WComp: Analyse de Performances (CPU)

Vincent Hourdin, Stéphane Lavirotte, Gaëtan Rey, Jean-Yves Tigli

Présentation générale du cadre de l'évaluation

Nous disposons actuellement de deux versions de WComp:

- **WCompAddIn**: Un container WComp UPnP inclus dans SharpDevelop (version actuellement distribuée et que vous utilisez, qui utilise l'interface graphique de SharpDevelop). C'est la version disponible sur le site web avec accès par mot de passe (Emeric y a un accès): https://www.wcomp.fr/nightly-builds/
- **WCompConsole**: Un container WComp UPnP, sans interface graphique (mode console / ligne de commande). Nous ne distribuons pas encore cette version à l'extérieure car elle n'est pas packagée, mais rien ne s'y oppose.

La version de **WCompAddIn** que vous connaissez, intégrée à SharpDevelop, souffre de temps de modifications de l'assemblage (création de composants et de liaisons) trop importants, ainsi qu'une consommation très élevée de mémoire. Ceci est dû à l'intégration de l'affichage graphique de l'état du container dans la plate-forme (via SharpDevelop) qui ralentit énormément les traitements. Cela dit, c'est une bonne plate-forme pour le prototypage de nos applications de test, et dans ce cas, les performances ne sont a priori pas la préoccupation principale.

Toutefois, nous sommes bien conscients que le temps d'adaptation est capital dans WComp, et c'est d'ailleurs un des points que nous mettons en avant dans nos publications, le temps de réactivité de l'adaptation (un article en cours de soumission à la revue "*LNCS Transactions on Aspect-Oriented Software Development*" pour l'évaluation de la complexité et du temps nécessaire pour le calcul de l'adaptation).

L'architecture SLCA de WComp nous permet de découpler le container (qui instancie les composants et liaisons entre composants) et le designer (affichage et contrôle/modification externe de l'assemblage). Nous disposons donc d'un container en ligne de commande **WCompConsole** (sans aucune interface graphique), et d'un designer qui permet d'afficher les containers découverts sur le réseau. Ceci nous permet de casser le couplage fort qui existe dans la version avec SharpDevelop. Les performances sont bien meilleures dans ce cas (plus de gestion de l'affichage qui coûte très cher en temps de calcul et en consommation mémoire dans le container). Si le container **WCompConsole** est utilisable dès maintenant, la partie de visualisation externe est, quant à elle, pour le moment non fiable. Donc nous ne distribuons pas encore cette version, mais à terme, cela devrait être le cas.

Mesures de performances

Conditions d'expérimentation

Pour réaliser les tests présentés ci-dessous, nous avons utilisé la configuration suivante:

- Intel Core i7 2.67GHz M 620 (ordinateur PC portable tout neuf)
- WCompConsole-2.4.822 compilé en mode release, sans affichage console des événements UPnP (WCompNetDevice.cs:343) et stdout redirigé sur /dev/null

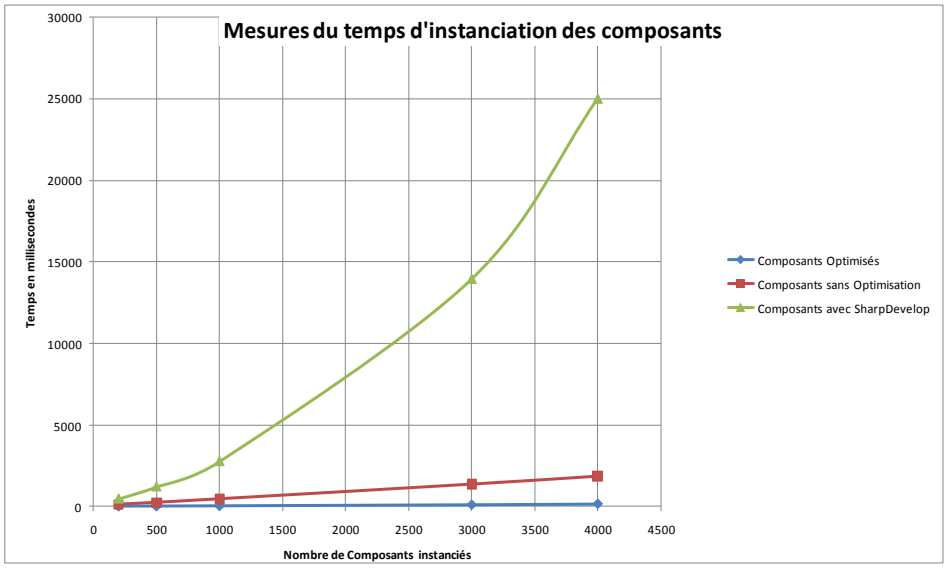
Les résultats qui sont fournis ici sont la moyenne de 10 exécutions réalisées lors de deux lancements de WComp (donc 2 exécutions du programme qui fait une boucle de 5 fois l'exécution à mesurer). Nous avons fait varier le nombre de composants instanciés (entre 200 et 4.000 composants) dans

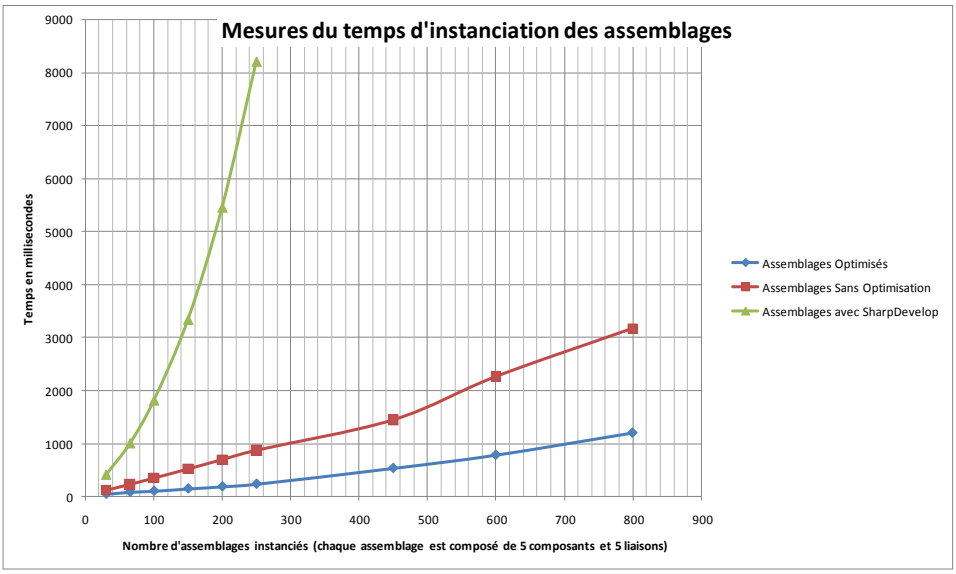
une première évaluation. La deuxième évaluation porte sur la variation du nombre d'assemblages créés (entre 30 et 800 assemblages, ce qui représente entre 150 composants et 150 liaisons d'une part et 4.000 composants et 4.000 liaisons d'autre part).

Résultats

Les résultats que nous présentons dans cette section sont l'évaluation des temps nécessaires à l'instanciation de composants (1er graphique) d'une part et d'assemblages (2ème graphique) d'autre part, soit composants et liaisons. Nous mesurons donc le temps de réponse pour effectuer ce travail par la plate-forme WComp.

Nous n'avons pas fait de mesures concernant le calcul de l'adaptation du tisseur d'aspects d'assemblage car nous n'avons pas encore la version finale à disposition, mais nous avons un calcul théorique présenté dans l'article précédemment cité. Nous pouvons vous le faire parvenir si cela est nécessaire.





Nous pouvons constater, sur les deux graphiques qui ont été générés à partir de ces deux campagnes de mesures, qu'il est clair que la version que vous (et nous utilisons) avec l'interface graphique de SharpDevelop (courbe verte) est sans aucune comparaison en terme de performance avec celle

(courbe bleue) que nous pourrons faire tourner sans l'interface graphique. Cela nous permet d'envisager sereinement le passage à l'échelle pour la plate-forme WComp.

En faisant cette campagne d'évaluation de performance, nous avons même trouvé des optimisations à faire sur la version sans interface graphique. La courbe rouge montre les temps pour la version non optimisée et la courbe bleue la toute dernière version optimisée (nous avons un gain de près d'un facteur 3). Cette dernière optimisation n'a pas encore été distribuée sur notre site, car nous devons l'évaluer.

Limitations des mesures présentées

Ces mesures actuelles ne prennent pas en compte la version avec les communications UPnP. Nous planifions de refaire ces mesures avec l'interface UPnP qui permet de communiquer entre le designer d'Aspects d'Assemblage et le container à adapter.

Toutefois, nous savons que de nombreuses optimisations sont possibles sur les communications UPnP par rapport à la version actuelle. En effet, lorsqu'une adaptation doit être effectuée, actuellement tous les ordres sont envoyés un par un (un ordre par composant ou liaison à créer), ce qui est très peu efficace. Dans la nouvelle version de SharpWComp (2.4) que nous préparons, nous envisageons de modifier l'interface UPnP du container pour ajouter une méthode qui permet de lui envoyer un ensemble de modifications à appliquer à l'assemblage. On ne paiera alors ce surcoût qu'une fois par cycle d'adaptation de l'application, et le bénéfice de l'utilisation du container non-graphique deviendra d'autant plus important que l'adaptation souhaitée sera composée d'un nombre important de nouveaux composants et liaisons.

D'autre part, les nombres présentés ici peuvent augmenter avec l'abonnement de designers aux événements du service UPnP du container (opérations supplémentaires à gérer), et le surcout exact lié aux structures de données du container n'a pas été mesuré actuellement.

Enfin, nous pensons aussi faire une évaluation concernant le gain sur la consommation mémoire qui devrait donner des résultats sensiblement identiques aux mesures de performance d'après nos premières mesures.

Nous pourrons aussi faire des tests de stress de la plate-forme pour voir combien de composants et d'assemblages nous pouvons instancier "au maximum" (nous avons fait un test avec 50.000 composants; **WCompConsole** fait bien l'instanciation, mais dans un temps conséquent que nous ne nous expliquons pas totalement pour le moment).

Annexe: Mesures Brutes

Voir le fichier Excel onglet des mesures.