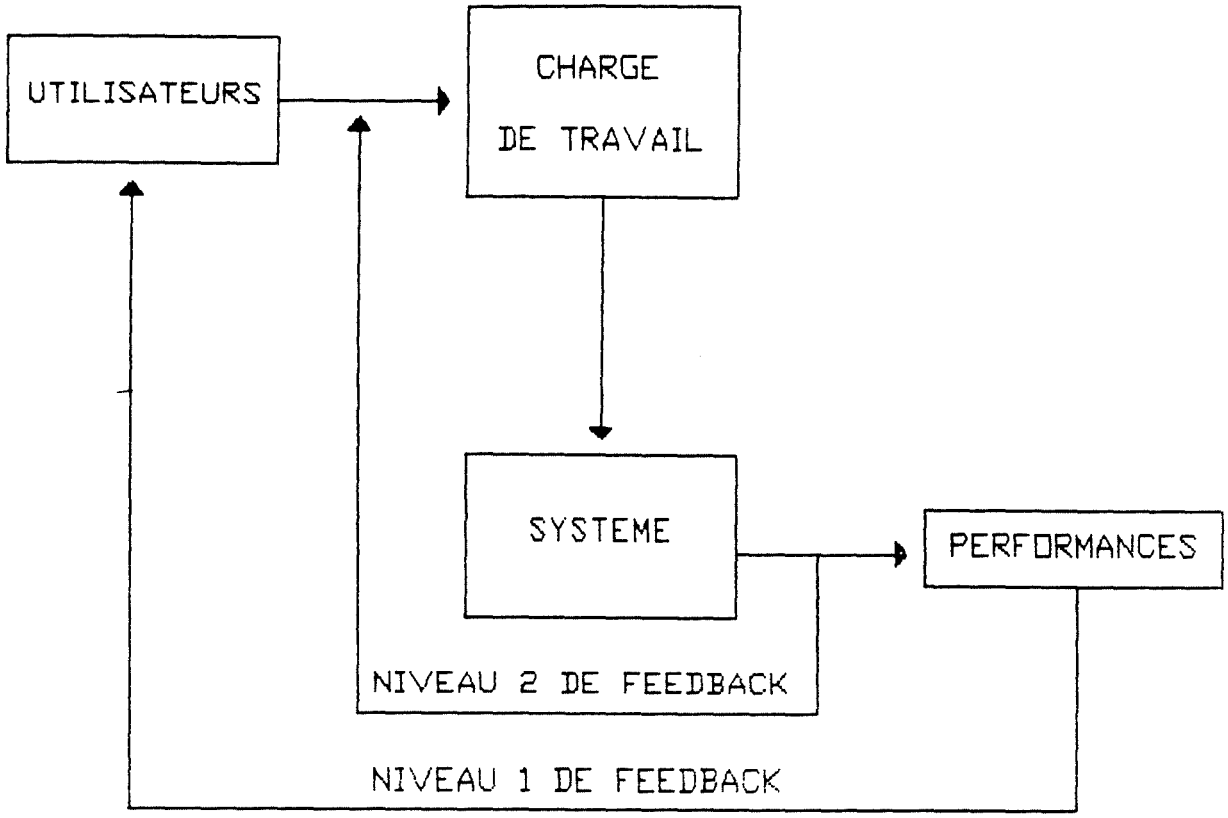


Figure 2.1 : Influence des phénomènes de feedback sur la charge de travail.

II. Implémentation d'un modèle de la charge de travail.

Cette section reprend les diverses opérations à effectuer pour concevoir et implémenter un modèle de la charge de travail (Fig 2.2.).

Celles-ci peuvent être groupées en trois phases :

1. La formulation.
2. La construction.
3. La validation.

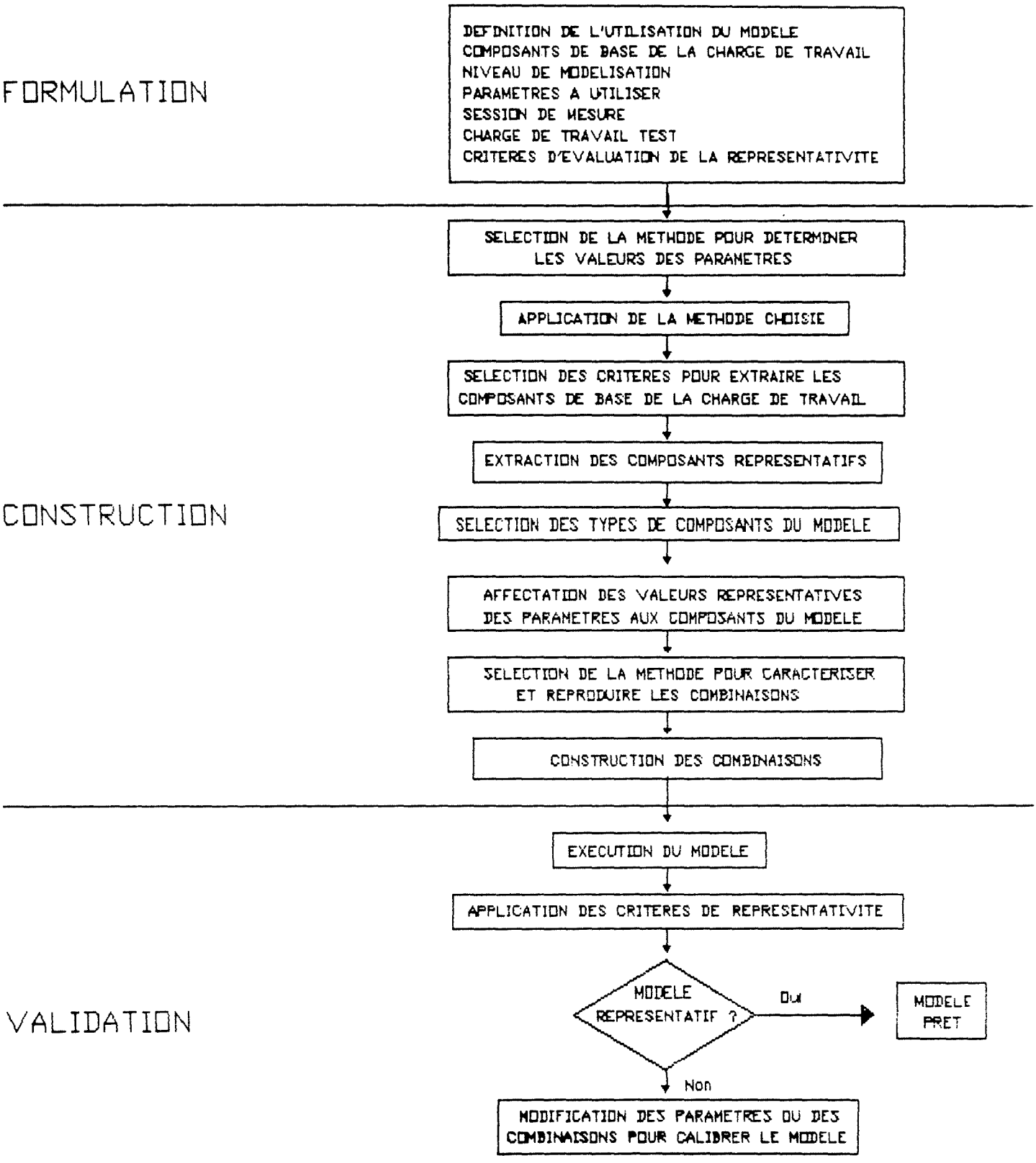


Figure 2.2 : Phases de conception et d'implémentation d'un modèle exécutable de la charge de travail.

1. La formulation.

Cette phase consiste principalement à prendre un certain nombre de décisions. Ces décisions seront fortement influencées par les objectifs de l'étude d'évaluation.

La phase de formulation peut être subdivisée de la manière suivante :

1.1. Définition de l'utilisation escomptée du modèle.
1.2. Définition des composants de base de la charge de travail.
1.3. Sélection du niveau de modélisation.
1.4. Détermination des paramètres à utiliser.
1.5. Session de mesure.
1.6. Charges de travail tests.
1.7. Critères pour l'évaluation de la représentativité du modèle de la charge de travail.

1.1. Définition de l'utilisation escomptée du modèle.

Définissons tout d'abord ce qu'est la caractérisation d'une charge de travail. Il s'agit de la description quantitative des caractéristiques de celle-ci.

La première opération consiste à définir l'utilisation du modèle et donc l'utilisation que l'on souhaite faire de la caractérisation.

1.2. Définition des composants de base de la charge de travail.

La définition des composants de base de la charge de travail consiste à identifier le plus petit niveau de détail que l'on désire modéliser.

Les composants de base qui peuvent être sélectionnés sont les programmes, les commandes, les instructions machine,...

Si notre choix se portait sur les programmes, les utilisateurs seraient caractérisés en terme de consommation globale des ressources réalisées par leurs programmes. Dans ce cas, nous ne tiendrions pas compte des différences entre programmes en ce qui concerne les demandes de ressources de chaque instruction du programme.

1.3. Sélection du niveau de modélisation.

On distingue généralement trois niveaux de modélisation d'une charge de travail :
* le niveau physique,
* le niveau virtuel,
* le niveau fonctionnel.

* Au niveau physique, le modèle de la charge de travail est basé sur les consommations ou les taux de consommation des ressources hardware et software du système.

La caractérisation est orientée "ressources" et chaque composant de base peut être caractérisé, par exemple, par le temps CPU consommé, l'espace mémoire requis, le nombre d'instructions exécutées,...

* Au niveau virtuel, ce sont les ressources logiques consommées qui sont considérées.

  Par exemple, chaque composant de base peut être caractérisé par le nombre et le type de commandes interactives, l'espace mémoire virtuel requis,...

* Au niveau fonctionnel, la charge de travail est caractérisée par ses applications. Le modèle doit contenir des programmes qui exécutent les mêmes fonctions que la charge de travail originale.

1.4. Détermination des paramètres à utiliser.

Rappelons que la caractérisation de la charge de travail est définie comme la description quantitative des caractéristiques de celle-ci.
Cette caractérisation est généralement réalisée à l'aide de paramètres quantitatifs.

La spécification des objectifs de l'opération de caractérisation permet de déterminer ces paramètres.

Les paramètres qui caractérisent une charge de travail sont généralement subdivisés en plusieurs classes suivant qu'ils se réfèrent à des composants de base de la charge de travail ou à la charge de travail complète.

PARAMETRES	DESCRIPTION
Pour les composants de base de la charge de travail	
Temps CPU	Temps CPU total consommé par le programme.
Opérations d'E/S	Nombre total d'opérations d'E/S effectuées.
Espace mémoire	Espace mémoire demandé par le programme.
Lignes imprimées	Nombre total de lignes imprimées.
Priorité externe	Priorité assignée par l'utilisateur.
Fichiers disque	Nombre total de fichiers disque utilisés.
Burst CPU moyen	Temps CPU moyen entre 2 opérations d'E/S.
Pour la charge de travail complète	
Taux d'arrivée	Inverse du temps moyen entre les arrivées de deux requêtes successives pour la ressource considérée.
Distribution du temps entre arrivées	Distributions des temps entre les arrivées de deux requêtes successives pour la ressource considérée.
Pour des charges de travail interactives.	
Nombre d'utilisateurs	Nombre d'utilisateurs connectés simultanément.
Temps de réflexion de l'utilisateur	Temps entre l'arrivée d'un message du système à un terminal et le début de l'introduction de la commande suivante par l'utilisateur.
Intensité des requêtes	Ratio entre le temps de réponse moyen et le temps de reflexion moyen.

1.5. Session de mesure.

La période pendant laquelle les mesures doivent être réalisées est déterminée à cette étape. En fonction des objectifs, nous serons amenés à choisir des périodes de pointes ou des périodes pendant lesquelles des programmes ayant certaines spécificités s'exécutent. La durée minimum dépendra du degré de précision que l'on souhaite obtenir.

1.6. Charges de travail tests.

La charge de travail test est la charge que le système traite pendant que l'on effectue des mesures du système. Elle peut toujours être considérée comme un modèle de la charge de travail réelle.

Les catégories de charges de travail tests sont :

* Les charges de travail réelles tests.
* Les charges de travail synthétiques tests.
* Les charges de travail artificielles tests.

* Les charges de travail réelles tests. Ce sont les données et programmes originaux traités pendant une session de mesure. Elles sont les plus représentatitives et les moins onéreuses. Elles peuvent être considérées comme des modèles de la charge de travail réelle dans la mesure où leur durée est généralement inférieure à la charge de travail à représenter.

Le seul choix à effectuer est la partie de la charge de travail réelle qui doit faire l'objet d'une étude d'évaluation. Il s'agit de la détermination de l'instant auquel une session doit commencer et finir.

* Les charges de travail synthétiques tests. Elles peuvent être subdivisées en deux classes :

  - les charges de travail synthétiques naturelles (benchmarks) qui sont un sous-ensemble des composants de base (programmes, commandes interactives) de la charge de travail réelle.

  - Les charges de travail synthétiques hybrides qui sont composées à la fois de composants de la charge de travail réelle et de composants conçus spécialement à cet effet.

Ce type de charges de travail tests est habituellement utilisé pour comparer différents systèmes et est confronté à des problèmes externes qui peuvent avoir une forte influence sur les performances mesurées sur ces systèmes.

Par exemple :

  - Les priorités d'exécution. Chaque système traite les priorités de façon différente. Certains systèmes altèrent la priorité attribuée par le scheduler pendant l'exécution.

- Le degré maximum de multiprogrammation. Ce paramètre varie de système à système et influence considérablement les indices de performance.

- Les routines de logging. Ces routines fournissent souvent les données utilisées dans les études d'évaluation. Chaque système a ses propres routines qui attribuent parfois un nom similaire à des variables n'ayant pas la même signification.

- Les paramètres de génération de l'O.S. A la génération du système, l'O.S. assigne des valeurs à certains paramètres qui influencent fortement les performances du système.

* Les charges de travail artificielles test. Il s'agit d'une charge de travail qui n'utilise pas de composant de la charge de travail réelle.

On distingue :

- les modèles artificiels exécutables,
- Les modèles artificiels non-exécutables (entrées d'un modèle analytique ou de simulation d'un système).

Comparons les caractéristiques principales des modèles de charge de travail artificielle avec les modèles de charge de travail réelle.

Caractéristiques	Modèles de charge de travail réelle	Modèles de charge de travail artif.
Représentativité	Virtuellement élevée	Habituellement plus basse
Coût d'implémentation	Bas	Plus élevé
Coût d'utilisation	Bas (pas d'interruption de service)	Plus élevé
Flexibilite	Faible	Elevée
Reproductibilité	Faible	Elevée
Compacité	Faible	Elevée
Indépendance % système	Faible	Habituellement plus forte
Coût de collecte et de réduction des données	Elevé	Plus faible

1.7. Critères pour l'évaluation de la représentativité du modèle de la charge de travail.

La représentativité d'un modèle correspond à la précision de ce modèle par rapport à la charge de travail réelle du système.

Un certain nombre de critères, dépendant du niveau de modélisation adopté, peuvent être définis pour évaluer la représentativité d'un modèle (M) de la charge de travail (C).

Généralement, on distingue les quatre critères suivants :

1. M est un modèle représentatif parfait de C s'il demande les mêmes ressources que C et dans les mêmes proportions.

2. M est un modèle représentatif parfait de C s'il demande les mêmes ressources que C et dans les mêmes taux.

3. M est un modèle représentatif parfait de C s'il exécute les mêmes fonctions dans les mêmes proportions que C.

4. M est un modèle représentatif parfait de C s'il produit les mêmes valeurs pour les indices de performance de C quand il tourne sur le même système.

Si la caractérisation de la charge de travail est basée sur la temps CPU total consommé et le nombre d'opérations d'E/S, le premier critère est satisfait si les ratios entre les valeurs de ces deux paramètres dans C et M sont identiques.
Le second critère spécifie par contre que la durée moyenne des bursts CPU et E/S de C doivent être égales à celles de M.

Si la charge de travail à modéliser est constituée de 400 heures de compilation, de 200 heures de tests et de 200 heures de calculs scientifiques, le critère 3 sera satisfait si M est par exemple constituée de 20 minutes de compilation, de 10 minutes de tests et de 10 minutes de calcul scientifique.

Le quatrème critère est plus orienté "performances".
L'utilisation de ce critère permet d'évaluer la validité d'un modèle en se basant sur les effets qu'il produit sur le système et pas seulement sur les causes possibles.

Il n'y a pas de critère unique pour construire et évaluer les modèles de la charge de travail. Le choix du niveau de modélisation et des paramètres de caractérisation est réalisé en fonction des objectifs de l'étude.

En résumé, la phase de formulation reprend un ensemble d'opérations qui sont principalement constituées de choix influencés par les objectifs de l'étude d'évaluation.

Après la définition des objectifs, nous avons celle des composants de base de la charge de travail. Les paramètres à utiliser pour caractériser les composants de base sont sélectionnés sur base du niveau de modélisation souhaité (déterminé après l'étude des objectifs) et la vérification de la disponibilité de ces paramètres.

Le nombre de ces paramètres doit être choisi judicieusement. L'objectif de l'analyse des valeurs de ceux-ci sera généralement d'identifier les affinités existant entre ces valeurs et de regrouper les composants présentant des caractéristiques similaires. Si le nombre est trop petit, il est difficile d'identifier des classes de composants distinctes. Le problème inverse se pose si le nombre est trop élevé.

2. Construction du modèle.

Cette phase est généralement constituée des quatre opérations fondamentales suivantes :

2.1. L'analyse des paramètres.

2.2. L'extraction des valeurs représentatives.

2.3. L'assignation des valeurs représentatives aux composants de base.

2.4. La reconstruction des combinaisons des composants significatifs.

2.1. L'analyse des paramètres.

Nous avons défini l'ensemble des paramètres à utiliser pour caractériser les composants de base de la charge de travail.

L'opération suivante consiste maintenant à identifier la méthode pour dériver les valeurs des paramètres pour les composants de base.

Comme le nombre de données collectées est habituellement considérable, leur analyse, effectuée dans le but d'identifier des groupes de composants avec des caractéristiques similaires, est souvent réalisée avec des techniques statistiques.

2.2. L'extraction des valeurs représentatives.

L'examen des valeurs des paramètres de la charge de travail réelle doit nous permettre de définir les valeurs qui appartiennent aux composants représentatifs de la charge de travail.

Le nombre de composants contenu dans un modèle influence directement la représentativité du modèle et sa compacité.

2.3. L'assignation des valeurs représentatives aux composants du modèle.

L'objectif de cette opération est de transformer des valeurs représentatives en composants exécutables.

Par exemple, si les composants de base de la charge de travail réelle doivent être utilisés, nous devons essayer de trouver les composants dont les valeurs des paramètres se rapprochent le plus des valeurs des composants représentatifs.

2.4. La reconstruction des combinaisons des composants significatifs.

Il s'agit de reproduire dans le modèle des situations similaires à celles que la charge de travail réelle produit.

3. Validation.

La phase de validation utilise les critères d'évaluation de la représentativité du modèle. La validité du modèle est donc analysée.

Après avoir conçu et implémenté un modèle exécutable de la charge de travail, il nous reste à évaluer les performances du système. Pour ce faire, des outils d'évaluation doivent être utilisés. Ces outils se basent sur les techniques décrites au chapitre suivant.

CHAPITRE 3 : OUTILS, PRINCIPES ET TECHNIQUES DE MESURES

Ce chapitre décrit les différents types d'outils que l'on peut utiliser pour mesurer un système. Les principes et techniques de mesure sur lesquels ces outils se basent seront également étudiés.

I. Les principes et techniques de mesure.

Lorsque l'on évalue les performances d'un système, un ensemble de données sont collectées afin de décrire l'activité du système de façon quantitative.

Ces observations de l'activité du système doivent permettre d'explorer les relations existant entre les événements afin de les expliquer et de les prédire.

Mesurer un système permet donc de traiter des phénomènes de façon quantitative.

On distingue généralement deux catégories de mesures :

- Les mesures demandées par les utilisateurs d'un système. Cette catégorie reprend toutes les mesures effectuées en vue d'évaluer les performances du système et de déterminer l'utilisation de ses ressources.

- Les mesures demandées par le système lui-même. Cette catégorie reprend toutes les mesures utilisées par le système afin de s'adapter dynamiquement aux facteurs conditionnant son activité. Ces mesures doivent normalement permettre au système de maintenir un niveau suffisant de performances.

Bien souvent, les mesures pouvant être effectuées sur un ordinateur sont restreintes par le fait que ni le hardware, ni le software d'un ordinateur n'est conçu à l'origine pour être mesuré. Pour vaincre ces restrictions, il est parfois nécessaire de modifier certaines fonctions du système ou d'ajouter des instruments coûteux.

Il existe différentes techniques permettant de mesurer un système. Le type d'analyse désirée ainsi que le niveau auquel on désire l'effectuer déterminent le choix de la technique.

On distingue deux types d'analyse :

- une analyse macroscopique. Dans ce cas, seuls des indices globaux tels que le turnaround time, le temps de réponse moyen,... nous intéressent.

- une analyse microscopique. Cette analyse est plus détaillée que la précédente et peut être requise lorsque, par exemple, on doit déterminer l'utilisation du CPU de chaque type d'instruction.

Les diverses techniques appliquées d'après le type d'analyse désirée et permettant de mesurer un système sont :

1. La détection d'événements.
2. L'échantillonnage.
3. La simulation.

1. La technique de détection d'événements :

Un événement peut être considéré comme un changement d'état du système. Des données sur certaines activités du système peuvent être collectées en enregistrant tous les événements associés à ces activités.

On distingue différents types d'événements :

- Les événements corrélés au software ou "software event". Ce sont les événements associés à une fonction d'un programme.

  Un événement de ce type peut se produire quand un programme atteint un certain stade de son exécution (début d'une opération d'E/S).

- Les événements corrélés au hardware ou "hardware event". Ce sont les événements indépendants du contenu logique du programme en exécution (mouvements du bras d'un disque).

Le principe de la détection d'événements software consiste à insérer un code spécial à certains endroits spécifiques de l'OS. Le déclenchement d'un évènement provoquera, via ce code, un transfert vers une routine appropriée. La fonction de cette routine sera d'enregistrer l'occurence de l'événement, de stocker les données interéssantes dans un tampon , données qui seront sauvées par la suite sur disque ou bande, et finalement de rendre la main à l'O.S.

L'ensemble des instructions et données utilisées à cet effet constituent un MONITEUR. Ce moniteur est donc un mécanisme permettant d'acquérir des données sur l'activité du système et est appelé moniteur "event driven" puisqu'il est activé par l'occurence d'un événement.

A la fin de la période d'observation, un programme d'analyse peut traiter les données enregistrées et effectuer une réduction de ces données afin de pouvoir les interpréter plus facilement.

Cette technique présente l'avantage de pouvoir fournir un grand nombre d'informations sur certains aspects du comportement du système, mais elle peut également conduire à la collecte d'une masse exorbitante de données.
Il est donc nécessaire de pouvoir sélectionner les événements à intercepter, d'une part pour permettre une réduction de données acceptable et d'autre part, pour ne pas ralentir le système au delà de certaines limites acceptables. Le tampon est aussi un élément critique car lorsque celui-ci est plein, il doit être transféré sur disque ou sur lecteur de bandes. Le nombre de ces transferts, qui est fonction de la fréquence des événements, peut également ralentir le système.

Cette technique est peu utilisée car elle est coûteuse en terme de ressources du système consommées et elle nécessite des modifications dans l'O.S.

2. La technique d'échantillonnage :

Moins dépendante du système, cette technique consiste à interrompre ce dernier, à intervalles réguliers, pour détecter l'état de certains de ses composants. La précision de la technique d'échantillonnage dépend du nombre d'échantillons collectés.

La méthode statistique d'échantillonnage n'examine donc qu'une partie de la population. Cependant, il est souvent possible d'estimer, avec un degré de précision élevé, certains paramètres caractéristiques de la population.

Les avantages offerts sont les suivants :

- le nombre de données produites est moins élevé. Cela a pour effet de simplifier et réduire l'analyse de celles-ci.

- la collecte des données par échantillonnage est moins contraignante pour le système. L'altération des performances du système due à l'outil de mesure ne doit plus être prise en considération.

La technique d'échantillonnage peut poursuivre deux objectifs différents :

1. Mesurer, dans un intervalle de temps donné, combien de temps chaque composant d'un système passe dans ses différents états.

   La détermination de ce qui se passe dans l'intervalle, de l'interaction des composants les uns avec les autres,... peut être réalisée par une analyse a posteriori des données collectées pendant la session de mesure.

   Des outils de mesure spécifiques évaluant l'efficacité avec laquelle les ressources sont employées, permettent d'atteindre ce premier but.

   La taille de l'échantillon détermine la précision des résultats.

2. Suivre l'évolution du système et prévoir son comportement afin de prendre des mesures ayant une influence positive sur ses performances.

   Cet objectif peut être atteint lorsque des fonctions de mesure font partie intégrante du processus de contrôle de l'activité de l'O.S.

   Un grand nombre de fonctions fournies par l'O.S. doivent s'adapter dynamiquement aux variations des conditions d'opération de l'O.S. Ces variations au niveau de l'O.S. sont induites par celles de la charge de travail.

3. La technique de simulation :

La technique de simulation consiste à mesurer un modèle du système. Les états du système sont reproduits et les transitions d'état sont étudiées afin de représenter le comportement dynamique du système.

Un simulateur a comme principal avantage de ne pas nécessiter l'existence ou la disponibilité du système.
Par contre, il est très difficile d'établir si les résultats obtenus sont suffisament précis.

II. Les outils de mesure.

Lorsque l'on observe l'activité d'un système, on utilise généralement des outils de mesure qui permettent de quantifier les résultats des diverses observations réalisées.

On distingue habituellement deux types d'outils :

1. Les moniteurs software. Ces outils sont constitués de programmes permettant de détecter les états du système.

2. Les moniteurs hardware. Ces moniteurs utilisent des appareillages électroniques connectés à certains endroits spécifiques du système afin de caractériser certains phénomènes. Ces appareils ont pour objet de détecter des signaux.

Les moniteurs hardware et software présentent chacun des avantages et des inconvénients. Ces deux types de moniteurs ne peuvent pas être réellement comparés dans la mesure où le choix du moniteur dépend du type de phénomène que l'on désire caractériser.

Toutefois, un moniteur software est généralement moins précis qu'un moniteur hardware. Ce dernier type de moniteur n'interfère pratiquement pas avec le système tandis qu'un moniteur software accroît la charge du système et modifie donc ses performances.

Il est donc nécessaire de réduire au maximum l'overhead system que ces moniteurs software engendrent.

Notre étude d'évaluation utilisera un moniteur software. Ce dernier est basé sur la technique d'échantillonnage. Les données sont collectées en analysant l'état des composants du système à certains moments prédéterminés.

CHAPITRE 4. METHODOLOGIE ET ETUDE DE CAS

Pour tenter d'améliorer l'efficacité d'un système, nous devons en analyser le fonctionnement et réaliser une étude d'évaluation des performances.

Nous avons déjà défini qu'une étude d'évaluation peut être considérée comme une procédure itérative essentiellement composée d'une phase de diagnostic et d'une phase de thérapie (cfr chap 1, Fig 1.1).

Nous allons étendre et spécifier davantage ce diagramme afin de détailler les différentes phases d'une méthodologie d'amélioration des performances (Fig 4.1). Les phases suivantes regroupent les différentes opérations à effectuer.

I . Problématique.

II . Définition des objectifs.

III. Caractérisation de la charge de travail.

IV . Instrumentation.

V . Conception des tests et expérimentation.

VI . Analyse des résultats.

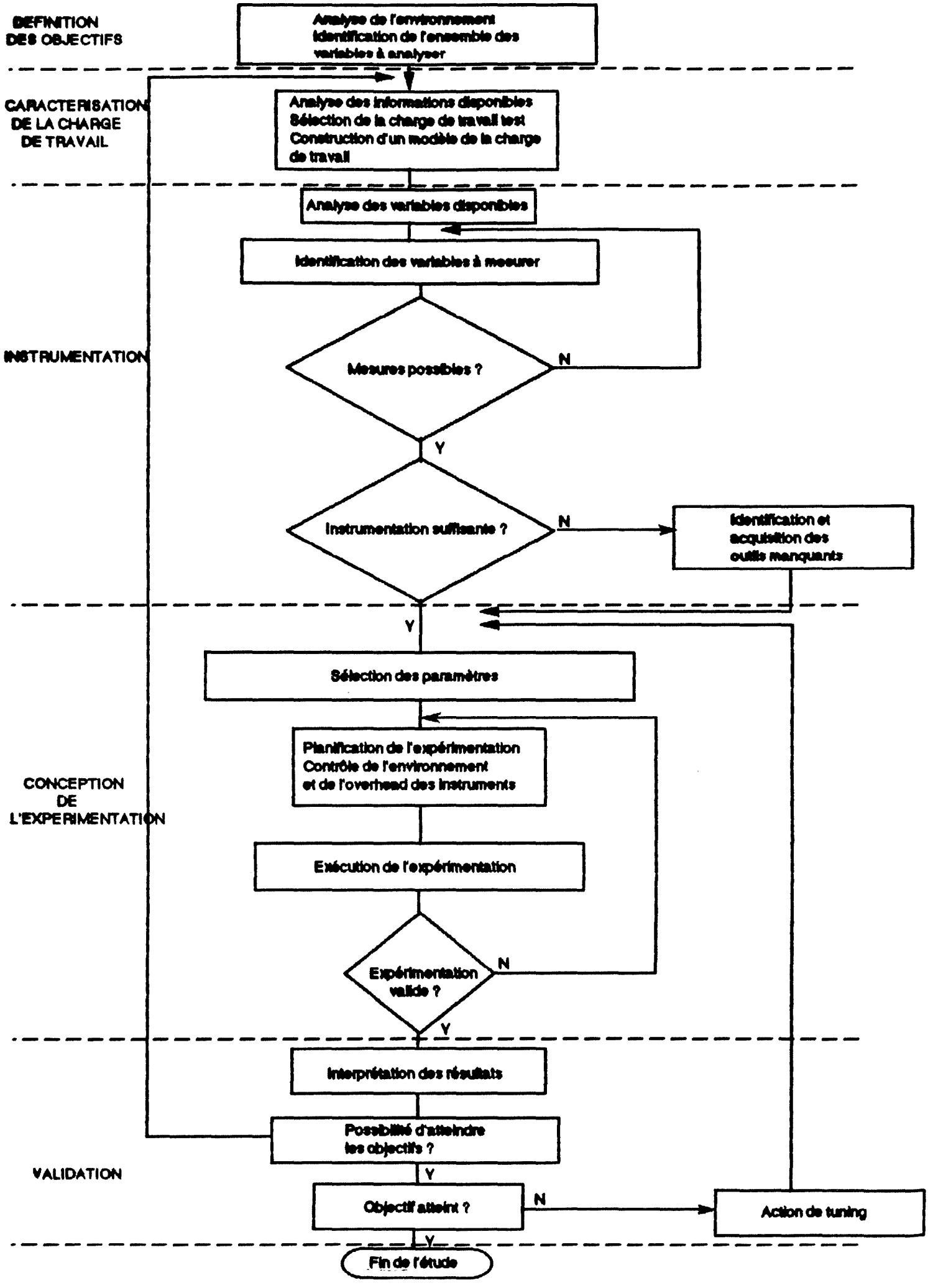


Figure 4.1. Phases d'une méthodologie d'amélioration des performances

I. Problématique.

L'étude que nous avons entamée est partie d'un problème rencontré chez A, grossiste en matériel de bureau.

Cette société, voulant accroître les performances de son système, a souhaité l'installation d'un tape streamer. Le but de ce dernier était évidemment de faire des backups du disque dur dans un délai plus court que les bandes et, de plus, de permettre que ces backups s'effectuent sans qu'aucune présence humaine ne soit requise.

Afin de satisfaire cette requête, la société ayant informatisé la société A a installé un tape streamer et une nouvelle version de l'OS, l'OS 7.40, l'ancienne version étant incapable de gérer les tape streamer. Sous ce nouvel OS, les copies de sauvegarde s'effectuent beaucoup plus rapidement et de manière autonome. Cependant un gros problème s'est présenté : les performances du système ont chuté au point d'en gêner les utilisateurs.

Nous allons donc tenter de localiser le problème et d'en établir les causes. La localisation exacte du problème nécessite une analyse de la configuration ainsi que diverses rencontres avec les personnes intéressées, à savoir, la société A, B (société responsable de l'installation, de la maintenance du système et du développement et de la maintenance des programmes d'application) et C (société constructeur du système utilisé chez A).

De plus, une première analyse des données collectées par des routines de logging permet de faire apparaître des aspects et problèmes inconnus (cfr infra, 6.1).

Ces problèmes sont généralement subdivisés en quatre classes :

- analyse de l'utilisation hardware,
- analyse de la charge de travail,
- analyse de l'efficacité des programmes,
- recherche des goulots d'étranglement.

Nous étudierons ces diverses classes à l'aide d'outils et techniques adéquats.

Dans un premier temps, nous ne pouvons que supposer et non déterminer avec précision les causes effectives des problèmes d'AGEROPA. Nous allons donc d'abord cerner le problème avant de l'identifier avec précision.

La première analyse doit permettre une première identification des problèmes. Les analyses suivantes portent principalement sur cette identification plus précise qui se fera communément avec B et/ou C. Toutes ces analyses permettent de localiser les problèmes et tentent d'apporter des solutions conformément aux objectifs que nous allons définir.

II. Définition des objectifs de l'étude d'évaluation des performances.

La seconde phase de définition des objectifs est nécessaire dès que des décisions doivent être prises quant à la méthode pour analyser les performances du système, son degré de précision et la quantité de ressources à utiliser.

Nous ne pouvons répondre de manière satisfaisante à ces problèmes si les objectifs de l'étude d'évaluation n'ont pas été préalablement identifiés.

En pratique, une étude de performances est généralement effectuée non pas après une analyse de la situation, mais après des suspicions ou inefficiences, par le sentiment que les performances du système sont moins élevées que celles attendues, ou par des plaintes émanant des utilisateurs. Avec des données aussi vagues, nous ne pouvons espérer découvrir immédiatement LE problème et LA solution adéquate. Il est donc préférable de choisir des objectifs plus modestes.

L'objectif général de notre étude concerne un problème de temps de réponse "insatisfaisant" en période d'utilisation "normale" du système.

Partant d'une définition aussi peu précise, des objectifs qui s'occupent d'aspects plus spécifiques et qui sont donc mieux définis, sont :

- réduire le temps de réponse moyen des terminaux.

- déterminer si l'activité de pagination est trop élevée et requiert certaines interventions (par exemple, une extension de la taille de la mémoire centrale).

- déterminer la meilleure allocation des volumes et fichiers sur les disques (par exemple, ajouter un disque afin de mieux répartir la charge).

- déterminer le taux d'utilisation du CPU afin d'établir si celui-ci est saturé.

Les objectifs suivants doivent aussi être considérés :

- vérifier s'il est possible d'éviter ou de postposer l'acquisition de nouveaux composants hardware.

- tenter de réduire l'overhead système et autres activités non productrices.

- essayer de réduire la charge de travail existante.

Après cette première définition des objectifs, l'environnement doit être étudié. Ce dernier comprend :

II.1. La configuration de la société A,

II.2. le hardware,

II.3. le système d'exploitation,

II.4. la charge de travail.

II.1. Configuration de la société A.

La configuration du système de la société A est la suivante :

* Hardware :

1. Système central :

- 1 ordinateur du type MIVRS, dont les caractéristiques techniques sont reprises en annexe B, avec une mémoire centrale de 6 MB.
- 1 cabinet d'alimentation.
- 1 unité disque fixe 354 MB.
- 1 unité bande magnétique 1600 BPI.
- 2 processeurs d'E/S - IAC/16 de 16 lignes asynchrones.

2. Périphériques :

- Ecrans-claviers : * 1 terminal D/411.
  * 2 micro-ordinateur.
  * 19 terminaux D/215.
- Imprimantes :
  * 4 imprimantes de 200 CPS.

* 2 imprimantes de 350 CPS.

* 1 imprimante de 400 CPS.

* 1 imprimante laser.

II.2. Le hardware.

Le système (que nous appelerons S) est constitué d'un double board CPU, d'une mémoire extensible jusqu'à 8 MB et d'un certain nombre de contrôleurs d'entrées/sorties. Ces E/S s'effectuent via le "Burst Multiplexor Channel", qui est un système d'E/S à très grande vitesse (accès de 5 Mégabytes/sec), et via un bus d'E/S spécial, qui est un système I/O à vitesse moyenne (taux de transfert de 2,5 Mégabytes/sec en entrée et de 1,25 Mégabytes/sec en sortie).

Les caractéristiques de l'architecture 32 bits de S sont :

- Un espace d'adressage logique de 4 Gigabytes.
- Une mémoire segmentée avec protection en anneau.
- Un ensemble d'instructions avec manipulation de données de 32 bits.
- Un maximum de 64 utilisateurs.

S possède donc un système de mémoire virtuelle puisque la mémoire logique est nettement supérieure à la mémoire physique. Ces 4 Gigabytes d'adressage logique sont segmentés en 8 segments de 512 Mégabytes, chacun de ces segments étant entouré d'un anneau de protection.

II.3. Le système d'exploitation : OS/VS (Operating System/Virtual storage).

L'OS/VS est un système d'exploitation à mémoire virtuelle et à demande/allocation de pages conçu pour les ordinateurs 32 bits de la société C.

Ses caractéristiques sont :

- Utilisation d'un espace d'adressage logique très grand.
- Gestion de mémoire en utilisant des techniques de demande/allocation de pages.
- Multiutilisateurs et multiprogrammation.
- Système de protection en anneau.
- ...

Ce système d'exploitation fait partie intégrante de l'espace d'adressage logique des utilisateurs. Il délègue certaines fonctions à des processeurs auxiliaires.

Certaines des spécificités d'AOS/VS ont été analysées afin d'évaluer l'impact des modifications que l'on devra effectuer. Le lecteur interessé trouvera en annexe B une description du fonctionnement du système S.

II.4. La charge de travail.

D'un point de vue général, la charge de travail est essentiellement composée de :

1. Logiciels de base :

   - 1 OS/VS (système d'exploitation).
   - 1 Business BASIC.

2. Logiciels comptables :

   - 1 logiciel de comptabilité générale.
   - 1 logiciel de comptabilité analytique.
   - 1 logiciel de tableau de bord de gestion sur la comptabilité générale et analytique.

3. Logiciels commerciaux :

   - 1 logiciel de gestion commerciale comprenant :
     * La gestion de stocks.
     * La gestion des commandes clients.
     * La gestion des commandes fournisseurs.
     * La gestion du magasin.
     * La gestion des représentants.
     * Les statistiques de vente.
     * La gestion des offres.
   - 1 logiciel de traitement de texte.

III. Caractérisation de la charge de travail.

Le système à évaluer existant et étant disponible, nous pouvons utiliser une techique basée sur des mesures directes du système à évaluer. Avant de décrire les outils de mesure utilisés ainsi que les techniques sur lesquelles sont basés ces outils, nous devons caractériser la charge de travail. Par caractérisation, nous entendons une description quantitative de la charge de travail.

L'étape de caractérisation doit être précédée par la détermination des composants de base de la charge de travail. Ces composants de base sont les programmes d'application. Un tel choix permet d'évaluer l'impact d'un utilisateur sur les performances du système. Nous pourrons ainsi déterminer quel utilisateur a tendance à saturer les ressources....

Les paramètres nécessaires caractérisant la charge de travail dépendent du niveau de modélisation choisi. Nous parlerons de niveau "physique-virtuel" car nous allons mesurer les consommations de ressources au niveau physique et également les consommations de ressources au niveau des programmes d'application.

IV. Choix des instruments.

Après une définition des objectifs de l'étude, nous devons mesurer tout ce qui peut l'influencer. Nous collecterons des données sur les activités du système par des techniques et outils adéquats. Nous devons déterminer quelles techniques et quels outils utiliser. Nous devons également déterminer si toutes les variables à analyser

sont mesurables et, si oui, si elles peuvent être mesurées avec suffisament de précision.

Les domaines d'investigations sont les suivants :

- les composants hardware du système,
- le software système,
- la charge de travail que nous définissons comme l'ensemble des programmes que le système doit traiter pendant une période donnée.

Nous avons utilisé un moniteur software permettant d'obtenir un ensemble d'informations sur certaines activités hardware et software du système et sur la charge de travail.

1. Hardware.

La détermination de l'utilisation de chaque composant hardware d'un système est essentielle pour savoir s'il est surchargé, souschargé, ou correctement utilisé. Nous devons déterminer quels sont les composants provoquant des goulots d'étranglement et quels sont ceux presque toujours inactifs ou non utilisés.
Une amélioration des performances peut être obtenue par l'élimination des goulots d'étranglement et par une distribution plus équilibrée des charges sur les composants hardware.
Cette détermination de l'utilisation des composants est aussi importante afin d'établir si le système peut accepter une augmentation de la charge de travail sans une dégradation trop importante des performances.

Les composants hardware qui seront analysés sont :

- le CPU,
- les BUS,
- les unités de contrôle,
- les disques,
- les imprimantes.

2. Software.

L'overhead du système influence toujours les performances du système autant que sa capacité. L'overhead peut varier considérablement avec les caractéristiques de la configuration, en particulier avec la taille de la mémoire disponible.

La connaissance de l'utilisation des modules du système peut être très utile quand on choisit les paramètres de génération du système.
Il est utile de savoir comment l'OS utilise l'espace d'adressage disque (par exemple, l'espace des pages) en vue de choisir au mieux les valeurs de ses paramètres.
Cette information permet de définir efficacement l'allocation des différents espaces et des fichiers les plus souvent utilisés sur les disques.
Un goulot d'étranglement dans une fonction de l'OS limite considérablement les performances du système.

3. La charge de travail.

Nous devons sélectionner la charge de travail test. Cette dernière sera la charge de travail traitée effectivement par le système pendant les périodes d'observation.

Les motifs qui ont commandé ce choix sont les suivants : ce type de charge de travail test a une représentativité élevée et ne nécessite pas d'interruption de service. D'autre part, nous disposons d'outils permettant d'analyser avec précision cette charge de travail réelle.

L'annexe C reprend une brève description du moniteur software ainsi que les diverses valeurs que l'on peut obtenir à partir de ce moniteur et qui nous paraissent significatives.

A partir des valeurs fournies par le moniteur lors du traitement des diverses charges de travail sélectionnées, nous pourrons localiser les problèmes et proposer des remèdes. Les analyses effectuées et les résultats obtenus (cfr annexe C.1, C.2) sont étudiés au point VI.

V. Conception des tests.

Nous avons défini les objectifs de l'étude et choisi les outils. Il nous reste à sélectionner les sessions de mesures pendant lesquelles nous allons évaluer les performances du système. Une attention particulière doit être accordée à la plannification des tests de mesurage avant leur exécution. Cette activité de planning se réfère à la charge de travail spécifique définie auparavant, ainsi qu'à la configuration et l'environnement décrit précédemment.

La planification d'une session doit tenir compte de la charge de travail et de son évolution dans le temps. De plus, la prise de mesures doit uniquement être réalisée dans un environnement contrôlé c'est-à-dire que la charge de travail doit non seulement être connue, mais aucune modification hardware ne peut être effectuée pendant la session de mesures.

Nous garantirons donc cette stabilité dans la configuration hardware. En outre, nous devons tenir compte des différentes pointes possibles dans la charge de travail. Ces pointes reflètent d'une part l'activité de l'entreprise et d'autre part, les variations de la charge de travail elle-même.

Ainsi, nous distinguerons les analyses effectuées pendant :

- le traitement normal,
- la clôture,
- une charge extrême de traitement.

L'étude de la charge de clôture et de la charge extrême devra nous permettre d'évaluer s'il est possible d'effectuer certaines actions de clôture pendant la journée.

Une distinction doit encore être faite entre la première analyse et les suivantes. La première analyse permet d'identifier un certain nombre de problèmes qui donnent des lignes de conduite à explorer. Les analyses suivantes, utilisant un moniteur software, localisent de façon plus précise les problèmes, permettent de déterminer s'il existe des goulots d'étranglement, et évaluent les performances du système d'une façon plus systématique.

Des mesures effectuées avec un moniteur software doivent tenir compte des ressources utilisées par ce dernier. L'annexe C.2 montre que les consommations de ressources de ce moniteur sont peu élevées en dehors de la période d'initialisation.

VI. Analyse des résultats.

Cette phase a pour but d'analyser les résultats et en fonction de ceux-ci de mener les actions qui devraient aboutir à une amélioration des performances du système.

Nous avons collecté des informations sur l'activité du système pendant son fonctionnement.

Plusieurs analyses ont été réalisées.

- La première analyse vérifie si le changement d'OS n'est pas responsable de la dégradation des performances et si d'autres modifications de l'environnement du système n'ont pas altéré le bon fonctionnement du système.

- Le nouvel OS gère les composants du système de façon différente et ralentit considérablement les E/S. De plus, la consommation mémoire de cet OS a nettement augmenté. La place disponible pour les programmes utilisateurs s'est donc restreinte et il a été nécessaire d'accroître la capacité mémoire de la machine de 2 Mbytes.

  Après la première analyse, certaines corrections ont été apportées au système (cfr point VI.1). Cependant, celles-ci n'ont permis d'améliorer les performances que de façon peu significative pour les utilisateurs.

Considérant le fait que l'OS/VS installé doit être utilisé malgré son influence néfaste sur les performances du système, nous devons essayer d'améliorer ces dernières en tenant compte des différents objectifs fixés.

Un des objectifs de l'étude étant d'éviter ou postposer l'achat de nouveaux composants hardware, nous devons tenter d'améliorer les performances du système sans effectuer de modifications hardware.

La première analyse nous a apporté des informations sur les composants de base de la charge de travail (programmes d'application) et donc sur l'activité engendrée par les différents utilisateurs. Elle nous a permis de mettre en exergue certains problèmes. Cependant, les modifications apportées n'ont pas éliminé le problème du temps de réponse, et l'ensemble des informations obtenues n'a pas apporté suffisament d'éléments pour pouvoir localiser les points possibles d'intervention qui devraient permettre d'améliorer les performances.

Pour cette localisation, il est nécessaire de disposer d'un moniteur permettant d'obtenir une description quantitative détaillée et complète des composants du système et de la charge de travail.

Les analyses suivantes utilisent cet outil et permettent de proposer, vérifier, établir des relations entre des quantités qui caractérisent un phénomène.

Examinons maintenant en détail les diverses analyses réalisées.

1. PREMIERE ANALYSE :

- Approche : la première analyse tente de localiser le problème. Le temps de réponse s'étant degradé après l'installation du nouvel OS, il nous faut vérifier si ce dernier n'en est pas responsable. Ainsi, l'installation de l'OS doit être contrôlée et les paramètres de génération de celui-ci doivent être identiques à ceux de la précédente version.

  Si le problème de temps de réponse est toujours présent après l'installation correcte de l'OS, les ressources consommées par ce dernier doivent être étudiées. Il est en effet possible que cet OS nécessite plus de ressources que l'ancienne version ou qu'il gère de façon différente certains composants du système.

  Après ces différents contrôles, la charge de travail doit être analysée afin de vérifier qu'elle est similaire à la charge du système avant l'installation de la révision 7.60 de l'OS.

  Finalement, cette première analyse doit encore vérifier si certains composants du système ne se sont pas dégradés suite à une activité normale, indépendamment de l'installation du nouvel OS.

- Réalisation : un des symptômes du système de la société A est, en fonction d'une augmentation de l'activité du système, un temps de réponse de plus en plus long.

Par augmentation de l'activité du système, nous entendons principalement l'utilisation de programmes effectuant des statistiques et autres traitements sur les achats.

* Vérification de l'OS : le temps de réponse étant acceptable avant l'installation du nouvel OS/VS et aucune autre adjonction ou modification n'ayant été faite au système, il est possible que l'OS soit responsable de la dégradation des performances.

    1. L'OS doit avoir été installé correctement. Nous avons constaté un problème au niveau des défauts de pages et des working sets.
    Il est d'ores et déjà certain que le nombre très élevé de défauts de pages ralentit fortement le système.

    Nous avons constaté l'absence de fichiers configurant correctement et avec un maximum d'efficacité le système d'exploitation. Malgré la mise en place de ces fichiers, le temps de réponse ne s'est pas amélioré de façon significative.

    Cette modification a eu pour effet de réduire le nombre de défauts de page. Cependant, ces derniers sont encore trop élevés.

    La gestion des working sets et défauts de pages doit être analysée en profondeur afin d'améliorer la

situation actuelle. Il est possible qu'une augmentation de la capacité mémoire soit nécessaire afin de réduire les défauts de pages.

Cependant, aucune analyse des performances n'ayant été effectuée avant l'installation de cet OS, il ne s'agit pour le moment que d'une hypothèse. Afin de l'infirmer ou de la confirmer, un contact avec la société C est nécessaire.

2. Nous avons vérifié les paramètres de génération du nouvel OS. Ceux-ci sont corrects et identiques à ceux de l'ancienne révision de l'OS.

   Une modification de ces paramètres devrait améliorer les performances du système mais elle ne doit pas être envisagée à ce stade afin de pouvoir comparer les deux OS.

* La charge de travail n'a pas été modifiée entre le changement de révision. Il n'est donc pas nécessaire de l'étudier de façon détaillée lors de cette analyse.

* Afin de mettre hors cause un problème tel qu'une forte désorganisation du disque, un compactage de ce dernier a été réalisé. Une petite amélioration a été constatée mais celle-ci n'a pas permis de résorber suffisament l'attente des utilisateurs interactifs.

Pour réaliser cette étude et obtenir des chiffres représentatifs, nous avons utilisé un logiciel permettant d'obtenir un ensemble de données relatives aux programmes d'application. L'annexe D décrit ces données et reprend un exemple concret.

Un autre logiciel permet également d'acquérir des informations sur les accès disque. La fragmentation des fichiers, la longueur de la file d'attente du disque (nombre moyen de demandes à tout instant), ainsi que d'autres paramètres peuvent être obtenus. Les données et un exemple sont repris à l'annexe E.

Cependant, nous ne pouvons que localiser partiellement les problèmes et nous ne pouvons déterminer s'il y a saturation des ressources d'un point de vue global.

- Conclusion : l'étude de l'ensemble des résultats obtenus a permis de mettre en évidence quelques problèmes, sans toutefois pouvoir éliminer le problème du temps de réponse.

Les raisons de ce quasi statu quo sont les suivantes :

  - le nouvel OS gère les E/S plus lourdement et utilise beaucoup plus de mémoire. Les effets directs de ces changements dans l'OS sont un accroissement des défauts de page et une augmentation de l'overhead système pour gérer ces derniers. De plus, les E/S sorties s'effectuant plus lentement, l'exécution d'un programme tel que celui des statistiques (nécessitant un grand nombre d'E/S) a

pour effet d'altérer considérablement le temps de réponse aux terminaux interactifs. Il n'est pas encore possible de déterminer à ce stade si ces défauts de page sont causés par un manque de mémoire physique ou par l'inefficience de certains programmes.

Malgré le compactage du disque, nous constatons toujours un problème de fragmentation.

Nous sommes confrontés à un autre problème : les valeurs fournies par les petits logiciels décrits aux annexes D et E sont-elles normales? Seule une longue expérience peut nous dire si ces valeurs ne sont pas dans des limites de tolérance acceptables. D'autre part, rien ne nous permet d'affirmer que telle ou telle ressource du système est saturée.

Cette analyse se révélant insuffisante, un moniteur s'avère indispensable afin de localiser avec précision les différents problèmes.

L'analyse des chiffres obtenus à partir de ces derniers devrait alors orienter les travaux et permettre d'effectuer judicieusement les diverses opérations et/ou achats...

2. SECONDE ANALYSE :

- Approche : afin de pouvoir effectuer des copies sur un streamer, une nouvelle version de l'OS qui est moins performante pour le type d'application de la société A doit être utilisée.

  Nous allons essayer de déterminer les facteurs susceptibles d'améliorer les performances du système sans accroître la configuration hardware.

  Pour ce faire :

  * nous devons étudier la consommation des ressources d'un point de vue global afin de déterminer s'il n'y a pas de goulots d'étranglement.

  * l'overhead système doit également faire l'objet d'une étude sérieuse pour déterminer si une fonction de l'OS ne sature pas le système.

  * enfin, les consommations de ressources individuelles effectuées par les programmes d'application doivent être analysées. Ces dernières analyses doivent permettre de mettre en exergue l'inefficience de certains programmes.

- Réalisation : la seconde évaluation des performances du système a été réalisée avec l'aide d'un ingénieur système de la société C. Un logiciel de monitoring a été utilisé pour analyser les performances. L'utilisation de ce logiciel est

décrite à l'annexe C tandis que le type de résultats fournis par ce dernier apparaît à l'annexe C.1 (écrans utilisés pour la détection "on-line" des divers problèmes) et à l'annexe C.2 (graphes représentants les variations de valeurs de divers paramètres observées pendant de longues périodes d'observation).

a. En jetant un bref coup d'oeil à l'annexe C.1, nous constatons rapidement qu'un grand nombre de valeurs ne sont pas, ou peu, significatives. La première étape de l'analyse consiste donc à déterminer les valeurs pouvant apporter des lumières sur les lacunes du système. Des valeurs "clés" sont à déterminer.

   Comme nous l'avons déjà souligné à maintes reprises, les domaines qui nous intéressent sont principalement :

   - les accès disque,
   - l'utilisation du CPU,
   - l'utilisation mémoire.

b. Le moniteur nous a permis de constater :

   b.1. Un problème au niveau du disque : les performances d'un disque sont principalement influencées par trois facteurs.

Ces facteurs sont :

* l'espace libre disponible.
* la fragmentation des fichiers.
* la hash frame size des directories.

Analysons individuellement chacun de ces facteurs :

* L'espace libre disponible. La capacité résiduelle du disque est d'environ 40 % . Quand plus de 70 % des blocs du LDU (Logical Disk Unit) sont occupés, des dégradations du temps d'accès apparaissent. Nous n'avons donc pas à nous préoccuper de l'espace disque.

* La fragmentation des fichiers. Chaque fichier AOS/VS a un ou plusieurs éléments de données. Par défaut, un élément occupe quatre blocs du disque. Nous pouvons cependant spécifier un "element size" différent c'est-à-dire créer un fichier avec un plus grand nombre de blocs contigus.

  Nous avons constaté que le LDU réalisait un grand nombre de "seeks" et était donc fragmenté. La première action qui doit être réalisée pour minimiser ou éliminer la fragmentation, est de sauver tous les fichiers sur bande et de reformatter le disque.

D'un point de vue spécifique, la fragmentation d'un fichier peut être éliminée en lui attribuant un element size plus grand. L'element size des grands fichiers (de plus de 10 MB) doit donc être augmenté.

En effet, une augmentation de l'element size de certains fichiers doit permettre de réduire le nombre de niveaux d'index-système et donc de diminuer le nombre d'opérations "lecture" pour accéder à un enregistrement. Divers fichiers "données" ont un niveau d'index de 3, ce qui implique quatre opérations "lecture" pour accéder à un enregistrement.

L'element size idéal d'un fichier est défini comme suit :

$$ES = \frac{\text{taille du fichier}}{64\ KB}$$ ajusté à la puissance de 2 supérieure.

Cet ajustement (qui sera réalisé par la société B) de l'element size diminuera la fragmentation du disque.

* Le hash frame size des directories : le hash frame size est utilisé par un algorithme de AOS/VS pour déterminer où il faut enregistrer les nouveaux noms de fichiers.
Le fonctionnement de cet algorithme n'est pas essentiel, mais nous devons savoir comment utiliser cet hash frame size pour accélérer les accès aux fichiers.

Chaque directory possède un hash frame size assigné lors de la création de la directory (par défaut 7). Le meilleure hash frame size d'une directory est fonction du nombre de fichiers dans la directory et de la longueur du nom des fichiers. Nous pouvons ainsi définir un bon hash frame size de la façon suivante :

$$HFS = \frac{\text{nombre de fichiers}}{\text{nombre de noms de fichiers dans un bloc disque}}$$

Généralement, le nombre de fichiers dans un bloc est égal à 20.

b.2. Au niveau de la mémoire : la taille de la mémoire cache est insuffisante et d'autre part, les défauts de page sont encore élevés.

* La mémoire cache composée de 128 buffers (valeur par défaut) est insuffisante; elle a été augmentée et est ainsi passée à 512 buffers. La cache est le nombre de buffers de 512 bytes alloués au système pour effectuer ses entrées/sorties. La taille de cette cache dépend :

    * de la charge du système,
    * du type de la charge,
    * du hardware (taille de la mémoire physique).

  La charge du système est relativement élevée et est principalement constituée de programmes d'application de type "gestion". Si le système possède de la mémoire en suffisance, nous avons tout intéret à augmenter le nombre de buffers de la cache de façon à réduire les demandes d'accès disque.

  La taille idéale de la cache est difficile à déterminer. La seule façon de procéder est la méthode "essai-erreur".

* Le nombre de défauts de page est toujours élevé. Nous pensons que le passage au BASIC 5.00 devrait réduire considérablement ces derniers et améliorer les performances des programmes d'application.

b.3. Au niveau du CPU : celui-ci n'est pas saturé en utilisation sous charge normale. Notons cependant que le pourcentage de temps CPU occupé par le système est trop grand. La gestion du grand nombre de défauts de page en est principalement la cause.

- Conclusion : les diverses modifications préconisées au niveau du disque devraient améliorer considérablement les performances au niveau des E/S. Une nette amélioration devrait se faire sentir dès que l'element size des grands fichiers sera ajusté.

  Les performances des programmes devraient également s'accroître fortement dès le passage au BASIC 5.00.

  L'augmentation de la taille de la cache a amélioré les performances. Notons que le temps de réponse s'est amélioré sensiblement mais pas de façon suffisante.

3. TROISIEME ANALYSE

- Approche : après application des recommandations résultant de la deuxième analyse, les performances du système ne se sont pas améliorées suffisament. Les modifications réalisées n'ont donc pas altéré le temps de réponse pour les terminaux interactifs. Aucune amélioration probante ne s'étant fait sentir après la seconde analyse, les actions suivantes ont alors été planifiées :

  Moniteuring du comportement du système sous différentes charges de travail :

  - pendant le traitement normal.
  - pendant l'opération de clôture.
  - pendant une charge extrême du traitement.

- Réalisation : utilisant ces analyses réalisées sous différentes charges, nous avons tiré des conclusions qui ne rejoignent pas toujours celles de la société C.

Remarquons que l'efficacité des programmes d'application ne fera pas l'objet d'une étude approfondie. Comme nous l'avons déjà souligné, une nouvelle révision du BASIC sera bientôt installée et devrait accroître les performances de ces programmes. Toutefois, certains programmes d'application doivent subir des modifications au niveau de la gestion par le système de leur working set.

Le nombre de défauts de page d'un programme est un indicateur important de l'efficacité de ce programme.

Les divers problèmes et solutions pouvant y remédier sont :

a.1. Au niveau du disque.

Nous constatons tout d'abord que l'element size des fichiers n'a toujours pas été modifié. La société B, même si elle affirme avoir effectué ces modifications, n'a pourtant rien réalisé à ce niveau.

Lorsque la valeur de l' "avg seek" est élevée, cela peut soit signifier un remplissage fort élevé du disque (ce qui n'est pas le cas) soit que la fragmentation des fichiers ralentit les accès. La valeur du seek doit normalement être comprise entre 70 et 100. Notre analyse fournit des chiffres nettement plus élevés.

La modification concernant l'element size des fichiers de grande taille n'ayant pas été réalisée, les accès disque s'effectuent toujours de manière peu performante.

Afin d'optimiser encore le temps d'exécution d'une opération disque, les fichiers données et index doivent être regroupés physiquement sur le disque. La longueur des déplacements des têtes de lecture sera ainsi considérablement diminuée.

Le hash frame size de certaines directories contenant un grand nombre de fichiers a été ajusté. Un hash frame size adéquat permet d'améliorer les accès fichier.

La société C estime que le nombre de requêtes envoyées au disque saturent celui-ci. Cependant, le temps de service moyen est de 0.03 sec. A partir de 28, 30 requêtes, des problèmes peuvent se présenter. Le nombre de requêtes ne dépassant pas 20, nous ne pouvons pas considérer que le disque est fortement saturé.

a.2. Au niveau des paramètres de génération de l'OS.

Les propositions de modifications qui vont suivre devraient permettre d'améliorer fortement le temps de réponse aux terminaux interactifs. Cependant, avant d'effectuer certaines de ces modifications, il est préférable d'attendre le passage au BASIC 5.00.

A cet effet, nous allons utiliser VSGEN. VSGEN est un utilitaire qui permet de générer un OS sur mesure en fonction d'un hardware spécifique.
Nous pouvons spécifier le modèle de CPU, ajouter des devices, changer les paramètres de l'AOS/VS,...
VSGEN permet donc de créer un système sur mesure pour :

- le modèle de CPU,
- les disques,
- les lecteurs de bandes,
- les diverses consoles (système et utilisateurs),

- l'unité d'alimentation de courant de secours,
- les paramètres de l'OS.

Aucune modification hardware n'ayant eu lieu, seuls certains des paramètres de l'OS nous interressent :

1. La cache (défaut=128).

   Pour évaluer l'efficacité de la cache, nous devons contrôler la valeur "chitpc". Il s'agit du "cache hit rate" qui doit être supérieur à 90 % . Il atteint presque 100 % et la valeur du "cache misses/sec" est de 0.

   La taille de la mémoire cache au moment des mesures est de 512. Elle peut donc être réduite.

2. Minimum number of system pageable pages (défaut=10). Il s'agit du nombre de pages de 2048 bytes que le système essaie de garder en mémoire pendant les traitements. AOS/VS utilise des pages pour ses propres traitements. Le minimum number of system pageable pages établit le nombre de pages qu'AOS/VS essaiera de garder résident en mémoire centrale (son propre working set). Ses valeurs sont comprises entre 4 et 64 pages. Comme la contention mémoire est

faible, nous avons tout intérêt à spécifier un nombre de pages relativement élevé (40). Certains processus pourront ainsi s'exécuter plus rapidement.

3. ACCESS (défaut=Y). Une réponse affirmative implique un contrôle d'accès des fichiers par AOS/VS. Nous imaginons sans peine qu'un tel contrôle est assez fastidieux. Les utilisateurs ne quittant pas leurs programmes d'application (en basic), ce mécanisme de contrôle d'accès n'est pas nécessaire. Dans un environnement multi-utilisateurs, si on élimine le contrôle d'accès (ACCESS=N), chaque utilisateur peut accéder, modifier, effacer n'importe quel fichier existant. Pour garantir un minimum de sécurité, il suffit d'interdir aux utilisateurs d'employer les commandes du CLI.

4. Max program load pages - non contention (défaut=0). Quand un processus est créé, son working set a seulement quelques pages. Nous pouvons établir le nombre maximum de pages qui peuvent être chargées comme working set lorsqu'un processus est créé. Cependant, en spécifiant par exemple le nombre 50 pour cette variable, les pages multiples ne seront pas chargées avant que SPRED n'ait été lancé (programme permettant à un programme d'utiliser le mécanisme de pages multiples).

   L'utilisation de cette variable se révèle intéressante quand des programmes utilisent beaucoup

de pages non partagées. Il est plus rapide de charger la majorité (ou toutes) des pages nécessaires quand le processus est créé que de faire des défauts de page par la suite.

Le choix de ce maximum dépend de la contention mémoire. Nous avons tout intérêt à spécifier un grand maximum lorsque la mémoire n'est pas utilisée totalement.

5. Max program load pages - contention (défaut=0). Nous pouvons augmenter la valeur de cette variable et ainsi permettre le chargement de pages multiples en mémoire lorsqu'un processus est créé et que l'on a de la contention mémoire.

6. Fault time prepaging maximum (défaut=0). Nous pouvons spécifier le nombre maximum de pages qu'un processus peut lire quand il a besoin de données ou de code qui ne sont pas en mémoire.

Normalement, lorsqu'un processus a besoin d'informations qui ne se trouvent pas en mémoire, il y a un défaut de page. Le système trouve alors une page, l'ajoute au working set du processus et lit les informations nécessaires et les inscrit sur la page.

Le fault time prepaging permet au système d'ajouter des pages multiples au working set d'un processus qui fait un défaut de page.

Permettre à un processus de fonctionner de la sorte peut s'avérer particulièrement utile quand on sait que ce processus a besoin habituellement de plus d'une page lorsqu'il fait un défaut de page.

Comme pour le maximum program load, nous devons utiliser l'utilitaire SPRED. L'évaluation de la taille du WSSMIN et du WSSMAX est effectuée à l'aide de l'utilitaire de Working Set Trace (ce programme permet d'estimer la taille du Working Set souhaitée par un programme).

Aucune contention mémoire moyenne ou forte n'étant constatée, toutes les variables permettant de gérer plus efficacement ces deux types de contention mémoire ne feront pas l'objet de modifications.

a.3. Au niveau du disque et de la mémoire.

Les directories SWAP et PAGE ont une grande influence sur les performances du système. Elles font partie intégrante de la gestion mémoire d'AOS/VS.

Dans la mesure du possible, ces directories doivent être sur un disque indépendant du disque contenant les fichiers AOS/VS et ceux des utilisateurs afin que les E/S de swapping et de pagination puissent s'effectuer en concurrence avec le processeur central.

Idéalement, le disque des directories PAGE et SWAP devraient posséder leur propre contrôleur. Cependant, les avoir sur un disque séparé (même avec un contrôleur multidisques) est très souhaitable.

Remarquons que l'augmentation de la taille mémoire a pour effet de réduire les activités de pagination et de swapping et est moins onéreuse que l'achat d'un nouveau disque.

Nous avons une option qui nous permet d'effectuer la troncature des fichiers swaps et pages mais nous ne devons l'utiliser que lorsque nous sommes fortement limités au niveau de l'espace disque.

Nous allons donc supprimer la troncature des fichiers de pages utilisés par le mécanisme de pagination (celle des fichiers swaps n'étant pas active).

a.4. Au niveau de certains processus particuliers.

La priorité des processus nécessitant un traitement plus rapide peut être augmentée. Ce changement de priorité permet ainsi d'obtenir des résultats dans un délai plus court.

- Conclusion : Si la nouvelle version du BASIC n'améliore pas les performances du système, la plupart des modifications préconisées ci-avant devront être appliquées et auront pour effet de réduire le temps d'attente aux terminaux interactifs.

Cependant, les modifications à effectuer sont celles concernant l'element size des fichiers importants, la mémoire cache et l'option de contrôle des accès (ACCESS).

CONCLUSION

Le passage au basic 5.00 a été réalisé et a permis d'améliorer très significativement le temps de réponse. Avec une charge normale, nous avons donc un temps de réponse plus que satisfaisant.

De plus, il est maintenant possible d'ajouter à la charge normale la charge de clôture sans que le temps de réponse ne soit fortement perturbé. Même en charge extrême, aucune plainte n'émane plus des utilisateurs.

Nous devons encore effectuer un grand nombre de modifications afin d'optimiser les performances du système. Un certain nombre d'ajustements à effectuer nécessite la monopolisation du système pendant un week-end. Une fois ces ajustements réalisés (cfr conclusion du point VI.3), il nous restera à évaluer leur impact sur les performances du système. Nous sommes toutefois certains que ces modifications auront une influence considérable sur les performances du système.

Même si aucune contention mémoire importante n'apparaît dans l'étude, le nombre de défauts de page générés par certains programmes est fort élevé. Pour avoir une gestion de la mémoire relativement bonne, le nombre de pages libres doit être approximativement égal à 1000. Le nombre relevé lors des diverses études précédant la nouvelle version du BASIC était compris entre 450 et 650 et était donc fort limite. Actuellement, nous avons en moyenne 1000 pages libres.

De nouvelles applications vont être installées dans la société A: des informations relatives aux représentants devront être obtenues et distribuées via modems et disquettes. Ces applications devront notamment permettre une gestion des prospects, l'édition des offres. Pour le service achat, elles serviront à la préparation des achats.

Il est fort probable que dans un avenir relativement proche, des problèmes de disponibilité d'espace mémoire se feront à nouveau sentir. Une extension mémoire devra donc être rajoutée au système si l'on veut garder un niveau de performance plus ou moins équivalent à celui constaté actuellement.

Une nouvelle amélioration est encore possible en optimisant les programmes d'application. Une telle optimisation n'ayant jamais été effectuée s'avère toujours très utile.

Un changement de système ne s'avère donc pas nécessaire malgré l'augmentation d'activité prévue. En effet, le composant qui sera probablement saturé à court terme est la mémoire centrale. Celle-ci pouvant être encore augmentée de 2 MB, un goulot d'étranglement insurmontable n'est pas à craindre. Il suffira donc d'accroître la mémoire pour permettre une gestion efficace de l'ensemble des applications.

Remarquons encore que toute modification doit être suivie d'une évaluation des performances afin d'évaluer l'impact de cette modification.

Que sont devenus les objectifs de l'étude d'évaluation ?

* Le temps de réponse moyen des terminaux a été réduit de façon à satisfaire les utilisateurs.

* Une extension de la mémoire va probablement s'avérer nécessaire.

* L'allocation des volumes et fichiers sur disque a été déterminée et nécessite des changements au niveau de l'element size des grands fichiers.

* Le CPU n'est pas saturé. Cependant, le taux d'activité du CPU est relativement élevé. Tout accroissement de l'activité doit donc tenir compte de ce taux. Une optimisation des programmes d'application devrait permettre de diminuer ce taux.

L'acquisition de nouveaux composants autre que la mémoire principale serait prématurée. Par contre, l'optimisation des programmes d'applications devrait être envisagée.

Il n'existe pas réellement de méthodologie pouvant être appliquée quel que soit le type d'évaluation des performances à effectuer. Cependant, nous croyons avoir présenté une méthode applicable pour la catégorie des améliorations.

BIBLIOGRAPHIE

K.M. CHANDY et M. REISER. (1977), Computer Performance, IFIP.

S. LAVENBERG. (1983), Computer Performance Modeling Handbook, Academic Press.

D. FERRARI, G.SERAZZI, A. ZEIGNER. (1978), Measurement and tuning of Computer Systems, Prentice Hall, Inc.

C.H. SAVER et K.M. CHANDY. (1981), Computer systems performance modeling, Prentice Hall, Inc.

Sources diverses de la société C.

Facultés Universitaires Notre-Dame de la Paix, Namur

Institut Informatique

Année académique 1988 - 1989

Measurement and Tuning of Computer Systems : Annexes

Bernard Mambour

Mémoire présenté en vue de l'obtention du grade de Licencié et Maître en Sciences Informatiques

ANNEXE A

INDICES DE PERFORMANCE

Les indices de performances sont généralement classifiés en :

- Indices internes : ils quantifient l'efficacité d'utilisation de chaque composant du système.

  Indices internes :

  - Utilisation du CPU.
  - Recouvrement des activités.
  - Niveau de multiprogrammation.
  - Taux de pagination.
  - Temps de réaction.

- Indices externes : ils sont orientés vers une évaluation externe de l'efficacité de traitement du système.

  Indices externes :

  - Turnaround time.
  - Temps de réponse.
  - Throughput.
  - Capacité.
  - Disponibilité.
  - Sécurité.

- Turnaround time :

On distingue :

* Le turnaround time externe d'un programme qui est l'intervalle de temps entre l'instant auquel un utilisateur soumet son programme et l'instant auquel il reçoit le résultat.

  On l'appelle "externe" car il comprend le temps requis pour les opérations manuelles d'entrées et de sorties ainsi que le temps requis par le système pour exécuter le programme.

* Le turnaround time interne d'un programme est représenté par le temps requis par le système pour exécuter le programme.

Le "terme turnaround time" sera utilisé pour désigner le turnaround time interne.

Celui-ci est défini par

$$T = P - R$$

où R est le moment où les instructions du programme commencent à être lues et P le moment où l'impression des résultats est terminée.

Le processing time T_p d'un programme est defini comme son turnaround time quand c'est le seul programme tournant sur le système. La présence de plusieurs utilisateurs implique généralement que $T > T_p$.

Le turnaround time T dépendant du processing time T_p ne nous permet pas d'obtenir d'information utile pour l'évaluation de l'efficacité avec laquelle le programme a été traité par le système.

Le turnaround time moyen défini comme

$$T_m = \frac{\sum_{i=1}^{n} T_i}{n}$$

où n = le nombre de programmes,

T_i = le turnaround time du ieme programme,

peut conduire à des conclusions imprécises quant à l'efficacité de traitement (surtout si n est petit).

Pour permettre une comparaison plus précise de l'efficacité de traitement de différents programmes, nous définissons le turnaround time pondéré T_w comme le ratio entre le turnaround time T et le processing time T_p :

$$T_w = \frac{T}{T_p}.$$

La pondération moyenne du turnaround time est définie comme

$$T_{wm} = \frac{\sum_{i=1}^{n} T_{wi}}{n}.$$

Le turnaround time pondéré est affecté par les politiques de gestion des ressources implémentées par le système ainsi que par les caractéristiques de la charge de travail.

- Response time :

Le temps de réponse d'un système interactif à une commande peut être défini comme l'intervalle de temps entre le moment où l'introduction d'une commande se termine et le moment où la réponse correspondante commence à apparaître au terminal (Schéma A.1).

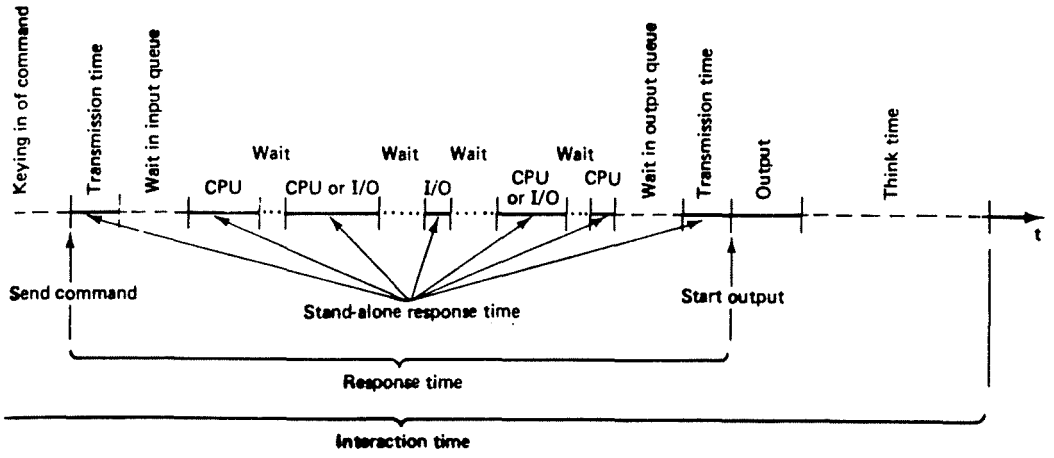


Schéma A.1 : Composition du temps de réponse.

On distingue généralement deux types de commandes :

- Les commandes légères souvent définies comme les commandes nécessitant moins de temps CPU que le quantum de temps qui lui est attribué.

- Les commandes lourdes qui nécessitent plus d'un quantum de temps pour être exécutées.

Le temps de réponse R_m moyen ne fournissant pas une information complète sur les performances d'un système interactif, nous pourrons également utiliser comme indice la distribution des temps de réponse.

Ainsi, certains descripteurs de la distribution tels que la variance, l'écart-type, la moyenne, les centiles, etc peuvent être utilisés si l'on désire obtenir une caractérisation plus complète et plus significative des performances d'un système interactif.

La variance σ des temps de réponse donne par exemple une certaine indication de la variabilité des temps de réponse, c'est-à-dire de leur degré de dispersion par rapport à la valeur moyenne. Elle peut être définie comme

$$\sigma = \frac{1}{n} \sum_{i=1}^{n} (R_i - R_m)^2$$

où R_i est le temps de réponse à la $i^{ème}$ commande.
Le temps de réponse à une commande dépend d'un certain nombre de facteurs tels que les caractéristiques des terminaux, la gestion

des terminaux, la façon dont l'application a été conçue, le nombre d'utilisateurs interactifs connectés et, finalement, de la charge de travail non interactive existant dans le système au moment du traitement des commandes.

- Throughput :

Les études d'évaluation ont généralement pour objectif principal l'amélioration de la productivité du système ou throughput.

Nous pouvons exprimer le throughput comme le nombre de programmes exécutés, le nombre d'opérations d'E/S effectuées, le nombre de transactions traitées, etc..., et cela par rapport à une unité de temps considérée.

Une définition générale du throughput X est donnée par

$$X = \frac{N_p}{t_{tot}}$$

où N_p est le nombre de programmes traités pendant l'intervalle de temps t_{tot}.

Lorsque des techniques de Benchmarks (cfr chapitre 2) sont utilisées, N_p est le nombre de programmes de la charge de travail et t_{tot} est le temps requis pour leur exécution.

Le throughput est également influencé par un certain nombre de facteurs :

- les caractéristiques de la charge de travail pendant la session de mesure,
- la configuration du système,
- le degré de multiprogrammation permis par le hardware,
- les algorithmes d'allocation des ressources,
- la vitesse des composants du hardware et du software.

Le throughput donne une indication globale de la puissance d'un système et ne fournit pas d'information utile pour l'évaluation des performances d'un seul programme.
Le throuthput d'un seul composant, par exemple, d'un sous-système d'E/S (buffer, canal, unité de contrôle, unité périphérique), si le programme utilise toujours un seul fichier, est donné par

$$X_{io} = \frac{b}{T_R + b/V_{io}} \text{ bits/s}$$

où b représente le nombre moyen de bits dans un record, T_R est le temps d'accès moyen à un record, et V_{io} est le nombre de bits théoriquement transmis en une seconde par le système.

Cette dernière équation indique que le throughput d'un sous-système des E/S est fonction de la longueur des records, de la vitesse hardware, de l'organisation des fichiers et de la vitesse de transmission du canal.

Une autre définition du throughput applicable à tout composant i est

$$X(i) = \frac{U(i)}{t_m(i) * N_{r-p}(i)}$$

avec U(i), l'utilisation du composant i,

$$U(i) = \frac{\text{temps pendant lequel i est occupé}}{\text{intervalle de mesure Ttot}}$$

$t_m(i)$, le temps de service de chaque requête,

$$t_m(i) = \frac{\text{temps pendant lequel i est occupé}}{\text{nombre de requêtes servies par i en Ttot}}$$

N_{r-p}, le nombre moyen de requetes faites pour le composant i par chaque programme,

$$N_{r-p} = \frac{\text{nombre de requêtes servies par i en Ttot}}{\text{nombre Np de programmes exécutés}}.$$

- Capacité :

La capacité est la valeur maximum théorique que le throughput d'un système peut atteindre.

Lorsque la charge de travail augmente graduellement, un phénomène de saturation peut survenir provoquant une réduction du throughput.

- Disponibilité :

C'est le pourcentage du temps total pendant lequel le système est à la disposition des utilisateurs.

- MTBF :

Le Mean Time Between Failure est l'intervalle de temps entre deux pannes consécutives du système.

- Utilisation du CPU :

Il s'agit du temps d'exploitation du système pendant lequel le CPU est actif.
Cette utilisation du CPU est subdivisée selon la contribution des différents types d'activité du CPU.

- Recouvrement :

Le recouvrement est considéré comme le pourcentage de temps de traitement du système pendant lequel plusieurs ressources sont occupées simultanément.

- Niveau de multiprogrammation :

C'est le nombre de programmes en exécution simultanée et donc en concurrence pour les ressources hardware et software du système.

- Taux de pagination :

Le taux de pagination se définit comme la fréquence avec laquelle les programmes de la charge de travail génèrent des défauts de page. Ces défauts de page provoquent le transfert de la page du support de stockage à la mémoire centrale, et parfois le transfert dans la direction opposée d'une page remplacée.

- Temps de réaction :

Temps dont le système a besoin pour réagir à une commande externe. C'est l'intervalle de temps entre le moment où le système reçoit une commande (du terminal ou en mode batch) et le moment où le CPU commence à exécuter cette commande.

ANNEXE B

MECANISMES DU SYSTEME S

Nous pouvons définir le système S au niveau de :

1. la configuration de la société A,
2. le hardware,
3. le système d'exploitation,
4. la charge de travail.

1. la configuration de A.

La configuration du système de la société A est la suivante :

* Hardware :

1. Système central :
   - 1 ordinateur central présentant les caractéristiques suivantes :
     - taille du mot : 32 bits.
     - temps de cycle : 200 nsecondes.
     - débit E/S-mémoire : 5 Moctets/secondes.
     - une mémoire centrale de 6 MB.
   - 1 cabinet d'alimentation.
   - 1 unité disque fixe 354 MB.
   - 1 unité bande magnétique 1600 BPI.
   - 2 processeurs I/O - IAC/16 de 16 lignes asynchrones.

2. Périphériques :
   - Ecrans-claviers :
     * 2 micro-ordinateurs.
     * 20 terminaux D/215.

- Imprimantes :

    * 4 imprimantes de 200 CPS.
    * 2 imprimantes de 350 CPS.
    * 1 imprimante de 400 CPS.
    * 1 imprimante laser.

* Système d'exploitation : 1 OS/VS.

2. Le hardware.

Le système S est constitué de double board CPU, d'une mémoire extensible jusqu'à 8 MB et d'un certain nombre de contrôleurs d'entrées/sorties. Ces E/S s'effectuent via le "Burst Multiplexor Channel", qui est un système E/S à très grande vitesse (accès de 5 Mégabytes/sec), et via un bus spécial d'E/S, qui est un système d'E/S à vitesse moyenne (taux de transfert de 2,5 Mégabytes/sec en entrée et de 1,25 Mégabytes/sec en sortie).

Les caractéristiques de l'architecture 32 bits du système S sont :

- Un espace d'adressage logique de 4 Gigabytes.
- Une mémoire segmentée avec protection en anneau.
- Un ensemble d'instructions avec manipulation de données de 32 bits.
- Un maximum de 64 utilisateurs.

Le systèmew S possède donc un système de mémoire virtuelle puisque la mémoire logique est nettement supérieure à la mémoire physique. Ces 4 Gigabytes d'adressage logique sont segmentés en 8 segments de 512 Mégabytes, chacun de ces segments étant entouré d'un anneau de protection.

3. Le système d'exploitation : OS/VS (Operating System/Virtual storage).

L'OS/VS est un système d'exploitation à mémoire virtuelle et à demande/allocation de pages conçu pour les ordinateurs 32 bits de la société C.

Ses caractéristiques sont :

- Utilisation d'un espace d'adressage logique très grand.
- Gestion de mémoire en utilisant des techniques de demande/allocation de pages.
- Multiutilisateurs et multiprogrammation.
- Système de protection sophistiqué.
- Un grand nombre d'utilitaires,...

Ce système d'exploitation fait partie intégrante de l'espace d'adressage logique des utilisateurs. Il délègue certaines fonctions à des processeurs auxiliaires. Ainsi, l'OS/VS permet à certains contrôleurs (processeurs de communication) de gérer des interruptions et des tampons de données.

Certaines des spécificités d'OS/VS doivent être décrites et analysées afin d'évaluer l'impact des modifications que l'on pourrait être amené à réaliser. Ainsi, nous verrons :

3.1. La structure de l'espace d'adressage logique.

3.2. La gestion de la mémoire.

3.3. La gestion des processus.

3.4. La gestion des entrées/sorties.

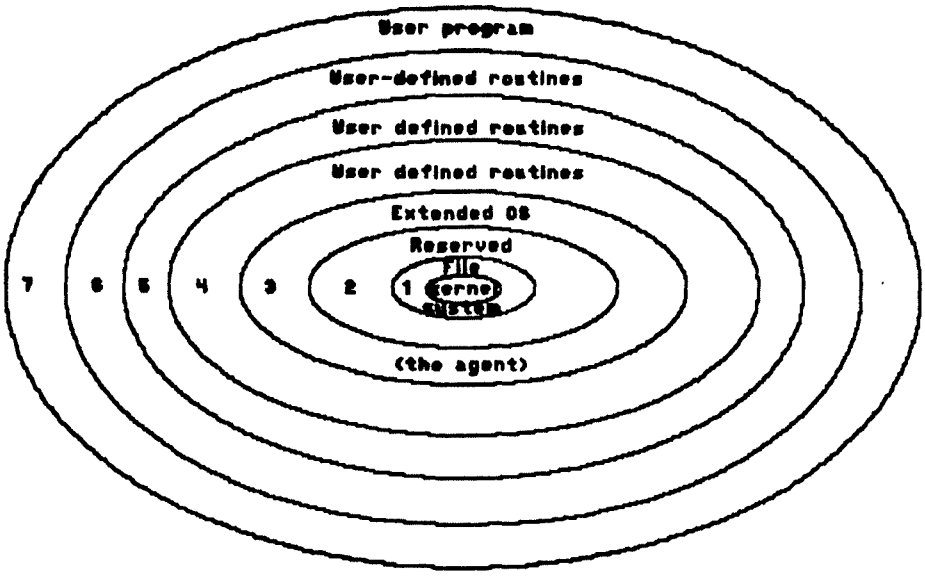
3.1. Structure de l'espace d'adressage logique :

Pour gérer les logiciels de base et les programmes des utilisateurs, l'OS utilise des segments, des anneaux et des portes.

- Segments : Nous avons vu que l'espace d'adressage logique est partagé en 8 segments de 512 Mégabytes.

- Anneaux : Un mécanisme hiérarchique d'anneaux de protection est utilisé pour assurer la sécurité du système. Par conséquent, un programme contenu dans un segment peut accéder à ce segment et aux segments supérieurs mais ne peut accéder aux segments de niveaux inférieurs.

- Portes : A l'intérieur même d'un segment, un programme peut définir un certain nombre de portes à travers lesquelles un programme appartenant à un autre segment peut demander l'accès.



STRUCTURE DES SEGMENTS ET ANNEAUX DE L'ECLIPSE MV/4000.

* Le segment 0 contient le noyau de l'OS/VS. Celui-ci s'occupe de la gestion de la mémoire (gestion des segments, demandes de page, et swapping de page), de la gestion des processus, et de la gestion des interruptions (rappelons toutefois que certaines interruptions sont contrôlées par des processeurs auxiliaires).

* Le segment 1 contient les données relatives aux utilisateurs. Le contenu de ce segment est mofifié lorsqu'un autre utilisateur exécute un programme.

* Le segment 2 est réservé.

* Le segment 3 contient l'interface utilisateur à l'OS/VS appelé "agent". Ce dernier valide tous les appels système avant de les adresser au noyau (niveau 0) où l'OS/VS les exécute.

* Le segment 7 contient les programmes des différents utilisateurs.

  Ceux-ci sont chargés par l'OS/VS. Ces programmes peuvent ensuite charger d'autres programmes dans les segments 4 à 6.

Après cette description de l'espace d'adressage logique, nous allons décrire les mécanismes de gestion des ressources utilisés par OS/VS. Pour effectuer cette gestion des ressources, l'OS/VS transforme les adresses logiques en adresses physiques, alloue la mémoire physique, maintient une structure de fichier, organise les tâches et les processus.

3.2. La gestion de la mémoire :

La gestion de la mémoire détermine en grande partie les performances d'un système. Nous allons donc décrire en profondeur le mécanisme de gestion implémenté par OS/VS. Cette description s'effectuera comme suit :

- Principe général.
- Accès aux segments et transformation d'adresses.

  * Segment Base Register (SBR).
  * Tables de pages.
  * Transformation d'adresse.
  * Accès aux pages.

- Principe général

Un système à mémoire virtuelle ne peut garder simultanément toute sa mémoire logique en mémoire physique. L'excédent de mémoire logique réside sur disque. L'architecture 32-bits du système S divise la mémoire en pages de 2 kilobytes. Disposant d'une mémoire physique de 6 Mégabytes, 3072 pages de mémoire logique peuvent résider à un moment donné en mémoire physique. L'OS/VS implémente les demandes de pages. OS/VS transfère les pages mémoires en mémoire physique lorsqu'elles sont requises. Pour chaque processus, OS/VS garde uniquement un working set de pages en mémoire physique.

Une unité spéciale, l'ATU (address translation unit), transforme une adresse logique en adresse physique. Lorsqu'un programme lit ou écrit un emplacement mémoire logique appartenant à une page qui ne se trouve pas en mémoire physique, un défaut de page survient. L'OS/VS doit alors charger en mémoire physique la page référencée.

Le système fournit à l'OS/VS un bit de page référencée et un bit de page modifiée pour chaque page du working set d'un programme. Il met à jour le bit de page référencée lors d'une lecture d'un emplacement mémoire appartenant à une page. Lors d'une écriture d'un emplacement mémoire appartenant à une page, les bits de page référencée et de page modifiée sont tous deux mis à jour.

Quand OS/VS doit lire une page, il contrôle d'abord si de la mémoire est disponible. S'il n'y en a pas, OS/VS recherche une page relativement peu utilisée en contrôlant son compteur de bits de page référencée. Quand OS/VS trouve cette page, le bit de page modifiée est testé. Si le programme n'a pas modifié la page, OS/VS peut réécrire sur cette page une nouvelle page. Sinon, il doit d'abord écrire sur disque la page modifiée.

- Accès aux segments et transformation d'adresses.

Pour accéder à un (plusieurs) mot(s) mémoire, le processeur accède à un segment, transforme une adresse logique en une adresse physique, et accède à la page physique qui contient le(s) mot(s).

* Segment Base Register (SBR). Avant que le processeur n'accède à un segment, il contrôle d'abord le SBR spécifié dans l'adresse logique. Le bit 0 du SBR est utilisé pour contrôler si l'accès au segment est autorisé.

  Si le processeur ne peut faire référence au segment, le processeur interrompt l'exécution de l'instruction et signale une violation de protection d'un segment.

Le processeur possède huit SBR (un pour chaque segment). Chaque SBR contient les informations suivantes :

- Validation de l'accès au segment.
- Validation d'un accès E/S.
- Spécification d'une table de page en simple ou double niveau.
- Spécification pour le segment de l'adresse de la première entrée dans la table des pages.

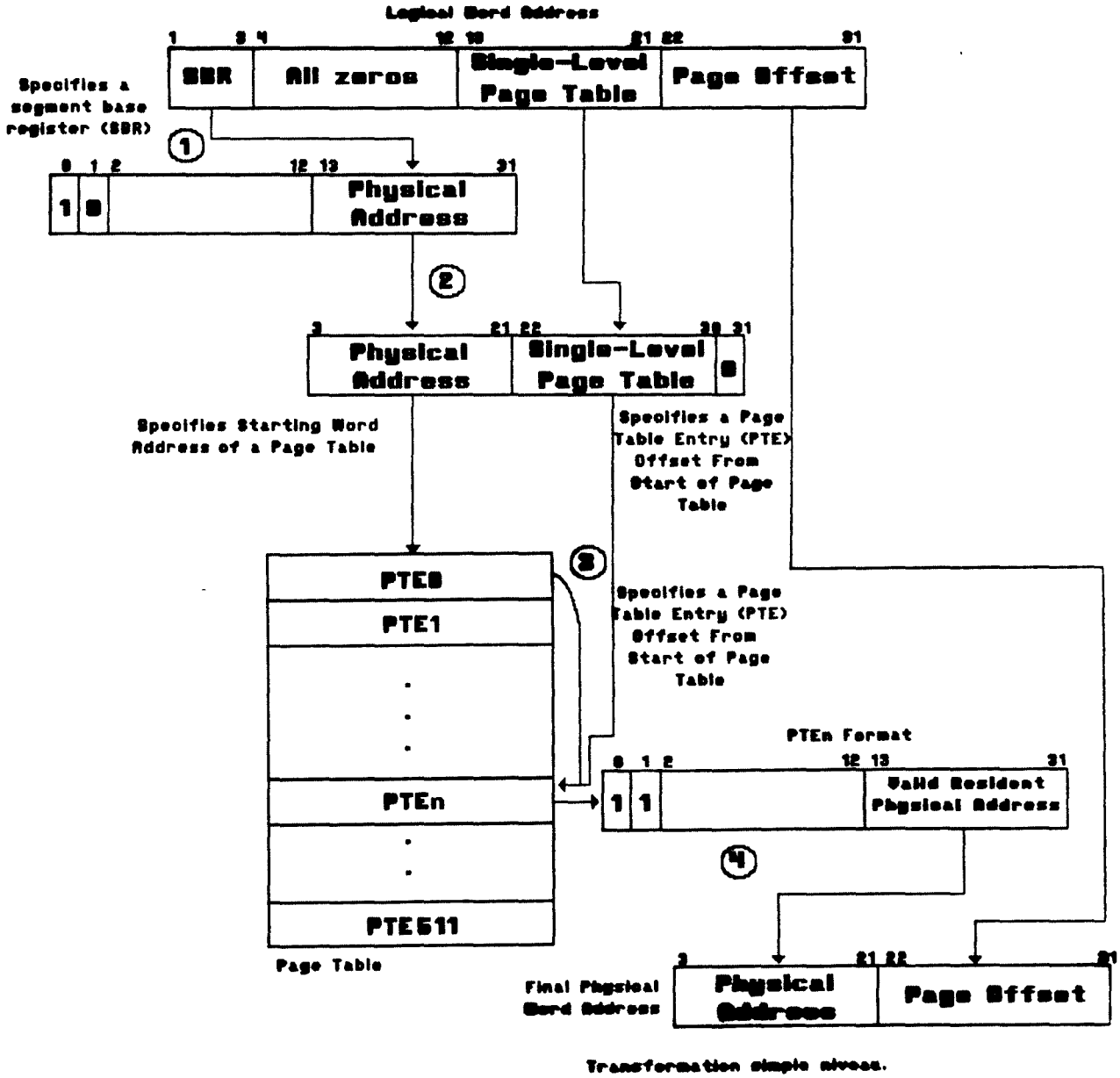
* Tables de pages. Pour chaque segment, le processeur accède à une table de pages qui spécifie l'état des pages du segment qui sont en mémoire. La table des pages contient une entrée (PTE) pour chaque page qui :

- indique si la page est une référence valide et le type d'accès,
- indique si une page est en mémoire physique,
- contient les informations nécessaires pour la transformation d'adresse.

* Transformation d'adresse. Des informations de transformation d'adresse sont conservées pour chaque segment. Les informations de transformation d'adresse d'un segment se retrouvent dans des tables de pages et résident en mémoire logique.

Les transformations d'adresse peuvent s'effectuer de deux façons :

A. En simple niveau. Dans ce cas, un espace d'adressage maximum de 1 Mégabyte est possible pour un segment.



Transformation simple niveau.

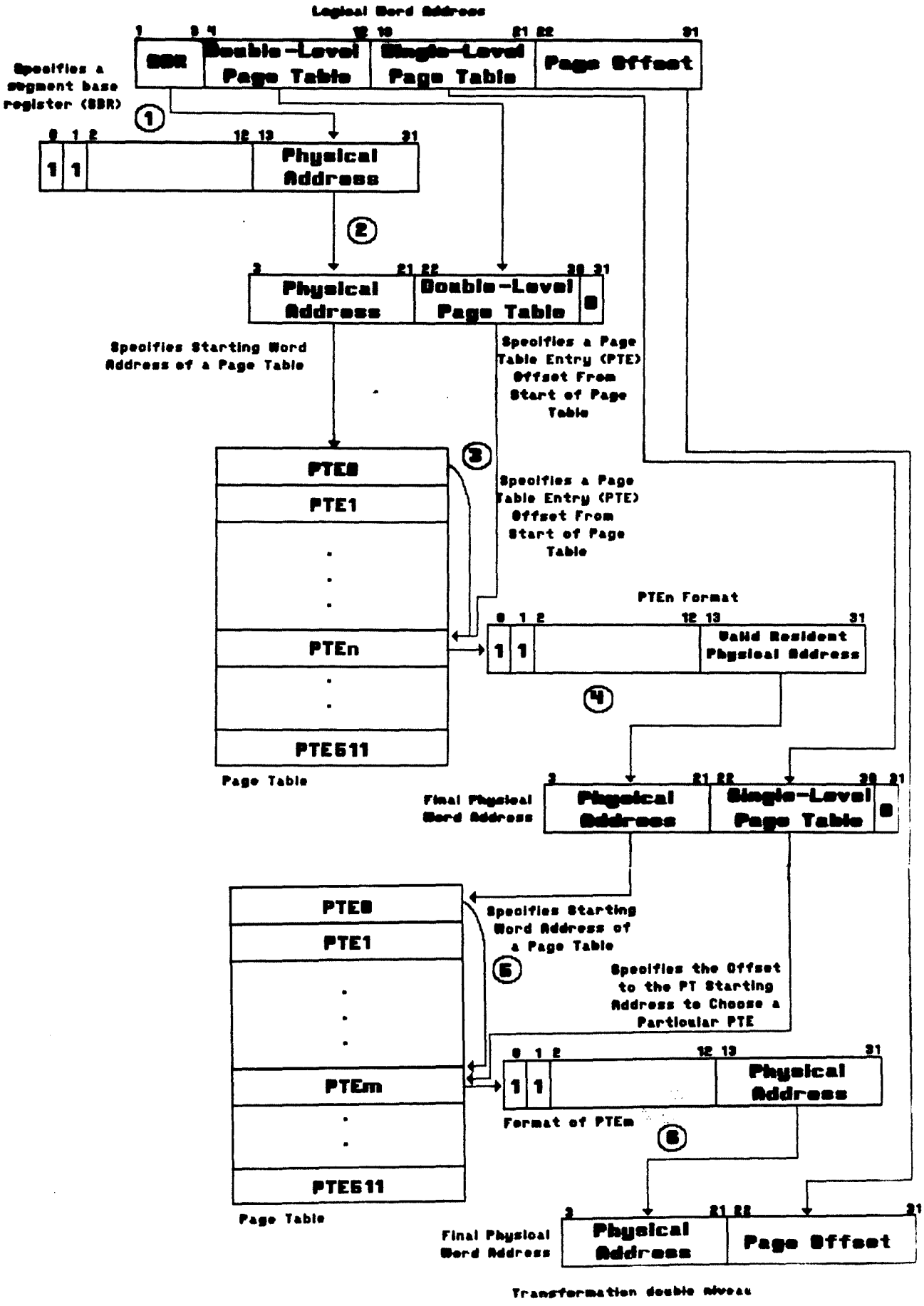
1. Les trois premiers bits de l'adresse logique spécifient un des huit SBRs. Le processeur utilise le contenu du SBR (supposé valide) pour former l'adresse physique d'un PTE.

2. Le SBR pointe au début d'une table de pages.

3. Un champ de 9 bits de l'adresse logique d'origine donne le déplacement dans la table des pages. Ce champs sélectionne une entrée parmi les 512 possibles de la table des pages.

4. Le MV/4000 constitue l'adresse physique. L'entrée de la table des pages fournit les 12 bits supérieurs qui spécifient alors une page physique. Les 10 derniers bits de l'adresse logique de départ donnent le déplacement dans la page.

B. En double niveau. Un espace d'adressage maximum de 512 Mégabytes peut être alloué à un segment.



Transformation double niveau

1. |
2. | Identique a la première transformation.
3. |

4. L'entrée dans la table des pages pointe au début d'une seconde page.

5. Un autre champ de 9 bits de l'adresse logique d'origine donne le déplacement dans la seconde table des pages.

6. L'adresse physique finale est alors constituée. L'entrée dans la seconde table des pages fournit les 12 bits supérieurs qui spécifient une page physique. Comme pour l'adressage en simple niveau, les 10 derniers bits de l'adresse logique d'origine donnent le déplacement dans la page.

* Accès aux pages :

1. Validation de l'accès : Quand une instruction fait référence à une page, le processeur vérifie la validité de l'accès en contrôlant la demande d'accès avec les bits de validation et d'accès dans l'entrée de la table de pages.

2. Demande de page : L'espace d'adressage logique étant nettemment supérieur à l'espace d'adressage physique, toutes les pages ne peuvent se trouver simultanément en mémoire physique. Quand une instruction fait

référence à une page valide qui ne se trouve pas encore en mémoire physique, il y a un défaut de page. L'état courant du processeur est sauvé en mémoire réservée (context block), une page en mémoire physique est sauvée sur disque (si nécessaire), et la page référencée est transférée du disque en mémoire.

3.3. La gestion des processus dans OS/VS

Nous allons décrire la gestion des processus effectuée par OS/VS. OS/VS gère ses ressources efficacement, allouant la mémoire et le CPU aux processus batchs et interactifs en fonction de leur type, priorités et groupes. L'OS/VS supporte jusqu'à 255 processus concurrents.

Les points suivants feront l'objet d'une étude plus approfondie:

- Processus et mémoire virtuelle.
- Processus et mémoire physique.
- Processus et CPU.
- Groupes de priorité.
- Processus et état.
- Processus et tâches.

- Processus et mémoire virtuelle.

Chaque programme est considéré comme un processus avec sa propre identification de processus (PID). Quand un processus est créé, l'OS constitue son espace d'adressage virtuel. Il place le programme dans le segment 7 et une copie de l'"agent" dans le segment 3 (une copie existe pour chaque processus actif). Il place également les données système de ce processus dans le segment 1. Chaque processus partage le noyau.

Un processus démarre avec un certain nombre de pages de mémoire virtuelle - son working set initial. Le working set est un sous-ensemble de l'espace d'adressage logique total du processus. Ce sont les pages du processus qui se trouvent en mémoire physique quand le processus est résident. Quand le processus essaie d'acquérir un plus grand nombre de pages (par ex, pour exécuter une routine qui ne se trouve pas en mémoire), un défaut de page survient; OS/VS augmente alors le working set du processus en lui ajoutant une page.

La taille du working set d'un processus est ajustée dynamiquement. L'OS/VS détermine la taille du working set en se basant sur le nombre de pages qu'un processus utilise et l'historique des défauts de page. Quand un processus génère un nombre excessif de défauts de pages, l'OS/VS augmente la taille du working set de ce processus, ce qui permet de diminuer les défauts de pages et d'améliorer les performances.

La taille minimale et maximale des working set est spécifiée pour chaque utilisateur et l'OS/VS ne peut altérer la taille du working set que dans la fourchette de valeur ainsi définie.
Des problèmes de contention mémoire surviennent lorsque tous les processus actifs désirent obtenir un working set supérieur à la taille de la mémoire physique. Ces problèmes de contention mémoire sont de deux types :

* En contention mémoire légère, OS/VS résoud la situation en enlevant des pages inactives de certains processus et en stockant leurs images dans la directory PAGE. Les processus restent en mémoire physique mais disposent d'un working set contenant moins de pages. Ces pages sont restaurées à la demande aux working sets. Nous parlons dès lors de mécanisme de pagination.

* En forte contention mémoire, OS/VS enlève des processus entiers (en sélectionnant d'abord les processus bloqués) et stocke leur image dans la directory SWAP. Les working sets complets sont retirés de la mémoire et sont restaurés par la suite par OS/VS. Ce phénomène est appelé "swapping".

En cas de contention mémoire, les processus sont pagés ou swappés sur disque en fonction de leur type.

- Processus et mémoire physique.

Les processus obtiennent la mémoire physique en fonction de leur type; leur priorité joue un rôle secondaire.

Les types de processus sont :

* Processus résident. Un processus résident a son working set en mémoire et peut disposer de plus de mémoire quand il en a besoin (il peut avoir le processus complet en mémoire). Ce type de processus obtient la mémoire sur demande et la conserve. Le processus peut être soumis au mécanisme de pagination mais pas à celui du swapping.

* Processus Swappable : Ce type de processus dispose uniquement de son working set en mémoire (système de demande de pages). Les processus swappables reçoivent les ressources du système après les processus résidents et préemptibles.

* Processus préemptible : Ce type de processus reprend des caractéristiques des deux autres. Il est traité comme un processus résident mais est soumis à la technique de demande de pages et peut donc être soumis au mécanisme de swapping si un processus résident demande plus d'espace mémoire.

- Processus et CPU.

Seul le processus de plus haute priorité et se trouvant dans l'état prêt obtient le CPU (le working set du processus est en mémoire physique, n'est pas bloqué et est en attente d'un événement externe).

Après avoir obtenu le CPU, un processus garde celui-ci pendant une période de temps déterminée (subslice) ou jusqu'à ce que le processus soit bloqué.

- Groupes de priorité.

Trois groupes existent :

* Groupe 1 : processus de haute priorité (de 1 à 255). Ces processus comprennent les processus résidents ou préemptibles avec une priorité 1, 2 ou 3. Le groupe 1 reprend également tout processus avec une priorité allant de 4 à 255.
  Les processus du groupe 1 ayant la même priorité obtiennent le CPU sur base du principe round-robin.

* Groupe 2 : processus de priorité moyenne (de 256 à 258). Les processus de ce groupe sont gérés en fonction de leur comportement. Dans ce groupe, la priorité et le nombre d'événements ayant provoqués le blocage du processus déterminent quel sera le prochain processus qui obtiendra le CPU.

* Groupe 3 : processus de faible priorité (de 259 à 511). Dans ce groupe, les processus de même priorité obtiennent le CPU sur une base round-robin.

- Processus et état.

Les processus peuvent se trouver dans les états suivants :

* Processus éligible. Un processus éligible est prêt à être exécuté, veut s'exécuter, et est en mémoire. Il est exécuté dès que l'OS lui alloue une période de temps (time slice).

* Processus non-éligible. Un processus non-éligible est prêt à être exécuté, souhaite s'exécuter, mais ne se trouve pas en mémoire. Il est exécuté dès que le processus est swappé en mémoire (il devient alors éligible).

* Processus bloqué. Un processus est bloqué jusqu'à ce qu'un événement le débloque.

* Processus prêt. Le working set du processus est en mémoire physique et n'est pas bloqué et en attente d'un événement externe.

- Processus et tâches.

L'OS/VS accepte jusqu'à 32 tâches par processus. Ces tâches ont également un état et une priorité.

Les états des tâches sont les suivants : inactif (dead), actif mais en attente, actif et prêt.

La priorité d'une tâche est comprise entre 0 et 255 et est indépendante de l'état dans lequel se trouve cette tâche.

Si plusieurs tâches ont la même priorité et sont dans l'état actif et prêt, l'ordre de considération des tâches s'effectuera de façon round-robin.

2.3.4. La gestion des entrées/sorties :

Le système S permet d'effectuer les E/S via le Burst Multiplexor Channel (BMC) et via le Data Channel (DC).

- Le Burst Multiplexor Channel. Le BMC est utilisé pour transférer des données avec des disques ou bandes à grande vitesse. Il transfère les données par blocs (bursts). Ces blocs varient de 1 à 256 mots en fonction de l'appareil de stockage.

  Le système S permet d'utiliser jusqu'à 4 appareils BMC. Un transfert via le BMC est établit à l'aide d'instruction programmées d'E/S.

- Le Data Channel. Le DC travaille à vitesse moyenne et transfère un seul mot à la fois. Comme pour le BMC, l'établissement du transfert est effectué avec des instructions programmées d'E/S.

- Les instructions d'E/S. Elles permettent le transfert de caractères et commandes aux périphériques.

ANNEXE C

LE MONITEUR

1. Introduction.

Le moniteur est un programme qui décrit l'activité d'un système après que le système d'exploitation (AOS/VS) ait été lancé. Le but de l'utilisation de ce moniteur est de rassembler un grand nombre d'informations sur les divers composants du système et sur la charge de travail.

Une fois collectées, les informations doivent être interprétées et doivent nous amener à effectuer les changements judicieux nous permettant d'accroître les performances du système.

Notre attention se portera principalement sur les domaines suivants :

- l'utilisation mémoire avec les défauts de page et divers éléments relatifs à la contention mémoire.

- l'utilisation du CPU avec les pourcentages d'utilisation du CPU par les utilisateurs et les routines système ainsi que le temps pendant lequel le CPU est dans l'état "idle".

- les entrées/sorties qui reprennent aussi bien les E/S disques que les E/S consoles.

II. Principe de fonctionnement du moniteur.

Le moniteur obtient des données en consultant certains emplacements mémoire de l'AOS/VS et en effectuant certains appels système qui lisent des données relatives aux E/S.

Un ensemble d'écrans sont définis et permettent d'afficher certains symboles ainsi que les valeurs associées à ces symboles. Pour accéder aux valeurs qui nous intéressent, nous devons sélectionner l'écran les contenant.

Au démarrage, le moniteur transfère un ensemble de pages mémoire d'AOS/VS dans un tableau interne. Après chaque cycle, les symboles contenus dans le tableau sont contrôlés. Si la valeur contenue dans un emplacement a changé par rapport au cycle précédent, le moniteur met à jour cette valeur.

Le moniteur est considéré comme un processus résident et utilise environ 150 pages mémoires. L'overhead qu'il provoque peut influencer considérablement les performances du système. Un système ayant une configuration mémoire relativement peu élevée peut voir sa contention mémoire s'accroitre fortement. Nous devons donc tenir compte et évaluer l'overhead que le moniteur provoque et, si nécessaire, diminuer cet overhead en spécifiant nos propres écrans d'évaluation et en scindant l'évaluation par type (mémoire, CPU, E/S).

Le moniteur peut être lancé en batch et ne nécessite dès lors aucune présence pendant son exécution. Les données collectés sont alors enregistrées sur un fichier. Ces données peuvent être présentées sous diverses formes. Nous distinguons :

* les écrans,
* les graphiques.

Les graphiques permettent parfois de mieux illustrer les relations entre certains nombres obtenus.

III. Les écrans du moniteur.

Un exemple d'écran du moniteur est repris à l'annexe C.1.

Pour une description plus détaillée des paramètres repris dans ces écrans, nous renvoyons le lecteur au manuel d'utilisation du moniteur.

ANNEXE C.1

ANALYSE ON-LINE.

EXEMPLE.

Jul 31,1989 AOS/VS Performance Monitor Revision 4.00 15:42:11
Function Key £10 for Menu Manager's Overview Function Key £11 to Exit

Run times		Environment	
System up time	076 07:18:42	6 Mbytes of memory	
Monitor up time	000 00:00:01	31 Pids	

I/O	Last cycle	Monitor start	System start	
Disk I/O	32.00	32.00	10.34	Blocks per sec
Pmgr I/O	632.00	632.00	214.35	Chars per sec

CPU usage	Last cycle	Monitor start	System start
User_%	0.00	0.00 %	
System_%	0.00	0.00 %	Data not collected
Idle_%	25.00	25.00 %	
Monitr_%	75.00	75.00 % (monitor)	

Memory Management	Last cycle	Monitor start	System start	
Free pgs	545	545	Data not collected	
Faults	11.00	11.00	3.57	Total
Pfaults	10.00	10.00	0.93	Physical
Swaps	0.00	0.00	0.00	
PFFs	0.00	0.00	0.00	
Steals	0.00	0.00	0.00	

Jul 31,1989 AOS/VS Performance Monitor Revision 4.00 15:42:11

Function Key £10 for Menu — General Info — Function Key £11 to Exit

AOSCN.W	533	LCBRDS.W	164739	PRDRQS.W	13946205
BFMIN	512	LMODCN.W	16	PROCFN.W	2831
BUFCN	512	LRUCN.W	545	PROCRQ.W	2922
BUFLD	505	LRUMD.W	454349	PWTRQS.W	3244021
CAPT	1024	LRURQ.W	4401749	RSTKCT	4
CCMX	0	MBLKCN	2	SCCNT	0
DFESZ	4	NCLRF.W	0	SCTIM	0
ELINT	0	NCPUPG.W	3072	SFAULT.W	0
ELNON	0	NEPSTL.W	0	SHUCN.W	1091
FBLKC.W	0	NERPG.W	0	SRBLC	27
FCBCN	231	Nwbytes	6	SRCNT	2
FCBRDS.W	37897892	NOBUFS	0	SSTKCT	13
FLSHCN.W	0	NPFF.W	0	SWPS1.W	0
FLSHRQ.W	0	NQDBHRS	1	SWPS0.W	0
Free pgs	545	NSRC.W	349378	TERMRQ.W	2818
GCORCM.W	5030	OVCNT	78	TERMTA.W	17
GCORRQ.W	5030	PBLKSR.W	58356512	TLGFLT.W	17371306
IEBCN	0	PBLKSW.W	9841679	TOTFLT.W	23549239
IELRS	0	Pids	31	TRAPCT.W	0
IELSW	0	PFFDIF.W	0	USRCN.W	1991
IEQCN	0				

Memory Statistics

Npages	3073	Total number of 2kb pages in the system
Nmbytes	6	Total number of megabytes of memory
FBLKC.W	0	Number of pages on the free memory chain
LRUCN.W	545	Number of 2kb pages on the LRU chain
Free pgs	545	Total free memory. (FBLKC.W + LRUCN.W)
Avgfre	545	Average number of free pages
AOSCN.W	533	Number of 2kb pages used by AOS/VS
OVCNT	78	Number of system pageable pages in AOS/VS
OVMIN	10	Minumum number of system pageable pages
SHUCN.W	1091	Number of 2kb shared pages in use
USRCN.W	1991	Number of 2kb pages in use by users

Demand Paging Statistics

Faults per second	Last cycle	Monitor start	System start	
Totsec	11.00	11.00	3.57	Total
Lgsec	1.00	1.00	2.63	Logical
Physec	10.00	10.00	0.93	Physical
Syssec	0.00	0.00	0.00	System
Usrsec	11.00	11.00	3.57	User

LRUpgsec	10.00	Used page from the LRU chain	LRUpg.ms	10.00
LRUmpsec	0.00	Used modified page from the LRU chain	LRUmp.ms	0.00

Time constants in states 1, 2/3, and 4 (sys start)			Since monitor start	
State1.W	219808	State 1 (no contention)	State1	0.00
State2.3	0	State 2/3 (medium contention)	State2/3	0.00
OFPCNT.W	0	State 4 (heavy contention)	State4	0.00
ONPCNT.W	219808	NOT in state 4		
TICON	30	The time constant in seconds		

Disk File I/O

Chits	38062631	Number of Cache hits
Cmisses	184639	Number of Cache misses
Chitsec	10	Cache hits per sec
Cmissec	0	Cache misses per sec
Chitpc	99.51	Cache hit rate as a percentage
PBLKSR.W	58356512	Number of phys. disk blocks read
PBLKSW.W	9841679	Number of phys. disk blocks written
Bkrsec	32.00	Number of phys. disk blocks read per sec
Bkwsec	0.00	Number of phys. disk blocks written per sec
BUFCN	512	Number of buffers on the LRU chain
BFMIN	512	System startup number of buffers in the cache
NQDBHRS	1	Number of buffer headers enqueued to disks
X3MAT.W	696933	Reused Index Block instead of buffer request
X2MAT.W	8642950	Reused 2nd level IB instead of buffer request
X1MAT.W	270603	Reused 3rd level IB instead of buffer request
X3matsec	0.00	Reused Index Block per sec
X2matsec	0.00	Reused 2nd level IB per sec
X1matsec	0.00	Reused 3rd level IB per sec

Page Fault Frequency

CRFCON	48	Copy reference bits threshold			
STLFLG	0	Steal inhibit flag			
PFFEN	0	PFF enable flag	SWPSI.W	0	Swap ins
TICON	30	Time constant	SWPSO.W	0	Swap outs
STLCON	15	Steal constant	NEPSTL.W	0	Page steal call
SWCON	5	Swap constant	NERPG.W	0	Page removes
			NPSTL.W	0	Old page steals
NPFF.W	0	Processes PFFed			
NPFOFF.W	0	Times PFF stopped			

LRURQ.W	4401749	Satisfied a page request from the LRU
LRUMR.W	454349	Satisfied a mod. page request from LRU
State1	0.00	% time in no contention since monitor startup
State2/3	0.00	% time in medium contention since monitor startup
State4	0.00	% time in heavy contention since monitor startup

Process Management

PROCRQ.W	2922	Number of process creation requests
PROCFN.W	2831	Number of processes actually created
TERMRQ.W	2818	Number of termination requests
TERMTA.W	17	Number of times term went to Agent
TRAPCT.W	0	Number of user traps
GCORRQ.W	5030	Number of requests for memory for new/swapped process
GCORCM.W	5030	Number of GCORRQs completed
SWPSI.W	0	Number of swap ins
SWPSO.W	0	Number of swap outs

Peripheral Manager Activity

Chisec	5.13	Last cycle, characters read per second
Chosec	619.81	Last cycle, characters written per second
Avgci	0.66	Running average of chars read per second
Avgco	426.66	Running average of chars written per second
Avgrl	2.04	Average read length
Avgwl	296.61	Average write length
RDHI	4580115	Total number of characters read
WRHI	1409006893	Total number of characters written
NRDRQ	2243843	Number of read requests
NWTRQ	4750286	Number of write requests
DPCT	37	Current number of devices assigned

Scheduler Statistics

Elque	0	Length of the eligible queue (ELINT + ELNON)
IELSW	0	Length of the ineligible swapped queue
SRBLC	27	Number of process tables on the blocked queue
IEBCN	0	Number of process tables on the SWAP IN queue
IELRS	0	Length of the ineligible resident queue
CBFAIL.W	0	Number of times sched could not get control blocks
Cbsec	0.00	Number of CBFAILs per second
SSTKCT	13	Number of swappable control blocks
SCBFL.W	0	Number of times no swappable CB when needed
RSTKCT	4	Number of resident control blocks
RCBFL.W	0	Number of times no resident CB when needed

C P U A c t i v i t y

User_%	0.00	Percentage of user time
System_%	0.00	Percentage of system time
Idle_%	25.00	Percentage of idle time
Monitr_%	75.00	Percentage of monitor time
HAPH	0	Number of ticks in a user process
HSBH	0	Number of ticks in AOS/VS
HSIH	2	Number of ticks in the idle loop
HPRH	6	Number of ticks in the monitor
Nticks	8	Number of ticks in the last cycle
HTTH	8	Running total number of ticks

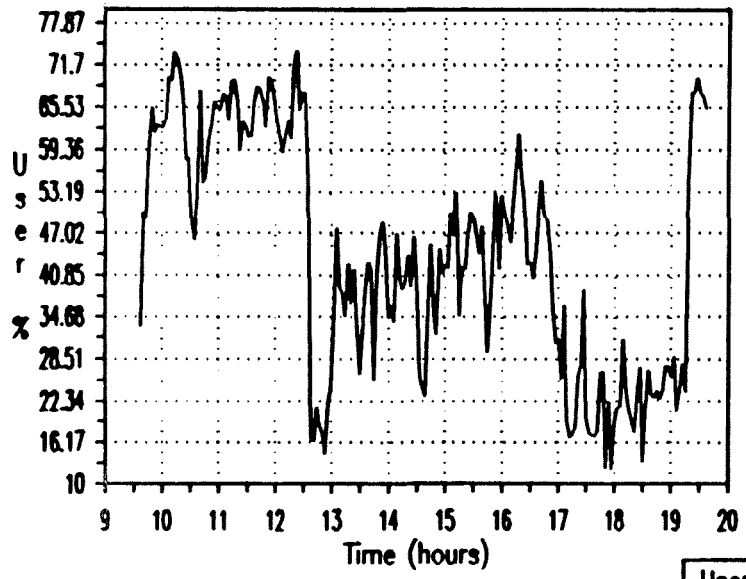
C P U A c t i v i t y

User_%	23.48	Percentage of user time
System_%	17.73	Percentage of system time
Idle_%	44.66	Percentage of idle time
Monitr_%	14.12	Percentage of monitor time
HAPH	143	Number of ticks in a user process
HSBH	108	Number of ticks in AOS/VS
HSIH	274	Number of ticks in the idle loop
HPRH	92	Number of ticks in the monitor
Nticks	609	Number of ticks in the last cycle
HTTH	617	Running total number of ticks

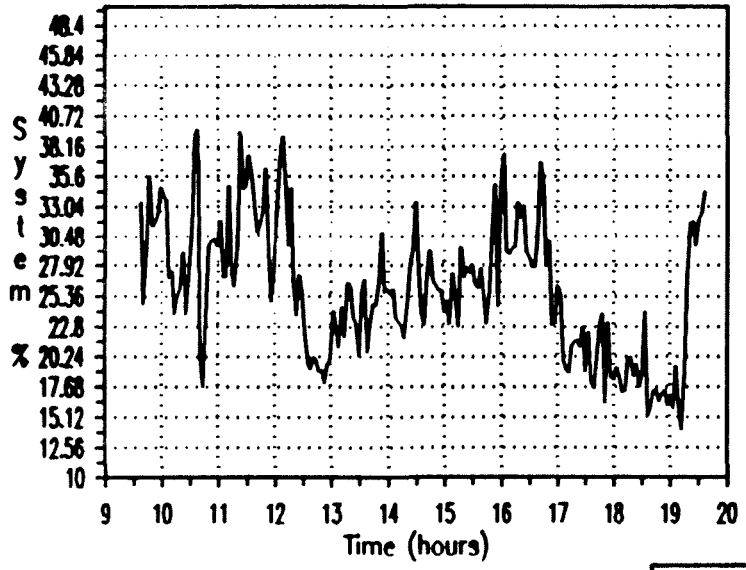
ANNEXE C.2

ANALYSE OFF-LINE.

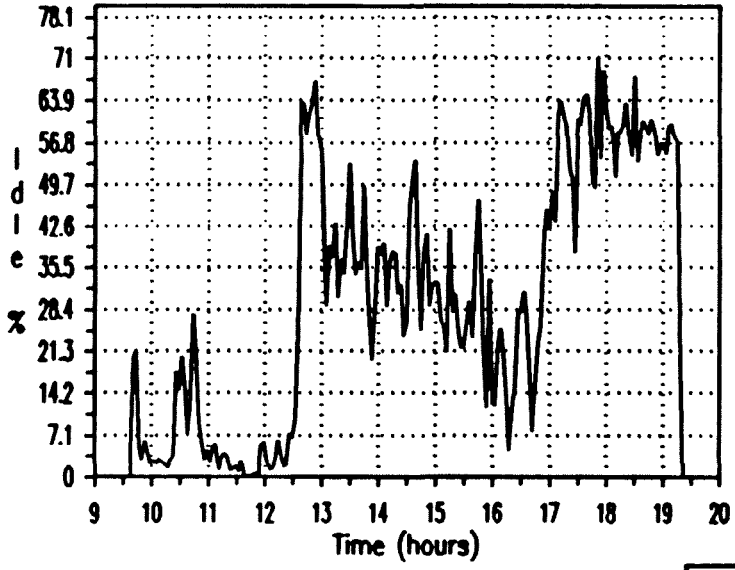
PRESENTATION GRAPHIQUE.



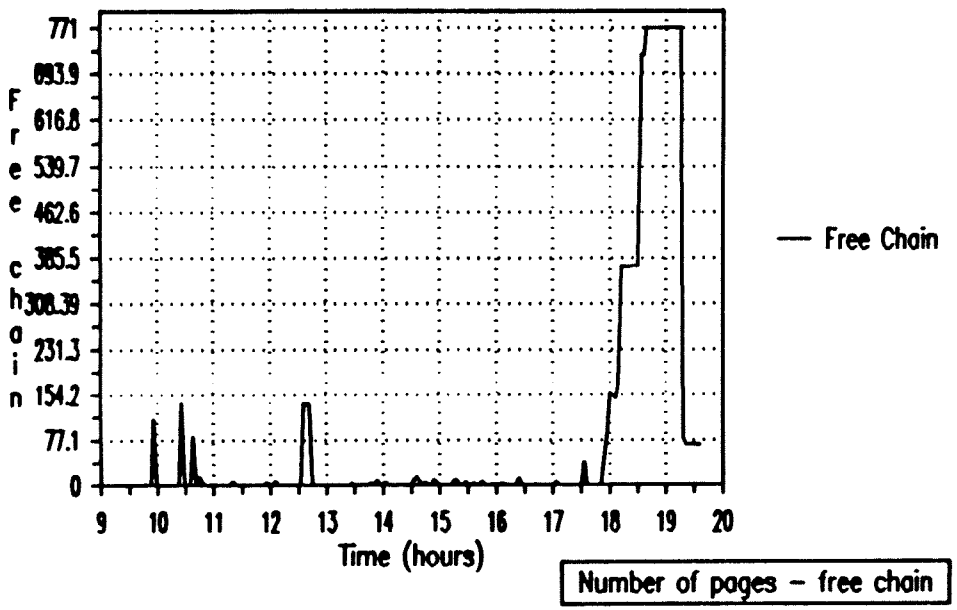
User cpu percentage

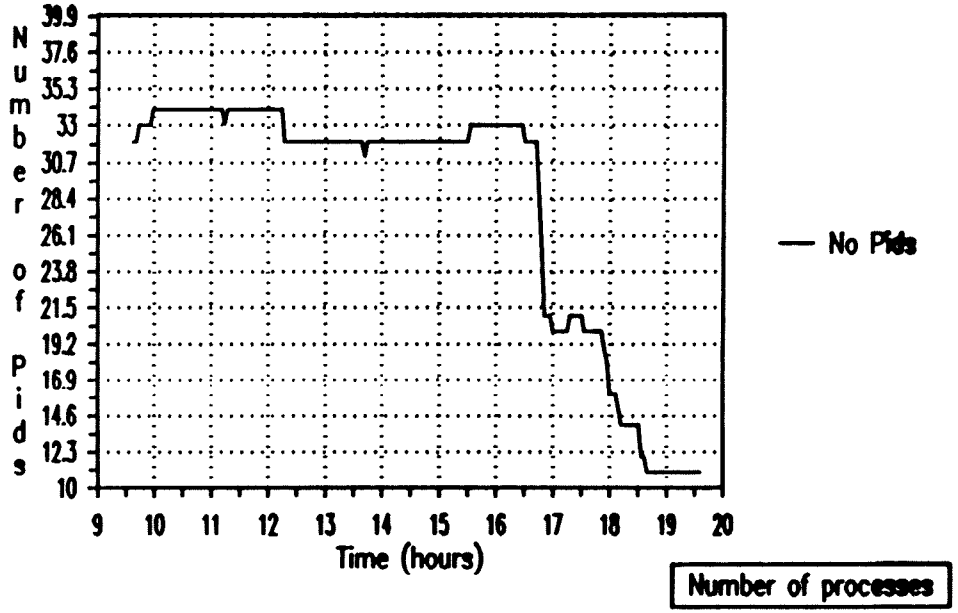


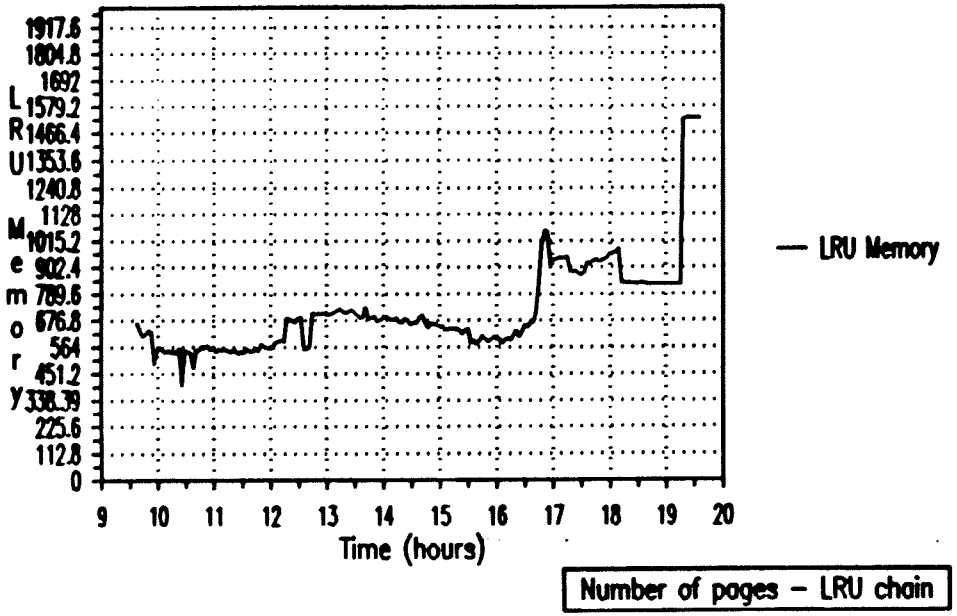
System cpu percentage

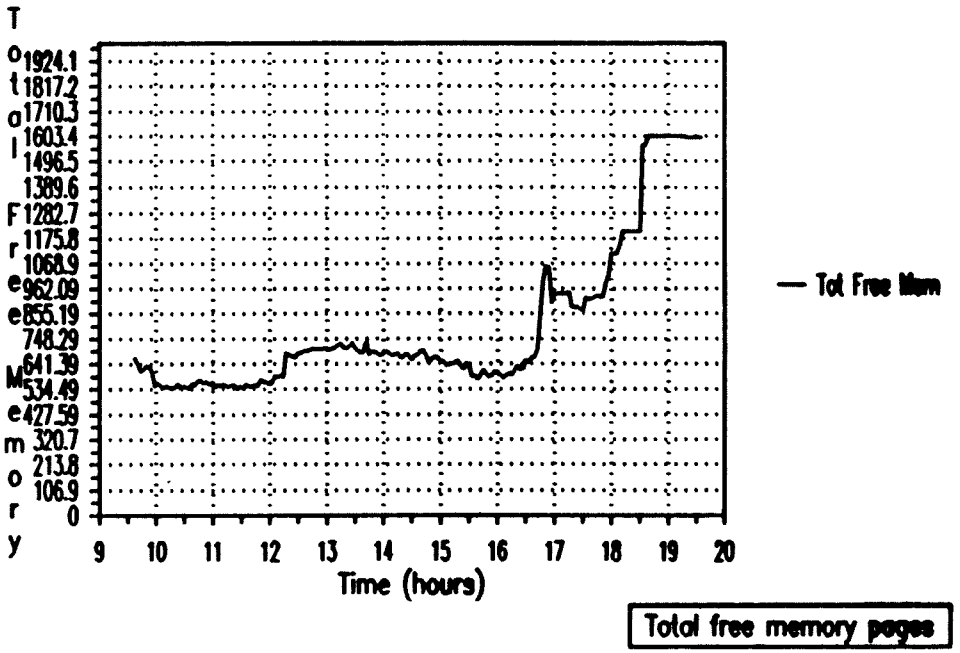


Idle cpu percentage


Free Chain
771
693.9
616.8
539.7
462.6
385.5
308.39
231.3
154.2
77.1
0
9 10 11 12 13 14 15 16 17 18 19 20
Time (hours)
Number of pages – free chain


No Pids
39.9
37.6
35.3
33
30.7
28.4
26.1
23.8
21.5
19.2
16.9
14.6
12.3
10
9 10 11 12 13 14 15 16 17 18 19 20
Time (hours)
Number of processes


LRU Memory
1917.6
1804.8
1692
1579.2
1466.4
1353.6
1240.8
1128
1015.2
902.4
789.6
676.8
564
451.2
338.39
225.6
112.8
0
9 10 11 12 13 14 15 16 17 18 19 20
Time (hours)
Number of pages – LRU chain


Tot Free Mem
1924.1
1817.2
1710.3
1603.4
1496.5
1389.6
1282.7
1175.8
1068.9
962.09
855.19
748.29
641.39
534.49
427.59
320.7
213.8
106.9
0
9 10 11 12 13 14 15 16 17 18 19 20
Time (hours)
Total free memory pages

ANNEXE D

PED : Process Environment Display

1. Description.

Ce programme fournit un certain nombre d'informations sur les processus actifs du système.

Les informations que l'on peut obtenir sont les suivantes :

- PID : l'identification du processus (process ID).
- PROCESS : le nom attribué par le système au programme.
- USERNAME.
- PROGRAM : le nom de fichier du programme.
- ELAPS : le temps pendant lequel le processus a existé.
- BS : processus bloqué/swappé.
- SH7 : le nombre de pages partagées dans l'anneau alloué par défaut pour les processus utilisateurs.
- US7 : le nombre de pages non partagées dans l'anneau 7.
- I/O : le nombre de blocs de 512 bytes transférrés via le DC (data channel).
- FTA : le nombre de défauts de page.
- FTAs : le nombre de défauts de page par seconde.
- PRIORITY : la priorité du processus.
- WSS : le nombre de pages dans le working set du processus.

2. Exemple.

```
 PID USERNAME PROCESS  PROGRAM   ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  16 COT      CON21    BASIC5.00  3/10   2:23     60   19  6727  7501   169
  34 OP       00034    PED        4.00   0.29     37   94     0   119   125
PED Rev 7.60.00.00 Monday 31-Jul-89  4:20:20 PM AOS/VS Rev 7.60.00.00

 PID USERNAME PROCESS  PROGRAM   ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  13 RTS      CON40    BASIC5.00  4/51  54.92     60   24  2538  2360   176
  16 COT      CON21    BASIC5.00  3/10   2:23 B   60   23  6767  7531   173
  23 OCC      CON38    BASIC5.00  6/59   3:38     60   25  8286  9617   179
  32 OP       00032    MONITOR    3:32   1:00     72  101   788   148   155
  34 OP       00034    PED        9.00   0.44     37   95     0   133   139
PED Rev 7.60.00.00 Monday 31-Jul-89  4:20:24 PM AOS/VS Rev 7.60.00.00

 PID USERNAME PROCESS  PROGRAM   ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   3 OP       EXEC     EXEC      76-08  13:34 B  170  132 23772 20663   349
   5 OP       CON24    XLPT      41-23  10:25 B    4    4 18400    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  13 RTS      CON40    BASIC5.00  4/51  55.27 B   60   24  2538  2377   177
  16 COT      CON21    BASIC5.00  3/10   2:24 B   60   19  6843  7546   169
  23 OCC      CON38    BASIC5.00  6/59   3:39 B   60   25  8297  9629   178
  32 OP       00032    MONITOR    3:37   1:02     72  101   812   148   155
  34 OP       00034    PED       14.00   0.70     37   95     0   133   139
PED Rev 7.60.00.00 Monday 31-Jul-89  4:20:30 PM AOS/VS Rev 7.60.00.00

 PID USERNAME PROCESS  PROGRAM   ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   5 OP       CON24    XLPT      41-23  10:25 B    4    4 18415    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  11 MB2      CON2     BASIC5.00  8/23  31:46 B   60   26 33248 81019   174
  13 RTS      CON40    BASIC5.00  4/51  55.52 B   60   24  2538  2390   176
  16 COT      CON21    BASIC5.00  3/10   2:25 B   60   23  6895  7619   173
  32 OP       00032    MONITOR    3:47   1:05 B   72  101   860   148   155
  34 OP       00034    PED       24.00   1.01     37   95     0   134   140
PED Rev 7.60.00.00 Monday 31-Jul-89  4:20:39 PM AOS/VS Rev 7.60.00.00

 PID USERNAME PROCESS  PROGRAM   ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   5 OP       CON24    XLPT      41-23  10:26 B    4    4 18418    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  13 RTS      CON40    BASIC5.00  4/51  55.56 B   60   24  2538  2390   176
  16 COT      CON21    BASIC5.00  3/10   2:25 B   60   19  6971  7634   169
  32 OP       00032    MONITOR    3:53   1:09     72  101   908   148   155
  34 OP       00034    PED       30.00   1.29     37   95     4   135   141
PED Rev 7.60.00.00 Monday 31-Jul-89  4:20:46 PM AOS/VS Rev 7.60.00.00
```

```
 PID USERNAME PROCESS  PROGRAM    ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   5 OP       CON24    XLPT      41-23  10:26 B    4    4 18423    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
   8 HDS      CON14    BASIC5.00  8/55   3:17 B   60   19  3380 15245   169
  11 MB2      CON2     BASIC5.00  8/23  31:46 B   60   23 33324 81049   170
  13 RTS      CON40    BASIC5.00  4/51  56.19 B   60   24  2538  2424   176
  16 COT      CON21    BASIC5.00  3/11   2:26     60   20  7047  7743   169
  32 OP       00032    MONITOR    4:01   1:11     72  101   936   148   155
  34 OP       00034    PED       38.00   1.54     37   95     4   135   141
PED Rev 7.60.00.00 Monday 31-Jul-89  4:20:54 PM AOS/VS Rev 7.60.00.00
```

```
 PID USERNAME PROCESS  PROGRAM    ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   5 OP       CON24    XLPT      41-23  10:26 B    4    4 18429    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
   8 HDS      CON14    BASIC5.00  8/55   3:17 B   60   19  3380 15266   169
  11 MB2      CON2     BASIC5.00  8/23  31:47 B   60   27 33388 81079   175
  13 RTS      CON40    BASIC5.00  4/52  56.19 B   60   24  2538  2424   176
  16 COT      CON21    BASIC5.00  3/11   2:28 B   60   23  7091  7846   173
  32 OP       00032    MONITOR    4:10   1:14     72  101   976   148   155
  34 OP       00034    PED       47.00   1.85     37   95     8   135   141
PED Rev 7.60.00.00 Monday 31-Jul-89  4:21:03 PM AOS/VS Rev 7.60.00.00
```

```
 PID USERNAME PROCESS  PROGRAM    ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   5 OP       CON24    XLPT      41-23  10:26 B    4    4 18433    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  10 OLM      CON9     BASIC5.00  8/27   5:11 B   60   18 14521 15985   170
  11 MB2      CON2     BASIC5.00  8/23  31:48     60   23 33436 81125   170
  13 RTS      CON40    BASIC5.00  4/52  56.60 B   60   24  2538  2441   177
  14 CVI      CON41    BASIC5.00  2/31   1:13 B   60   18  2844  5364   157
  16 COT      CON21    BASIC5.00  3/11   2:29 B   60   19  7167  7866   169
  18 MVM      CON4     BASIC5.00  8/15   3:44 B   60   18 16358  9557   170
  19 RCR      CON11    BASIC5.00  6/28  20.37 B   60   18   884   801   158
  22 EMB      CON37    BASIC5.00  8/02   8:44 B   60   18  3749 31109   169
  32 OP       00032    MONITOR    4:18   1:16     72  101  1004   148   155
  34 OP       00034    PED       55.00   2.15     37   95     8   135   141
PED Rev 7.60.00.00 Monday 31-Jul-89  4:21:11 PM AOS/VS Rev 7.60.00.00
```

```
 PID USERNAME PROCESS  PROGRAM    ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   5 OP       CON24    XLPT      41-23  10:26 B    4    4 18442    61    67
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  10 OLM      CON9     BASIC5.00  8/27   5:11 B   60   22 14553 15999   171
  11 MB2      CON2     BASIC5.00  8/24  31:48 B   60   18 33464 81131   166
  13 RTS      CON40    BASIC5.00  4/52  56.88 B   60   24  2538  2454   176
  16 COT      CON21    BASIC5.00  3/11   2:30 B   60   23  7219  7957   173
  32 OP       00032    MONITOR    4:30   1:21     72  101  1068   148   155
  34 OP       00034    PED        1:07   2.56     37   95    12   136   142
PED Rev 7.60.00.00 Monday 31-Jul-89  4:21:22 PM AOS/VS Rev 7.60.00.00
```

```
 PID USERNAME PROCESS  PROGRAM    ELAPS    CPU BS  SH7  US7   I/O   FTA   MBS
   1 PMGR     PMGR     PMGR      76-08   7/18      0    1   105   117   127
   6 OP       RLS2     RLS2      76-08   1/50 B    9    1    32    51    57
  13 RTS      CON40    BASIC5.00  4/52  57.09 B   60   24  2538  2471   177
  32 OP       00032    MONITOR    4:30   1:21     72  101  1072   148   155
  34 OP       00034    PED        1:08   2.84     37   95    12   136   142
PED Rev 7.60.00.00 Monday 31-Jul-89  4:21:23 PM AOS/VS Rev 7.60.00.00
```

ANNEXE E

DISCO

1. Description.

Le programme DISCO fournit des informations sur les I/O disque : les accès, la longueur de la file d'attente, la longueur moyenne des seeks, le nombre de blocs lus et écrits depuis l'initialisation du disque.

DISCO permet d'avoir une perspective de la fragmentation du disque. Lorsque un système a plusieurs disques, DISCO donne la répartition de la charge sur les différents disques.

Les divers paramètres utilisés dans DISCO sont :

- Dev : le "device code" du contrôleur.
- Unit : le numéro d'unité du disque.
- # of Accesses : le nombre total d'I/O demandées à l'unité depuis qu'elle a été initialisée.
- % of Total : le taux d'I/O disque effectuées par cette unité.
- % of Busy : le taux d'I/O demandées à cette unité alors qu'une ou plusieurs demandes étaient déjà en attente.
- % of Interf : le taux de requêtes d'I/O faites à cette unité qui ont dû attendre au niveau du contrôleur car le contrôleur du disque traitait une autre requête.
- Avg Queue : la longueur moyenne de la file d'attente pour l'unité considérée.

- Max Queue : la plus longue file d'attente que l'unité ait rencontré.
- Avg Seek : Longueur moyenne des seeks.
- Blocks Read : le nombre total de blocs lus depuis l'initialisation de l'unité.
- Blocks Written : le nombre total de blocs écrits sur cette unité depuis son initialisation.

2. Exemple

```
Dec 21,1988          ADS/VS  DISC  MONITOR  PROGRAM                  17:18:34
                                                      Function key 11 to Exit

Actual time     3.1 seconds                           Cycle time     3 seconds
    U
 D  n                                                                Avg   Avg
 e  i    # of    % of  % of % of  Avg/Max Avg   Blocks    Blocks    % of Serv  Resp
 v  t    Reqs    Total Busy Intf  Queue   Seek  Read      Written   Util Time  Time
-------------------------------------------------------------------------------
24  0  4450809 100.0  13.4  0.0 0.17 25  57.0 13732978  2816197

Dec 21,1988          ADS/VS  DISC  MONITOR  PROGRAM                  17:18:34
                                                      Function key 11 to Exit

Actual time     3.1 seconds                           Cycle time     3 seconds
    U                                Per Cycle
 D  n                  Reqs                                                Avg
 e  i    # of    % of  per   % of  % of  Avg/Max  Avg   Blocks/sec  % of Serv
 v  t    Reqs    Total sec   Busy  Intf  Queue    Seek  Read Write  Util Time
-------------------------------------------------------------------------------
24  0       1 100.0   0.3   0.0   0.0  0.00/25 258.0   0.0   1.2
```

Dec 21,1988 AOS/VS DISC MONITOR PROGRAM 17:18:37
Function key F11 to Exit

Actual time 3.1 seconds Cycle time 3 seconds

Dev	Unit	# of Reqs	% of Total	% of Busy	% of Intf	Avg/Max Queue	Avg Seek	Blocks Read	Blocks Written	% of Util	Avg Serv Time	Avg Resp Time
24	0	4450815	100.0	13.4	0.0	0.17 25	57.0	13732988	2816199			

Dec 21,1988 AOS/VS DISC MONITOR PROGRAM 17:18:37
Function key F11 to Exit

Actual time 3.1 seconds Cycle time 3 seconds

Dev	Unit	# of Reqs	% of Total	Reqs per sec (Per Cycle)	% of Busy	% of Intf	Avg/Max Queue	Avg Seek	Blocks/sec Read	Blocks/sec Write	% of Util	Avg Serv Time
24	0	6	100.0	1.9	0.0	0.0	0.00/25	106.3	3.2	0.6		

Dec 21,1988 AOS/VS DISC MONITOR PROGRAM 17:18:40
Function key F11 to Exit

Actual time 3.0 seconds Cycle time 3 seconds

Dev	Unit	# of Reqs	% of Total	% of Busy	% of Intf	Avg/Max Queue	Avg Seek	Blocks Read	Blocks Written	% of Util	Avg Serv Time	Avg Resp Time
24	0	4450818	100.0	13.4	0.0	0.17 25	57.0	13732988	2816205			