ERCIM "ALAIN BENSOUSSAN"
FELLOWSHIP PROGRAMME

ERCIM
European Research Consortium
for Informatics and Mathematics

Scientific Report

First name / Family name	Marek Materzok
Nationality	Polish
Name of the *Host Organisation*	INRIA
First Name / family name of the *Scientific Coordinator*	Alan Schmitt
Period of the fellowship	01/01/2015 to 31/12/2015

I – SCIENTIFIC ACTIVITY DURING YOUR FELLOWSHIP

My fellowship was carried out in the Celtique team at Inria Rennes, under the supervision of Dr. Alan Schmitt. The subject of my work was formal semantics for the JavaScript programming language, which is the most popular programming language for client-side Web scripting and applications, and is currently gaining popularity in other fields, like server-side network programming (e.g. Node.js), databases (MongoDB), mobile apps (Cordova, jQuery Mobile), and others. Specifically, my work involved two JavaScript semantics – JSCert and λ_{JS}. The first one, which was developed in Inria and the Imperial College in London, attempts to directly model JavaScript semantics in Coq, staying close to the official ECMA specification of the language. The other, created in Brown University, defines the semantics by translation to a simple core language. It is intended to be used in analysis tools. My task was to formally prove that λ_{JS} correctly defines JavaScript by relating it to JSCert. The task would increase the confidence we have in the correctness of both JSCert and λ_{JS}, enabling development of better program analysis tools for JavaScript.

The first step of my work was formalizing λ_{JS} in Coq. The starting point was an interpreter of core λ_{JS} in Coq written by Valentin Lorentz. I formalized the semantics of core λ_{JS} in Coq and proved the interpreter sound and complete with respect to the semantics. Having both a declarative and executable model of the language, I proceeded to formalize the JavaScript to λ_{JS} translation in Coq. The translation and the λ_{JS} interpreter together form a

JavaScript interpreter, which was extracted to OCaml. The next step was formally specifying the relationship between JSCert and λ_{JS}. The heaps of JSCert and λ_{JS} were related using a (strong) bisimulation. Having the relationship formalized, I could state the correctness theorem I wanted to prove. Proving the theorem – which involved showing that each of many JavaScript language features was correctly modeled in λ_{JS} – was a time consuming process, due to the complexity of JavaScript and the technical challenges involved in developing the proof in Coq. During the fellowship period, the λ_{JS} implementation of most of the core JavaScript (strict mode) features were proven correct; proving correctness for the library functions (ECMAScript 5, chapter 15) was intentionally left for future work.

While trying to prove the correctness theorem, a large number of bugs (as in – deviations from the behavior specified by the ECMAScript standard) were found, some of them very serious. Fixing them required changes to λ_{JS}, including core language changes. As the original authors did not assist in fixing the problems, I ended up developing a dialect of λ_{JS} for the purpose of proving the correctness theorem.

An interesting consequence of my work was finding an issue in the ECMAScript specification. The issue, which involved the semantics of while loops and the *break* control statement, manifested itself as a discrepancy between the specification and the actual behavior of JavaScript implementations (including Google's V8 and Mozilla's SpiderMonkey). The problem was found to interfere with loop unrolling, an important program transformation, so we decided to contact the ECMA committee; as a result, the issue was fixed in ECMAScript 6, the latest revision of the specification. (See: https://esdiscuss.org/topic/loop-unrolling-and-completion-values-in-es6).

My work was presented on several events (see point III), and it is due to be presented on CoqPL workshop in St. Petersburg, Florida, in January.

II – PUBLICATION(S) DURING YOUR FELLOWSHIP

- Accepted talk on CoqPL 2016: M.Materzok, “Certified Desugaring of Javascript Programs using Coq”, 23 January 2016, St. Petersburg, USA

III – ATTENDED SEMINARS, WORKHOPS, CONFERENCES

- Informal Workshop on Formal JavaScript, 23 March 2015, Inria Paris (talk)
- Celtique group seminar, 15-16 June 2015, Sables-d’Or-les-Pins (talk)
- Journées Scientifiques Inria 2015, 17-19 June 2015, Inria Nancy (attended)
- AJACS group meeting, 26-27 November 2015, Inria Paris (talk)

IV – RESEARCH EXCHANGE PROGRAMME (REP)

During the time spent at CWI in Amsterdam I have met several people from the SWAT (Software Analysis and Transformation) team, including the designers of the Rascal programming language (http://www.rascal-mpl.org/): Prof. Paul Klint, Dr. Jurgen Vinju and Dr. Tijs van der Storm. During the REP I was learning the Rascal programming language for the purpose of using it to transform and simplify λ_{JS} programs. This has resulted in a proof-of-concept implementation, which is available on my Github profile (https://github.com/tilk/lambdajs-rascal). During one of the last days of my visit, I have presented a talk: the first part was a summary of the work I had done in Inria, the second contained my impressions on Rascal. The stay was very fruitful overall.