

Performance des systèmes :

Étalons et méthodes

Janvier 2003

René J. Chevance

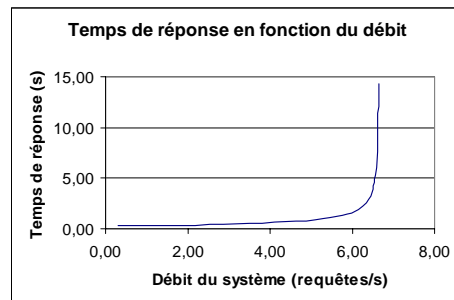
- Remarques préliminaires
- Grandeurs caractéristiques
- Approches des performances
- Niveaux d'expression de la performance
- Performances au niveau processeur
 - Facteurs de la performance des processeurs
 - SPEC CPU2000 (System Performance Evaluation Corporation)
- Performances au niveau système
 - SPEC JVM98, SFS97, WEB99, JBB2000
 - TPC
 - Transactionnel et transactionnel Web
 - Décisionnel
- Prise en compte des aspects performance dans les projets
- Techniques de modélisation des performances
 - Réseaux de files d'attente
 - Analyse opérationnelle
- Méthodologie de modélisation
- Prévion de la charge

Remarques préliminaires

- **Objectif de cette présentation**
 - Présentation des étalons (benchmarks ou bancs d'essai de performance)
 - Approche méthodologique pour la prise en compte des aspects performance dans les projets
- **Quelques évidences**
 - La performance des serveurs dépend d'un grand nombre de paramètres : puissance de traitement, architecture du système, caractéristiques de l'application,.....
 - Toute activité de mesure et d'évaluation de performances suppose la compréhension du fonctionnement du système
 - Les activités de mesure et de modélisation sont indispensables pour la bonne conduite des projets
 - La maîtrise des performances est à ce prix
- **Chiffres**
 - Bien que la performance s'exprime à l'aide de chiffres, cette présentation n'en contient pas car ils sont très souvent révisés. Nous indiquons les sites Web qui sont à la source des résultats.

Grandeurs caractéristiques

- **Deux grandeurs permettent de qualifier la performance des systèmes:**
 - Débit du système : nombre de requêtes exécutées par unité de temps
 - Temps de réponse : temps nécessaire au traitement d'une requête par le système
- **Ces deux grandeurs sont liées**
- **Les phénomènes de saturation limitent le débit du système et conduisent à une augmentation du temps de réponse. Le schéma ci-dessous montre l'évolution du temps de réponse lors que l'on augmente le flux de requêtes soumises au système**



Approches des performances

Constat

- **L'aspect performance est, assez souvent, le parent pauvre dans les projets**
- **Parmi les difficultés classiques rencontrées suite à une absence de considération des aspects « performance » dans les projets, on peut citer:**
 - le traitement tardif et "à chaud" de ces problèmes conduit à des résultats médiocres voire même désastreux : impact négatif sur coûts et délais, dé-structuration du système,...
 - difficulté à prévoir les conséquences de changements sur le comportement du système (e.g. autres missions pour le système, changements de technologies,...)
 - difficulté à analyser les problèmes rencontrés dans ces domaines
- **La même remarque s'applique aussi à la disponibilité. La démarche « modélisation », qui sera développée dans la seconde partie de ce cours, s'applique tout aussi bien aux aspects disponibilité**

- Identifier les caractéristiques de l'application
- Caractériser l'architecture proposée:
 - Architecture système
 - Architecture d'application
- Dimensionner les composants
- Estimer la performance
- Itérer si nécessaire
- On reviendra en détails sur cette démarche dans la seconde partie du cours

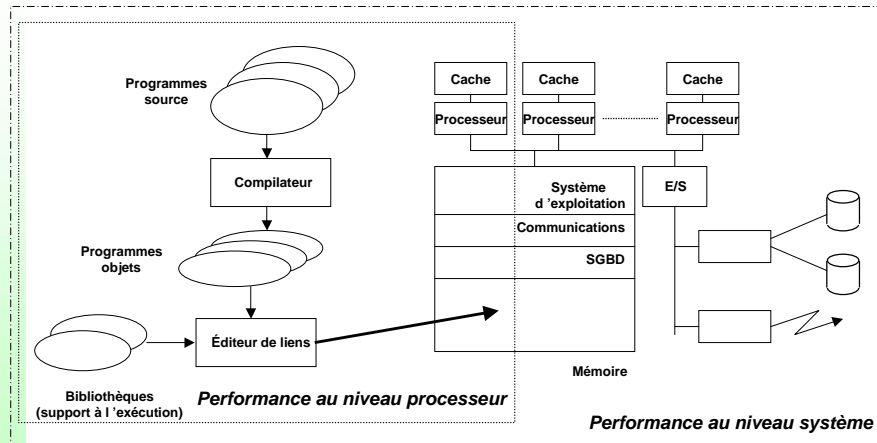
- Approches
 - Étalons de performance
 - Comparaison entre différents systèmes
 - Le cas d'utilisation représenté par l'étalon ne correspond pas toujours à l'utilisation envisagée
 - Intuition
 - Simple, peu coûteux, limitée
 - Fondée sur l'expérience
 - Manque de fiabilité
 - Mesures
 - Précises
 - Difficiles à mettre en place
 - Coûteuses
 - Interprétations difficiles
 - Le système doit exister
 - Modélisation
 - Ne nécessite pas l'existence du système
 - Retour sur investissement
 - Nécessite une expertise

Étalons de performance (Benchmarks)

Étalons de performance

- Les étalons de performance permettent de comparer les performances
- Les étalons sont composés de programmes reflétant des utilisations particulières des systèmes
- Différents types d'étalons existent. Ils correspondent à des cas d'utilisation différents
- Si la sincérité des étalons n'est en général pas contestable (rigueur de leur définition, règles d'évaluation, mécanismes de contrôle, ...), la question de fond concerne la représentativité des programmes composant l'étalon vis-à-vis des applications envisagées
- La liste des étalons décrits ci-après n'est pas exhaustive
- Les résultats des tests réalisés avec ces étalons varient en permanence, nous ne donnerons pas de chiffres ici. On se contentera d'indiquer les sites sur lesquels les résultats sont régulièrement publiés

■ Deux niveaux d'expression de la performance :



Étalons

Performances au niveau processeur

Performances "niveau processeur"

- **Éléments évalués par les étalons « niveau processeur »**
 - Performance intrinsèque du processeur
 - Performance de la hiérarchie de mémoire (caches et mémoire)
 - Compilateurs
- **Facteurs de la performance des processeurs**
 - Équation de base de la performance d'un processeur

$$\text{temps_tache} = \text{nombre_instructions_tache} \times \text{nombre_cycles_instruction} \times \text{temps_cycle}$$

- **Termes contribuant à cette équation:**

- **Nombre d'instructions à exécuter pour la tâche:**
 - algorithme, compilateur optimisant, répartition des fonctions entre programme d'application et système d'exploitation, adéquation de l'architecture au problème à traiter
- **Nombre de cycles par instruction:**
 - compilateur optimisant, définition de l'architecture, implémentation de l'architecture (pipeline, superscalaire, superpipeline,....)
- **Temps de cycle:**
 - définition de l'architecture, technologie, implémentation de l'architecture

Performances "niveau processeur" (2)

- **4 types de "benchmarks":**
 - Programmes synthétiques (e.g. Whetstone, Dhrystone) résultant de l'analyse du comportement de différents programmes
 - Programmes "noyaux" (e.g. Linpack) correspondant à l'isolation des parties de programmes les plus fréquemment utilisées
 - Programmes "jouets" (e.g. tours de Hanoi): programmes de très petite taille
 - Programmes réels (e.g. SPEC)
- **Note:** Exprimer la performance d'un processeur en nombre d'instructions exécutées par secondes (e.g. mips = millions d'instructions par seconde) n'a pas de signification en dehors de quelques cas très particuliers. Du fait de la diversité des répertoires d'instructions, l'expression de la performance d'un processeur en nombre de mips est comparable à l'expression de la vitesse des véhicules automobiles en nombre de tours de roues par unité de temps dans des conditions de route non définies sur un parcours non spécifié. Rappelons que l'on rencontre pratiquement, des diamètres de roues variant de 10" à 18" pour les automobiles.
- **Il convient donc d'exprimer la performance des processeurs en fonction de l'exécution de programmes réels et représentatifs de l'utilisation.**

Performances "niveau processeur" (3)

■ SPEC : Standard Performance Evaluation Coporation

- Groupement de constructeurs de systèmes et de microprocesseurs établissant des standards en matière de mesure de performance (pour systèmes UNIX). Publication périodique des résultats (site Web de SPEC). Adaptation continuelle des standards pour une meilleure représentativité.
- Les étalons SPEC les plus connus concernent la performance des systèmes pour les traitements en nombres entiers (CINTxx) ou bien en nombres flottants (CFPxx). Ces benchmarks (programmes en C et en Fortran) servent essentiellement à évaluer la performance des processeurs, de la hiérarchie de mémoire (c-à-d caractéristiques des caches et de la mémoire) ainsi que la capacité des compilateurs à engendrer un code optimisé. En d'autres termes, ces benchmarks ne servent pas à exprimer la performance des systèmes. SPEC a développé d'autres types de benchmarks pour exprimer la performance au niveau système.

■ Définitions et résultats disponibles sur

<http://www.spec.org>

Performances "niveau processeur" (4)

■ SPEC - Historique :

- **SPEC89** : premier étalon, sélection d'un ensemble de programmes "entiers" et "flottants" permettant d'exprimer la performance d'un système par rapport au VAX 11/780 (valant 1 en entier et en flottant).
- **SPEC92** : la sélection des programmes pour SPEC89 et la définition des conditions de mesure permettant aux acteurs de fausser les résultats, SPEC a défini un nouveau standard: sélection d'un ensemble de programmes réels représentatifs des traitements portant soit sur des nombres entiers (6 programmes), soit sur des nombres flottants (14 programmes). Le système de référence étant toujours le VAX 11/780.
- **SPEC95** : les fournisseurs de systèmes ayant "appris" à développer des optimisations, au niveau des compilateurs, permettant de fausser les résultats, SPEC a défini un nouvel ensemble de programmes et de nouvelles règles d'évaluation. La machine de référence est maintenant un SPARCstation 10/40 (SuperSPARC @40 Mhz, pas de cache de second niveau, 64 Mo de mémoire) valant 1 en entier et en flottant.
- **SPEC2000** : la suite de programmes composant SPEC95 a été modifiée et complétée. La machine de référence est un Sun Ultra5_10 (UltraSPARC@300 Mhz, 256 Mo de mémoire) et valant 100 en entier et en flottant.

■ Ensemble de programmes de référence:

- CINT2000 : 12 applications (11 C et une C++)
- CFP2000 : 14 applications (6 Fortran77, 4 Fortran90, 4 C)

■ Procédure d'évaluation (SPEC2000):

- mesure du temps d'exécution pour chaque programme sur le système à mesurer (moyenne sur 3 exécutions)
- calcul du SPEC ratio pour chaque programme (avec deux types d'options de compilation, voir ci-après):
 - $\text{SPEC_ratio} = (\text{temps_d'exécution_Sun Ultra5_10}) / (\text{temps_d'exécution_système}) \times 100$
- calcul des valeurs:
 - SPECint2000 : moyenne géométrique des ratios obtenus (programmes "entiers") avec toute liberté en ce qui concerne les possibilités d'optimisation au niveau du compilateur
 - SPECint_base2000 : moyenne géométrique des ratios obtenus (programmes "entiers") avec des options de compilation identiques
 - SPECfp2000 : moyenne géométrique des ratios obtenus (programmes "flottants") avec toute liberté en ce qui concerne les possibilités d'optimisation au niveau du compilateur
 - SPECfp_base2000 : moyenne géométrique des ratios obtenus (programmes "flottants") avec des options de compilation identiques

■ Pas de règle simple pour faire la correspondance entre les valeurs SPEC95 et SPEC2000 (programmes différents).

Note : un Sun Ultra10 Model 333 (UltraSPARCIIi @333 Mhz) vaut 14,1 SPECint95/18,3 SPECfp95 et 133 SPECint200/126 SPECfp2000 .

Étalons

Performances au niveau système

■ Deux ensembles d'étalons pour les serveurs

□ SPEC

- SPECsfs97 pour les applications de type serveur de fichiers
- SPECjvm98 pour le support de la machine virtuelle Java
- SPECweb99 pour les applications de type serveur Web
- SPECjbb2000 pour les applications Java « serveur »
- SPECipc96 pour les applications de calcul intensif
- SPECcomp2001 pour les applications de calcul intensif exécutées sur des systèmes SMP
- SPECmail2001 pour le support du courrier électronique

□ TPC - Transaction Processing Council

- TPC-C pour les applications de type OLTP
- TPC-W pour les applications transactionnelles dans le contexte du Web
- TPC-H (et TPC-R) pour les applications de type décisionnel

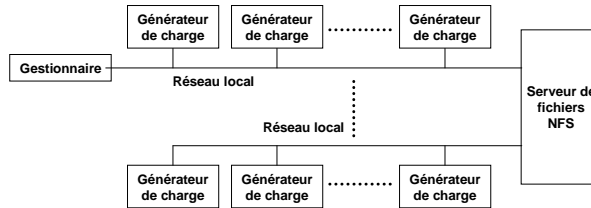
■ *Note : On ne détaillera pas ici les étalons HPC96, SPECcomp2001 et SPECmail2001.*

■ *Note : Il existe aussi des étalons pour les stations de travail (PC) tels que : BAPCo SYSmark 2000, BAPCo/MadOnion.com WebMark 2001, Ziff-Davis*

- SPECjvm98 permet de comparer l'efficacité des implémentations de la jvm (Java Virtual Machine) pour l'exécution d'applets ou de servlets
- Étalon composé de 8 tests dont 5 sont des applications réelles ou des dérivés d'applications réelles
- SPECjvm98 consiste à mesurer le temps nécessaire au chargement du programme, à la vérification des classes utilisées par le programme chargé, à la compilation " juste à temps " (JIT, pour just-in-time) et à l'exécution du test
- L'étalon est exécuté plusieurs fois, ce qui permet d'enregistrer deux types de mesures : le plus mauvais temps d'exécution et le meilleur. Le résultat final est une moyenne géométrique. La performance des systèmes est exprimée par rapport à une machine de référence : une station de milieu de gamme à base de Power PC avec un processeur 604 cadencé à 133 MHz.
- La performance des implémentations de la machine virtuelle Java étant dépendante de la capacité mémoire de la station, trois classes de mémoire sont considérées : de 0 à 48 Mo (32 Mo étant considéré comme un minimum), de 48 à 256 Mo et au-delà de 256 Mo.

■ SPECsfs97 permet de comparer l'efficacité des serveurs de fichier NFS

- Profil déterminé par l'observation de la charge des serveurs NFS
- Configuration de test :



- Temps de réponse du serveur <40 ms
- Résultats :
 - le nombre moyen d'opérations par seconde
 - le temps de réponse moyen par opération.

■ Évolution de SPECweb96, il permet de caractériser la performance des serveurs Web

■ Profil d'utilisation dérivé de l'observation de l'activité des sites Web

- 35 % des requêtes concernent des fichiers de moins de 1 ko.
- 50 % des requêtes concernent des fichiers dont la taille est comprise entre 1 et 10 Ko.
- 14 % des requêtes concernent des fichiers dont la taille est comprise entre 10 et 100 Ko.
- 1 % des requêtes concernent des fichiers dont la taille est comprise entre 100 Ko et 1 Mo.

■ Nombre de répertoires de fichiers proportionnel à la puissance du serveur

■ Profil des actions

- GET statique 70%
- GET dynamique standard 12,45%
- GET dynamique standard (CGI) 0,15%
- GET dynamique spécifique 12,6%
- POST dynamique 4,8%

■ Débit du serveur compris entre 40 et 50 Ko/sec

■ Résultat : le nombre maximum de connexions que peut supporter un serveur

- SPECjbb2000 (Java Business Benchmark) permet de comparer les performances des serveurs pour l'exécution des parties « serveurs » des applications Java
- Modèle d'une architecture d'application Client/Serveur à 3 niveaux (inspirée de TPC-C) dont seul le niveau « application » est mesuré. La partie client et la partie base de données sont simulées de façon tout à fait élémentaire : la partie client engendre des requêtes et la partie base de données accède à des données en mémoire structurées sous forme d'arbre.
- Exerce le support de Java (JVM ou JIT), le « ramasse miettes », le support des threads ainsi que certains aspects du système d'exploitation. Permet de caractériser la performance des processeurs, la hiérarchie de mémoire et la scalabilité multiprocesseur symétrique
- Résultat : une caractérisation du débit moyen du serveur exprimé en nombre d'exécutions de l'application par unité de temps; ce serveur étant soumis à des flux de requêtes le chargeant bien au delà de son débit maximal
- Procédure :
 - On détermine le débit maximal (nombre d'exécutions de l'application Java) dans une unité de temps donnée (2 minutes) en faisant varier le nombre de générateurs de requêtes
 - On mesure ensuite les débits en augmentant à chaque fois de 1 le nombre de générateurs de requêtes depuis ce point jusqu'au double du nombre de générateurs de requêtes (pour lequel le maximum de débit a été obtenu)
 - Le résultat est la moyenne des débits observés entre le débit maximal et tous ceux qui ont été relevés jusqu'au double de générateurs

- La création du TPC (Transaction Processing Council) fait suite aux premiers efforts de standardisation d'un étalon transactionnel dans la seconde moitié des années 80 (étalon TP1)
- TPC = Groupement de constructeurs de systèmes, de sociétés de logiciel (SGBD)
- Plusieurs types d'étalons
 - TPC-A (1989): transaction simple et unique de mise à jour d'un compte bancaire (obsolète);
 - TPC-B (1990): partie "base de données" de TPC-A (obsolète);
 - TPC-C (1992): transactions complexes et multiples;
 - TPC-D (1995): aide à la décision (obsolète);
 - TPC-H (1999) : aide à la décision
 - TPC-R (1999) : aide à la décision
 - TPC-W (2000) : transactionnel sur le Web (commerce électronique, B-to-B,...)
 - TPC-E (pas approuvé): environnement serveur d'entreprise
- Métriques
 - Performance
 - Prix/performance
- Définitions et résultats disponibles sur <http://www.tpc.org>

Performances « niveau système » - TPC-C

■ TPC-C (1992)

- Applications transactionnelles de gestion ;
- 5 types de transactions et 9 tables ;
- Exerce l'ensemble des dimensions du système (sous-système central, base de données, télécommunications) ;
- Spécification générique à implémenter sur chaque système ;
- Respect des propriétés ACID (Atomicité, Consistance, Isolation et Durabilité) ;
- Deux métriques :
 - Performance : tpm-C (TPC-C transactions par minute)
 - Prix/Performance : \$/tpm-C (coût de possession sur 3 ans rapporté à la transaction par minute)
- Dimensionnement de la base de données et des communications dépendant de la performance du système ("scalabilité") e.g. pour chaque tpm : 1 terminal et 8 Mo mesurés (72 Mo facturés) ;
- Contraintes de temps de réponse (à 90%) en fonction du type de transaction.

Performances « niveau système » - TPC-H & R

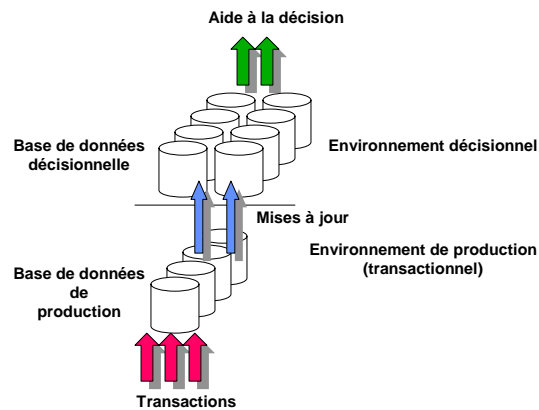
■ TPC-H (1999) Applications décisionnelles

- Base de données disponible 24x24h et 7jx7j
- Mise à jour, en permanence, de la base décisionnelle à partir de la base de production sans interruption du traitement décisionnel
- Plusieurs tailles de base de données possibles (typiquement 100 Go, 300 Go, 1 To)
- Ensemble de requêtes que l'on doit considérer comme non déterminé *a priori* (H pour *had oc*) ce qui a pour conséquence que les optimisations au niveau du SGBD ne sont pas possibles
- Métriques (exprimées en relation avec la taille de la base de données)
 - QphD@<taille base> : Composite Query-per-Hour mesure du débit du système
 - \$/QphD@<taille base> : coût de possession sur 5 ans rapporté à la performance du système

■ TPC-R (1999) Applications décisionnelles

- Identique à TPC-H sauf que les requêtes sont considérées comme connues *a priori* (c-à-d des optimisations au niveau du SGBD sont possibles)
- Métriques
 - QphR et \$/QphR

■ Modèle applicatif TPC-H



Page 27

© RJ Cheavance

Performances « niveau système » - TPC-W

- Applications commerciales sur le Web
- 3 types de profils :
 - Recherche d'information et achat (B-to-C)
 - Recherche d'information
 - Transactionnel sur Web (achats de type B-to-B)
- 5 types de transactions
 - Recherche d'information (Browse), Sélection d'articles (Shopping Cart), Achat, Enregistrement d'information sur le client, Recherche d'information sur un article
- Respect des propriétés ACID, sécurisation des transactions
- Contraintes de temps de réponse pour chacune des transactions
- Taille de la base de données : à choisir entre différents facteurs d'échelle représentant le nombre d'articles du « magasin » (1000, 10 000, 100 000, 1 000 000 et 10 000 000) et le nombre de navigateurs supportés (nombre de clients). Les données concernant les achats doivent être conservées « en ligne » durant 180 jours
- Résultats (exprimés en fonction de la taille du « magasin »)
 - WIPS (Web Interactions Per Second). Note : le temps « réseau » n'est pas mesuré
 - Prix/Performance : coût de possession sur 3 ans/WIPS
 - Autres résultats :
 - Recherche d'informations WIPSB (Web Interactions Per Second-Browsing)
 - Transactionnel sur Web WIPSO (Web Interactions Per Second-OLTP)

Page 28

© RJ Cheavance

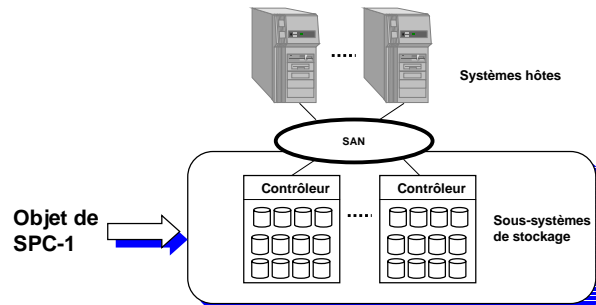
Étalons

Entrées-sorties

Étalons « Entrées-Sorties »

- **Tant dans le domaine de l'architecture que des étalons de performance, les entrées-sorties ont été le parent pauvre cf. le nombre d'articles et le nombre de thèses sur le sujet**
- **Il faut toutefois noter que certains étalons (e.g. TPC-C ou TPC-H) exercent les entrées-sorties**
- **L'essor récent du SAN a conduit l'industrie à créer le Storage Performance Council (SPC) qui a développé un étalon spécifique:**
 - **Le SPC-1 (SPC Benchmark-1)**
 - **Voir <http://www.StoragePerformance.org>**

■ Environnement système de SPC-1



- SPC-1 composé de ensemble de 8 flux représentant chacun le profil d'entrées-sorties d'une application. Un flux est appelé une « execution instance » et l'ensemble des 8 flux est appelé un BSU (Business Scaling Unit). Pour simuler la montée en charge, on augmente le nombre de BSU

■ Composition des 8 flux:

- 5 flux aléatoires en lecture et en écriture
- 2 flux séquentiels en lecture
- 1 flux séquentiel en écriture.

■ Composition des flux:

	Lectures/s	Ecritures/s	Total (opérations/s)
Aléatoire	14,5	16,2	30,7
Séquentiel	5,3	14	19,3
Total (opérations/s)	19,8	30,2	50

■ Résultats:

- Le temps de réponse à faible charge appelé SPC-1 LRT (Least Response Time)
- Le débit maximal que peut supporter le sous-système de stockage (SPC-1 IOPS)

■ Procédure de mesure:

- Augmenter la charge pour déterminer le point de saturation -> SPC-1 IOPS
- Mesurer SPC-1 LRT à 10% de cette charge

■ Vérification complémentaires:

- Persistance du stockage
- Caractère reproductible des résultats

- **SPC prépare un autre étalon ayant pour objectif de caractériser les applications suivantes:**
 - **Serveur Web**
 - **Charges de travail séquentielles**
 - **Opérations de sauvegarde et de restauration**
 - **Flux de données synchrones (Data Streaming tel que vidéo)**
 - **Serveur NAS**
- **SPC a l'intention de développer d'autres étalons par la suite**

- **Assez bonne couverture du domaine**
- **Solidité des processus de définition et d'auto-contrôle**
- **Représentativité (dans certaines limites)**
- **Signe de bonne santé financière et de confiance dans les performances de leurs systèmes pour les sociétés qui publient des résultats car l'investissement est très important**
- **Incitation à l'amélioration des performances**
- **Représentativité aux limites (e.g. \$/tpmC les plus bas, tpmC les plus élevés)**
- **Le coût de réalisation des mesures peut dissuader certains acteurs**

Prise en compte des aspects performance dans les projets

Prise en compte des aspects performance dans les projets

- **L'aspect performance (et aussi disponibilité) est, assez souvent, le parent pauvre dans les projets.**
- **Difficultés classiques:**
 - le traitement tardif et "à chaud" de ces problèmes conduit à des résultats médiocres voire même désastreux : impact négatif sur coûts et délais, déstructuration du système,...
 - difficulté à prévoir les conséquences de changements sur le comportement du système (e.g. autres missions pour le système, changements de technologies,...)
 - difficulté à analyser les problèmes rencontrés dans ces domaines
- **Approches**
 - **Intuition**
 - Simple, peu coûteux, limitée
 - Fondée sur l'expérience
 - Manque de fiabilité
 - **Mesures**
 - Précises
 - Difficiles à mettre en place
 - Coûteuses
 - Interprétations difficiles
 - Le système doit exister
 - **Modélisation**

■ Compromis entre intuition et mesures

- Plus fiable que l'intuition
 - Reflète la dynamique du système
 - Permet de détecter des effets secondaires
- Plus souple que l'expérimentation
 - Modification aisée des configurations
 - Analyse de sensibilité aux différents paramètres
- Ne nécessite pas l'existence du système

■ Avantages

- Formalisme rigoureux et fiabilité des résultats
- Implique la compréhension de l'architecture du système et de sa dynamique

■ Inconvénients

- Démarche peu répandue
- Nécessité de comprendre l'architecture du système et sa dynamique
- Difficulté d'obtention des paramètres

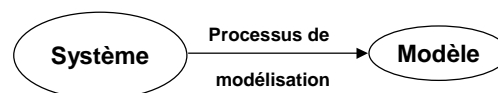
■ Deux approches non-exclusives:

- "dégrossissage": on procède à des calculs simples;
- "affinage": on a recours aux techniques de modélisation.

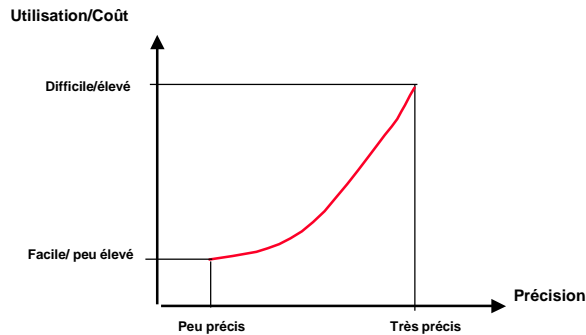
■ Modélisation et architecture de système

- la modélisation est une activité de base de l'architecte de système : c'est le moyen de valider -qualitativement et/ou quantitativement- l'architecture avant les phases de développement
- l'architecte étudie le comportement du système sur le modèle
- la modélisation peut s'appliquer à différents aspects : performance, disponibilité, coût et temps de développement.

■ Concept de modèle



■ Recherche d'un compromis entre la précision et le coût d'un modèle



■ Caractéristiques d'un « bon » modèle :

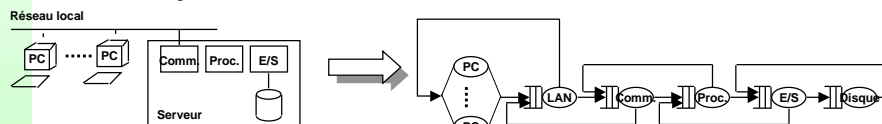
- Utilité
- Homogénéité
- Simplicité
- Compréhensibilité

■ Trois techniques:

- Analyse opérationnelle
- Réseaux de Pétri stochastiques
- Réseaux de files d'attente

- On dispose d'outils informatiques pour les deux dernières techniques. L'analyse opérationnelle constitue une première approche, elle permet d'obtenir des estimations de performance au moyen de calculs simples.

■ Concept de réseau de files d'attente

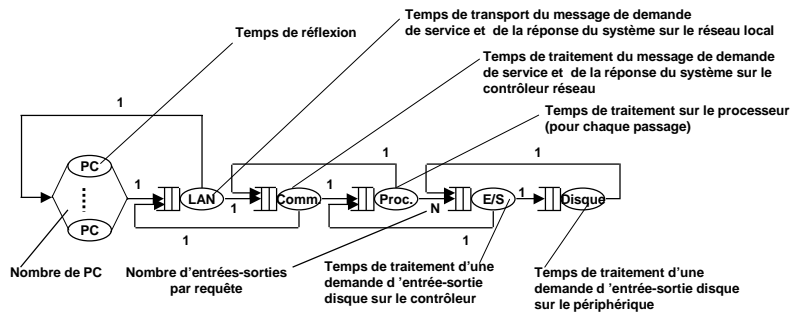


□ Notions de :

- Client et de classe de clients
- Station (= file d'attente + serveur) et de service
- Transition

Réseaux de files d'attente

■ Paramètres à spécifier (exemple)



■ Méthodes de résolution: modèles mathématiques à base de files d'attente (solutions exactes), chaînes de Markov, simulation à événements discrets,

■ Exemple d'outil de résolution : MODLINE de Simulog (<http://www.simulog.fr>)

■ Résultats :

- temps de réponse moyen
- charge des différentes stations
-

Analyse opérationnelle

■ Une première approche de la modélisation des systèmes

■ Introduite par Buzen (1976) et améliorée par P.J. Denning et J. P. Buzen (1978)

■ Le terme "opérationnelle" signifie que les valeurs utilisées dans cette analyse sont directement mesurables :

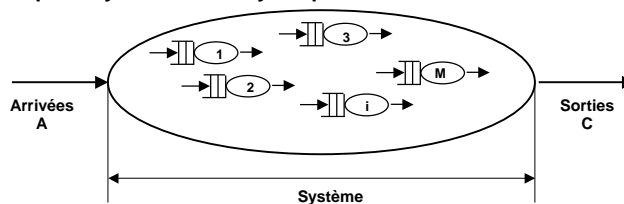
- pendant une expérience unique
- pendant un intervalle de temps fini

■ L'analyse opérationnelle établit des relations entre ces valeurs

■ L'un des intérêts de l'analyse opérationnelle est qu'elle permet de vérifier l'homogénéité de mesures faites sur un système

■ Les valeurs utilisées dans l'analyse opérationnelle sont produites par des outils tels que Performance Monitor (Windows) ou vmstat et sar (UNIX)

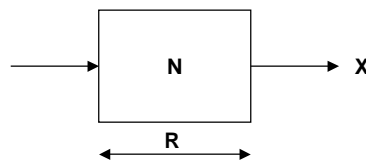
■ Concept de système en analyse opérationnelle :



- L'analyse opérationnelle suppose que les hypothèses suivantes soient vérifiées :
 - Équilibre des flux : le nombre total des arrivées doit être égal au nombre total des sorties pendant la période d'observation.
 - Évolution pas à pas : l'état du système évolue uniquement sous l'influence d'un événement à la fois, arrivée ou départ.
 - Homogénéité des ressources : le flux de sortie d'une ressource ne dépend que du taux d'arrivée des clients, si elle n'est pas saturée. À saturation, le flux de sortie dépend du temps de service de la ressource saturée.
 - Routage homogène : le taux de visite d'une station ne dépend pas de l'état du système.

■ Types de modèles

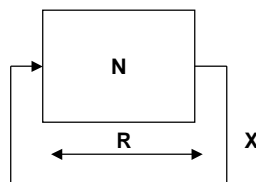
- Modèle ouvert :



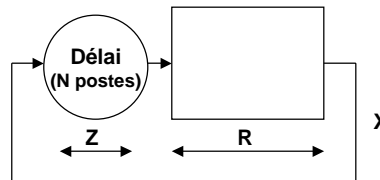
Notations :

N : nombre de clients dans le système
 X : débit du système (clients/s)
 R : Temps de réponse

- Modèle fermé :



a) - Modèle de base



b) - Modèle avec station de type délai

Analyse opérationnelle (4)

- Quantités opérationnelles (observées au niveau de la ressource i pendant la période T)
 - Taux d'arrivée $\lambda_i = \text{nombre d'arrivées} / \text{temps} = A_i / T$
 - Débit $X_i = \text{nombre de terminaisons} / \text{temps} = C_i / T$
 $X = \text{débit global du système} = C_0 / T$
 - Utilisation $U_i = \text{temps d'occupation} / \text{temps} = B_i / T$
 - Temps de service moyen $S_i = \text{temps de service total} / \text{nombre servi} = B_i / C_i$
 - Demande de service $D_i = S_i V_i$ (demande par utilisateur, V_i nombre de visites à la ressource i pour une interaction)
 - V_i se définit comme le rapport entre le nombre de terminaisons de la ressource i et le nombre total de terminaisons du système, noté C_0 (donc $V_i = C_i / C_0$). C_0 est relatif au système complet : c'est le débit global du système.
 - Demande totale de service $D = \sum_{i=1}^M D_i$
- Loi d'utilisation (Utilization Law)
 - $U_i = X_i S_i = X D_i$
- Loi de conservation (Forced Flow Law)
 - $X_i = X V_i$

Page 45

© RJ Chevanec

Analyse opérationnelle (5)

- Loi de Little
 - $Q_i = X_i R_i$ avec $Q_i = \text{nombre de clients dans la station } i$ et $R_i = \text{temps de réponse par visite à la station } i$

Dans le cas d'un modèle fermé de « base » ou d'un modèle ouvert :

 - $N = X R$
- Loi générale du temps de réponse (General Response Time Law)
 - $R = \sum_{i=1}^M R_i V_i$ R est le temps de réponse global du système
- Loi du temps de réponse interactif (Interactive Response Time Law). S'applique aux modèles fermés
 - $R = (N/X) - Z$ avec $N = \text{nombre d'utilisateurs}$ et $Z = \text{temps de réflexion (au niveau de la station de type délai)}$

Page 46

© RJ Chevanec

Analyse opérationnelle (6)

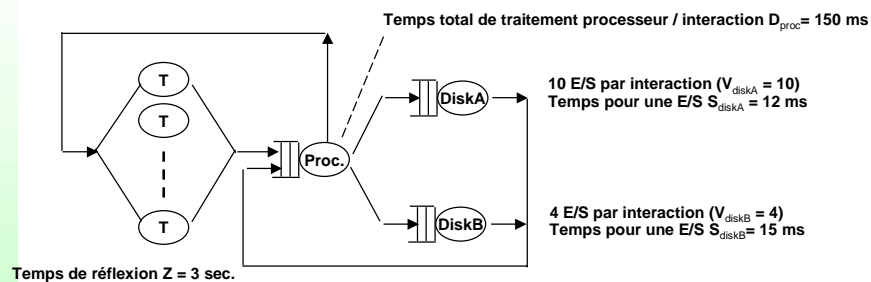
- **Analyse des goulets d'étranglement (Bottleneck Analysis)**
 - L'analyse des goulets d'étranglement nécessite de déterminer le serveur le plus chargé dans le système (i.e. la plus forte demande de service D_i).
Soit, sur l'ensemble des serveurs, D_{\max} la demande la plus forte (ne pas considérer les stations de type délai telles que les postes de travail). *Rappel* $D = \sum_{i=1}^M D_i$
 - Les bornes sont :
 - Pour un modèle ouvert :
 - Débit : $X \leq 1 / D_{\max}$
 - Pour un modèle fermé (avec station de type délai) :
 - Débit : $X(N) \leq \text{Min} \{ 1 / D_{\max}, N / (D + Z) \}$
 - Temps de réponse : $R(N) \geq \text{Max} \{ D, N D_{\max} - Z \}$
 - Nombre maximal de postes avant que la saturation apparaisse (coude de saturation) dans un modèle fermé :
 - $N^* = (D + Z) / D_{\max}$

Page 47

© RJ Cheavance

Analyse opérationnelle (7)

■ Exemple d'un système interactif

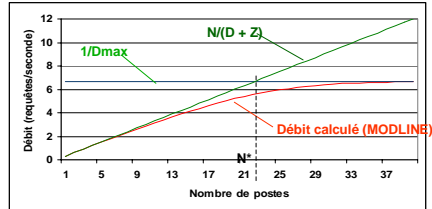
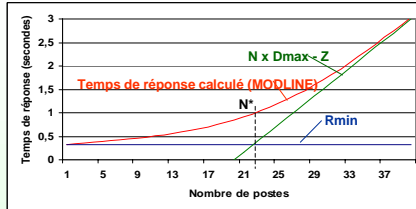


Page 48

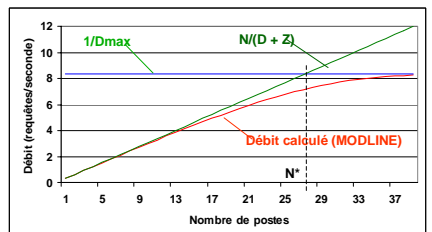
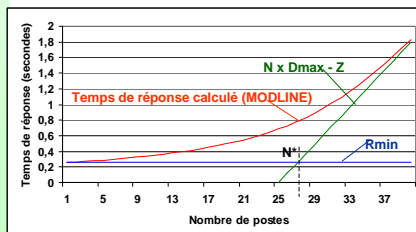
© RJ Cheavance

Analyse opérationnelle (8)

■ Comportement du système



■ Comportement du système avec un processeur 2 fois plus rapide



Page 49

© RJ Cheavance

Méthodologie de modélisation

■ De façon synthétique, deux cas sont à considérer :

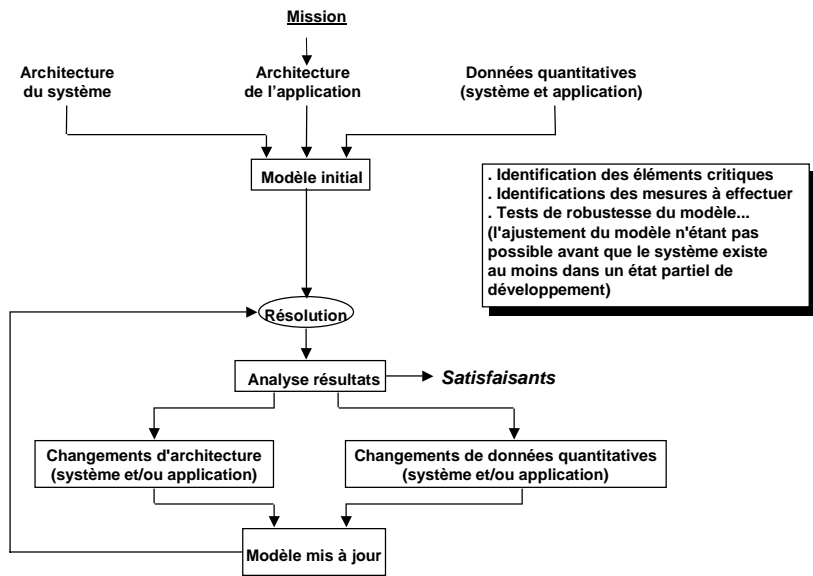
- Pour un nouveau système (cas idéal : on part d'une "feuille blanche")
 - identifier la mission que le système doit remplir
 - développer un modèle du système;
 - étudier le comportement et les propriétés du système sur le modèle
 - valider les choix d'architecture et d'implémentation
 - aussi tôt que possible, valider les différentes hypothèses et ajuster le modèle en fonction des observations faites sur le système et ses composants
- Pour un système existant (un cas commun : tout ou partie du système existe déjà, le système ré-utilise des composants existants,...):
 - identifier la mission que le système doit remplir
 - dériver un modèle à partir de l'existant
 - ajuster le modèle vis à vis du système
 - étudier et valider le comportement du système sur le modèle (et non sur le système)

■ Dans tous les cas, à la fin du processus, le modèle représente le système résultant. L'impact de modifications (e.g. changement de mission du système, nouvelles applications, nouveau dimensionnement, nouveau composants matériel ou logiciel, changement d'architecture...) peut être étudié sur le modèle et donc être validé avant sa mise en œuvre ou sa réalisation.

Page 50

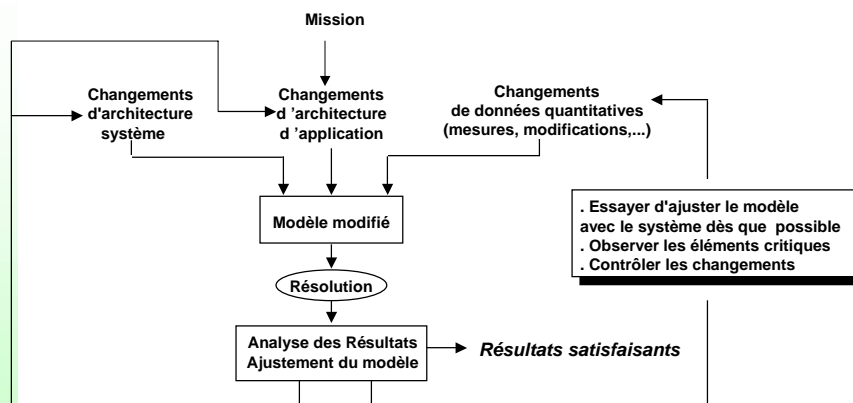
© RJ Cheavance

« Cas idéal » - Phase de conception



. Identification des éléments critiques
 . Identifications des mesures à effectuer
 . Tests de robustesse du modèle...
 (l'ajustement du modèle n'étant pas possible avant que le système existe au moins dans un état partiel de développement)

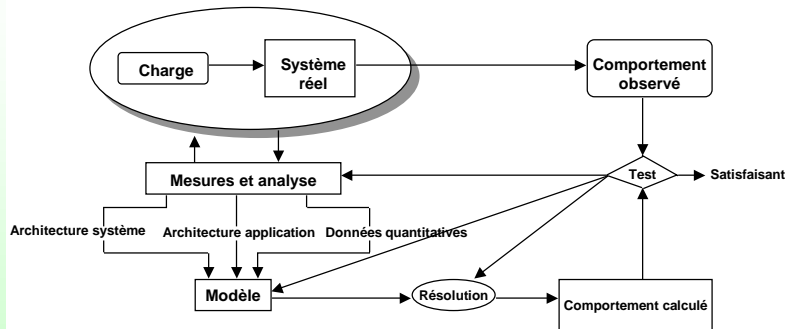
« Cas idéal » - Phase de développement



. Essayer d'ajuster le modèle avec le système dès que possible
 . Observer les éléments critiques
 . Contrôler les changements

Cas d'un système existant

- L'objectif est d'obtenir un modèle ajusté pour un système existant dont on ignore tout ou partie de son architecture et de ses données quantitatives.



- Ne pas espérer obtenir de mesures significatives avant d'avoir appréhendé l'architecture du système et sa dynamique. Cette compréhension peut se faire grâce aux moyens suivants:

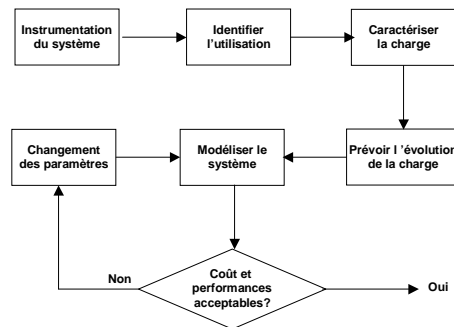
- interviews;
- analyse de code;
- traces,

Stratégie de modélisation

- Définir une stratégie de modélisation:
 - Isoler les parties correspondant à différents types de modélisation (e.g. simulation, analytique);
 - Rechercher une hiérarchie de modèle;
 - Définir des modèles;
 - Identifier les paramètres nécessaires;
 - Obtenir les paramètres (mesures, estimations,...) et vérifier leur vraisemblance;
 - Évaluation et ajustement des différents modèles en allant du plus élémentaire au plus complet;
 - Évaluation et ajustement du modèle global.
- Remarques générales :
 - Les activités de mesure et (surtout) de modélisation sont une partie intégrante de l'activité d'architecture système.
 - Les modifications d'architecture système, en particulier pour répondre à des problèmes de performance, conduisent en général à des résultats médiocres (en terme de performance) ou catastrophiques (en terme de structuration du système) si elles sont décidées sans que l'on ait une bonne compréhension de l'architecture du système et de ses paramètres essentiels.
 - Les activités de mesure et de modélisation étant étroitement reliées à l'architecture système, les outils associés (mesures et modélisation) doivent être assimilés et, idéalement, utilisables directement par les architectes système.

Prévision de la charge

- **Objectif : aménager la configuration du système en fonction de l'évolution de sa charge (implique de caractériser l'évolution de la charge)**
- **Deux approches possibles :**
 - **Modèle de comportement du système**
 - **Mesures et anticipation**



Page 55

© RJ Chevance

Ouvrages et sites de référence

- **Tim Browning « Capacity Planning for Computer Systems »**
AP Professional 1995
- **René J. Chevance « Serveurs multiprocesseurs, clusters et architectures parallèles »**
Eyrolles 2000
- **René J. Chevance « Performances des systèmes et techniques d'estimation »**
Les Techniques de l'Ingénieur 2001
- **René J. Chevance « Méthodologie en matière des performances des systèmes »**
Les Techniques de l'Ingénieur 2001
- **Raj Jain « The Art of Computer Systems Performance Analysis »**
John Wiley & Sons 1991
- **Georges Kéryvel « Apport des méthodologies de modélisation de performance à la conception des systèmes »**
CNAM Versailles 1999
- **Daniel A. Mencias, Virgilio A. Almeida « Capacity Planning for Web Services »**
Prentice Hall 2002
- **Sites Web :**
 - <http://www.spec.org> (System Performance Evaluation Corporation - SPEC)
 - <http://www.tpc.org> (Transaction Processing Council - TPC)
 - <http://www.StoragePerformance.org> (Storage Performance Council - SPC)
 - Outil de modélisation <http://www.simulog.fr> (Simulog)
 - Pour les performances des stations de travail :
 - <http://www.bapco.com>, <http://www.madonion.com>, <http://www.zdnet.com>

Page 56

© RJ Chevance