Microsoft Research Limited
Roger Needham Building
7 J J Thomson Avenue
Cambridge
CB3 0FB

Telephone: +44 1223 479 700
Direct dial: +44 1223 479 778
Fax: +44 1223 479 999
http://research.microsoft.com/~nick
nick@microsoft.com

19 September, 2012

EPSRC
Polaris House
North Star Avenue
Swindon SN2 1ET

I am writing to express my support for, and willingness and enthusiasm to act as a project partner on, the EPSRC proposal “Logical Relations for Program Verification” by Dr Patricia Johann and Prof Neil Ghani.

I am a senior researcher in the Programming Principles and Tools group at Microsoft Research in Cambridge. The group works on foundational theory, programming languages and analysis tools, and also has a strong record of transferring research into tools and products (including the F# programming language, the GHC Haskell compiler, parametric polymorphism in .NET, and the SLAyer tool for analyzing Windows device drivers). We have established links with the MSP group at the University of Strathclyde, as well as with the other partners in Edinburgh and Copenhagen. Nearly all of our research is published in peer-reviewed conferences and journals.

Logical relations are mathematical constructs that are key to formulating and proving most interesting properties of programs and programming calculi with higher-order features (including first-class functions, polymorphism, and modern module systems). Originally defined and used in the context of rather idealized, theoretical lambda-calculi, the last decade or so has seen many researchers adapt and extend the definition to develop models and reasoning principles for much more realistic programming languages, in particular ones featuring various forms of side effects, such as mutable storage and concurrency. Logical relations are even being applied to machine language, for example in work on compiler correctness. At the same time, logical relations have been refined to treat subtler program properties, including information-flow security and dimensional invariance. For the simplest idealized calculi, the underlying theory of logical relations has been extensively studied and is now fairly well-understood. The more recent extensions, whilst demonstrably successful in tackling important practical problems, are generally much more ad hoc: we generally experiment with variations on the basic pattern until we find a definition that works, without having a clear understanding of what we’re really doing. Even in the case of `purer’ mathematical calculi, there is a need to extend logical relations to more sophisticated dependently-typed systems, including those that underlie interactive proof assistants such as Coq and Agda.

This proposal outlines an unusually well thought-out research programme that aims to apply category-theoretic techniques, notably fibrations, to understand and generalize logical relations, including those for languages with effects. Category theory is the only language in which one could carry out such a study at the appropriate level of abstraction, but the proposal also includes a number of specific instantiations of the general theory and domain-specific logical system, which will bring the foundational work to a wider audience and enhance its applicability. If successful, this work will significantly deepen our understanding of logical relations and lay the foundations for future applications to new languages and program properties. Dr Johann and Prof Ghani are exceptionally well-qualified to carry out this programme, having an extensive track record in logical relations, the relevant category theory and its applications to programming language semantics.

I plan to collaborate with the proposers mainly on WP6, logical relations for effects. Together will colleagues here in Microsoft and in Munich, I have worked for some years on the application of logical relations to give semantics and equational reasoning principles for languages whose type systems refine conventional types with extra information tracking which side-effects an expression may have. Such type-and-effect systems have many applications, including optimizing compilation (for example in commuting computations whose side effects do not interfere, or hoisting an effect-free computation out of a loop), and we have even built an ML compiler which works this way. We have had considerable success applying logical relations to formulate the meaning of type-and-effect systems, but have hitherto had to re-build the whole theory from the ground up for each system. I am excited by the prospect of a general theory and am convinced that the combination of the algebraic theory of effects with logical relations, as outlined in this proposal, is the right way to go about developing one. I anticipate hosting at least two week-long visits to Microsoft by Dr Johann and Prof Ghani to collaborate on the project, and greatly look forward to working with them.

Yours,

Dr Nick Benton, Senior Researcher