PROJECT SUMMARY

Need for speed. Appetite for computing power is insatiable, fueled by multimedia, the Internet and constantly appearing new applications. As advances in electronics are reaching natural limits, the only way to continue to sustain Moore’s law is by providing systems that perform many computations concurrently; hence the introduction of multicore architectures and grid computing.

In reality, most experts will readily concede that we do not know how to program concurrent applications correctly and efficiently. The programming models are complex and error-prone; concurrent systems hang, fall prey to “data races” and deadlocks, and fail to deliver the performance improvements that to which concurrent hardware should entitle us. This leads to widespread industry disappointment with multicore architectures.

This project proposes to bridge the gap between the available hardware potential for concurrent computing and the current support for developing concurrent software exploiting that potential by integrating a simple yet powerful model for concurrency with an established programming language. Devised for contract-enabled languages such as Eiffel, the simple and lean SCOOP model is the ideal candidate for this. SCOOP takes object-oriented programming as given, in a simple and pure form based on contracts which have proved highly successful in improving the dependability of sequential programs, and extends them in a minimal way to cover concurrency and distribution.

Intellectual merit. The integration of SCOOP with the current commercial version of the Eiffel programming language will include tool support in the EiffelStudio IDE. Having been investigated through simple prototypes so far, research will be required on SCOOP for this integration, in particular to mesh with independent recent advances in Eiffel. In addition, refinements will be proposed to relax the original SCOOP model: SCOOP has been designed to preclude data races at any cost, which it achieves by avoiding simultaneous executions on objects. This strong isolation model disregards certain sources for potential speedups, which is addressed in this project by proposing lock passing and sharing mechanisms. Another important contribution is deadlock detection support -- in the runtime (dynamic) as well as in the IDE (static).

These contributions will have an impact beyond the present context. Programming and specification languages based on axiomatic semantics have namely been among the most promising endeavors towards provably correct software. Their integration with concurrency is however incomplete at this point, due to the lack of formal frameworks which are also practical. The Secure Software Systems laboratory at Purdue University leading these research efforts gathers strong expertise in concurrent programming and contracts. Research results will be published.

Broader impact. The product will reduce development time for novel cycle-intensive applications for example in health care or finance. By the integration of a well-elaborated concurrency model into a popular programming language, the delivery of the computational power of multicore architectures to our society becomes imminent. In contrast, major industrial players are conducting research on new programming paradigms and languages for highly concurrent computing, which are however in premature states and have met mainly scepticism in industry.

Research and development efforts will be lead by Eiffel Software, securing its position in the market and furthering employment opportunities. Through a dual licensing model, Eiffel Software will be earning through product licenses and support; at the same time, by “open sourcing” it, other research institutions may contribute to the integration of contracts with concurrency. This also allows academic institutions to benefit from the outcome for teaching. In particular, the product will be tied in to an undergraduate class being created at Purdue University on concurrent programming with emphasis on multicore architectures, and in to the core Software Engineering class. Furthermore, undergraduate internships will be offered for students to participate in the research by designing benchmark suites to evaluate the performance of SCOOP programs.

Keywords. Software engineering, programming languages, object-oriented programming, concurrency, axiomatic semantics, Eiffel, Design by Contract

Topic and subtopic. Software and Services (SS) – 7. Tools that enable user-generated content