Conservatoire National des Arts et Métiers

Centre d'Enseignement de Paris

MEMOIRE
POUR L'EXAMEN PROBATOIRE

en

INFORMATIQUE

par

EPEE NGANDO Benjamin

Le Point sur l'analyse des performances des systèmes informatiques

---

Soutenu le Mardi 9 Mai 2006

JURY

Professeur LEMAIRE Bernard : Président

M. DELAHAYE David : Membre
M. LIGNELET Patrice : Membre

TABLE DES MATIERES

Introduction générale et problématique

La croissance rapide des systèmes informatiques entraîne de nos jours des problèmes critiques de performances, avec par exemple des phénomènes d'effondrement, une qualité de service non garantie, etc. Pour garantir les performances de ces systèmes, une analyse fine du comportement est nécessaire afin d'identifier les problèmes et de les résoudre. Il est donc fondamental de disposer de techniques, de méthodologies et d'outils permettant d'analyser et de comprendre le comportement de ces systèmes.

La problématique de ce mémoire concerne donc l'analyse des performances des systèmes informatiques. Une attention particulière sera portée sur les systèmes distribués en architecture client serveur (section 2.4 du chapitre 2). L'objectif est de faire le point sur l'analyse des performances en présentant l'état de l'art essentiellement en terme de techniques et de méthodologie d'analyse des performances des systèmes informatiques.

Nous commençons par définir l'analyse des performances ainsi que les concepts fondamentaux qui la sous-tendent. Ensuite, nous effectuons un tour d'horizon des différentes techniques actuelles d'analyse des performances des systèmes informatiques puis nous détaillons une méthodologie d'analyse des performances. Enfin nous déroulons cette méthodologie à l'aide d'un exemple concret.

1. Introduction à l'analyse des performances des systèmes informatiques

L'objet de ce chapitre est de donner une définition de l'analyse des performances des systèmes informatiques et d'énumérer les concepts de base qui la sous-tendent.

1.1 Définition de l'analyse des performances des systèmes informatiques

Les performances de systèmes informatiques inférieures aux standards admis peuvent créer d'énormes problèmes. Il est nécessaire d'utiliser un processus d'analyse des performances pour rechercher d'éventuels problèmes, et s'assurer que le système répond ou dépasse les objectifs prévus lors de sa conception. Ce processus s'appelle *l'analyse des performances*. Dans l'acceptation informatique du terme, l'analyse des performances consiste donc à examiner un système pour s'assurer que chacun de ses composants fonctionne efficacement et conformément à sa conception, en observant plus particulièrement l'utilisation du processeur, les services réseau et système, ainsi que l'enregistrement et les entrées-sorties.

1.2 Concepts de base de l'analyse des performances des systèmes informatiques

1.2.1 Le Système

C'est l'entité dont on analyse les performances. On s'intéresse particulièrement aux systèmes informatiques tels que les ordinateurs, les serveurs de données (Web, Multimédia, …). En gros, un système est considéré comme étant un ensemble de ressources partagées entre différentes tâches. La caractéristique commune pour de tels systèmes est la présence de temps d'attente pour l'accès à ces ressources partagées.

1.2.2 Les Métriques

Ce sont les critères utilisés pour quantifier et analyser les performances d'un système.

1.2.3 La Charge

Ou *workload* en anglais. Elle représente généralement le trafic en entrée, qui pourrait être par exemple l'ensemble de messages à servir par un dispositif du réseau ou le nombre de tâches à exécuter par un processeur, etc. Il est généralement décrit par des lois probabilistes (Poisson, Exponentiel, …).

2. Etat de l'art de l'analyse des performances des systèmes informatiques

La première décision à prendre avant d'analyser les performances d'un système est de décider *Quoi* analyser, c'est à dire de définir le système, sa configuration, et les programmes sujets de l'analyse de performance. Ces décisions englobent la sélection des métriques de performance, des paramètres de performance, des facteurs de performance, de la charge (workload) et le choix des méthodes à utiliser. Cet ensemble de choix de base représente les entrées du processus d'analyse de performance d'un système. Après cette étape cruciale, toute étude de performance est basée sur un ensemble d'assomptions homogènes, explicites ou implicites.

2.1 Approches candidates pour l'analyse des performances

Dans la littérature, il existe globalement trois approches d'analyse de performance de systèmes : les modèles analytiques, les simulations et les mesures. Le choix d'une approche ou d'une autre pour analyser les performances d'un système dépend de plusieurs critères. Nous pouvons classifier ces techniques en deux grandes catégories (voir Figure 1) : les techniques basées sur la mesure et les techniques basées sur la modélisation.

La première catégorie permet de quantifier les critères de performance en les mesurant directement sur un système réel. Dans cette catégorie, nous distinguons deux variantes d'approches de mesure : l'instrumentation du code source de l'application en insérant des codes de mesure (in-side), ou l'utilisation de moniteurs externes à l'application pour quantifier les mesures au cours de cycles d'exécution de l'application (out-side).

La deuxième catégorie d'approches basées sur la modélisation, permet de spécifier le système à analyser afin de prédire ses performances. Les modèles élaborés par ces approches seront utilisés pour effectuer des simulations ou une étude analytique du système. La différence entre ces deux approches est que les simulations mettent en œuvre des modèles conceptuels du système sous test (représentation conceptuelle du système) qui nécessitent leur développement dans un outil de simulation. En revanche, la deuxième méthode (l'étude analytique) met en oeuvre des modèles calculables (files d'attentes, réseaux de Pétri, ...) sous la forme de formules mathématiques.

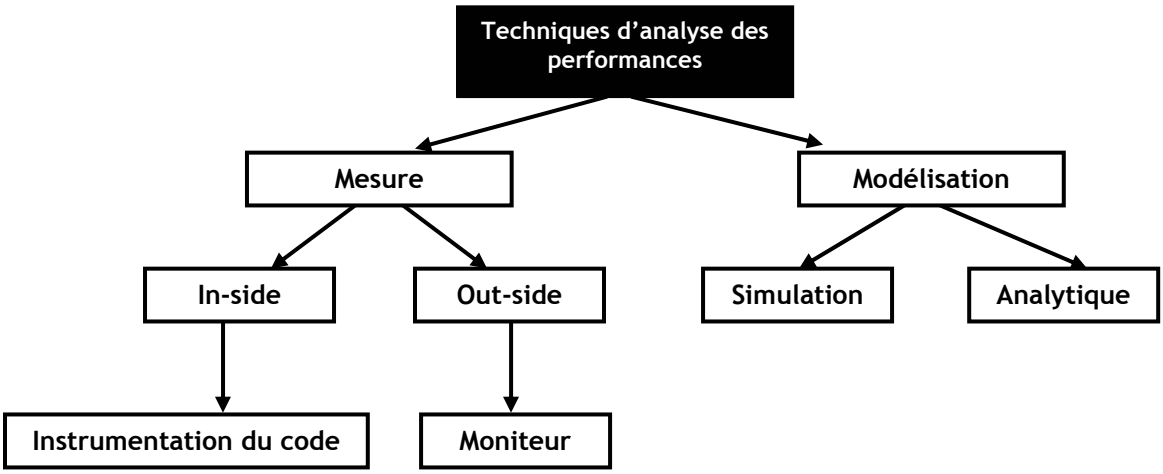


FIGURE 1 - Classification des techniques d'analyse des performances

2.1.1 Les techniques basées sur la mesure

Les techniques basées sur la mesure permettent d'analyser un système existant ou au moins un prototype de ce système, et elles se basent sur des outils d'instrumentation. L'instrumentation d'un système est une méthode indispensable pour mesurer ses performances au cours de son exécution, en insérant des codes de mesure dans son code source, ou en utilisant des outils d'observations collectant des mesures. L'un des principaux avantages de ces techniques est la réalité de ses résultats, puisque les données sont prises sur un système réel. Ce réalisme des mesures est l'un des arguments majeurs en faveur de ces techniques pour analyser les performances d'un système.

Cependant, ces résultats, malgré leur réalisme, sont relatifs et variables. Ils dépendent de plusieurs paramètres, comme la configuration du système, la charge et le temps de mesure, qui sont généralement uniques et relatifs à chaque test d'analyse de performance. Ces résultats, ainsi, ne sont pas généraux mais spécifiques à un test particulier. Un autre inconvénient majeur de cette technique est le coût nécessaire pour sa réalisation. En effet, ces techniques d'analyse introduisent un coût important dans un projet puisqu'elles nécessitent des équipements réels, l'instrumentation des applications et un temps important de développement et de collecte de mesures. Un troisième inconvénient de ces techniques est qu'elles ne sont pas toujours réalisables.

Bref aperçu sur les types de mesures

Il faut noter qu'il existe deux type de mesures : la mesure passive et la mesure active. La mesure passive ne perturbe pas le système sous test (idéalement). En revanche, la mesure active induit, généralement, des perturbations sur le système. Il existe deux approches de la mesure passive :

- Une approche de vraie mesure passive, où l'outil de mesure n'induit aucune perturbation sur le système sous test. A titre d'exemple un sniffer des paquets dans un réseau LAN sans fil (WLAN) ne perturbe pas le fonctionnement des autres équipements dans le réseau.

- Une approche de faible mesure passive, où le système sous test doit être modifié (par exemple instrumenté) mais sans changer son état de fonctionnement. A titre d'exemple, certains switchs Ethernet fournissent des ports spécifiques vers lesquels tous les paquets seront copiés sans perturber leurs transmissions. Un analyseur réseau peut s'attacher à ce type de port.

L'utilisation de la mesure active ne nécessite pas seulement la modification du système sous test mais impacte sa charge et son état interne de fonctionnement. Dans des cas extrêmes, l'utilisation de cette approche de mesure active peut induire le phénomène de Heisenbug[1] en faisant apparaître des bugs dans le système sous test.

En définitive les techniques basées sur la mesure utilisent deux méthodes : l'instrumentation de code du système sous test ou le monitorage du système au cours de son exécution via les moniteurs.

[1] Terme utilisé pour nommer des bugs extrêmement difficiles à reproduire et à identifier.

2.1.1.1 L'instrumentation du code

Cette méthode consiste à placer des capteurs dans le code source du système pour mesurer ses performances sous une charge réelle. La facilité offerte par cette technique parvient à combler certaines lacunes d'autres techniques qui n'arrivent généralement pas à exécuter une telle charge réelle et complexe. Des bibliothèques d'instrumentation (par exemple ARM : Application Response Measurement [1] ou WMI : Windows Management Instrumentation) existent et elles s'intègrent facilement dans le code source de l'application sous test en offrant un cadre efficace et transparent pour le représentation de données d'instrumentation. D'autres outils ne nécessitent pas une modification ou une recompilation du code source de l'application mais instrumentent l'application à l'exécution. Parmi ces outils[2], nous citons le profiler fourni par Netbeans [2] ou Gprof (Gnu profiling Tool, [3]).

2.1.1.2 Les moniteurs

Le monitorage du système au cours de son exécution utilise donc des moniteurs qui se positionnent comme des processus indépendants de l'application sous test. Les moniteurs sont donc des outils qui observent l'activité du système au cours de son cycle d'exécution, collectent les mesures et les affichent ou les placent dans un fichier.

a) Terminologies des moniteurs

- **Evènement :** Tout changement d'état du système. Par exemple arrivée d'un paquet
- **Trace :** Objet gérant l'historisation et la persistance des évènements.
- **Overhead :** Données de service. De manière générale l'overhead décrit la durée du temps de traitement (temps de communication + temps de synchronisation + …).
- **Domaine :** Ensemble des activités observables par le moniteur.
- **Fréquence d'entrée :** nombre maximum d'évènements que le moniteur peut correctement observer par unité de temps.

b) Classification des moniteurs

Moniteur logiciel : Moniteur utilisé pour mesurer les performances d'un logiciel système (Système d'exploitation) ou d'une logiciel d'application. Il est approprié lorsque la fréquence d'entrée est raisonnable.

Moniteur matériel : Moniteur utilisé pour mesurer les performances d'un matériel (Disque dur, Mémoire, Processeur, …)

Moniteur firmware : Moniteur implémenté par modification du microcode du processeur. Par exemple un microprogramme embarqué dans une interface réseau pour mesurer le trafic réseau.

[2] Ces outils sont généralement spécifiques aux applications Java et utilisent les facilités offertes par l'instrumentation de la machine virtuelle Java.

Au total, les techniques basées sur la mesure perturbent le système sous test et engendrent un coût (overhead) sur les performances réelles du système sous test. Ce coût est dû aux instructions supplémentaires à exécuter par l'application en utilisant la première méthode (instrumentation) ou aux ressources consommées par un moniteur externe à l'application dans le cas de la deuxième méthode (moniteurs).

2.1.2 Les techniques basées sur la modélisation

Le processus de modélisation

Le modèle est une représentation de la réalité dans un formalisme. Il est développé pour répondre à des questions déterminées et comporte certaines limitations, c'est une abstraction du système réel. Ce *système réel* n'existe pas forcément lors du processus de modélisation, il se peut que ce soit un système que nous cherchons à développer. Pour cela nous désirons effectuer une étude préliminaire de ses performances.

A partir du modèle, nous voulons obtenir des résultats sur le système étudié, c'est l'étape de résolution. Certains résultats peuvent nous amener à modifier notre vision du système, et nous inciter à calculer de nouveaux résultats sur le modèle. Pour cela, nous pouvons être amenés à complexifier le modèle pour rajouter des informations. De même, nous pouvons nous rendre compte que certains résultats ou éléments du modèle sont inutiles pour calculer les indices de performances recherchés, et nous pouvons ainsi parfois simplifier le modèle pour faciliter sa résolution.

Pour finir, nous comparons les résultats obtenus avec le comportement du système réel (comportement espéré si le système n'est pas encore créé), et nous pouvons alors concevoir le système qui donne les performances optimales.

Le processus de modélisation est résumé dans la Figure 2 suivante

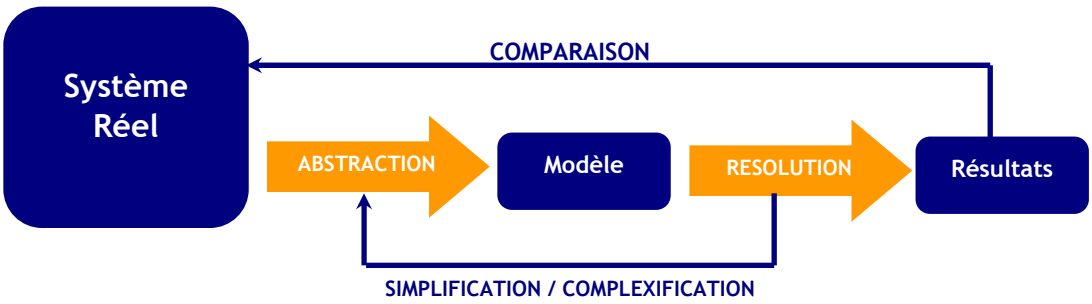


FIGURE 2 – Processus de Modélisation

2.1.2.1 L'analyse par simulation

C'est probablement la technique la plus utilisée pour analyser les performances de systèmes informatiques. Elle représente un moyen utile pour prédire les performances d'un système et les comparer sous plusieurs de ses configurations. Un atout majeur de cette technique est sa flexibilité puisqu'elle permet d'analyser le système sous plusieurs conditions et configurations. De même, si le système est déjà implanté la technique de simulation reste favorable puisqu'elle offre une flexibilité, difficile à réaliser avec la technique de mesure. En revanche, la technique de simulation nécessite une excellente maîtrise des techniques statistiques pour une analyse pertinente des résultats, ainsi qu'une maîtrise de la programmation pour développer le modèle à simuler sous un langage approprié. La confiance ou une validation préalable des modèles utilisés par cette technique est indispensable.

L'inconvénient majeur de cette technique, est qu'elle nécessite un temps potentiellement important et des ressources de calcul conséquentes. La simulation d'un système nécessite la réalisation des étapes suivantes (voir Figure 3) :

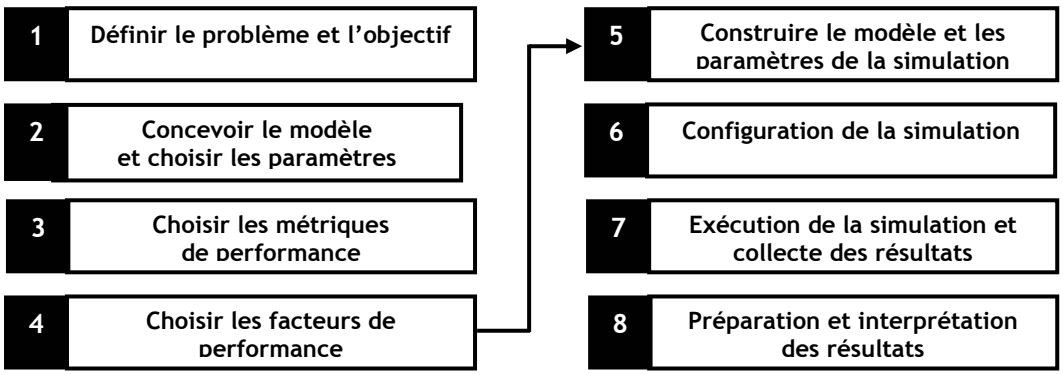


FIGURE 3 - Les étapes de réalisation d'une simulation.

1. Définir le problème et l'objectif de la simulation : dans cette étape il faut essentiellement spécifier le système à analyser et ses différents niveaux de détails.

2. Concevoir le modèle du système et définir l'ensemble de ses paramètres de performance. Notre objectif est d'analyser les performances, ainsi, nous n'avons pas besoin de modéliser le fonctionnement exact du système. Par exemple, la modélisation d'un agent de supervision ne nécessite pas la représentation de la récupération des informations de gestion du système d'exploitation, mais nous considérons juste les valeurs du temps nécessaires pour récupérer ces informations ou sa distribution mathématique (exponentielle, géométrique,...).

3. Choisir l'ensemble des métriques de performance du système.

4. Choisir l'ensemble des facteurs de performance du système.

5. Développer le modèle du système dans l'outil de simulation et fixer l'ensemble des paramètres de performance.

6. Configurer l'application développée dans l'outil de simulation afin de produire les résultats de performance.

7. Exécuter la simulation et collecter les données relatives aux performances du système

8. Représenter les données collectées et procéder à leur interprétation.

Plusieurs modèles de systèmes existent dans la littérature [4,11] et sont utilisés pour la simulation. Essentiellement, les modèles sont temporels avec des valeurs des états du système continues ou discrètes en fonction du temps. D'autres modèles à base d'évènements existent qui spécifient les états du système qui peuvent être continus ou discrets. Ces derniers modèles peuvent utiliser les premiers pour spécifier la nature de valeurs des états (discrètes ou continues). Une variété d'approches de simulation existe, aussi, dans la littérature [4,11]. Ces approches sont essentiellement : l'émulation[3], la simulation dirigée par trace et la simulation discrète.

L'outil de simulation est un choix important lors de l'utilisation de cette technique. Comme exemple dans le domaine de l'analyse de performances des systèmes de supervision nous pouvons citer :

- Network Simulator (NS) [5] qui est libre source
- OPNET [6] est un produit commercial[4]

Et dans le domaine de l'analyse de performances des logiciels d'applications nous pouvons citer :

- Load Simulator (Loadsim) [10], qui est libre source et est surtout dédié aux simulations de charges pour les serveurs web. Il existe une version pour Microsoft Exchange qui n'est pas libre source.

[3] L''émulation représente l'une des approches de la simulation. En revanche, elle se présente sous la forme d'un coeur de simulation avec des interfaces d'applications standard. Dans ce cas les applications ne sont pas des modèles.

[4] Une version gratuite de OPNET est disponible pour les chercheurs et les académiciens

2.1.2.2 La technique analytique

La technique analytique est l'un des moyens les plus rapides, comparé aux deux autres, pour évaluer à moindre coût les performances d'un système. Elle se base sur la modélisation des systèmes sous forme de paramètres, de variables et d'un ensemble des formules mathématiques qui régissent leurs relations.

Il s'agit de réduire le système en un modèle mathématique et de l'analyser numériquement. L'approche analytique est parfois rapide à réaliser, mais présente l'inconvénient de la représentation fidèle du système. Il est parfois très complexe voire impossible de modéliser le comportement réel du système mathématiquement. Généralement, il est nécessaire de poser des hypothèses qui simplifient l'étape de modélisation du système et rendent l'évaluation numérique faisable. Ces hypothèses simplificatrices peuvent toucher à la fidélité de représentation du système, mais permettent toutefois de traduire son comportement approché.

Cette technique se base sur plusieurs approches (Réseaux de files d'attente, Réseaux de pétri, chaînes de Markov,...) pour modéliser un système et prédire ses performances. La théorie de réseaux de files d'attente est la plus utilisée dans l'analyse de performances des systèmes informatiques. Elle représente et analyse, généralement, des systèmes à ressources partagées. Un modèle de réseau de files d'attente se présente sous la forme d'une collection de serveurs qui interagissent entre eux, représentant les ressources du système et d'un ensemble de clients qui représentent les utilisateurs partageant ces ressources. Ce modèle se représente formellement comme un graphe orienté direct avec des noeuds qui représentent ces serveurs et les liens entre eux qui représentent le comportement des sollicitations de clients à ces serveurs. Ces modèles sont analysés par plusieurs programmes de façon efficace pour obtenir des valeurs moyennes des indicateurs de performance du système modélisé.

La construction de ces modèles nécessite la réalisation des étapes suivantes :

1. La définition des centres de service (serveurs). Cette définition inclut leur nombre, leur classe, leurs clients et leur topologie.

2. La définition des paramètres du modèle : processus d'arrivée des clients, taux de service et le nombre des clients.

3. L'analyse pour obtenir une description quantitative du système modélisé en calculant la valeur de ses indicateurs de performance (utilisation de ressource, débit du système et temps de réponse). Ces indicateurs peuvent être locaux pour un serveur spécifique ou globaux pour tout le système.

A l'aide des techniques basées sur la modélisation, nous pouvons ainsi modéliser, et analyser les aspects variables d'un système. Cependant ces techniques nécessitent beaucoup de simplifications et d'hypothèses pour arriver à un modèle cohérent du système. Ces simplifications se présentent comme l'une de ses limites majeures et mettent en cause le degré de précision des résultats obtenus avec cette technique.

La Figure ci-après (figure 4) montre un réseau de files d'attente simplifié d'une architecture de supervision. Nous identifions trois centres de service[5] : le superviseur, le réseau et l'agent. Ce réseau de files d'attente représente les routes que les requêtes, provenant du superviseur, doivent suivre afin d'accomplir leurs taches de supervision. Les paramètres du modèle sont les taux de services (t_i, $1 \leq i \leq 3$) des différents centres de service.

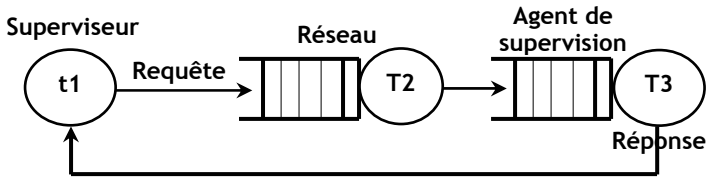


FIGURE 4 – Un réseau de files d'attentes d'un système de supervision.

2.1.3 Comparaison des techniques

Le tableau suivant représente une comparaison qualitative de trois techniques d'analyse des performances présentées précédemment.

Critère	Technique analytique	Technique de simulation	Technique de mesure
Etape	Ne nécessite pas l'existence d'un système réel	Nécessite l'existence d'un système réel ou prototype	Nécessite l'existence d'un système réel ou prototype
Temps requis	Peu	Moyen	Variable
Outils	Des analystes	- Langages de programmation - Simulateurs	- Instrumentation du code source - Moniteurs
Précision	Faible	Moyen	Variable
Compromis de l'analyse	Facile	Moyen	Difficile
Coût	Faible	Moyen	Elevé
Autres	Reproductible, mais pas très représentatif de la réalité	Reproductible, mais moyennement représentatif de la réalité	Non reproductible, mais réaliste

[5] Il s'agit bien d'un modèle de supervision de type manager/agent.

2.2 Les métriques de performance

Une métrique de performance est un critère de mesure choisi pour quantifier les performances d'un système. Dans cette section, nous allons essentiellement donner la définition de certaines métriques de performance qui existent dans la littérature [4,11], indépendamment des systèmes distribués en architecture client Serveur.

Les métriques de performance mettent en relation les trois paramètres relatifs à un service d'un système temps – débit - ressource : le temps mis par le système pour réaliser un service, le débit avec lequel le service est réalisé et les ressources consommées lors de la réalisation du service. Les métriques de performance sont classées en trois familles :

1. Les métriques de réponse : Ces métriques caractérisent la rapidité d'un système. Elles mesurent le temps écoulé entre l'invocation et la fin d'une opération. Généralement deux métriques sont utilisées dans cette catégorie :

– Le temps de réponse d'une simple opération. Par exemple, le temps nécessaire à un manager SNMP pour récupérer la valeur d'un ou d'un ensemble d'attributs depuis un ou plusieurs agents.

– Le temps de début à l'achèvement d'un ensemble d'opérations. Par exemple, le temps nécessaire pour résoudre un problème de congestion dans un réseau ou celui nécessaire à la renumérotation d'un réseau de grande taille.

2. Les métriques de production : Ces métriques mesurent la productivité d'un système. Elles représentent la quantité de travail accomplie par le système par unité de temps. Généralement deux métriques sont utilisées dans cette catégorie :

- Le nombre d'opérations réalisées par unité de temps. Par exemple pour un manager SNMP, le nombre de problèmes résolus par jour.

– La quantité de données générée pendant une opération donnée. Par exemple pour un manager SNMP, le nombre d'attributs scrutés par seconde.

Dans nos exemples précédents, pour qualifier un manager SNMP de performant, il faut qu'il réalise le maximum d'opérations de gestion tout en produisant un minimum de données de gestion. Ainsi, La première métrique doit être maximisée et la deuxième minimisée. Les points de mesure de ces métriques se situent donc au niveau du manager.

3. Les métriques d'utilisation : Ces métriques mesures les ressources consommées par un système. Nous identifions les métriques d'utilisation suivantes :

- L'utilisation du processeur
- L'utilisation des disques
- L'utilisation mémoire
- L'utilisation réseau.
- …

2.3 Métriques de performances utilisées dans les systèmes distribués

L'objectif de cette section est de présenter des métriques utilisées dans l'analyse des performances des systèmes informatiques distribués en architecture client-serveur. Dans cette catégorie nous citerons un ensemble non exhaustif de ces métriques :

a) Accélération

Accélération relative et Accélération absolue

L'accélération permet de déterminer de combien un programme parallèle est plus rapide qu'un programme séquentiel équivalent. On distingue deux façons de définir l'accélération : Relative et Absolue.

Définition 1 : *Soit T*(P) *le temps requis par un programme sur une machine parallèle à P processeurs. L'accélération* relative $S_r(P)$ *est alors défini comme :* $\frac{T(1)}{T(P)}$

En d'autres mots, on compare le programme s'exécutant sur une machine multi-processeurs avec P processeurs par rapport au même programme s'exécutant sur la même machine mais avec un seul processeur.

Définition 2 : *Soit T*(P) *le temps requis par un programme sur une machine parallèle à P processeurs. Soit* $T^*(1)$ *le temps requis pour* le meilleur programme séquentiel possible. *L'*accélération absolue $S_a(P)$ *est alors définie comme :* $\frac{T^*(1)}{T(P)}$

En d'autres mots, on compare le programme s'exécutant sur une machine multi-processeurs avec P processeurs avec le meilleur programme séquentiel permettant de résoudre le même problème.

Interprétation de l'accélération

- S(P) < 1 : On ralentit ! Autrement dit, l'utilisation de plusieurs processeurs ne diminue pas le temps d'exécution.
- 1 < S(P) < P: C'est «normal».
- S(P)=P : Nous sommes face à une accélération linéaire. C'est aussi l'accélération idéale. En d'autres mots, l'utilisation de P processeurs permet d'obtenir un programme P fois plus rapide. En général, de telles accélérations linéaires sont assez rares.
- S(P) > P: hyper accélération. Ce n'est pas magique, et ce n'est pas normal. On doit analyser le phénomène et l'expliquer; Corriger une erreur ou exploiter une optimisation. Deux situations expliquent généralement ce genre d'accélération :

→ La première est lorsque le programme est non déterministe (par exemple, une fouille dans un arbre) : dans ce cas, le fait d'utiliser plusieurs processus peut faire en sorte que l'un des processus est «chanceux» et trouve plus rapidement la solution désirée.

→ La seconde, mais qui s'applique à l'exécution du programme plutôt qu'au programme lui même, est la présence d'*effets de cache*.

La Figure ci-après résume l'interprétation de l'accélération.

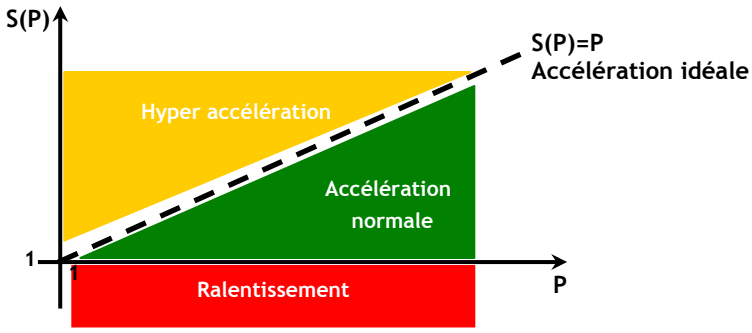


FIGURE 5 – Résumé interprétations de l'accélération.

b) Efficacité

Alors que l'accélération nous indique dans quelle mesure l'utilisation de plusieurs processeurs permet d'obtenir une solution plus rapidement, l'efficacité nous indique dans quelle mesure les divers processeurs sont utilisés. Il mesure donc le taux d'utilisation des processeurs.

L'efficacité E(P) *est alors définie comme* S(P)/P. Alors que pour P processeurs, l'accélération idéale est P, l'efficacité idéale est de 1, c'est-à-dire, le cas idéal est lorsque les P processeurs sont utilisés à 100 %.

Interprétation de l'efficacité

- En cas de ralentissement, c'est-à-dire dans le cas où l'accélération est inférieure à 1 : E(P) < 1/P, c'est-à-dire que E(P) < 50 % (puisque P[6] > 1). Donc moins de la moitié (50%) des ressources disponibles sont utilisées. Nous sommes donc face à ce nous appelons une « inefficacité » du système;

- Dans le cas « normal », c'est-à-dire dans le cas où 1 < S(P) < P, alors 1/P < E(P) < 1, c'est-à-dire 50 % < E(P) < 100 %. Donc plus de la moitié des ressources disponibles sont utilisées. Mais elles ne sont pas toutes utilisées;

- Dans le cas idéal, c'est-à-dire dans le cas où S(P)=P donc E(P)=100 %, toutes les ressources disponibles sont utilisées. C'est l'efficacité idéale;

- Dans le cas d'une hyper accélération du système, c'est-à-dire dans le cas où S(P) > P, alors E(P) > 100 %. Nous sommes donc face à une hyper efficacité. Ce n'est pas magique, et ce n'est pas normal. On doit analyser le phénomène et l'expliquer, corriger une erreur ou exploiter une optimisation. Deux situations expliquent généralement ce genre d'accélération :

  → La première est lorsque le programme est non déterministe (par exemple, une fouille dans un arbre) : dans ce cas, le fait d'utiliser plusieurs processus peut faire en sorte que l'un des processus est «chanceux» et trouve plus rapidement la solution désirée.

  → La seconde, mais qui s'applique à l'exécution du programme plutôt qu'au programme lui même, est la présence d'*effets de cache*.

[6] P est bien entendu un entier naturel

La Figure ci-après résume l'interprétation de l'efficacité.

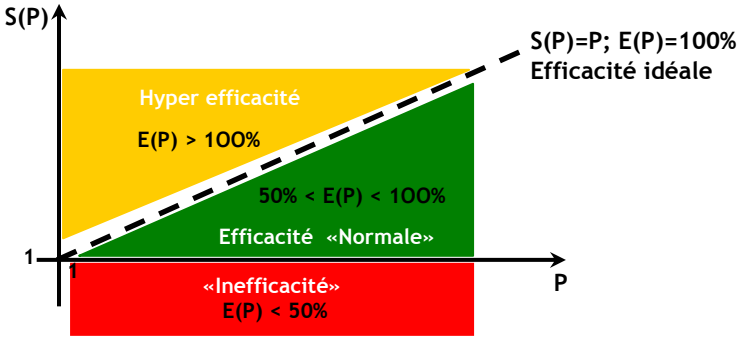


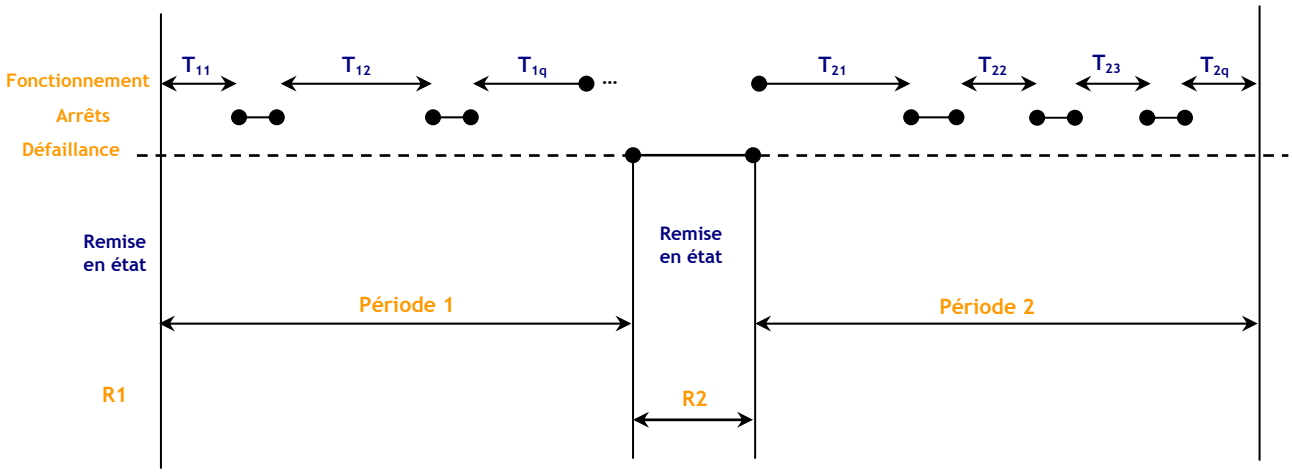
FIGURE 6 – Résumé interprétation de l'efficacité.

Remarque :

- L'utilisateur s'intéresse surtout à l'accélération obtenue
- L'acheteur de la machine s'intéresse beaucoup à l'efficacité
- Le développeur s'intéresse aux deux

c) Fiabilité

C'est une métrique liée à la sûreté de fonctionnement du système. La fiabilité définit donc la capacité du système à rendre continûment un service correct. Elle est mesurée à l'aide du MTBF (Mean Time Between Failure), c'est-à-dire le temps moyen entre l'apparition des erreurs dans le système.



MTTF (Mean Time To Failure):

$$MTTF1 = T_{11} + T_{12} + T_{1q}$$
$$MTTF2 = T_{21} + T_{22} + T_{2q}$$
$$\dots$$
$$MTTFn = T_{n1} + T_{n2} + T_{nq}$$

$$MTTF = \sum_{i=1}^{n} MTTF_i$$

MTTR (Mean Time To Repair):

$$MTTR = \sum_{i=1}^{n} R_i$$

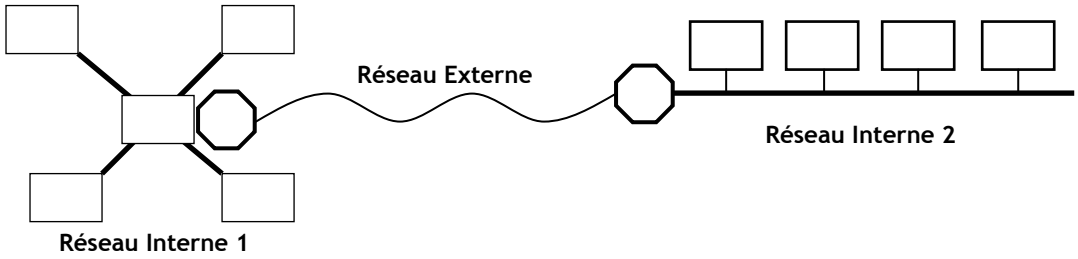
MTBF (Mean Time Between Failures): $MTBF = MTTF + MTTR$

d) Disponibilité

C'est une métrique liée à la sûreté de fonctionnement du système. La disponibilité définit donc la capacité du système à rendre un service correct à un instant donné.

Disponibilité = MTTF/MTBF

e) La bande passante



La bande passante « interne »

C'est une métrique liée à la performance du réseau interne. Cette métrique mesure donc la quantité d'information pouvant être véhiculée par unité de temps de chaque réseau «interne».

La bande passante « externe »

C'est une métrique liée à la performance du réseau «externe». Cette métrique mesure donc la quantité d'information pouvant être véhiculée par unité de temps dans réseau externe.

f) Temps de Transfert

C'est une métrique liée à la performance du réseau. Nous considérons que c'est le temps que met une requête pour traverser le réseau. Nous ne considérons pas le temps de traitement dans les nœuds d'extrémité.

g) Temps de latence

C'est une métrique liée à la performance du réseau. Nous considérons que c'est le temps entre le lancement d'une requête et la réception du résultat. Cette métrique mesure donc la réactivité.

h) Temps d'exécution

Dans les systèmes distribués en architecture client/serveur, c'est une métrique qui permet de mesurer les performances du serveur. Autrement dit le temps que met un serveur pour traiter des requêtes de clients.

3. Méthodologie d'analyse de performances des systèmes informatiques

L'objet de ce chapitre est de présenter une méthodologie d'analyse des performances des systèmes informatiques. Il existe dans la littérature [4, 11] des méthodologies, mais aucune d'elles ne constitue un standard. Devant ce constat, nous proposons dans ce chapitre certes une méthodologie, mais surtout un ensemble d'actons de «bonnes pratiques».

3.1 Choix de techniques d'analyse des performances

L'objectif de cette section est de définir les critères de choix d'une ou plusieurs techniques adéquates pour l'analyse de performance d'un système informatique. Comme nous venons de le voir dans le chapitre précèdent (section 2.1), il existe trois techniques candidates d'analyse de performance : mesure, analytique et simulation. L'utilisation d'une seule technique pour analyser les performances d'un système n'est pas recommandée. Il faut envisager d'utiliser au moins deux techniques d'une façon séquentielle, pour que l'une puisse valider les résultats de l'autre. Idéalement, la technique utilisée, devrait répondre aux critères suivants :

- *Le **coût de la technique** :* ce coût ne doit pas être considérable en terme de temps de réalisation des tests et de leur coût financier. La technique de mesure est la plus coûteuse puisqu'elle nécessite l'existence d'un prototype du système. La technique analytique est la moins coûteuse en terme de coût financier puisqu'elle ne nécessite pas le développement ou le déploiement d'un prototype du système. La technique de simulation possède un coût variable. Son coût dépend du type de simulation utilisé, de l'existence ou non d'un modèle de l'architecture à simuler dans l'outil de simulation utilisé.

- *La **faisabilité de la technique** :* le choix de la technique dépend de l'environnement du système sous test. Certains environnements peuvent défavoriser l'utilisation d'une technique par rapport à une autre. Par exemple l'analyse des performances d'un système embarqué dans un satellite défavorise l'utilisation de la technique de mesure. Les mesures vont altérer et dégrader le fonctionnement du système (consommation excessive des batteries). Dans ce cas, les techniques de simulation ou analytique sont plus adéquates. En revanche, la non disponibilité des modèles communs pour les architectures favorise l'utilisation de la technique de mesure, étant donné l'existence de prototypes de ces architectures.

- *La **flexibilité de la technique** :* le modèle utilisé par une technique d'analyse doit être facile à modifier et à étendre. Les architectures des systèmes informatiques et leurs environnements sont constamment en évolution. Ainsi, la technique d'analyse utilisée, doit offrir la possibilité d'une intégration facile de ces évolutions dans les modèles élaborés.

- *La **précision de la technique** :* elle se présente sous la forme du degré de précision des résultats obtenus par une technique d'analyse de performance d'un système donné. La technique analytique est généralement la moins précise. Elle nécessite des simplifications du système sous test afin de construire un modèle calculable sous forme de formules mathématiques pour donner une estimation des métriques de performance. Les modèles élaborés par la technique de simulation incluent plus de détails du système sous test et nécessitent moins de simplifications. Ainsi les résultats seront potentiellement plus précis. La précision de la technique de mesure est variable. Le degré de précision de résultats obtenus peut être fort comme faible. Cela est dû au choix de l'ensemble des paramètres, des facteurs et de la charge de performance qui sont relatifs à l'expérimentation réalisée par cette technique.

- ***L'impact de la technique** :* la technique choisie ne doit pas altérer les performances du système. Nous constatons que ces critères peuvent être en conflit, lors du choix d'une technique adéquate pour analyser les performances d'un système. Par exemple, la technique analytique est la moins coûteuse mais elle est la moins précise. Systématiquement, on peut se baser sur les deux critères suivants pour choisir une technique parmi les trois :

- Si le système sous test existe et qu'il est accessible avec un effort raisonnable, nous pouvons utiliser la technique de mesure.

- Si le système sous test n'est pas implanté ou s'il est trop compliqué d'accès, alors un modèle de performance doit être développé en utilisant la technique analytique ou en construisant un modèle de simulation pour analyser ses performances.

3.2 Choix de métriques d'analyse des performances

Le but ultime des métriques d'analyse des performances est de fournir aux « analystes » une compréhension commune des performances des systèmes analysés indépendamment des architectures qu'ils utilisent. Pour atteindre ce but, ces métriques doivent satisfaire à des critères qu'il est recommandable d'appliquer avant toute analyse de performance de systèmes informatiques. Nous définissons cet ensemble de critères de choix des métriques d'analyse de performance :

– Les métriques doivent être concrètes et bien définies : il faut essentiellement éviter les métriques stochastiques (probabilités) et se focaliser sur des métriques déterministes. Par exemple, il est recommandé d'éviter des métriques du genre probabilité de réponse d'un agent de supervision à des requêtes envoyées par un superviseur, mais plutôt définir une métrique du genre taux de réponse de l'agent en terme de nombre de requêtes correctes servies par seconde. Par exemple, une mesure selon la définition stochastique de cette métrique nous donne une valeur de genre 0.80, et la deuxième définition nous donne une valeur de genre 80 requêtes correctes servies sur 100. En effet, une définition déterministe nous permet d'éviter certaines hypothèses stochastiques qui peuvent être implicites pour une définition probabiliste.

– La méthodologie de mesure d'une métrique doit avoir un caractère répétitif : si une méthodologie de mesure d'une métrique est utilisée plusieurs fois sous des conditions identiques, les résultats obtenus doivent être quasi identiques.

– Les métriques ne doivent pas être aberrantes pour des architectures de systèmes implantés par des technologies identiques. Elles doivent être valables pour ces différentes architectures. Par exemple les métriques d'analyse des performances d'un système de gestion de bases de données doivent être identiques que l'on soit dans une architecture Windows, ou Unix/Linux.

– Les métriques doivent fournir une compréhension claire des architectures de systèmes implantés par des technologies non identiques.

3.3 Paramètres de sélection de la charge

La charge représente l'ensemble des services demandés au système au cours du processus d'analyse de ses performances. La charge est caractérisée selon la technique d'analyse utilisée. Dans les modèles analytiques, elle se caractérise par un modèle représentant les probabilités d'arrivée des requêtes sur le système. Dans les études basées sur les simulations, nous pouvons utiliser soit des traces de requêtes mesurées sur un système réel ou un modèle représentant ces requêtes. Dans la technique de mesure, nous pouvons utiliser soit une charge de gestion réelle basée sur un trafic opérationnel, soit des injecteurs de requêtes demandant des services au système. Dans tous les cas, il est important que la charge soit représentative d'une utilisation réelle du système.

Deux types de charges existent : les charges réelles et les charges synthétiques

La charge réelle : Elle correspond à une charge observée sur un système réel au cours de son utilisation pour des opérations ordinaires. Ce type de charge ne peut pas être régénéré en raison de son caractère variable au cours de temps.

La charge synthétique : Elle possède les mêmes caractéristiques qu'une charge réelle, à la seule différence qu'elle peut être régénérée plusieurs fois d'une façon contrôlée. C'est ce type de charge qui convient le mieux pour les études d'analyse de performance. Une raison majeure d'utiliser ce type de charge pour l´analyses de performances, est qu'elle représente un modèle de charge réelle avec la caractéristique d'être facilement modifiable. La charge d'une application distribuée gestion de réseau peut être représentée sous la forme de l'ensemble des requêtes envoyées par les programmes de gestion vers les agents. Par exemple Pattinson [8] a évalué les performances de SNMPv1 avec une charge qui se caractérise par des tailles respectives de requêtes et de réponses SNMPv1 de 90 et 100 octets. Le choix de ces valeurs d'intervalles n'est pas arbitraire, puisqu'elles sont utilisées par des systèmes de gestion réels (SUN's Network manager, …).

Les trois paramètres à considérer lors de la caractérisation d'une charge d'analyse sont les suivants :

- Le service exercé par la charge
- Le niveau de détail de la charge
- Sa représentativité d'une certaine réalité.

3.4 Mise en œuvre de la solution d'analyse des performances

Il est surtout question dans cette partie de voir concrètement comment la solution d'analyse des performances peut être mise en oeuvre. Nous résumons cette mise en œuvre en deux grandes étapes :

1- Le choix de l'outil d'analyse des performances
2- La mise en œuvre proprement dite

Le choix de l'outil d'analyse des performances

Le choix de l'outil d'analyse dépend de la technique d'analyse choisie et du contexte dans lequel l'analyse des performances est faite. Il est conseillé de ne surtout pas chercher à réinventer la roue. Autrement dit, il est recommandé d'utiliser les standards du marché en lieu et place des développements « maison » ou « personnel ». Dans la section 2.1.3 nous avons dressé un tableau comparatif des techniques d'analyse des performances des systèmes informatiques. Ce tableau nous montre qu'au niveau du coût, les outils basés sur les techniques de mesure sont relativement plus onéreux que ceux basés sur les techniques analytiques et ceux basés sur la simulation. En revanche ces derniers offrent dans l'ensemble un bon compromis et restent de loin les plus utilisés.

Dans le marché bon nombre d'outils combinent la possibilité :

- de faire une analyse des performances par simulation en prenant en entrée une charge synthétique;
- de faire une analyse des performances basée sur la mesure (réelle) par « branchement » sur le système réel ce qui permet de prendre en entrée la charge réelle du système.

La mise en œuvre proprement dite

La mise en œuvre dans notre entendement consiste à déployer la solution en veillant à ce qu'elle remplisse pleinement, correctement et efficacement son rôle. Par exemple un moniteur, c'est-à-dire un outil basé sur la technique de mesure (out-side) qui mobilise une part importante des ressources du système pour analyser les performances du même système est à proscrire, quand bien même les résultats produits par l'outil sont pertinents.

3.5 Actions de « bonnes pratiques »

Ayant constaté l'absence de standard en ce qui la méthodologie dans le domaine de l'analyse des performances des systèmes informatiques, nous proposons dans cette partie une ensemble d'actions de « bonnes pratiques » pour mener à bien un projet dans ce domaine.

Etapes- Actions	Description
1- Définir le problème et l'objectif.	Qu'essayez vous exactement d'analyser ? En quoi un processus d'analyse des performances va-t-il vous aider à prendre une décision ultérieure ? Combien de temps et quelles ressources voulez-vous consacrer à cet effort ?
2- Définir le système	
3- Choisir les métriques et les facteurs de performance	L'objectif n'est pas d'être exhaustif, mais de choisir les métriques et les facteurs de performances les plus pertinents.
4- Sélectionner et dérouler la technique d'analyse	Il n'y pas de techniques dédiées à des problèmes précis. La sélection d'une technique d'analyse dépendra du contexte et des objectifs
5- Choisir l'outil d'analyse des performances	Ne pas réinventer la roue. Utiliser de préférence un outil standard du marché.
6- Analyser et interpréter des données	Bien analyser les données produites. Ne pas hésiter à repasser l'analyse plusieurs fois.
7- Présenter les résultats	Utiliser une présentation claire et compréhensible. L'utilisation des graphiques est fortement conseillée.

Aux étapes et actions du tableau précédent, nous ajoutons l'action transversale qui consiste à utiliser séquentiellement une technique basée sur la modélisation (pour élaborer un prototype du futur système) et une autre basée sur la mesure (pour mesurer le système résultat) afin de comparer les résultats obtenus avec ceux qui avait été prédits lors de la modélisation.

La figure ci-après résume cette action transversale.

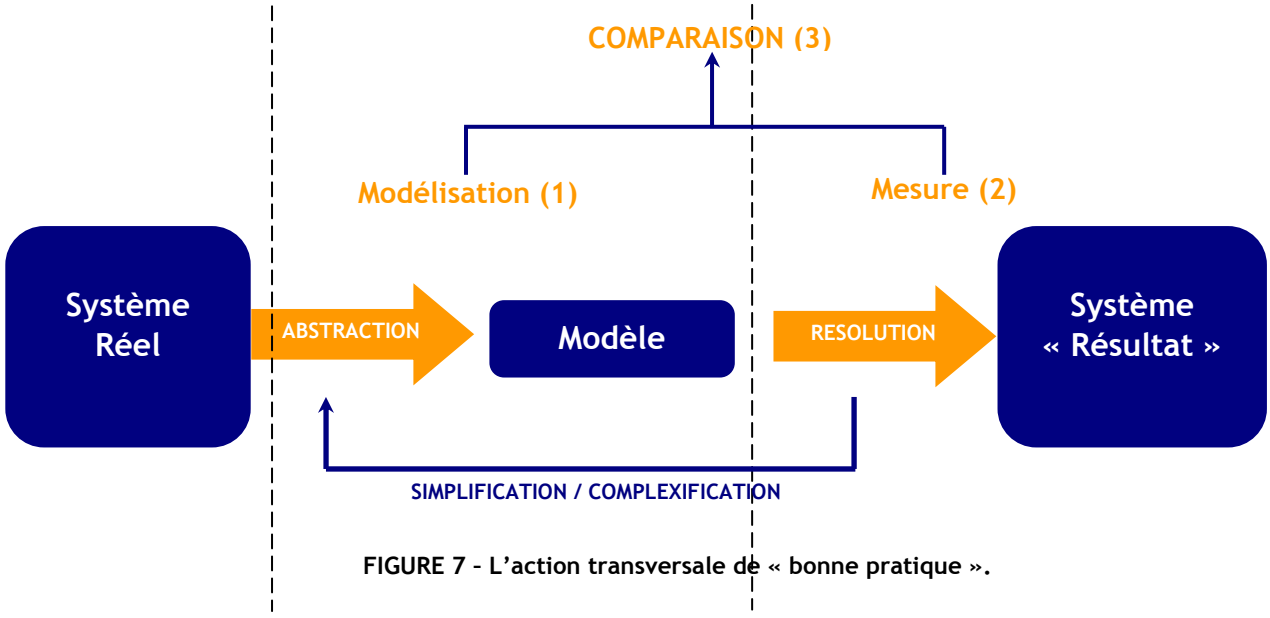


FIGURE 7 – L'action transversale de « bonne pratique ».

Certes les étapes et les actions doivent être déroulées de manière incrémentale, mais aussi de manière itérative.

La figure ci-après résume le déroulement de ces étapes et actions

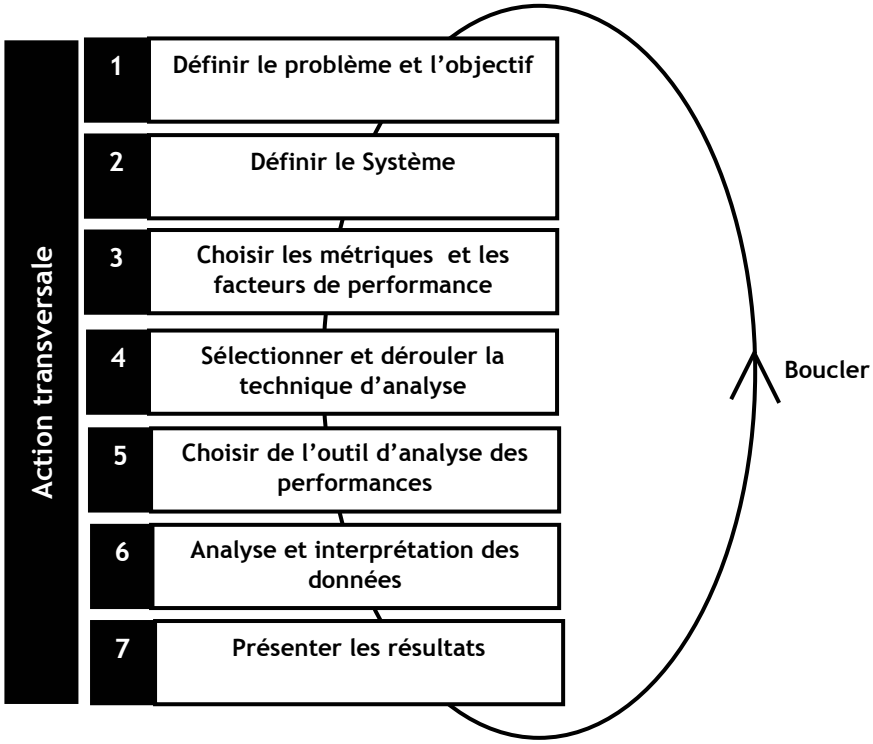


FIGURE 8 - Les étapes et actions de « bonne pratique ».

Quelques erreurs généralement rencontrées

- Pas d'objectifs
- Pas d'analyse ou analyse erronée
- Analyse sans compréhension du problème
- La charge non représentative de la réalité
- Faire trop de simplifications lors de la phase de modélisation
- Choix des métriques de performances inadaptées et non pertinentes
- Choix d'une technique d'analyse inadaptée au problème
- Choix d'un outil d'analyse des performances inadapté
- Négliger les performances de l'outil. Quelles ressources l'outil d'analyse des performances consomment-il pour effectuer ses requêtes ?
- Mauvaise interprétation des données
- Présentation impropre des résultats

4. Exemple d'analyse performances

Dans cette partie, nous proposons de voir concrètement l'analyse des performances des systèmes informatiques à l'aide d'un exemple qui portera sur l'analyse des performances d'un système d'exploitation réseau, en l'occurrence Microsoft Windows 2003 dans le cadre d'une alimentation d'un DataWareHouse hébergée par le système sus-cité.

- Définition du problème et objectif

Le but ici est d'analyser le comportement du système d'exploitation Microsoft Windows 2003 pendant la phase de chargement d'une base de données orientée décisionnelle (entrepôt de données). Une attention particulière sera faite sur l'analyse des composants suivants :

- Processeur
- Mémoire
- Disques physiques
- Cache

- Définition du Système

Architecture du système

L'architecture du système est résumée dans la Figure ci-après

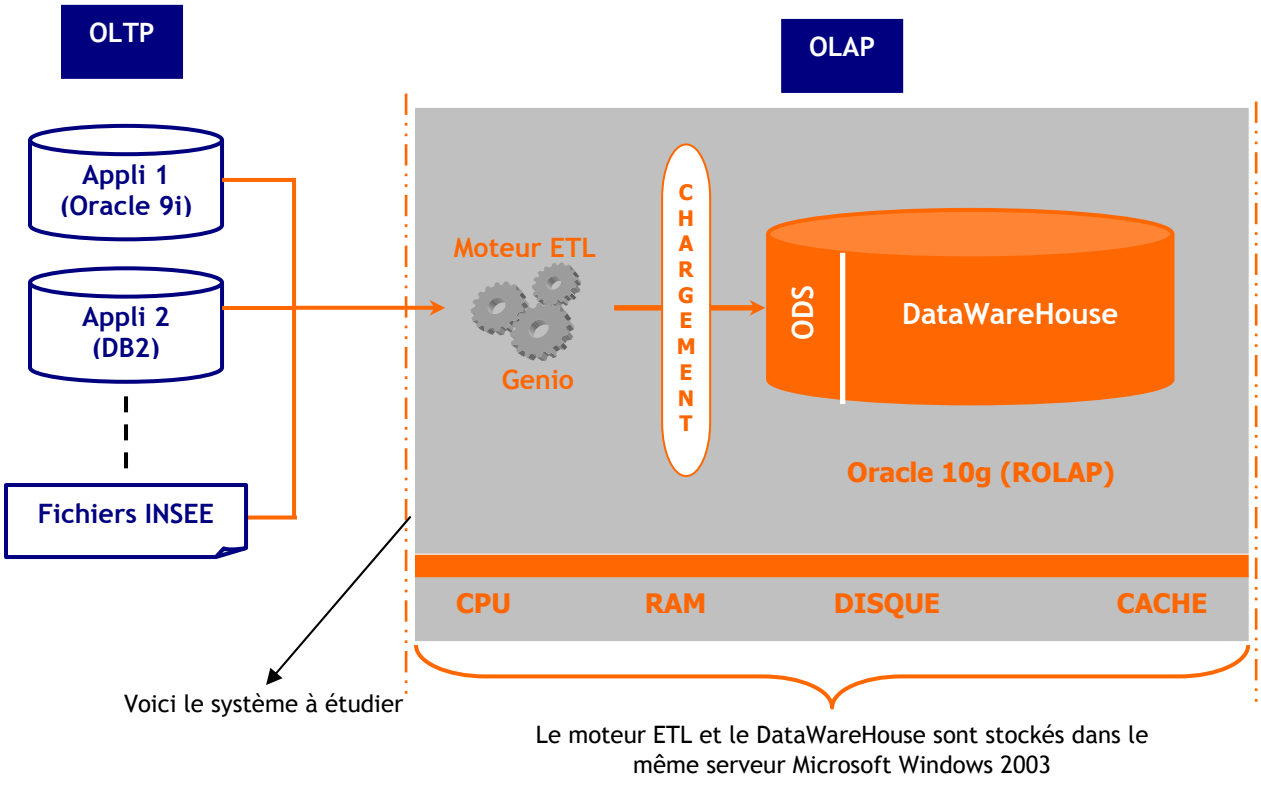


FIGURE 9 - Architecture Système de l'étude de cas

Nous résumons ci-après quelques caractéristiques techniques du système

Serveur :	Fabricant : Dell Inc. Modèle : PowerEdge SC1420 N° de série : B1DCC1J ID : 4C4C4544-44100031-C0C24380-4A31434F
OS :	Windows 2003 Standard Edition N° Version: 5.2.3790
CPU :	Intel(R) Xeon Fréquence : 2.80 GHz Mono CPU
Mémoire	Capacité totale 1,00 GO Vitesse du bus RAM : 4 * 100 MHz (400 MHz taux de transfert)
Disque physique	Maxtor 6Y160M0 (149 Go)
Disque logique	C : 12 Go (NTFS) D : 137 Go (NTFS)
Services	Serveur d'application et de base de données orientée décisionnelle

- Choix des métriques et des facteurs de performance

Ci-après quelques composants et pour chacun une liste non exhaustive de métriques (compteurs) associés :

Composant	**Aspect des performances en cours d'analyse**	**Compteurs à analyser**
Mémoire	Utilisation	- Mémoire\Octets disponibles - Mémoire\Octets du cache
	Engorgements ou fuites	- Mémoire\Pages/s - Mémoire\Lectures de page/s - Mémoire\Défauts en transit/s - Mémoire\Octets de réserve paginée - Mémoire\Octets de réserve non paginée. Les compteurs suivants, bien qu'ils ne soient pas spécifiques aux objets de mémoire, sont également utiles pour analyser la mémoire : - Fichier d'échange\% objet Usage (toutes les instances) - Cache\Présence des données mappées - Serveur\Octets de réserve paginée et - Serveur\Octets de réserve non paginée
Processeur	Utilisation	- Processeur\% Temps processeur (toutes les instances)
	Engorgements	- Système\Longueur de la file du processeur - Processeur\Interruptions/s - Système\Changements de contexte/s

Disque	Utilisation	- Disque physique\Lectures disque/s - Disque physique\Écritures disque/s - Disque logique\% Espace libre
	Engorgements	Disque physique\Long. moy. de file d'attente du disque
Fichier d'échange	Fichier d'échange\Pourcentage d'utilisation	Analyser cette valeur en même temps que celle du compteur Octets disponibles et Pages/s pour comprendre l'activité de pagination sur votre ordinateur.
Réseau	Débit	Compteurs de transmission de protocole (varie avec le protocole de mise en réseau) ; pour TCP/IP : - Interface réseau\Total octets/s - Interface réseau\Paquets/s - Serveur\Total octets/s - Serveur\Octets transmis/s - Serveur\Octets reçus/s

- Choix de la technique d'analyse

Notre système existe déjà. Donc il est plus judicieux de choisir une technique basée sur la mesure et plus exactement celle basée sur les moniteurs. Comme outil nous utiliserons L'analyseur de performance (rebaptisé «La console des performances» depuis Microsoft Windows 2003) livré en standard avec Microsoft Windows 2003.

- Type de moniteur

Moniteur matériel et logiciel

- Lancement de l'outil :

Le lancement se fait via « Démarrer\Programmes\Outils d'administration\performances »

- Présentation brève de l'outil

La console des performances Microsoft Windows 2003 comprend les outils suivants :

- Le Moniteur Système
- Les journaux et alerte de performance

Les outils Moniteur système et Journaux et alertes de performance fournissent des données détaillées sur les ressources utilisées par des composants spécifiques du système d'exploitation et par les programmes destinés à collecter les données de performance. Les courbes offrent une représentation graphique des données d'analyse des performances. Les journaux permettent d'enregistrer les données. Les alertes envoient des notifications aux utilisateurs lorsqu'une valeur de compteur atteint un seuil défini, chute sous ce seuil ou le dépasse. Il possède deux modes de fonctionnement : en temps réel ou en différé avec un journal. Par défaut, toute l'analyse se fait en temps réel. Nous avons utilisé le mode «temps réel».

- Analyse proprement dite

Nous ne détaillerons dans cette partie que l'exemple d'analyse des performances du composant «processeur» plus exactement son utilisation et son engorgement.

Choix des composants et métriques

L'analyse consiste essentiellement à dérouler les trois étapes suivantes :

- Choix de l'ordinateur hébergeant le système
- Choix de l'objet de performances (c'est le composant). Il peut s'agit d'un composant «physiques» comme le processeur, la mémoire ou d'un composant «logiciel» comme un serveur d'application (IIS,…)
- Pour chaque composant, choisir des métriques (compteurs)

Type de charge : Charge réelle

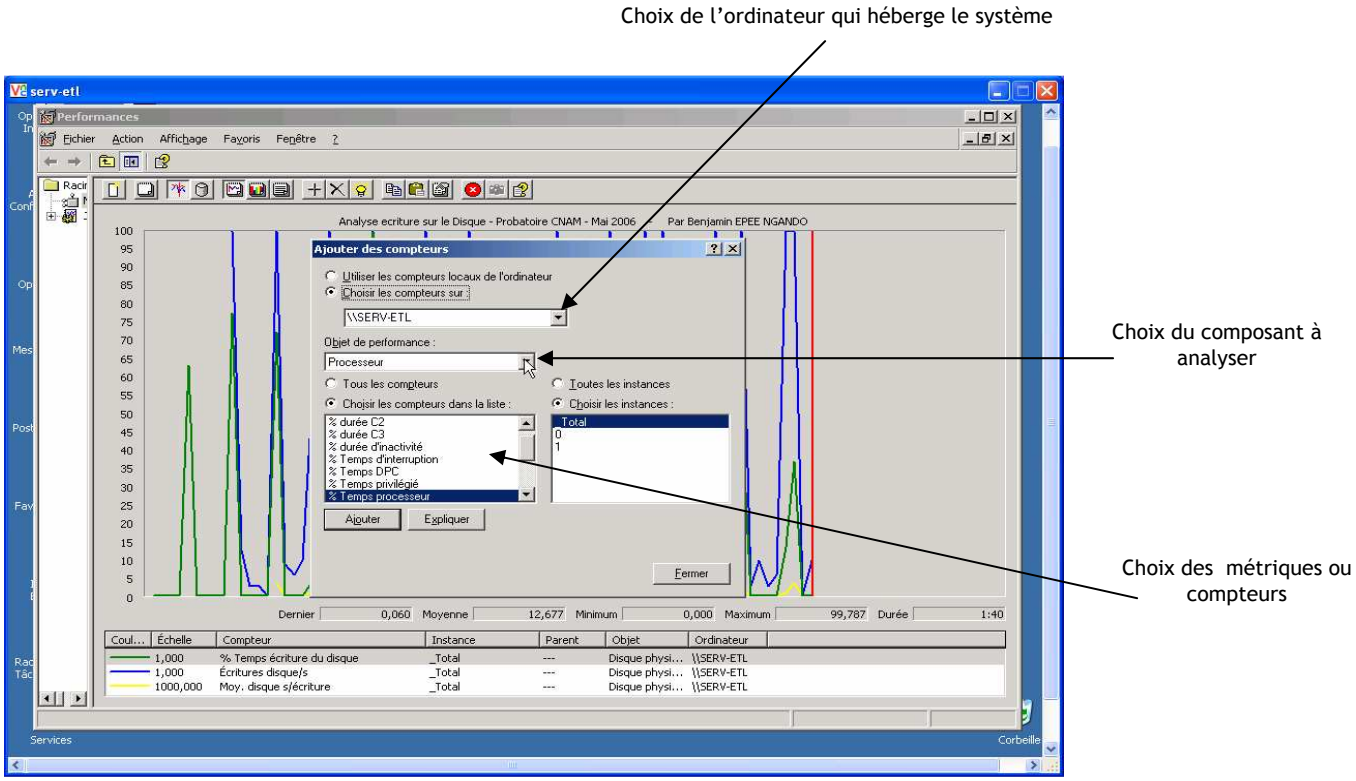


FIGURE 10 – Paramétrage de composants dans l'analyseur de performances

Finalisation des paramétrages

Dans cette partie nous achevons le paramétrage en fixant les valeurs de certains paramètres importants comme la période (respectivement la fréquence) d'échantillonnage. Dans notre exemple elle est d'une seconde.

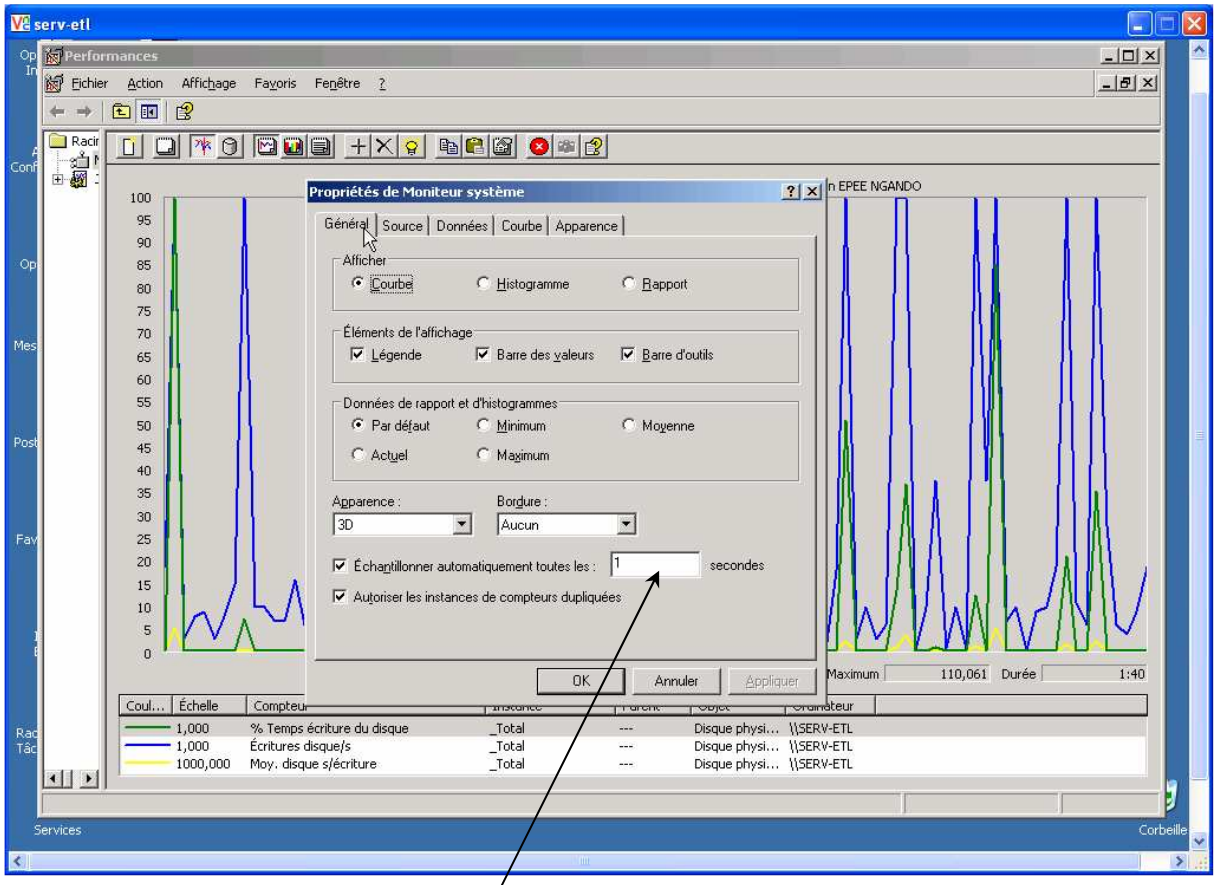


FIGURE 11 – Paramétrages des propriétés du moniteur Système dans l'analyseur de performances

Analyse des performances du composant « Processeur »

Métriques utilisées

Le tableau ci-après résume les métriques que nous avons utilisées pour analyser les performances du composant processeur. Ces métriques permettent de mesurer essentiellement l'utilisation et l'engorgement du processeur.

Métriques (Composant\Compteur)	Description
Processeur\% Temps processeur	C'est le pourcentage de temps que le processeur utilise pour exécuter des threads actives. Il est calculé en mesurant la durée d'activité de la thread inactive dans un intervalle de temps et en soustrayant ce temps à la durée de l'intervalle. (Chaque processeur a une thread inactive qui consomme des cycles lorsque aucune autre thread n'est prête à s'exécuter.) Ce compteur est l'indicateur principal de l'activité du processeur et affiche le pourcentage de temps sur l'intervalle échantillon passé à faire un travail utile. Il est calculé en surveillant le temps d'inactivité du service et en y soustrayant la valeur de 100%.
Processeur\ Interruptions/s	C'est le taux moyen, en incidents par seconde auquel le processeur a reçu et corrigé des interruptions matérielles. Cette valeur est un indicateur indirect de l'activité des périphériques tels que l'horloge système, la souris, les pilotes de disque, les lignes de communication de données, les cartes d'interface réseau et d'autres périphériques. Ces périphériques interrompent normalement le processeur quand ils ont fini une tâche ou qu'ils ont besoin d'assistance. L'exécution d'une thread normale est interrompue. L'horloge système interrompt le processeur toutes les 10 millisecondes, créant un arrière-plan à l'activité de l'interruption. Ce compteur affiche la différence entre les valeurs observées dans les deux derniers échantillons, divisé par la durée de l'intervalle d'échantillon.
Système\Longueur de la file du processeur	C'est le nombre de threads dans la file du processeur. À la différence des compteurs disque, ce compteur ne dénombre que les threads prêtes et non les threads en cours d'exécution. Il n'y a qu'une seule file pour le temps processeur, même sur les ordinateurs avec plusieurs processeurs. Par conséquent, si un ordinateur a plusieurs processeurs, vous devez diviser cette valeur par le nombre de processeurs s'occupant de cette charge de travail. Une file de processeurs maintenue à moins de 10 threads par processeur est normalement acceptable, selon la charge de travail.
Système\Changements de contexte/s	C'est le taux combiné auquel tous les processeurs de l'ordinateur commutent d'une thread à une autre. Les commutations peuvent intervenir lorsqu'une thread en cours d'exécution abandonne un processeur, et est devancée par une thread de plus haute priorité ou commute du mode Utilisateur au mode Privilégié pour utiliser un service. Ce compteur est la somme des valeurs Changements de contexte/s de l'objet Thread pour toutes les threads s'exécutant sur tous les processeurs de l'ordinateur et est mesuré sur les objets Thread et Système. Il affiche la différence entre les valeurs observées dans les deux derniers intervalles de temps, divisée par la durée de l'intervalle échantillon.

La figure ci-après nous montre une capture d'écran de l'analyse des performances du processeur à l'aide des métriques sus citées.

FIGURE 12 – Visualisation des métriques d'analyse des performances du processeur

Conclusion

En définitive, l'analyse des performances est un nouveau défi dans le domaine des systèmes informatiques, notamment les systèmes informatiques distribués. Un projet d'analyse des performances de systèmes informatiques consiste à faire le choix sur un ensemble d'éléments de base. Ces éléments incluent essentiellement les techniques d'analyse de performances, les métriques pour quantifier ces performances et la charge de test.

Concernant le choix de la technique d'analyse, une seule technique peut être suffisante, mais nous recommandons l'utilisation de plusieurs techniques. Par exemple l'on pourra utiliser une technique basée sur la modélisation (pour élaborer un prototype du futur système). Ensuite une autre technique basée sur la mesure (pour mesurer le système résultat), puis enfin comparer les résultats obtenus avec ceux qui avait été prédits lors de la modélisation.

Un autre choix important pour l'analyse des performances est celui des métriques de performance. Dans ce mémoire, ces métriques représentent les indicateurs de performance des systèmes informatiques. Une attention a été portée sur les métriques utilisées dans les systèmes informatiques distribués. Aussi important que les deux autres choix, la charge caractérise et représente les services d'un système informatique dans ses différents cas réels. Dans l'analyse des performance des systèmes informatiques, nous avons identifié deux types de charge : la charge réelle et la charge synthétique (détaillés dans la section 3.3 du chapitre 3)

Nous avons également constaté l'absence de standard en ce qui concerne la méthodologie d'analyse des performances des systèmes informatiques. Nous sommes persuadés qu'il est nécessaire d'avoir une méthodologie standard spécifiant cet ensemble de choix de base pour toute étude future de performance portant sur les systèmes informatiques. Dans ce mémoire, nous avons défini une méthodologie d'analyse des performances des systèmes informatiques en spécifiant un ensemble de recommandations et d'actions pour ces choix de base. Enfin nous avons appliqué cette méthodologie sur un exemple d'analyse de performance du système d'exploitation réseau Microsoft Windows 2003 dans le cadre d'une alimentation d'un DataWareHouse, en utilisant l'outil «La console des performances» livré en standard avec Microsoft Windows 2003.

Glossaire

Architecture distribuée : L'architecture d'un environnement informatique ou d'un réseau est dite distribuée quand toutes les ressources ne se trouvent pas au même endroit ou sur la même machine.

Benchmark : Mot signifiant à l'origine repère, point de référence. En résumé, effectuer un benchmark revient à effectuer un test pour analyser et comparer les performances d'un système informatique à celles d'un ou plusieurs autres systèmes informatiques.

CORBA (Common **O**bject **R**equest **B**roker **A**rchitecture) : C'est une architecture logicielle, pour le développement de composants et d'objets distribués en architecture client serveur. Ces composants, qui sont assemblés afin de construire des applications complètes, peuvent être écrits dans des langages de programmation distincts, être exécutés dans des processus séparés, voire être déployés sur des machines distinctes.

DataWarehouse: D'après Bill Inmon [12] un DataWarehouse est une « Collection de données orientées sujet, intégrées, non volatiles et historisées, organisées pour le support d'un processus d'aide à la décision».

ETL (Extract **T**ransform and **L**oad) **:** Il s'agit d'un outil qui a pour vocation de se connecter à un ensemble de données sources, d'y appliquer des transformations afin d'harmoniser les formats de données et de charger ces données homogénéisées dans une base de données cible qui est habituellement un DataWarehouse.

JMX (Java **M**anagement e**X**tensions) **:** C'est une spécification de Sun fournie à travers le JCP (Java Community Process) qui permet d'administrer à distance des applications Java, mais également des équipements d'administration de réseau (firewall, routeur,...).

ODS (Operational **D**ata **S**tore**) :** Espace de préparation, réceptacle de données issues des systèmes opérationnels. Base de données intermédiaire avant spécialisation, modèle d'intégration, structure intermédiaire qui stocke les données issues des systèmes de production opérationnelle dans un format proche de ces derniers. C'est un stockage tampon (un sas) avant l'intégration des données dans le DataWarehouse proprement dit. Elle est à vocation plus tactique que stratégique.

OLAP (On-**L**ine **A**nalytical **P**rocessing) : Désigne une catégorie d'applications et de technologies permettant de collecter, stocker, traiter et restituer des données multidimensionnelles, à des fins d'analyse.

OLTP (On-**L**ine **T**ransactional **P**rocessing) : Désigne une catégorie et d'applications et de technologies permettant de gérer des données transactionnelles.

SNMP (Simple **N**etwork **M**anagement **P**rotocol) : Ou protocole simple de gestion de réseau en Français. Il s'agit d'un protocole de communication qui permet aux administrateurs réseau de gérer les équipements du réseau et de diagnostiquer les problèmes de réseau.

Table des figures

Bibliographie

[1] The Open Group. Application Response Measurement.
http://www.opengroup.org/tech/management/arm/.

[2] SUN Microsystems. The netbeans profiler project.
http://profiler.netbeans.org/.

[3] The GNU project. The GNU profiler, gprof.
http://www.gnu.org/software/binutils/manual/gprof-2.9.1/html_mono/gprof.html

[4] Raj Jain. The art of Computer Systems Performance Analysis. John Wiley & Sons, Inc, 1991. ISBN : 0-471-50336-3.

[5] Information Sciences Institute. Network Simulator.
http://www.isi.edu/nsnam/ns/doc/

[6] OPNET Technologies. The OPNET modeler.
http://www.opnet.com/products/modeler.

[7] C. Bohoris, A. Liotta, and G. Pavlou. Software agent constrained mobility for network performance monitoring. In 6th IFIP Conference on Intelligence in Networks: SmartNet, 2000.

[8] Collin Pattinson. A study of the behaviour of the simple network management protocol. In DSOM (International Workshop on Distributed Systems: Operations & Management), October 2001.

[9] Abdelkader Lahmadi, Laurent Andrey, Olivier Festor. Évaluation de performance des architectures de gestion de réseaux: état de l'art et perspectives. INRIA (Institut National de Recherche en Informatique et en Automatique), Juin 2005. Rapport de recherche N° 5598, ISSN 0249-6399

[10] Open Souce Web testing tools in Java.
http://java-source.net/open-source/web-testing-tools/loadsim

[11] Raj Jain. The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. Dernière modification, Août 2003
http://www.cse.wustl.edu/~jain/books/perfbook.htm

[12] Corporate Information factory. Resources by Inmon Data Systems
http://www.inmoncif.com