Rapport sur la thèse

FLUXIONAL COMPILER : SEAMLESS SHIFT FROM DEVELOPMENT PRODUCTIVITY TO PERFORMANCE EFFICIENCY IN THE CASE OF REAL-TIME WEB APPLICATIONS

présentée par Etienne Brodu

Domaine de recherche

Les applications web sont une partie de plus en plus importante de l'ensemble des applications logicielles utilisées. Les sociétés qui proposent de nouvelles applications doivent pouvoir rapidement rencontrer (ou non) leur public. Pour cela il est indispensable de disposer de langages de programmation proposant une très grande productivité pour les applications web. Javascript en est un. Il est le plus souvent associé à un modèle d'exécution dirigé par les évènements.

Lorsqu'une application rencontre le succès, l'équipe de développement doit alors faire face à l'augmentation, parfois extrêmement rapide, du nombre de requêtes utilisateurs. Il est alors indispensable de recourir au parallélisme. Toutefois ceci nécessite le plus souvent de changer de langage et outils de développement ce qui impacte très négativement la productivité.

L'objectif du travail d'Etienne Brodu est de proposer un outil permettant de transformer des applications web écrites Javascript et suivant un modèle d'exécution dirigé par les évènements en application répartie sous forme de pipeline. Cette thèse considère les applications web qui traitent des flux de requêtes utilisateurs en temps réel souple.

Contenu et contribution

Le mémoire comprend six chapitres incluant un chapitre introductif et un chapitre de conclusion, et deux annexes.

Le chapitre 2 pose en détails le contexte du travail et ses objectifs.

Le chapitre 3 est un chapitre d'état de l'art. Il s'applique tout d'abord à définir les termes importants dans le contexte : d'une part ceux liés aux aspects de productivité, et d'autre part ceux liés aux aspects d'efficacité d'exécution. Etienne Brodu analyse ensuite différentes plateformes sous ces deux angles, ainsi que celui de l'adoption de la plateforme par les communautés et l'industrie. L'efficacité ne pouvant être obtenue que par des extensions concurrentes et parallèles, une sélection de telles extensions est examinée, ainsi que les outils pouvant faciliter le développement d'applications concurrentes ou parallèles. Les modèles de parallélisme considérés sont peu nombreux. En particulier ceux qui sont utilisés pour le calcul haute performance. Même s'ils sont peu utilisables directement pour les applications considérées, il aurait été intéressant de les mentionner ne serait-ce que par ce que les données présentées par certaines applications

web, sont produites en utilisant de tels modèles.

Le chapitre 4 présente la proposition qui consiste à transformer une application dirigée par les évènements en une application organisée sous forme de pipeline réparti. Informellement des règles de transformation sont proposées pour passer d'un modèle à l'autre, à sémantique préservée. Pour ce qui est des applications sous forme de pipeline réparti, Etienne Brodu propose un modèle d'exécution nommé *Fluxional Execution Model* et un langage pour décrire les programmes. Ce modèle et ce langage peuvent être utilisés directement plutôt que comme cible d'une transformation. Le chapitre se termine ainsi par une application développée directement avec des fluxions.

Le chapitre 5 présente les implantations de la proposition du chapitre 4 : deux compilateurs nommés respectivement Due et Fluxional. Le premier a pour objectif remplacer des chaînes de continuations de l'application d'origine pour produire des séquences d'une sorte de *future* nommée *due* et qui diffère légèrement de celle qui existe dans la norme ECMA de Javascript. Toutes les continuations ne sont pas compatibles avec cette transformation. Ce compilateur introduit donc de l'asynchronisme dans l'application. Le second compilateur transforme l'application en une application « fluxionnelle ». Les problèmes considérés ici sont ceux du passage d'une mémoire globale à une mémoire répartie. Les limitations de ce compilateur sont dues à la nature possiblement extrêmement dynamique des programmes Javascript et de l'incapacité des analyses statiques mises en place pour les détecter.

Le chapitre de conclusion établit très précisément les limites de l'approche, et annonce sans détour que les outils proposés sacrifient encore de la productivité pour obtenir de la performance. Néanmoins Etienne Brodu dégage des perspectives intéressantes pour surmonter ces limitations.

Évaluation et recommandation

Le mémoire est bien construit, les coquilles peu nombreuses. La présentation est soignée.

Le travail de Etienne Brodu aboutit à une proposition pragmatique qui, bien qu'elle nécessite encore un travail non négligeable de la part des développeurs, facilite le développement continu d'une application web vers une application web efficace.

Outre sa valeur intrinsèque, ce travail a été en partie présenté au *Workshop on All-Web Real-Time Systems* en 2015, et aux colloques *Middleware 2014* et *ACM Symposium on Applied Computing* en 2016.

Pour toutes ces raisons, je donne un avis favorable à la soutenance de la thèse de Etienne Brodu.

Frédéric Loulergue
Université d'Orléans
Orléans, le 3 juin 2016