Optimisation de la lecture des fichiers XML

dans OASIS4

J.M. Epitalon,

sous la supervision de S. Valcke

CERFACS Working Notes WN-CMGC-10-71

1) Introduction

Dans une étude préliminaire (Cf. CERFACS Working Notes WN/CMGC/10-20), on avait analysé la couche logicielle de OASIS4 pour la lecture des fichiers XML, sasa. On avait fait ressortir ses limitations et en particulier celle qui consistait à ne pouvoir lire qu'un seul fichier XML à la fois. Cette limitation avait pour conséquence de ralentir la lecture des fichiers de configuration par OASIS4.

On a corrigé ce problème en modifiant la couche sasa et les parties d'OASIS4 qui ont un rapport avec cette couche. Puis on a fait des tests d'exécution pour mesurer le gain obtenu.

Ce document présente succinctement les modifications faites et en détail les tests et leurs résultats.

2) Rappel du problème

Au démarrage, OASIS4 cherche plusieurs informations de configuration par ordre d'importance décroissante. Malheureusement, ces informations étant réparties dans plusieurs fichiers SMIOC, il devait passer de l'un à l'autre puis revenait au premier, puis au second, etc.... Il devait faire plusieurs aller-retours entre les fichiers.

3) Nature des modifications

3.1) La couche sasa

3.1.a) Modifications fonctionnelles

La couche sasa offre trois services que la couche logicielle appelante utilise toujours dans cet ordre :

- ouverture d'un fichier XML
- extraction d'un ou plusieurs éléments XML
- fermeture du fichier XML

Initialement, un seul fichier XML pouvait être ouvert à un instant donné. A l'ouverture, le fichier XML était analysé et son contenu XML conservé en mémoire pour consultation.

Maintenant, la couche sasa permet l'ouverture simultanément de plusieurs fichiers. Le service d'ouverture de fichier garde le contenu de plusieurs fichiers en mémoire. Lorsque l'application demande l'extraction d'un élément XML, elle précise de quel fichier il s'agit.

3.1.b) Modifications techniques

Techniquement, les modifications sont mineures car la bibliothèque libxml2 utilisée est capable de gérer automatiquement la mémoire de plusieurs fichiers.

3.2) L'application

Les modifications sont mineures :

L'ordre des appels à la couche sasa a été très peu modifié : un appel pour ouvrir explicitement chaque fichier SMIOC au début et un appel à la fin pour le fermer. Quant aux autres appels, seule la syntaxe a changé : à chaque appel, il y a un paramètre de plus : la référence au fichier XML concerné.

4) Résultats

On a mesuré la vitesse des traitements des fichiers XML sur deux plateformes :

- un PC sous Linux
- un ordinateur multiprocesseur Intel

On a fait trois mesures :

- mesure du temps CPU écoulé par le driver d'OASIS pendant toute la phase d'initialisation concernant les composants de modèles (SMIOC)
- mesure de la somme totale du temps écoulé pendant la lecture par le driver d'OASIS de tous les fichiers XML
- mesure du nombre de fichiers XML lus par le driver

Pour toutes ces mesures, on a utilisé le modèle jouet Toyoa4. Les résultats de ces trois tests sont exposés dans les paragraphes suivants.

4.1) Temps CPU pendant l'initialisation des composants

On a mesuré le temps CPU utilisé par le processus maître (le driver d'OASIS) pendant la phase d'initialisation, grâce à l'appel de la primitive *clock()*, en langage C.

La phase d'initialisation consiste en trois parties :

- la lecture des fichiers XML de configuration
- le calcul des données qui découlent des données lues, comme par exemple, les connexions entre champs de couplage
- la distribution des données lues et calculées aux composantes des application.

On suppose que la première partie est la plus couteuse en temps d'exécution. Ou tout au moins, que c'était la plus couteuse avant optimisation. De toutes façons, le gain de temps obtenu dans la phase

d'initialisation après optimisation est une mesure exacte du gain de temps obtenu dans la première partie puisque seule cette partie a été modifiée.

4.1.a) PC sous Linux

Sur le PC sous Linux, la précision de la fonction *clock()* n'est que de 10 ms. En moyennant sur plusieurs essais, on trouve que le gain de temps est de 10 ms ou plus (entre 10 et 20 ms).

La durée totale de cette phase d'initialisation est de 120 ms.

4.1.b) Multiprocesseur Intel

Sur le multiprocesseur Intel, la mesure n'est pas précise : aucune certitude n'a pu être retirée de ces essais quant au gain de temps.

4.2) Temps CPU total de lecture

On a mesuré le temps réel écoulé (pas le temps CPU) pendant l'exécution de la fonction qui lit, interprète le contenu d'un fichier XML et met son contenu en mémoire pour une extraction ultérieure de ses informations une par une.

La primitive employée est *gettimeofday(),* en langage C.

Le temps réel écoulé est plus long que le temps passé en exécution mais c'en est une mesure approchée. Le temps écoulé est la somme du temps passé en exécution et du temps passé en attente pendant l'exécution des autres processus.

Mais la mesure du temps écoulé faite ici ne concerne qu'une opération élémentaire, assez courte, faite par le programme, à savoir la lecture d'un fichier. Pendant cette lecture, les autres processus du couplage OASIS4 sont eux-même en attente. Les seuls autres processus pouvant s'exécuter en même temps que la lecture du fichier sont ceux du système Linux, dont la part dans le temps total d'exécution peut être considéré comme négligeable.

4.2.a) PC sous Linux

Sur le PC sous Linux, la précision de la fonction *gettimeofday()* est en µs. En moyennant sur plusieurs essais, on trouve que le temps de lecture total est passé de 6500 à 1500 µs, soit une division de la durée de traitement par un facteur de 4,3.

4.2.b) Multiprocesseur Intel

Sur le multiprocesseur Intel, la mesure n'est encore une fois pas suffisamment précise : cela semble dû au fait que le test est fait dans un environnement multitâches multi-utilisateurs. Les processus qui s'exécutent pendant le test doivent être mis en veille trop fréquemment pour obtenir une mesure de temps d'exécution significative.

En moyenne, sur plusieurs tests on observe quand-même une réduction du temps total de lecture.

En moyennant sur plusieurs essais, et en éliminant les valeurs trop élevées qui ne sont pas significatives, on trouve que le temps de lecture total moyen est passé de 25000 µs (deux tests

significatifs : 28607 et 23148 µs) à 21000 µs (trois tests significatifs : 26358, 17383, 19168 µs), soit 16%.

4.2.c) Nombre de fichiers XML lus par le driver

Sur les deux plateformes, le nombre de fichiers XML lus est passé de 19 à 4.

5) Conclusion

L'optimisation du temps de lecture des fichiers XML semble prouvée par les tests mais difficilement quantifiée. On peut raisonnablement penser que pour une simulation couplant de nombreux champs de nombreux modèles, et sur un gros ordinateur ayant un accès lent à son système de fichiers, le gain de temps sera significatif.