**PhD Project Proposal**

School of Electronics, Electrical Engineering and Computer Science

|  |
| --- |
| **Proposed Project Title: Automated Software Refactoring for Concurrency and other Qualities** |
| **Principal Supervisor: Des Greer Second Supervisor: Peter Kilpatrick** |
| **Project Description:**  In recent years there has been a steadily increasing amount of research into the use of Search Based Software Engineering (SBSE). SBSE is appropriate where there is a very large solution space for a given problem in the Software Engineering lifecycle. For example, it has been applied to such things as determining how to maximise test coverage and completeness, how to prioritise testing, how to maximise value and/or stakeholder satisfaction in software releases, how to allocate resources to a software project and how to design code to reduce coupling. As well as deterministic techniques like linear programming there are many classes of search algorithm that are applicable such as Simulated Annealing, Hill Climbing, Genetic Algorithms and many more.  This project will investigate such things as how to identify functional and non-functional quality attributes in existing code and to investigate if it is possible to specify these, set goals to improve them, measure the improvements and then to recommend refactoring actions or better still automatically refactor the code. With automatic refactoring, changes are made without human intervention and then measurements are made to find if improvements have been made, typically using quality metrics measured statically. In existing work at QUB, a many-objective approach has been employed**,** meaning that several [objective function](https://en.wikipedia.org/wiki/Loss_function)s can be optimized simultaneously. As you might expect this is not straightforward since some of the attributes that must be optimised may be in conflict or competition. Previous work concentrated on design quality and used a Multi Objective Genetic Algorithm. This PhD project is to extend on that work but to look at other objectives such as performance, testability, comprehensibility etc. One big opportunity is to automatically provide concurrency to improve better performance, efficiency, energy consumption, etc. Refactoring to parallelise code has been a long-standing aim but automatic refactoring towards an optimum would represent a significant step forward on existing approaches to retrofit parallelism to sequential code and is highly relevant to modern architectures.  The validation can be carried out be using open source software or with an industrial partner on live code.  Reading:   1. Danny Dig, John Marrero, and Michael D. Ernst. 2009. Refactoring sequential Java code for concurrency via concurrent libraries. In Proceedings of the 31st International Conference on Software Engineering (ICSE '09). IEEE Computer Society, Washington, DC, USA, 397-407. <https://ieeexplore.ieee.org/document/5070539> 2. Systematic Literature Review of Search Based Software Engineering: found here <http://www0.cs.ucl.ac.uk/staff/mharman/ACM-surveys-sbse.pdf> 3. A Survey of Search-Based Refactoring for Software Maintenance. Mohan, Michael; Greer, Desmond.   Journal of Software Engineering Research and Development, Vol. 6, No. 3, 2018. <https://jserd.springeropen.com/articles/10.1186/s40411-018-0046-4>   1. Greer, D., Ruhe, G., [Software Release Planning: An Evolutionary and Iterative Approach](http://www.cs.qub.ac.uk/~Des.Greer/greer%20ruhe%20IST.pdf), Journal Information and Software Technology, 46/4 pp. 243-253, 2004 - <http://www.cs.qub.ac.uk/~Des.Greer/greer%20ruhe%20IST.pdf> *as an example of SBSE* 2. <http://refactoring.com/catalog/>   Existing Work: https://github.com/mmohan01/MultiRefactor |
| **Contact details**  Supervisor Name: Dr Des Greer Tel: +44 (0)28 9097 4656  QUB Address: School of EEECS, CS Building Email: des.greer@qub.ac.uk |