Indiquer dans ce cadre une éventuelle

mention spéciale (Cotutelle, confidentiel)

***DOCTORAT DE L'UNIVERSITE DE LILLE 1 SCIENCES ET TECHNOLOGIES***

**N° d'ordre :**

**NOM/PRENOM DU CANDIDAT : Polito Guillermo**

Ecole doctorale : EDSPI

Laboratoire : Ecole des Mines de Douai

Discipline : Informatique

Si cotutelle, établissement partenaire :

**JURY :**

- Directeur(s) de thèse : Stéphane Ducasse

- Rapporteurs : Cristophe Dony, Oscar Nierstrasz

- Examinateurs : Noury Bouraqadi, Luc Fabresse

**SOUTENANCE : 13/04/2015 - Ircica**

**TITRE DE LA THESE :**

Virtualisation pour Specialisation et Extension d’Environnements d’Execution

**RESUME :**

Un environnement d’exécution est l’ensemble des éléments logiciels qui représentent une application pendant son éxecution. Les environnements d’exécution doivent être adaptables à differents contextes. Les progrès des technologies de l’information, tant au niveau logiciel qu’au niveau matériel, rendent ces adaptations nécessaires. Par exemple, nous pouvons envisager d’étendre un language de programmation pour améliorer la productivité des developpeurs. Aussi, nous pouvons envisager de réduire la consommation memoire des applications de manière transparente afin de les adapter à certaines contraintes d’exécution e.g., de la mémoire limitée.

Nous proposons Espell, une infrastructure pour la virtualisation d’environnement d’execution de langages orienté-objets haut-niveau. Espell fournit une infrastructure généraliste pour le contrôle et la manipulation d’environnements d’exécution. Une représentation de ’premier-ordre’ de l’environnement d’exécution orienté-objet, que nous appelons object space, fournit une interface haut-niveau qui permet la manipulation de ces environnements et clarifie le contrat entre le langage et la machine virtuelle. Un hyperviseur est client d’un object space et le manipule soit directement au travers d’objets ’miroirs’, soit en y exécutant des expressions arbitraires.

Nous avons implementé un prototype de Espell sur Pharo. Nous montrons au travers de notre prototype que cet infrastructure supporte l’amorçage des languages et la construction sur-mesure d’environnement d’éxecution. En utilisant l’amorçage nous initialisons un language orienté objet haut-niveau qui est auto-décrit, se montrant donc plus simple à étendre. Nous avons amorcé quatre langages qui présentent des modèles de programmation différents e.g., avec des ’traits’, avec des variables d’instance de ’premier-ordre’ ou avec une couche réflexive basé sur le concept de ’miroirs’. La taille d’environnements d’éxecution est une technique qui génère une application spécialisé en extrayant seulement le code utilisé pendant l’éxecution d’un programme. Une application taillée inclut seulement les classes et méthodes qu’elle nécessite, et évite que des librairies et des frameworks externes surchargent inutilement la base de code. Notre technique de taille basé sur Espell, créé des versions spécialisées des applications,en sauvant entre un 95% et 99% de la memoire en comparaison avec la distribution officielle de Pharo.

Enter here any special mention

(Co-tutorial thesis, confidential)

***DOCTORAT DE L'UNIVERSITE DE LILLE 1 SCIENCES ET TECHNOLOGIES***

**N° order :**

**NAME/SURNAME OF THE CANDIDATE : Polito Guillermo**

Doctoral School : EDSPI

Laboratory : Ecole des Mines de Douai

Discipline : Informatique

In case of co-tutorial thesis, provide the partner institution :

**THESIS COMMITTEE :**

- Thesis supervisor(s) : Stéphane Ducasse

- Referees : Cristophe Dony, Oscar Nierstrasz

- Examiners : Noury Bouraqadi, Luc Fabresse

**DEFENSE : 13/04/2015 - Ircica**

**TITLE OF THE THESIS :**

Virtualization Support for Application Runtim e Specialization and Extension

**ABSTRACT :**

An application runtime is the set of software elements that represent an application during its execution. Application runtimes should be adaptable to different contexts. Advances in computing technology both in hardware and software indeed demand it. For example, on one side we can think on extending a programming language to enhance the developers’ productivity. On the other side we can also think on transparently reducing the memory footprint of applications to make them fit in constrained resource scenarios e.g., low networks or limited memory availability.

We propose Espell, a virtualization infrastructure for object-oriented high-level languages runtimes. Espell provides a general purpose infrastructure to control and manipulate object-oriented runtimes in different situations. A first-class representation of an object-oriented runtime, namely an object space, provides a high-level API that allows the manipulation of such runtime and clarifies the contract between the language and the virtual machine. A hypervisor is the client of an object space and manipulates it either directly through mirror objects, either by executing arbitrary expressions into it.

We implemented a Espell prototype on Pharo. We show with this prototype that this infrastructure supports language bootstrapping and application runtime tailoring. Using bootstrapping we describe an object-oriented highlevel language initialization in terms of itself. A bootstrapped language takes benefit of its own abstractions and shows easier to extend. We bootstrapped four languages presenting different programming models e.g.,

traits, first-class instance variables and mirror-based reflection. Application runtime tailoring is a technique that generates a specialized application by extracting the elements of a program that are used during execution. A tailored application encompasses only the classes and methods it needs and avoids the code bloat that appears from the usage of third-party libraries and frameworks. Our run-fail-grow tailoring technique based on Espell succeeds in creating specialized versions of applications, saving between a 95% and 99% of memory in comparison with Pharo’s official distribution.