Mathieu Bivert, CSSR, bivert@essi.fr

PFE : Rapport de management (D_4)

7 Mars 2013

Placement constraints for a better QoS in clouds

POLYTECH NICE-SOPHIA

Coût du livrable 38 heures

Budget total du projet 304 heures

Entreprise Université de Nice-Sophia Antipolis

Lieu Sophia-Antipolis, France

Responsable Fabien Hermenier, équipe OASIS, fabien.hermenier@unice.fr

Contents

1 Description du projet

On renvoye le lecteur à la section *Description du projet* du DOW pour description plus précise de l'environnement, du vocabulaire et des besoins de l'utilisateur.

BtrPlace est un logiciel développé à l'INRIA par Fabien Hermenier qui place efficacement un ensemble de machines virtuelles sur un ensemble de serveurs physiques à l'aide de contraintes. Celles-ci sont principalement de deux natures:

1. les contraintes spécifiées par l'utilisateur, qui répondent à des cas d'utilisations particuliers, par exemple avoir à disposition un serveur sous un hyperviseur donné pour y migrer des VMs en cas de problèmes sur un autre serveur;
2. les contraintes imposées par les ressources disponibles, portant par exemple sur la mémoire, la puissance de calcul disponibles sur un nœud.

Dans le cadre de ce projet, on a travaillé en deux temps:

1. ajouter au modèle théorique derrière BtrPlace la possibilité de *typer* les nœuds et les VMs pour rendre compte de la présence de multiples hyperviseurs (Xen, VMWare, etc.);
2. concevoir et implémenter des *contraintes* mettant en œuvre ce typage.

Pour ce faire, le travail a suivi une progression incrémentale:

1. on a commencé par chercher un cas simple du typage, et on a implémenté une contrainte le décrivant;
2. dans un second temps, on a exprimé le typage dans un cas plus général, d'abords mathématiquement, puis, toujours d'une façon abstraite, sous forme de contraintes;
3. enfin, on implémente des contraintes dans BtrPlace avec Choco.

2 Synthèse des résultats obtenus

2.1 Support du typage

2.1.1 Cas particulier

On a ajouté une contrainte *Platform* permettant d'associer à des nœuds un nouvel type, c'est-à-dire en pratique, d'effectuer une action de retypage sur un ensemble de nœuds. Cette dernière s'assure que:

1. les VMs sont parties avant le redémarrage du nœud;
2. les VMs pouvant migrer sur ce nœud ne le feront qu'après le redémarrage de celui-ci.

2.1.2 Cas général

On associe à chaque nœud un vecteur contenant un élément pour chaque hyperviseur possible. Sémantiquement, les éléments de ce vecteur représentent le nombre de machines virtuelles d'un type donné tournant sur ce nœud. Une valeur nulle indique que l'hyperviseur n'est pas utilisé; si le vecteur ne contient que des zéros, alors le nœud est considéré comme éteint.

Il s'en suit qu'une seule composante du vecteur peut-être non nulle : un nœud ne peut pas faire tourner deux hyperviseurs simultanément.

2.2 Nouvelles contraintes

En plus de la contrainte du cas particulier, deux nouvelles contraintes peuvent être implémentée. La première, *MaxVM* permet de répondre aux limitations induites par les licences des logiciels de virtualisation[1], en fixant une limite au nombre de machines virtuelles tournant sur un nœud. Afin de s'assurer de la présence de certaines plateformes pour faciler les opérations de déploiement future, la contrainte *MinPlatform* s'assure qu'il existe au moins un nombre donné de nœuds faisant tourner un hyperviseur.

[1] `https://www.vmware.com/support/licensing/per-vm/`

3 Implication des personnes

3.1 Fabien Hermenier

Suite à une mauvaise compréhension de éléments du DOW, en partie due à une rédaction un peu hâtive de celui-ci, suite à une assignation tardive au projet, les heures passées avec l'encadrant n'ont pas été intégrées dans le DOW.

Quelques entretiens avec Fabien à l'INRIA m'ont permis de me débloquer sur certains points, mais surtout d'établir des lignes directrices et des méthodes d'avancement. Enfin, une correspondance assez régulière par emails m'a permis d'avoir des retours sur le travail effectué, ce qui m'a fortement aidé à progresser.

3.2 Mathieu Bivert

Avec Fabien, nous avons définis un plan d'action pour effectuer le travail demandé, puis nous avons:

- mis au clair les besoins concernant l'augmentation du modèle;
- proposé une modélisation mathématiques du typage des nœuds et des VMs;
- enfin, proposé une implémentation en deux temps du typage.

4 Synthèses des livraisons

Les livraisons sont effectuées pour mercredi 7 mars 2013, ce qui n'est pas conforme au planning prévu. L'un des problèmes rencontrés repose sur l'approche incrémentale utilisée. Bien que celle-ci permette de progresser facilement, elle a tendance à lier fortement les différentes composantes du projet. L'interdépendance entre contraintes et implémentation du modèle limite la possibilité de rendre au temps fixé dans le DOW les différents livrables.

Pour la même raison, la structure des livrables D_2 et D_3 n'est pas optimale : en effet, la majeure partie de l'implémentation du modèle décrit dans D_2 est à placer dans les contraintes présentes dans D_3. Ces problèmes n'avaient pas été envisagés dans le DOW. Une solution aurait été de découper les rapports différement, par exemple décrire la partie théorique dans D_2 et l'implémentation dans D_3, ou encore fusionner les deux documents. Dans les deux cas, le DOW n'aurait pas été respecté.

Enfin, des incohérences au sein du DOW rendent son interprétation difficile.

5 Suivi budgétaire

5.1 Consommation du budget

Le temps de rédaction des rapports et pour la gestion du management du projet ont été surévalués : il aurait été plus judicieux de passer plus de temps sur l'implémentation du code.

5.2 Synthèse

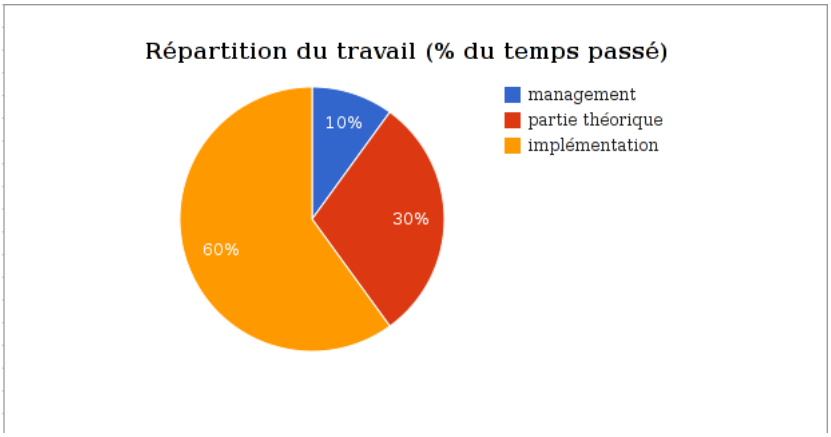
Globalement, le budget fixé a été respecté. Le projet était peut-être un peu trop ambitieux au vu du temps difficile à prévoir pour les cours et évènements para-scolaires.

6 Suivi des lots

Cette section ne correspond pas à celle annoncée dans le DOW. En effet, des incohérences sur le découpage de projet rendent la partie difficilement utilisable. On se propose ici, de décrire les rendus plus que les lots prévus. La description du travail du DOW est toujours d'actualité.

Une quinzaine d'heures de travail n'ont pas été consommée par rapport au budget de 304 heures (environ 5mauvaise compréhension de la date de fin, qui s'est avérer être deux jours plus tôt.

Le code du cas particulier et le modèle du cas général ont été validés par l'encadrant.



6.1 Management du projet

La planification du projet (DOW) ainsi que le rapport ci-présent sont rendus en temps, et ont pris environ un tiers du temps total passé sur le projet (30%).

6.2 Ajout du typage

L'extension du modèle théorique a été relativement rapide, et a pris dans les 10% du temps passé à travailler.

6.3 Implémentation des contraintes

L'implémentation a pris la plus grosse partie du temps (environ 60%), et n'est malheureusement pas complète : la validation des contraintes n'est pas toujours effectuée, et l'intégration du cas général pour le typage est incomplète.

La section suivante illustre plus en détails les difficultées rencontrées pour cette partie, et propose une solution.

7 Synthèse et retour d'expérience

En ne prenant en compte que le projet, la gestion du temps était loin d'être optimale. Bien que la partie théorique soit assez facile à comprendre, sa mise en œuvre dans BtrPlace est bien plus difficile. La complexité de l'implémentation est à mon avis l'un des facteurs majeurs justifiant le temps passé à implémenter le modèle.

Une solution envisageable à ce problème et peu coûteuse en temps, serait de décrire BtrPlace à l'aide de deux cartes:

- une première de « haut niveau », présentant la structure du logiciel, accompagnée d'exemples de bases et de quelques options, permettant à un novice de savoir où il est et où il va;
- une deuxième de « moyen niveau », qui explique comment le modèle est généré, et comment coder quelques contraintes.

Une API de programation est déjà présente, et fait déjà office de carte « bas niveau ». Celle-ci n'est cependant pas suffisante pour prendre en main efficacement le logiciel.

À l'heure actuelle, les ajouts en terme de code sont incomplets, mais:

1. l'implémentation du typage telle que décrite plus haut permet de résoudre un autre projet proposé par Fabien Hermenier, à savoir la gestion de la limitation sur le nombre de VMs pour un hyperviseur donné pour des questions de licence;

2. le modèle théorique peut-être facilement étendu pour gérer d'autres limitations au niveau des ressources, en fonction du type de l'hyperviseur. En effet, les licences des systèmes de virtualisation imposent généralement des limites sur la quantité de RAM utilisable, sur la puissance CPU ou encore limite le nombre d'interface réseau, la quantité de RAM, etc.

Au niveau des compétences, le projet m'a permis de revisiter certains éléments, tels que :

- Java;
- les tests unitaires pour la validation;
- git comme gestionnaire de version;
- la modélisation mathématiques
- programmation par contraintes

et d'en acquérir de nouvelles, comme:

- IDE (IntelliJ);
- outil de déploiement (Maven);
- framework Choco;
- gestionnaire de ressources;
- problèmes combinatoires

Du point de vue du management, j'ai appris qu'il me faut mieux gérer mon temps, notamment en essayant de mieux évaluer les évènements exogènes.