

# Proof transformation for Separation logic

## *Master thesis PROJECT PLAN*

Project period: December 16st 2009 – June 15st 2010

Student name: Tang Mei

Status (N-th semester):

Email address: mavis.tangtang@gmail.com

Supervisor name: Martin Nordio

## 1. PROJECT DESCRIPTION

### *Overview*

As the emerged threaten caused by unsafe execution of mobile code, Proof-Carrying Code (PCC) was adopted to establish “trust” between the code producer and consumer. In PCC, the code producer generates a formal proof automatically by Certifying compiler which indicates the code’s adherence to the security properties specified by the code consumer.

Proof- Transforming Compilers (PTC)[3,4] is a similar approach to Certifying compiler in PCC, but take a source proof as input and produce the bytecode proof, and with interactive source code verification compared to Certifying compiler, more complex properties can be handled. In order to simplify the translation from source proof to the bytecode level, they presented a Hoare- style logic for bytecode similar with the source code logic, proof transformation was formalized and soundness result was proved in their works.

Separation logic[5] is an extension to Hoare logic that permits reasoning about shared mutable heap structures, and it has been utilized to modular reasoning in object- oriented languages in many works[6,7]. In order to automatically translate source code proof to bytecode logic by PTC, a separation logic for bytecode is needed. This project consists of the separation logic for bytecode, and the translation from source to bytecode level.

### *Scope of the work*

This project will develop a proof transformation for separation logic from source to bytecode level, the task of this project consists of developing separation logic for CIL/java bytecode, and the proof transformation from the source separation logic to the bytecode separation logic.

### *Intended results*

The result will be a bytecode logic using separation logic, and proof translation functions which takes the source code proof and translates it to the bytecode level.

### *Optional part*

An optional part of the project is to develop the soundness of the translation

## 2. BACKGROUND MATERIAL

### *Reading list*

- Nordio, M. and Müller, P. and Meyer, B.: Proof-Transforming Compilation of Eiffel Programs. TOOLS-EUROPE, 2008.
- P. Müller and M. Nordio: Proof-Transforming Compilation of Programs with Abrupt Termination. 6th Workshop on Specification and Verification of Component-Based Systems (SAVCBS), 2007.
- F.Y. Bannwart and P. Müller. A Logic for Bytecode. In F. Spoto, editor, Bytecode Semantics, Verification, Analysis and Transformation (BYTECODE), volume 141 of ENTCS, pages 255–273. Elsevier, 2005

## **3. PROJECT MANAGEMENT**

### *Objectives and priorities*

The goal of this project is proof transformation for separation logic from source to bytecode level. The first step is choosing a subset of bytecode, and developing a separation logic for it which shares the same heap model with the source logic in [6], then transformation functions will be defined to yield bytecode proof. Finally, if the time allows, the soundness of our transformation system will be proved.

### *Criteria for success*

The project will succeed if for a source code proof would be translated to bytecode proof using separation logic.

### *Method of work*

A termly meeting via skype with Martin Nordio is held to keep track of the progress

### *Quality management*

#### **Documentation**

Master thesis report

Examples of bytecode proof generated from source proof using separation logic.

#### **Validation steps**

Continuous feedback from the supervisor will guide the development process

## **4. PLAN WITH MILESTONES**

### *Project steps*

1. Familiar with the background materials
2. Choosing a subset of bytecode
3. Developing separation logic for bytecode
4. Choosing a subset of source code supporting separation logic
5. Translate the proof from source to bytecode level
6. Optional: soundness proof of transformation
7. Writing thesis report

### *Deadline*

June 15st 2010

*Tentative schedule*

Week	51	52	53	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
letting started	■	■																							
			■	■	■	■	■																		
								■	■	■	■														
												■	■												
roof transformation														■	■	■	■								
optional: soundness proof																		■	■	■	■				
writing report																							■	■	■

**REFERENCES**

- [1] Chair of Software Engineering: *Semester-/Diplomarbeiten*; Online at: <http://se.inf.ethz.ch/projects/index.html>
- [2] Bertrand Meyer: *Object-Oriented Software Construction*, 2nd edition, Prentice Hall, 1997.
- [3] Nordio, M. and Müller, P. and Meyer, B.: *Proof-Transforming Compilation of Eiffel Programs*. TOOLS-EUROPE, 2008.
- [4] P. Müller and M. Nordio: *Proof-Transforming Compilation of Programs with Abrupt Termination*. 6th Workshop on Specification and Verification of Component-Based Systems (SAVCBS), 2007.
- [5] Reynolds JC. *Separation logic: A logic for shared mutable data structures*. In: Proc. of the LICS 2002. Copenhagen, 2002. 55–74.
- [6] Parkinson MJ, Bierman GM. *Separation logic, abstraction and inheritance*. In: Proc. of the POPL 2008. New York: ACM Press,
- [7] Chin WN, David C, Nguyen HH, Qin SC. *Enhancing modular OO verification with separation logic*. In: Proc. of the POPL 2008.