Chargement d'un enregistrement (II)

Par
Tim TONOOKA, Technicien Support Clients 4DUS
Note technique 4D-200012-32-FR
Version 1

Date 1 Décembre 2000

Résumé

Cette note technique traite de l'occupation mémoire et des différents réglages possibles (Mémoire Cache/Mémoire moteur, 4D monoposte/4D serveur, etc.)

4D Notes techniques

Introduction

Cette note technique est la seconde partie traitant du chargement des enregistrements. Avant de lire cette note, nous vous conseillons de consulter la première partie (note technique 2000-21 Chargement d'un enregistrement (1ère partie))

I- Comment trouver la taille du cache et de la mémoire principale (User heap) ?

Lors de l'ouverture d'une base, la quantité de mémoire qui sera allouée à 4D par le système ne dépendra pas uniquement du paramétrage réservé à cet effet (soit à l'aide de Customizer plus, soit dans les propriétés de la base), dans 4D, mais aussi de la quantité réelle de mémoire que possède la machine. Si les paramétrages ou la quantité de mémoire que possède la machine sont insuffisants, l'application ne se lancera pas.

1- L'utilisation du debogger ou de l'explorateur d'exécution permet d'obtenir les informations suivantes.

a- Détails sur le cache.

Dans une base en cours d'exécution, il est possible de voir la taille allouée au cache :

- Par l'explorateur d'exécution, dans l'onglet "évaluation" rubrique "statistiques du cache".
- Par le debugger dans la fenêtre dans le coin supérieur gauche rubrique "statistiques du cache".

Nous pouvons voir à cet endoit, par exemple, l'information suivante : 163 K octets / 2048 Ko (8%), 59 handles

Cela signifie que la taille réservée au cache est de 2048 Ko dont 163 K octets sont actuellement occupés par 59 objets.

(Voir l'image au paragraphe d)

b- Détails sur la mémoire disponible.

La mémoire disponible correspond à la mémoire principale :

- Par l'explorateur d'exécution, dans l'onglet "évaluation" rubrique "informations", item "Mémoire disponible".
- Par le debugger dans la fenêtre, dans le coin supérieur gauche, rubrique "informations", item "Mémoire disponible".

(Voir l'image du paragraphe d)

c- Détails sur l'utilisation de la mémoire.

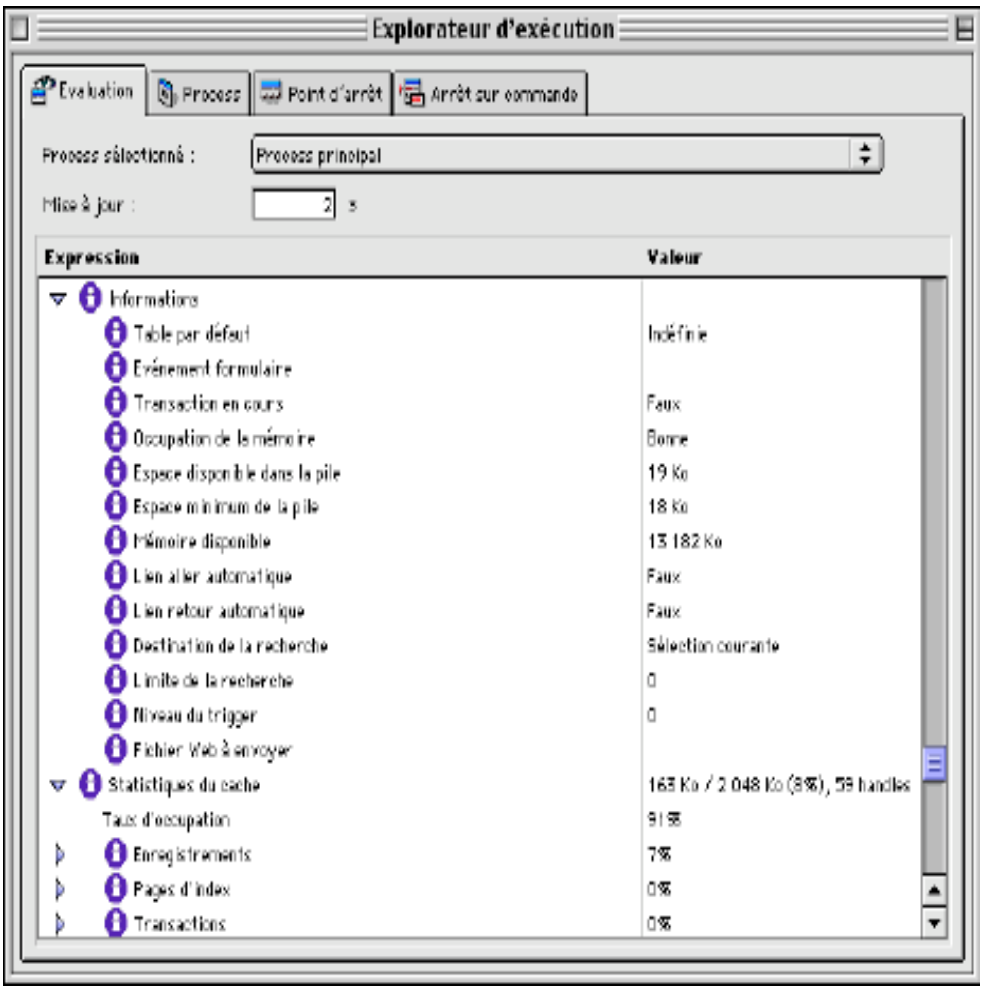
- Par l'explorateur d'exécution, dans l'onglet "évaluation", rubrique "informations", item "Occupation de la mémoire"

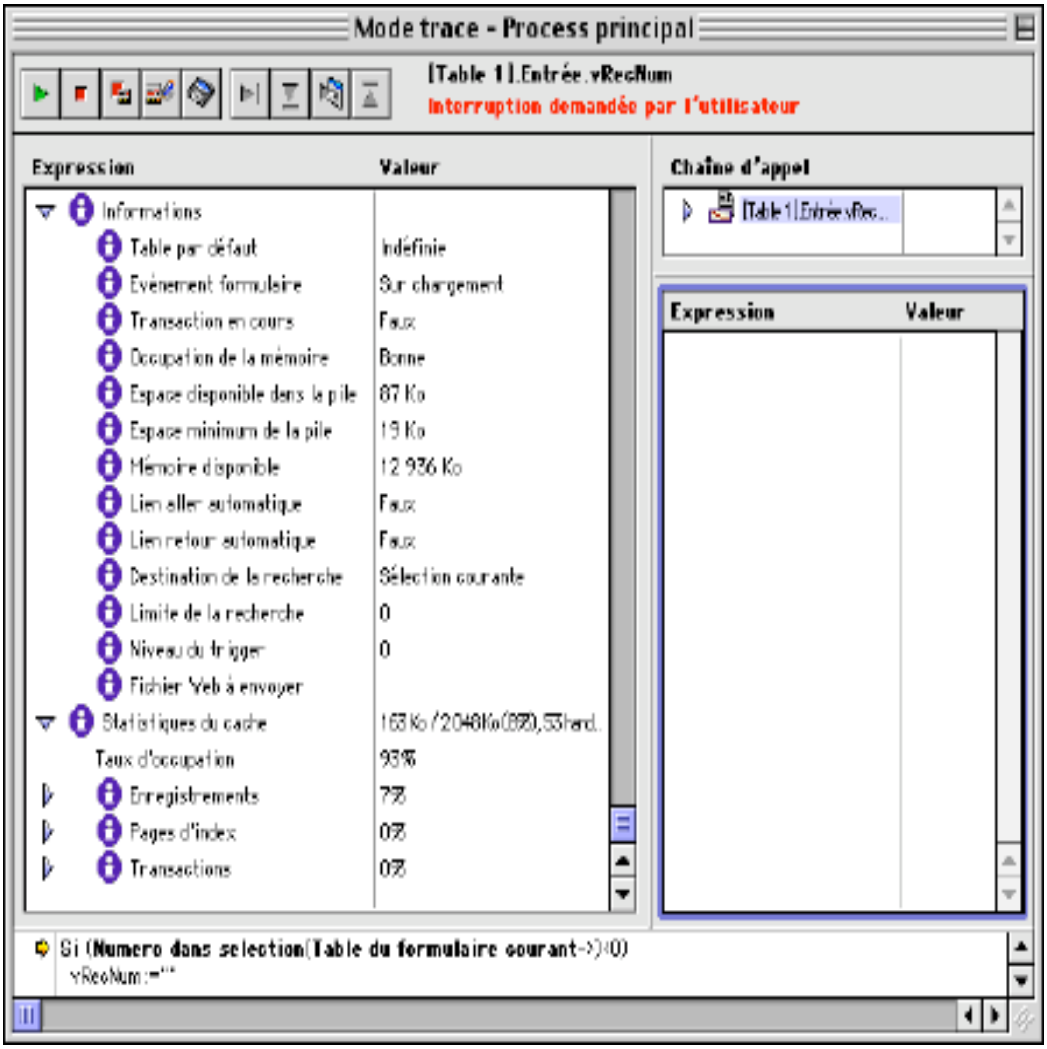
- Par le debugger dans la fenêtre dans le coin supérieur gauche, rubrique " informations ", item "Occupation de la mémoire".

(Voir l'image au paragraphe d)

L'ccupation de la mémoire vous indique si la mémoire est correctement occupée. Cette valeur traduite par "bonne" ou "mauvaise" est basée sur le nombre d'appels par seconde qu'effectue 4D vers les routines systèmes, qui s'occupent de purger la mémoire.
Si la valeur "mauvaise" apparaît cela signifie que 4D essaye continuellement de purger sa mémoire. Il faut en déduire que l'espace mémoire alloué à 4D est probablement insuffisant.

d) Présentation du mode trace et de l'explorateur d'exécution.





Pour des raisons d'encombrement, les images correspondant à la plate-forme Windows, étant identiques à la plate-forme Mac OS, ne sont pas présentées.
Cependant, les exceptions seront présentées sur les deux plates-formes.

2- Obtenir les mêmes informations par la programmation.

a- Détails sur le cache.

Pour obtenir la taille du cache dans une base de données, vous pouvez utiliser la commande "Lire paramètres base" en indiquant la valeur 9 pour le sélecteur.
Cette commande, ainsi paramétrée, retourne la taille du cache mesurée en octets.

b- Détails sur la mémoire disponible.

Pour connaître la quantité de mémoire principale disponible, il faut utiliser la commande "AP AVAILABLE MEMORY". La valeur du troisième paramètre correspond à cette quantité.

3- Mise en garde.

a- L'utilisation du debogger ou de l'explorateur d'exécution.

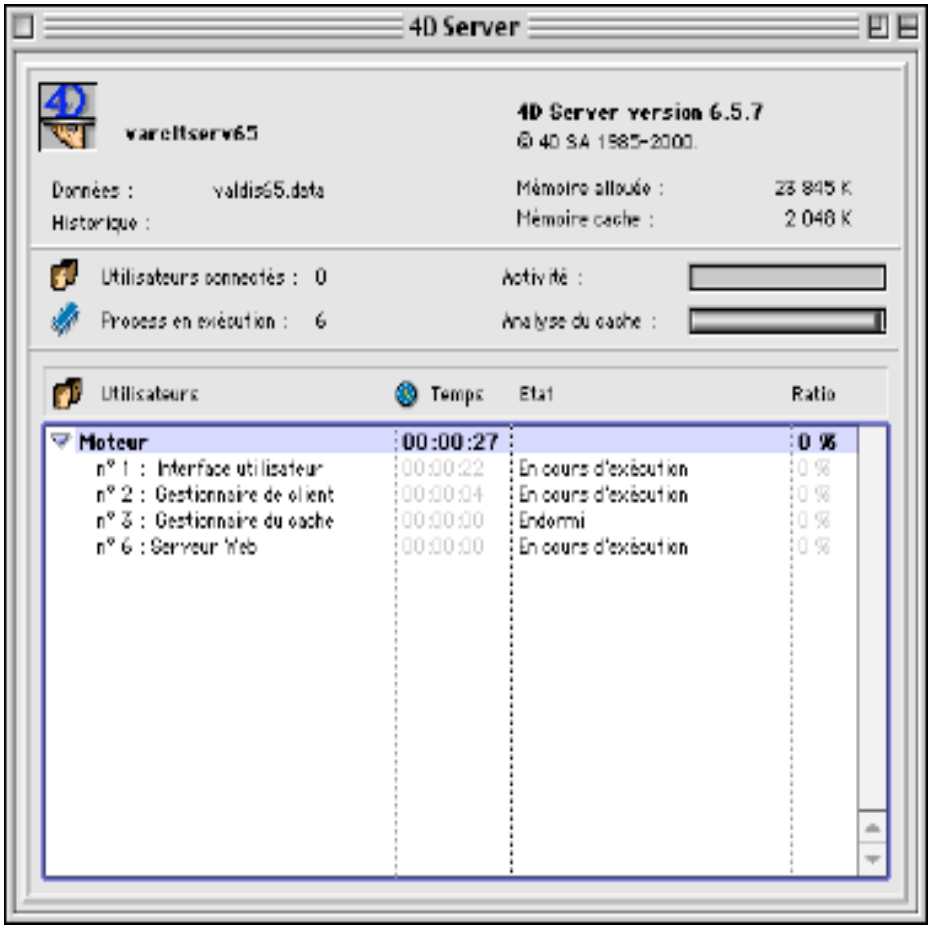
La lecture de ces informations est valable à un instant précis. En effet dès qu'il y aura une nouvelle activité (exemple : création d'un enregistrement), les informations seront rafraîchies. Vous pouvez régler le taux de rafraîchissement avec l'explorateur d'exécution.

b- La programmation.

La lecture de ces informations est valable à un instant précis. En effet dès qu'il y aura une nouvelle activité (exemple : création d'un enregistrement), les informations recueillies précédemment dans les différentes variables ne seront plus exactes.

4- Cas particulier : 4D serveur.

La figure ci-dessous montre la fenêtre de 4D serveur sous Mac OS.



La partie supérieure droite de la fenêtre montre :

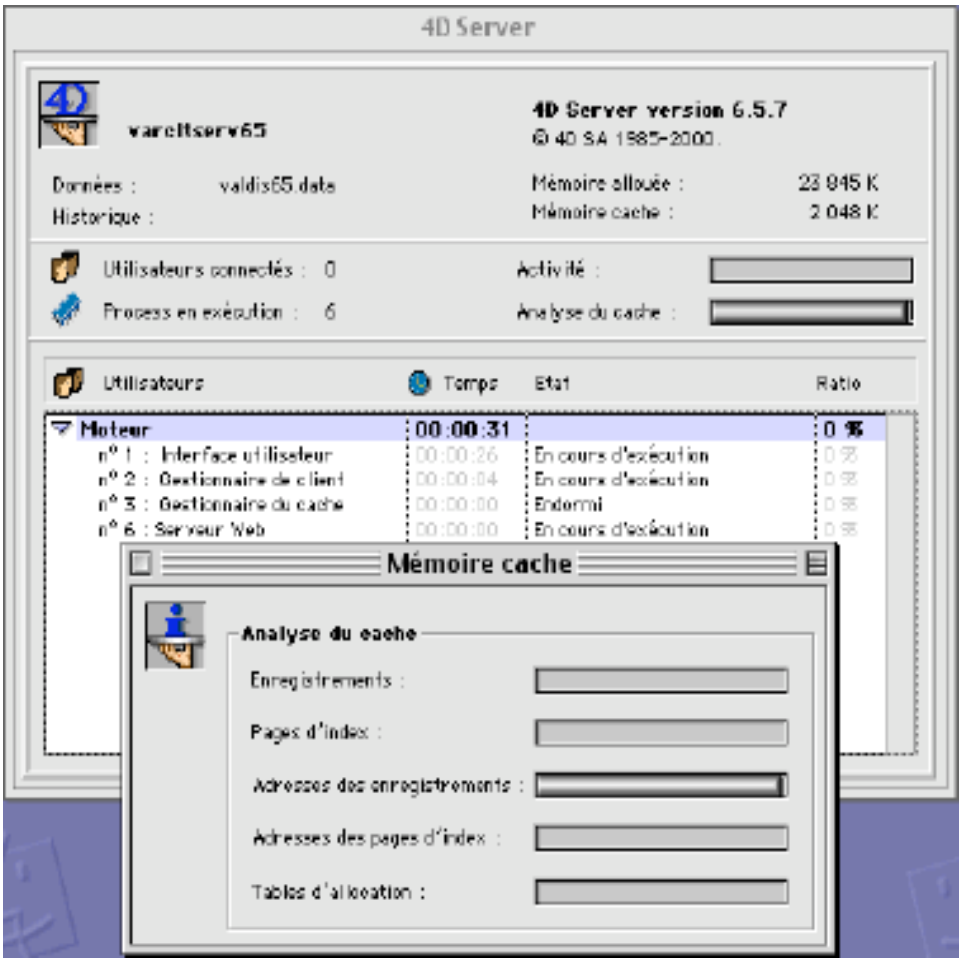
- La taille de la mémoire allouée à 4D serveur :
(Remarque : la valeur indiquée est valable pour un instant T sous Windows uniquement. Ce point va être détaillé ci-dessous.)

- La taille de la mémoire allouée réellement au cache qui a été défini dans les propriétés de la base ou dans les ressources adéquates (ressources mem et pref) à l'aide de customizer plus).

Les deux thermomètres représentent :

- L'activité qui montre la sollicitation du serveur par les clients à travers le réseau. Plus le trafic réseau augmente, plus la valeur du thermomètre croit.

- L'analyse du cache qui est basée sur un ratio. Si vous cliquez sur le thermomètre inférieur, vous obtiendrez une fenêtre nommée "mémoire cache" avec 4 autres thermomètres (voir figure ci-dessous). La graduation est exprimée en %.



Il faut remarquer que les thermomètres indiquent l'activité sur des catégories d'objets comme les index, les enregistrements, etc.

Ces différentes informations vont être décrites dans un prochain paragraphe.

5- Les différents mécanismes.

a) Qu'est ce que le moteur de 4D ?

C'est le code source de 4D. Ce code est chargé et exécuté dans une partie de la mémoire principale (User Heap) appelée aussi mémoire moteur.
Cependant si le système utilisé est paramétré pour utiliser la mémoire virtuelle, le code source sera stocké et lu directement sur la partie du disque qui est réservée à la mémoire virtuelle.
Le paramétrage de cette partie sera détaillé dans le chapitre III.

Le mécanisme d'allocation de la mémoire :

Sous Mac OS
le réglage de la mémoire s'effectue avec l'option "mémoire" du dialogue "info" correspondant à l'icône du moteur 4D utilisé.
A moins que le système ne puisse allouer la mémoire demandée, cette quantité restera fixe durant toute la durée d'utilisation de l'application 4D.
Il est possible de vérifier cela à travers les solutions proposées ci-dessus.

Sous windows
la valeur correspondant à la taille allouée peut varier.
Il est possible de la vérifier grâce aux différents outils qui ont été présentés précédemment.

Ce mécanisme dit dynamique est valable pour tous les moteurs 4D.
En effet, au lancement de 4D, le système allouera un bloc de mémoire correspondant à la taille du bloc qui a été défini dans les propriétés de la base ou dans les ressources adéquates (ressources mem et pref) à l'aide de customizer plus.

Si, par la suite, 4D a besoin de mémoire supplémentaire pour gérer l'augmentation du nombre de ressources, c'est le système qui attribuera dynamiquement des blocs supplémentaires de la même taille que le premier et dont le nombre maximum peut atteindre celui qui a été défini dans les propriétés de la base ou dans les ressources adéquates (ressources mem et pref), à l'aide de customizer plus.

Remarque : Il est possible de connaître la taille réellement utilisée par 4D sous windows NT à l'aide du "gestionnaire des taches", dans l'onglet "processus". La ligne correspondante à l'application 4D qui est lancée, donne comme valeur en Ko, la taille réellement occupée (Voir le paragraphe 6a pour plus d'informations).

Le mécanisme du ratio pour le cache : le ratio compare le nombre de fois où 4D trouve un objet dans le cache avec le nombre de tentatives d'accès pour trouver cet objet.
Pour illustrer ce mécanisme voici un exemple :

Chargement d'un enregistrement.
(précision : la base possède deux tables dont la seconde contient un seul enregistrement. Aucun index dans la base).

Première opération
lancement de la base en ayant préalablement pris soin que la première table, en mode utilisation, ne possède aucun enregistrement.

Puis consulter les statistiques du cache à l'aide de l'explorateur d'exécution ou par un autre moyen. Les valeurs sont à zéro. Ceci est normal car il n'y a aucun enregistrement de chargé.

Deuxième opération
en mode utilisation, passage de la table sans enregistrement sur une table avec un enregistrement.

Constatation sur les statistiques du cache : L'Item " accès aux enregistrements" du debboger ou de l'explorateur d'exécution donne comme valeur 66 %, c'est-à-dire 1 succès pour 1,5 tentatives.

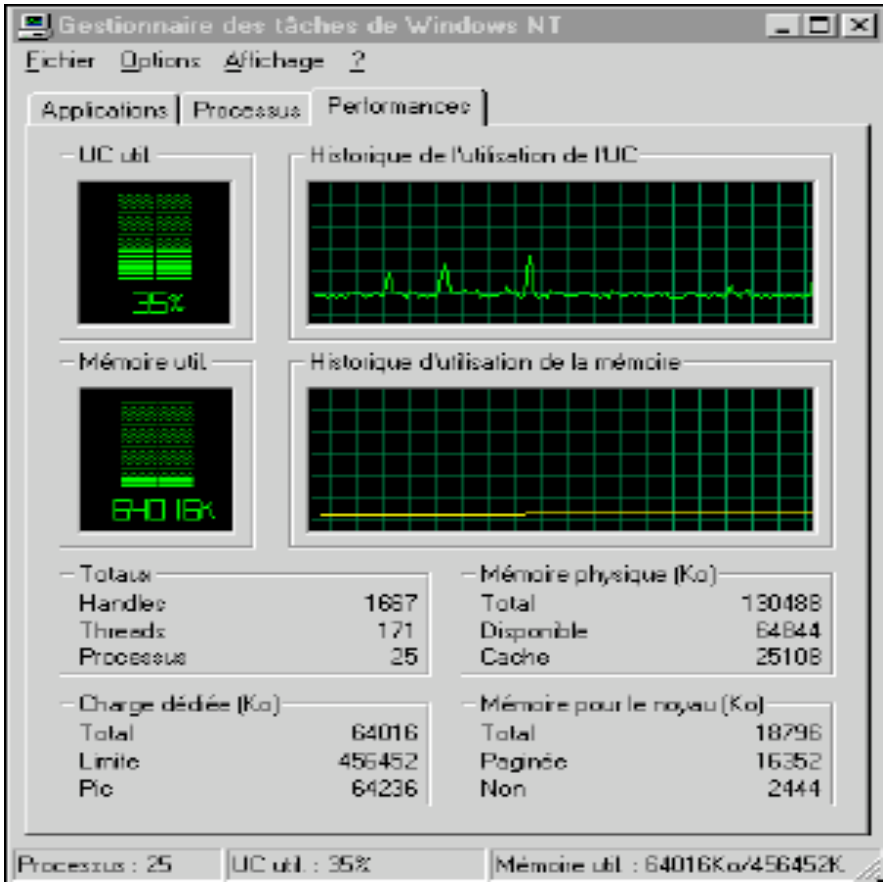
Pourquoi une valeur de 66 % et non 100% : Tout d'abord, l'enregistrement affiché n'est pas dans le cache, il faut donc le charger.
Si nous supposions qu'il possède un index, il faudrait auparavant charger l'index. Ceci réduirait encore le nombre de tentatives avec succès.
De plus, vous pouvez constater que si vous renouvelez l'opération de chargement de cet enregistrement, le ratio pourra tendre vers les 100%.
Ceci est rendu possible car l'enregistrement est resté dans le cache.

Remarque : la graduation de la fenêtre "mémoire cache" de 4D serveur donne ces mêmes informations.

6- Les informations procurées par le système.

a- Sous Windows.

Il est possible, par l'intermédiaire du gestionnaire des taches, d'obtenir des informations sur la place qu'occupe 4D en mémoire. Voir la figure ci-dessous.



Il existe un autre outil complémentaire que nous pouvons obtenir par "Démarrer, Programmes, Outils d'administrations (communs), Analyseur de performances".

b- Sous Mac OS.

L'image ci-dessous représente l'occupation mémoire. Il est possible d'obtenir cette information grâce au menu "pomme", item "à propos de votre ordinateur".

Cette fenêtre montre la liste des applications lancées sur la machine. Chacune d'elles est associée à un thermomètre qui indique la place occupée, en mémoire, par ce programme. Sur le coté gauche du thermomètre est indiquée la taille maximum dont dispose l'application.
Pour 4D, cette taille maximale correspond à trois éléments : la taille du cache ajoutée à la taille réservée pour la mémoire principale, qui inclue celle occupée par le moteur.

II- Configurer la mémoire.

1- Généralités.

Donner le maximum de mémoire à 4D peut sembler être la bonne solution pour obtenir les meilleures performances.
Il existe un point critique à ne pas dépasser. En effet, 4D manipule des objets qui se trouvent soit dans le cache soit dans la mémoire principale. Ces objets ont une date de péremption et sont éliminés quand 4D a besoin de place pour installer de nouvelles ressources.

Plus l'objet est utilisé, plus la date de péremption est éloignée.
Ce mécanisme est activé quand le moteur de 4D juge qu'il n'y a plus assez de place pour intégrer les nouveaux objets.

ATTENTION : si 4D est saturé et qu'il lui est impossible de purger (éliminer de la mémoire) des objets, un crash peut se produire.

Si vous allouez une grande quantité de mémoire à 4D, ce mécanisme est peu sollicité.

4D, pour manipuler ces objets grâce au mécanisme décrit ci-dessus, doit maintenir l'ensemble des objets qu'il possède sous forme de liste.
Ce maintient des listes est plus long s'il y a plusieurs objets à conserver en mémoire.
Il est donc possible d'atteindre le paradoxe suivant : 4D passe plus de temps à maintenir les listes d'objets plutôt qu'à recharger des objets directement du disque.
Ce comportement dégradant les performances de l'application.

Remarque : Il ne faut pas oublier que la machine, pour fonctionner, a besoin d'un système d'exploitation qui possède son propre cache, ce qui occupe une certaine place en mémoire.
Après avoir effectué de nouveaux réglages mémoire (moteur 4D, mémoire principale et cache), il est nécessaire de relancer l'application.

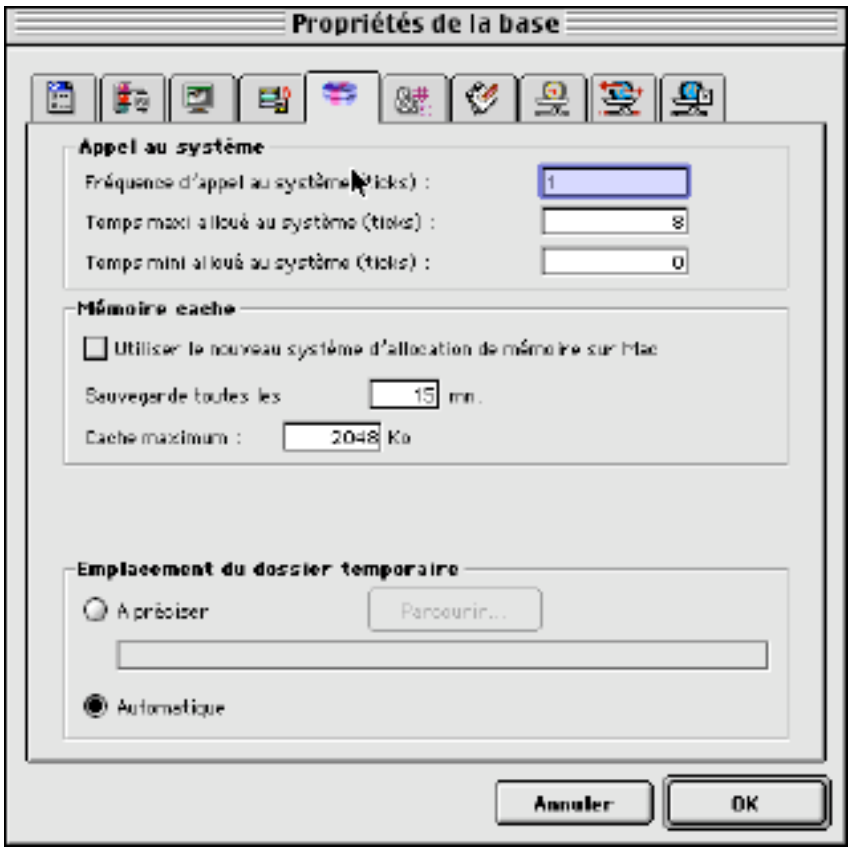
Mise en garde (Mac OS uniquement) :

A partir de la version 60XX, il est possible de configurer la taille allouée au cache de 4D de deux façons :

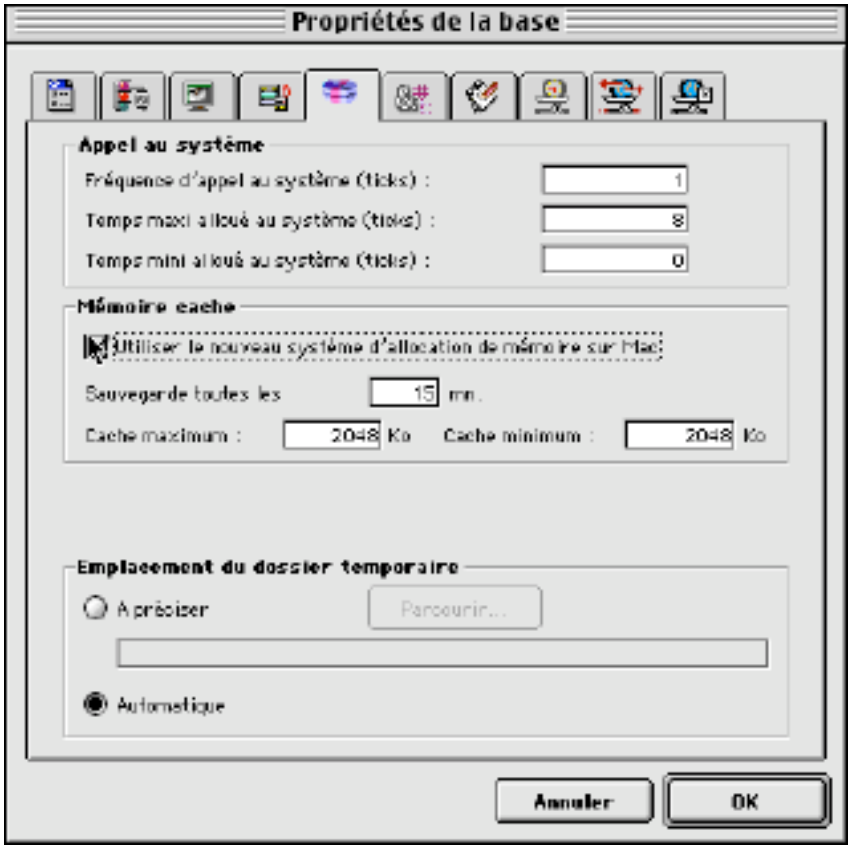
Ancien système
le cache fait partie de la mémoire allouée à l'application

Nouveau système
le cache est en dehors de la taille allouée à l'application. Il faut donc faire l'opération suivante (mémoire principale + mémoire cache) pour obtenir la taille réellement occupée.

Cette particularité s'obtient par l'intermédiaire des propriétés de la base (onglet "réglages système", thème "mémoire cache"). (Voir les images ci-dessous.)



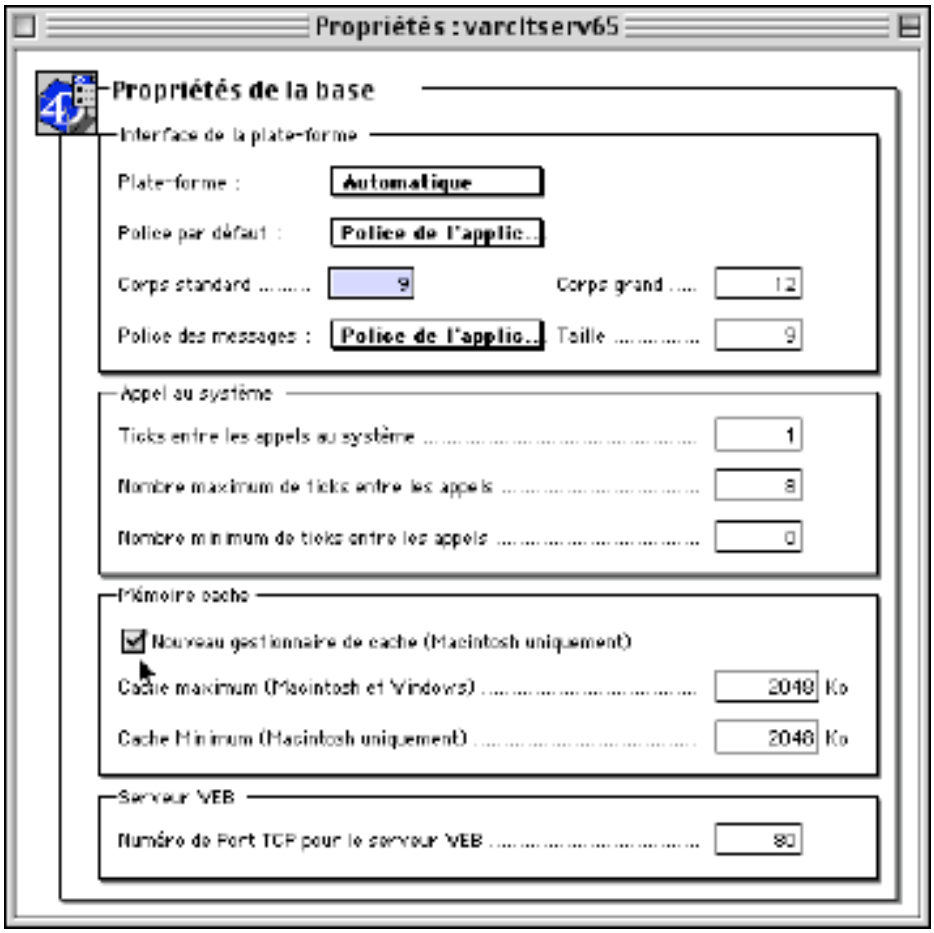
Ancien mode



Nouveau mode

Ce nouveau principe est donc identique à la gestion de la mémoire sous Windows. Nous parlerons donc de gestion dynamique de la mémoire pour Windows et pour le nouveau gestionnaire sur
Mac OS, et de gestion statique de la mémoire pour l'ancien gestionnaire sur Mac OS.

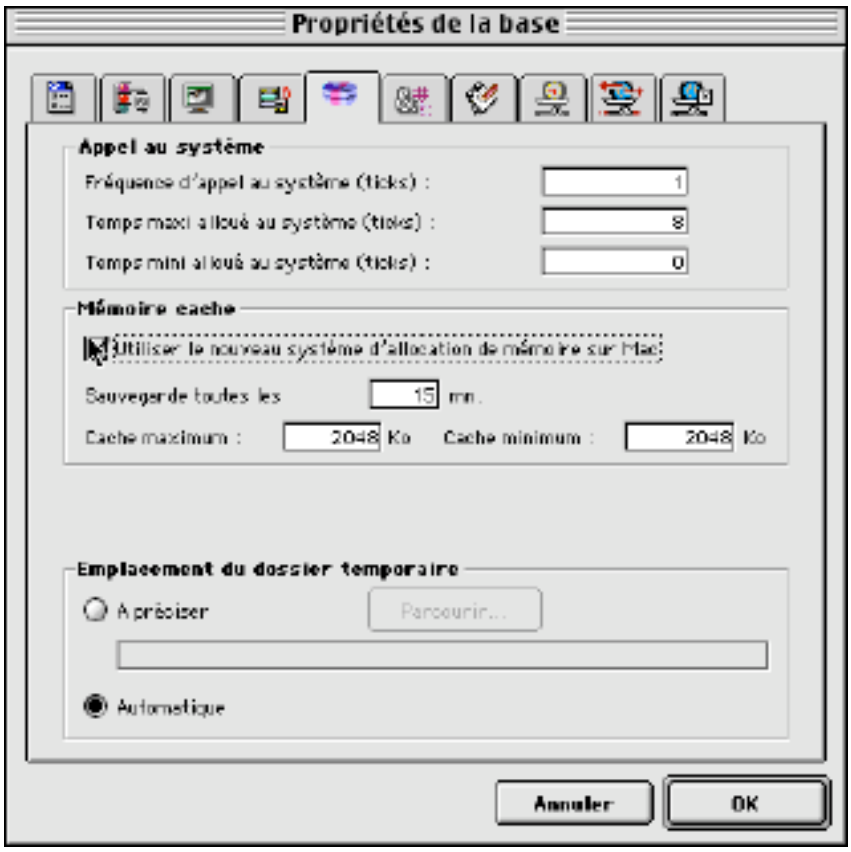
Remarque : il est possible de choisir le nouveau gestionnaire en passant une structure au customizer. Il faut éditer la ressource "propriétés" comme le montre la figure ci-dessous.



III- Mode d'emploi du réglage de la mémoire (User heap et cache).

1- Généralités.

A partir de la version 60xx de 4D, il est possible d'effectuer des réglages sur la taille de la mémoire allouée au cache (sous Mac OS et Windows) et pour la mémoire principale (Windows uniquement), en mode structure, dans les propriétés de la base, onglet "réglages système" thème
"mémoire cache". Voir les images ci-dessous.



Note : rubrique "cache maximum" : la valeur doit être comprise entre 500K et 3 276 700K.

2- Les cas particuliers.

Voir les copies d'écran pour la partie Windows.

- Sous Windows, vous pouvez constater qu'il existe un thème supplémentaire "mémoire principale de l'application".
- La rubrique "blocs utilisés" : la valeur doit être comprise entre 2K et 256K.
- La rubrique "taille du bloc" : la valeur doit être comprise entre 500K et 64000K.

Si 4D devait charger un objet dont la taille est supérieure à un bloc, il augmenterait la mémoire du nombre de blocs nécessaire au chargement de cet objet. En effet, la taille d'un bloc est indivisible et l'allocation mémoire s'incrémentera toujours par un multiple de la taille minimum d'un bloc.

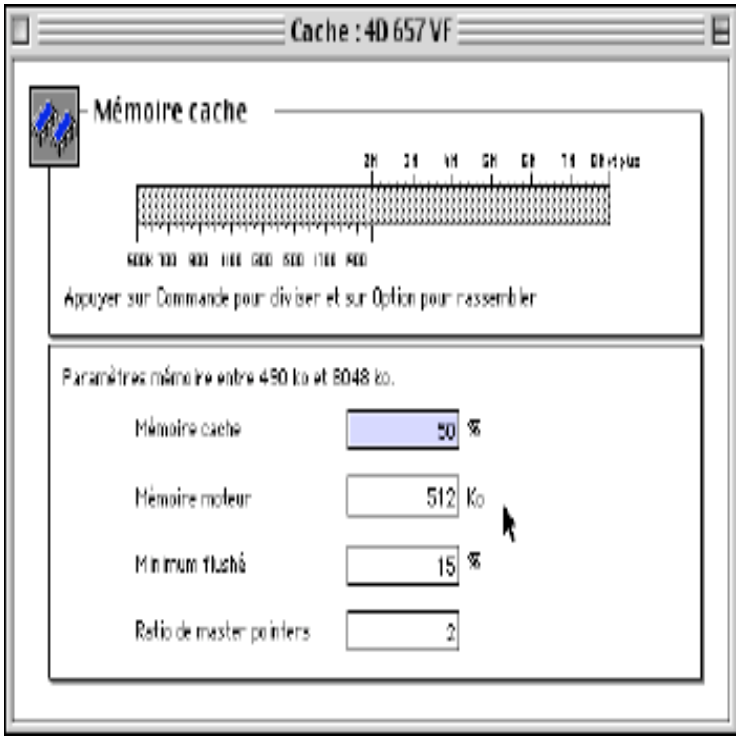
La rubrique "cache minimum" : cette rubrique apparaît uniquement quand vous utilisez le nouveau système d'allocation de la mémoire. Par contre, ces deux rubriques (cache maximum et cache minimum) suivent la même règle que pour le cache. Ils doivent être compris entre 500K et 3 276 700K.

3- Autres solutions pour configurer la mémoire (cache et mémoire principale)

Les différentes possibilités vont êtres détaillées dans le chapitre V intitulé : "Paramétrage mémoire : localisations et priorités."

4- Où trouver le paramétrage de la mémoire moteur ?

Cette partie est seulement consacrée à Mac OS.
Il faut passer l'un des moteurs de 4D au customizer et consulter la ressource "cache" comme le montre les images ci-dessous.



La rubrique "minimum flushé" correspond au pourcentage de mémoire cache que 4D peut purger quand le cache est plein ou quand il souhaite y charger de nouveaux objets.

5- L'influence de l'utilisation de la mémoire virtuelle.

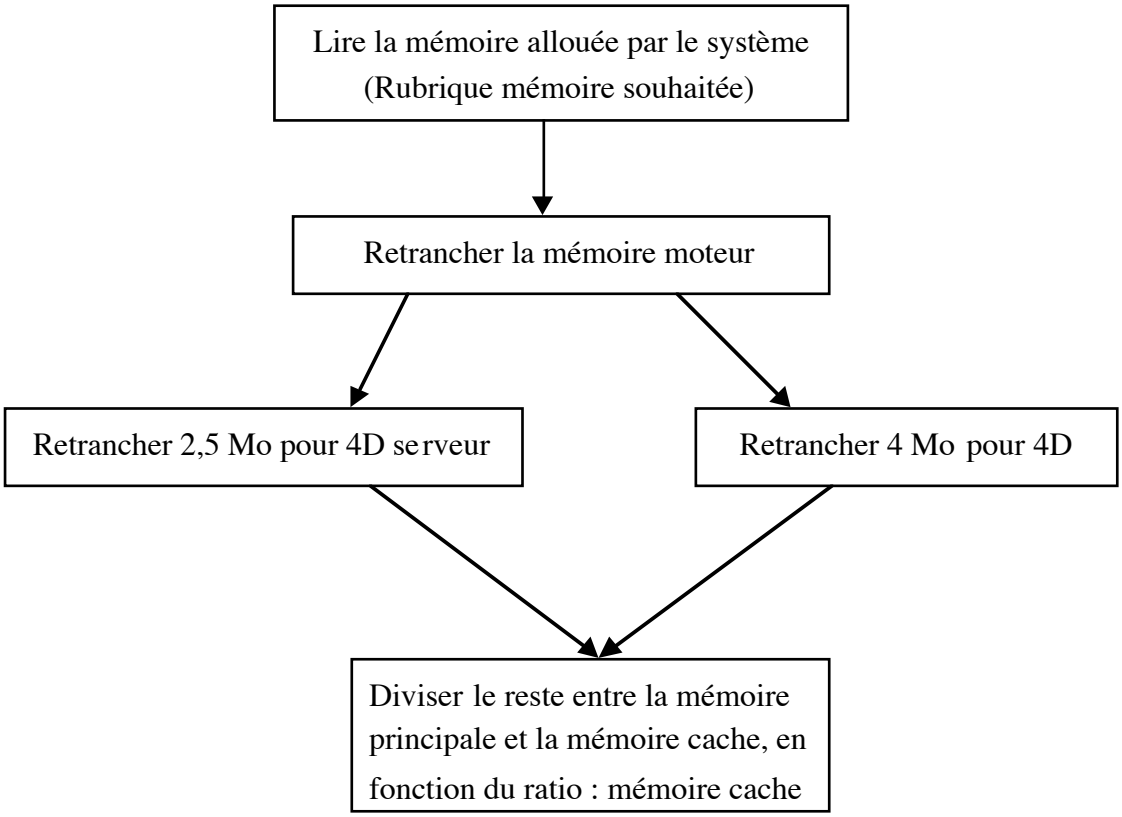
Présentation sous Mac OS des calculs à réaliser pour obtenir la taille de la mémoire principale et du cache.

a- Sous Mac OS

Il est possible, sous Mac OS, d'utiliser la mémoire virtuelle comme le montre la figure ci-dessous :

Comme nous l'avons vu dans le II, 1, ?mise en garde-, il est possible d'utiliser 2 systèmes pour la gestion interne de la mémoire de 4D : gestion statique ou dynamique de la mémoire.
Ci-dessous sont présentés les différents cas, en fonction de l'utilisation de la mémoire virtuelle et de la gestion de la mémoire interne de 4D.

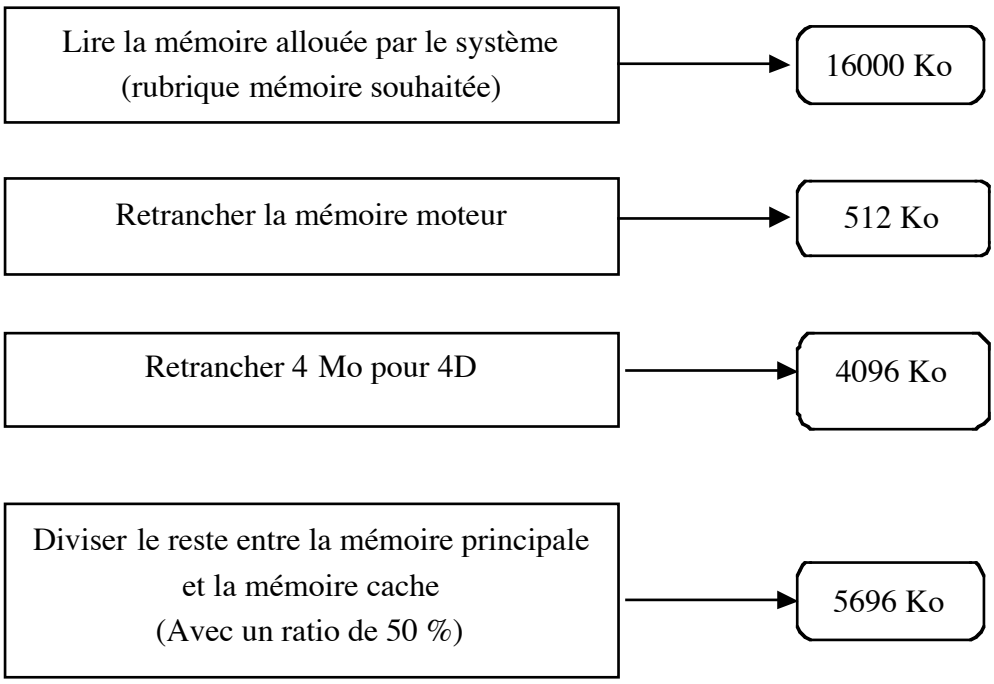
- Partie n'intégrant pas la mémoire virtuelle.



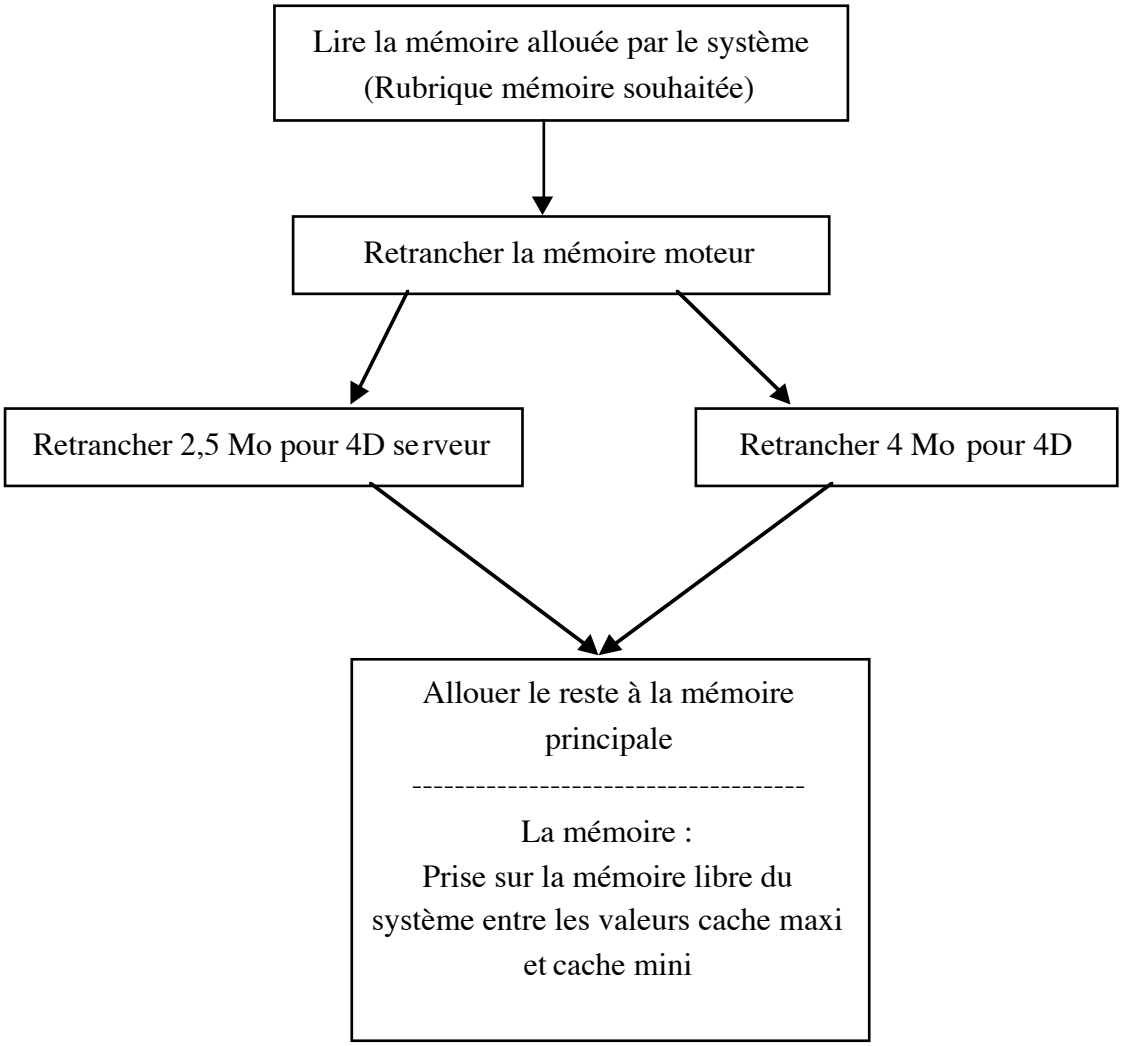
- Gestion statique de la mémoire.

Par "lire les informations", il est possible de connaître la mémoire allouée au moteur de 4D.
Mémoire cache = ((mémoire allouée-512Ko)/(100/ratio mémoire cache))
Mémoire principale = mémoire allouée - mémoire cache

- Exemple



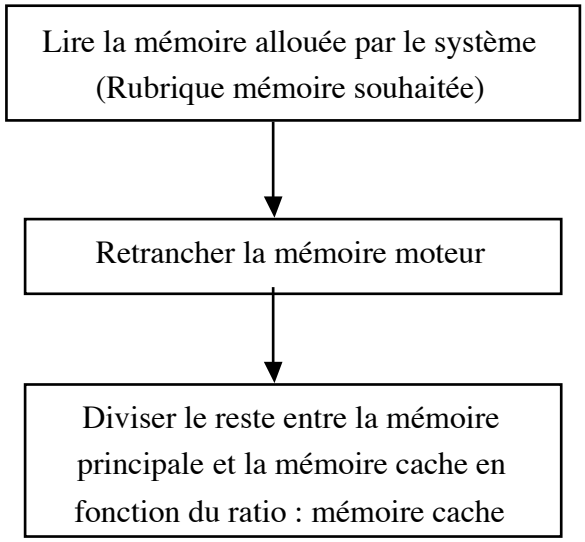
Soit : pour la mémoire principale (User Heap) : 6208 Ko (5696+512).
pour le cache : 5696 Ko.



- Gestion dynamique de la mémoire.

Remarque : si le moteur de 4D n'arrive pas à allouer la valeur du cache minimum, il découpe la partie réservée à la mémoire principale en deux, afin d'y mettre le cache. Cela revient en gros à utiliser le système non dynamique.

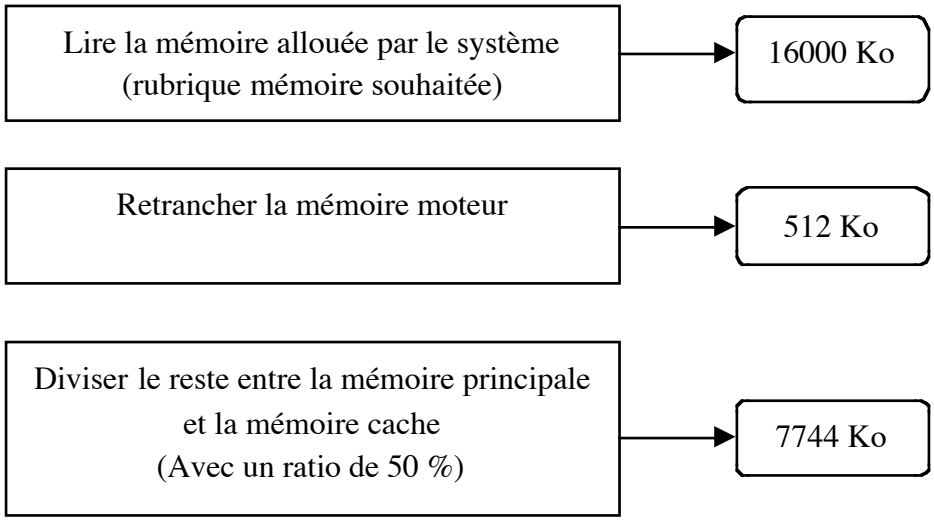
- Partie intégrant la mémoire virtuelle.

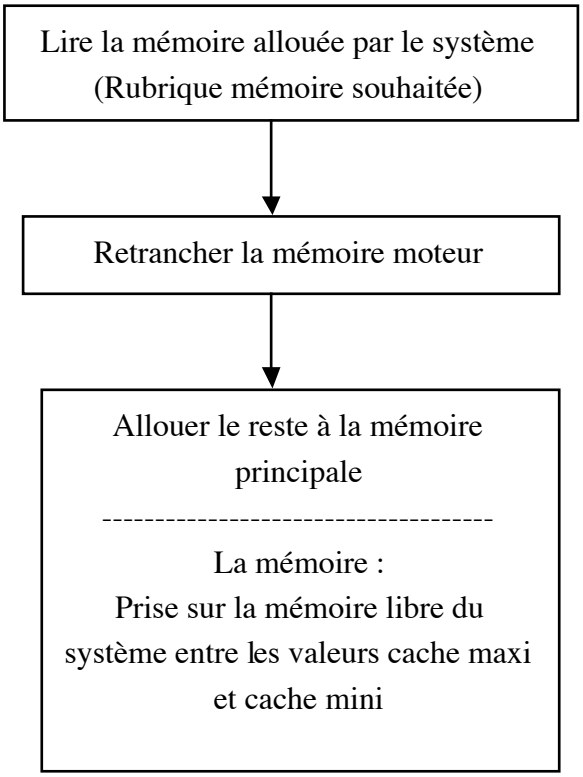


- Gestion statique de la mémoire.

Sous Mac OS par "lire les informations", il est possible de connaître la mémoire allouée au moteur de 4D
Mémoire cache = ((mémoire allouée-512Ko)/(100/ratio mémoire cache))
Mémoire principale = mémoire allouée - mémoire cache

- Exemple



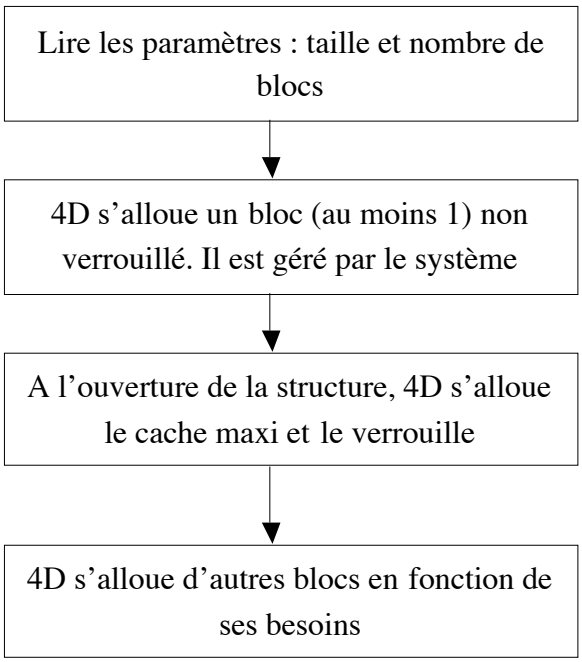


- Gestion dynamique de la mémoire.

Remarque : si le moteur de 4D n'arrive pas à attribuer la valeur du cache minimum, il découpe la partie réservée à la mémoire principale en deux afin d'y mettre le cache.
Cela revient en gros à utiliser le système non dynamique.

d- Sous Windows.

Gestion de la mémoire.



- Le cache n'est jamais swappé.
- Si l'allocation du cache est trop importante en fonction de la mémoire, 4D ne se lancera pas.

Remarque : Sur l'utilisation de la mémoire virtuelle :
Si le système requis est paramétré pour utiliser la mémoire virtuelle, le code source sera stocké et lu directement sur la partie du disque qui est réservée à la mémoire virtuelle.
Il ne sera donc pas chargé en mémoire vive comme pour un paramétrage de mémoire non virtuelle.

IV- Cas particulier sous Windows : la gestion de la taille des blocs.

Comme vu ci-dessus, la mémoire est allouée par bloc. La question est : faut-il allouer de la mémoire avec des blocs de grande ou de petite taille ?

- Généralement il vaut mieux utiliser des blocs de grande taille. Ceci permet d'éviter au moteur de 4D de demander au système de nombreuses allocations de blocs mémoire. Ces demandes d'allocation prennent un temps machine non négligeable. Plus 4D sollicite le système pour des allocations de blocs plus cela nuit aux performances de l'application.

- Si la solution des petits blocs est retenue, le système peut allouer ces blocs à quelque endroit que se soit de la mémoire. Cela ayant pour effet de réduire les performances de l'application quand 4D doit passer d'un bloc à l'autre (fragmentation).

- Ajouter à cela, dans le cas du choix des petit blocs, si 4D n'est pas le seul logiciel à s'exécuter sur la machine, le système peut attribuer d'autres blocs appartenant à une autre application, parmi les blocs 4D. Les effets engendrés sont les mêmes qu'au paragraphe précédent.

- Si 4D a besoin de charger des objets de grande taille (par exemple des blobs), il lui sera plus aisé de le faire s'il dispose de mémoire contiguë.
Remarque : un objet ne peut être segmenté sur plusieurs blocs.

- Pour de meilleures performances, la taille d'un bloc doit être comprise entre 7 et 15 Mo. Elle doit faire au minimum entre 5Mo et 7Mo.
Exemple : allocation de 40Mo à 4D soit 4 blocs de 10Mo.

- Par contre, il faut souligner que le système peut avoir des difficultés pour trouver des blocs de grande taille. En effet, un bloc est constitué d'une partie de mémoire contiguë et insécable.

Il n'y a pas vraiment de règles, cela dépend de l'application utilisée et surtout des objets manipulés par celle-ci. Plus la taille des objets est importante, plus la taille d'un bloc doit être grande.

Précision : il faut entendre par objet : les champs, images, blobs, index, ensembles, sélections, etc.
La programmation peut aussi influer sur le choix à effectuer (petit ou grand bloc) suivant le type d'objet privilégié.

V- Paramétrage mémoire : localisatons et priorités.

1- Localisation sous Mac OS.

a- Rappel sur la mémoire principale.

Sous Mac OS, la taille attribuée à l'application se paramètre grâce au dialogue "informations" du moteur.
Si le nouveau système d'allocation du cache est utilisé ainsi que la mémoire virtuelle, nous pourrons, et seulement dans ce cas, considérer que la mémoire allouée sera réellement la mémoire réservée pour la mémoire principale.

b- Le cache.

- Dans la ressource "cache" qui se trouve dans chaque moteur 4D et 4D util. Cette ressource est accessible par le customizer.

- Dans les propriétés de la base.

Emplacement du système utilisé (dynamique ou statique) :

- Dans la ressource "propriétés" qui se trouve dans chaque structure. Cette ressource est accessible par le customizer.

- Dans les propriétés de la base.

2- Localisation sous Windows.

a- La mémoire principale.

- Dans la ressource "préférences" qui se trouve dans chaque moteur 4D et 4D util. Cette ressource est accessible par le customizer.

- Dans la ressource "mémoire principale" qui se trouve dans les fichiers de préférences associés au moteur de 4D utilisé, et qui se situent dans le dossier "ACI" du dossier "système". Cette ressource est accessible par le customizer.

- Dans la ressource "préférences" qui se trouve dans chaque structure (*.4DB). Cette ressource est accessible par le customizer.

- Dans les propriétés de la base dans l'onglet : réglages système.

b- Le cache.

- Dans les propriétés de la base dans l'onglet : réglages système.

3- Les règles de priorités.

a- Sous Mac OS.

- Mémoire principale : Il n'y a pas de règle de priorité.

Attention à l'utilisation du gestionnaire mémoire interne du cache.

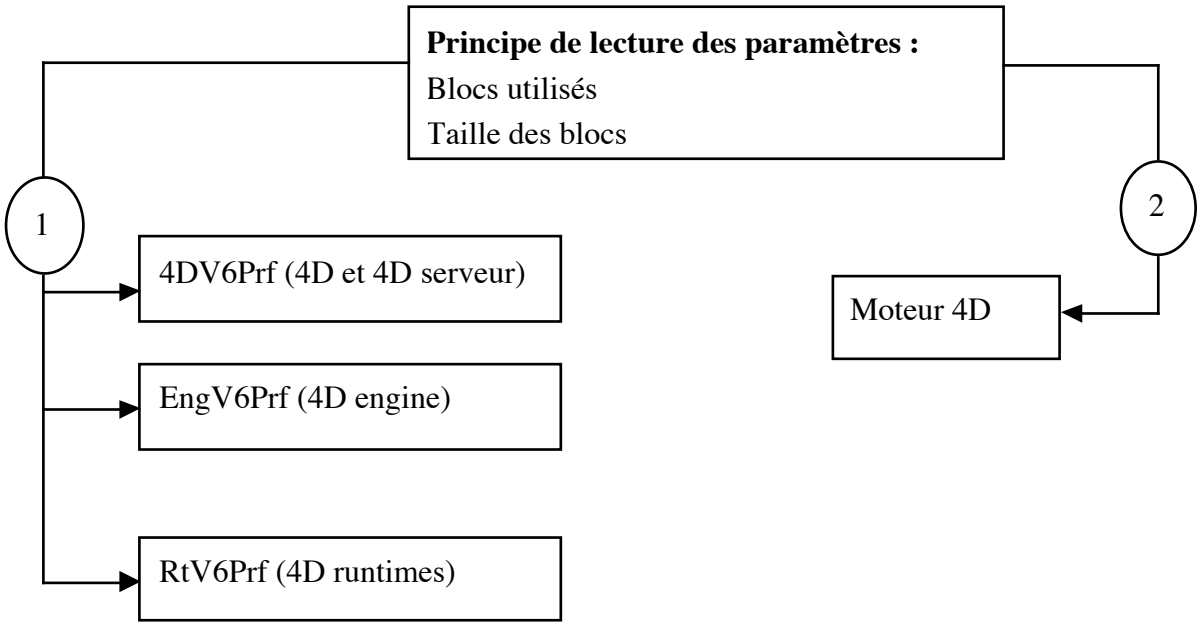
- Le cache :

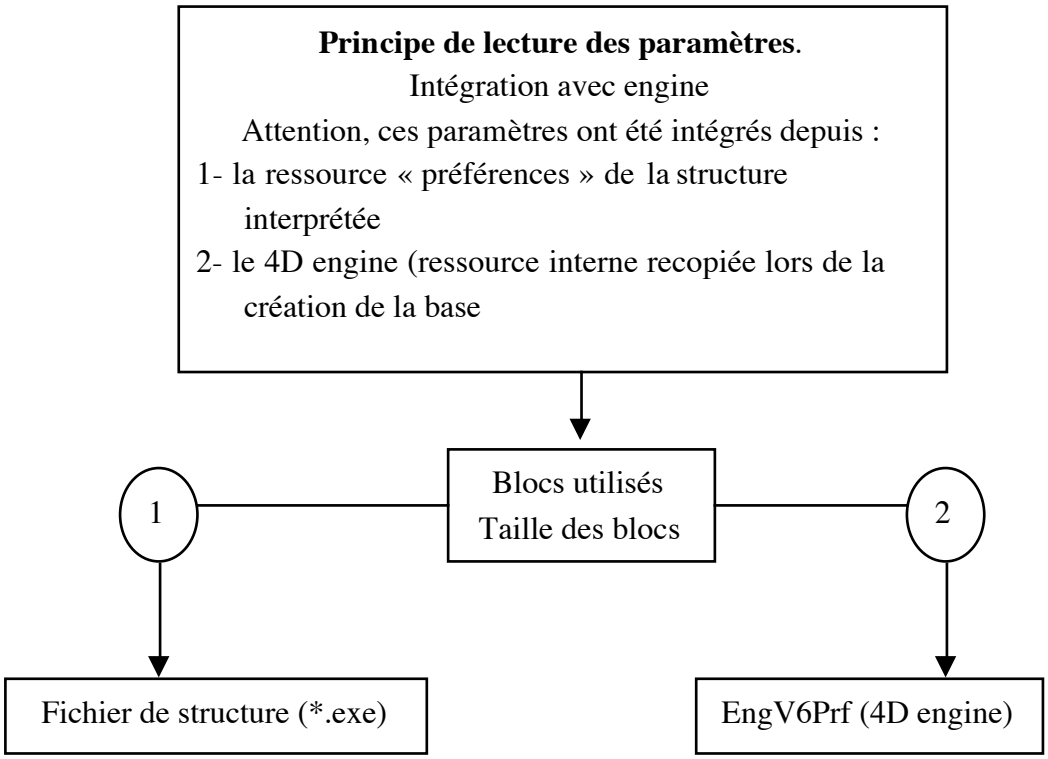
- Pour le système statique : Les valeurs sont lues dans la ressource "cache" qui se trouve dans chaque moteur 4D .
- Pour le système dynamique : Les valeurs sont lues dans les propriétés de la base de chaque structure.

A savoir (pour un cache dynamique) : Si 4D n'arrive pas à obtenir de la mémoire pour le cache (cette demande peut varier entre le cache maxi et le cache mini), 4D réintégrera automatiquement le cache dans la partie de la mémoire réservée à l'application. Cela reviendra à l'utilisation du système statique.

b- Sous Windows.

- Mémoire principale.

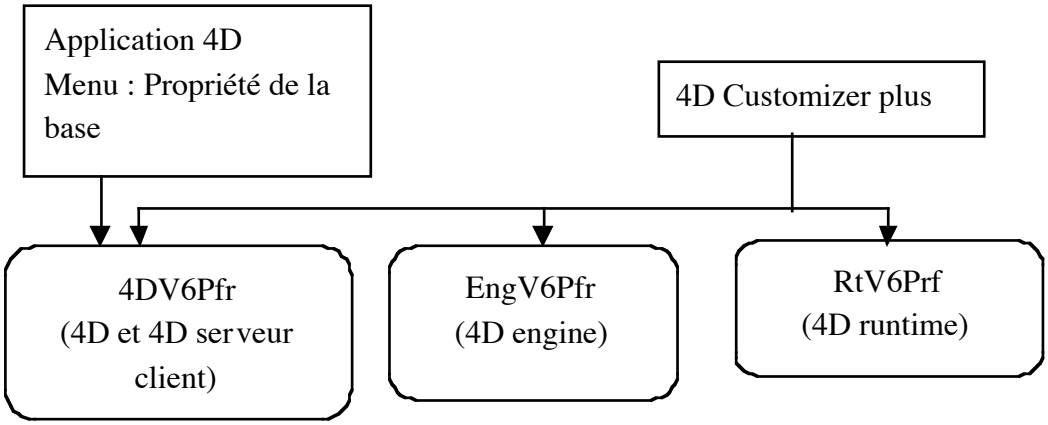




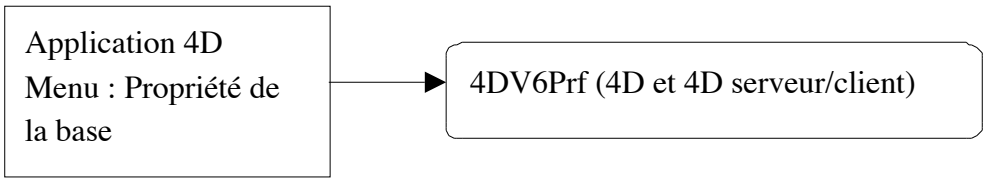
- Le cache : Valeur contenue dans la structure de la base utilisée.

4- Modification des paramètres (sous Mac OS et Windows).

(voir ressources préférence de structure)



5- Particularité de la modification des paramètres de la mémoire principale au travers des propriétés de la base dans une structure (Mac OS et Windows).



Le fait de modifier la taille de la mémoire principale entraîne la création ou la modification de la ressource "préférences" dans le fichier des préférences associé au moteur de 4D utilisé.

6- Quelques remarques.

Il existe également des fichiers de préférences sous Mac OS qui sont situés dans le dossier "ACI" du dossier "Préférences" du dossier "Système". Le contenu de ces dossiers n'est pas le même sous Mac OS que sous Windows.

Les fichiers de préférences sont conçus à l'ouverture d'un moteur 4D. Certaines ressources de ces fichiers ne sont pas créées systématiquement.

Une ressource créée volontairement dans un fichier de préférences ne peut être détruite.

Attention, les fichiers de préférences antérieurs à la version 6.5 contiennent les numéros de licence ainsi que les numéros d'expansion pack.
En 6.5, associé au moteur 4D monoposte, il stocke la liste des bases récemment ouvertes.

4D client possède son propre fichier de préférences qui est modifié au travers des propriétés de la base par le 4D client. Aucune information sur celui-ci, sur 4D serveur et sur les autres clients, n'est transmise via le réseau. La modification des paramètres n'affectera que le client pour lequel la modification a été effectuée.

Si une ressource n'existe pas, elle n'est pas prise en compte.
Si un fichier de préférences n'existe pas il n'est pas pris en compte.

Sous Windows :

A l'ouverture d'une base, le cache est créé en un bloc de mémoire qui est verrouillé par le système. Un bloc verrouillé réside obligatoirement en mémoire, il ne peut être écrit sur le disque (swappé). Par contre les blocs réservés à la mémoire principale ne sont pas verrouillés. Si 4D n'obtient pas la taille minimum requise pour le cache, il quitte.

Sous Mac OS :
A l'ouverture d'une base, dans le cadre d'une gestion dynamique du cache, il est créé à partir de la valeur maximum. Si le système n'est pas en mesure d'allouer cette taille mémoire, 4D décrémentera la taille du cache, ceci par paliers de 32 Ko, jusqu'à ce qu'il atteigne au moins cette valeur. Dans le cas contraire l'intégration du cache se fera dans la partie réservée à la mémoire principale de 4D. Le cache est créé dans un bloc de mémoire qui est verrouillé par le système. Un bloc verrouillé réside obligatoirement en mémoire, il ne peut être écrit sur le disque (swappé).

7- Les cas de dysfonctionnement.

S'il n'existe pas de ressource "préférences" dans le fichier 4DV6prf.rsr (ressources grisées), 4D n'affiche pas les bonnes valeurs dans les propriétés de la base s'il y a une ressource "préférences" active dans la structure de la base. En effet 4D prend les réglages du moteur et affiche la ressource
"préférences" active de la structure.

Sous Mac OS uniquement : quel système de gestion de mémoire interne choisir ?

Il faut paramétrer les propriétés de la base pour gérer le cache de manière dynamique car 4D ne le fait pas par défaut.

Voici les deux avantages de ce système :

- Il est possible de définir la taille du cache pour chaque application car les propriétés sont propres à chaque base.

- Quand 4D utilise les réglages associés au moteur (gestion statique), le fait de changer de moteur, ou de version de moteur, implique de nouveaux réglages. Si vous utilisez le nouveau système, les paramètres sont locaux à la base.