THE ROYAL SOCIETY
CELEBRATING 350 YEARS

e-GAP²

Dr Miguel Wermelinger / Dr Miguel Goulao

International Exchanges Scheme - 2011/R2 (inc. RIA)

Title:	Dr
First Name:	Miguel
Surname:	Wermelinger
Preferred Name:	Michel
Address:	Computing Department The Open University Walton Hall
Town:	MILTON KEYNES
Postcode:	MK7 6AA
Country:	United Kingdom
Nationality:	**Nationality** Portuguese
Email Address:	m.a.wermelinger@open.ac.uk
Telephone (home):	
Telephone (mobile):	
Telephone (work):	01908 659539
Fax:	01908 652335

Applicant Career Summary

Statement of qualifications and career:

Qualification	Date
Keynote speaker, 7th Int'l Conf. on Quality of ICT (QUATIC)	2010
Steering committee chair, Intl. Workshop on Principles of Software Evolution (IWPSE)	since 2009
Module team chair of MSc module ' Managing the Software Enterprise'	since 2008
Director of research of the Computing Department	2008
Supervisor of 6 PhD and 6 Msc students	since 2005
General Chair, European Softw. Eng. Conf. / ACM Symp. on Foundations of Softw. Eng. (ESEC/FSE)	2005
Programme committee co-chair, Fundamental Approaches to Softw. Eng. (FASE)	2004
Member of the Board, European Association for Software Science and Technology	2002 - 2006
Member, network 'Research Links to Explore and Advance Software Evolution', European Science Foundation	2002 - 2005
Consultant on secondment from university, ATX Software SA, Portugal	2001 - 2003
External examiner of 6 PhD and 1 MSc students in the UK, Spain, Belgium and Portugal	since 2000
PI, project 'Formal Approach to Software Architecture', Portuguese Science and Technology Foundation	2000 - 2004
Assistant Professor, Computing Department, New University of Lisbon	1999 - 2004

Field of Specialisation: Software engineering

Publications: The following papers are related to this project and available from http://oro.open.ac.uk/view/person/mw4687.html

[ICSM'11a] S. Butler, M. Wermelinger, Y. Yu, Yijun and H. Sharp (2011). Mining Java class naming conventions. Proceedings 27th International Conference on Software Maintenance, pp. 93-102, IEEE.

[ICSM'11b] A. González, R. Therón, F. García-Peñalvo, M. Wermelinger, and Y. Yu (2011). Maleku: an evolutionary visual software analytics tool for providing insights into software evolution. Proceedings 27th International Conference on Software Maintenance, pp. 594-597, IEEE.

[GTSE'11] M. Wermelinger and Y. Yu (2011). Some issues in the 'archaeology' of software evolution. In: Generative and Transformational Techniques in Software Engineering III. Lecture Notes in Computer Science vol. 6491, pp. 426-445, Springer.

[ESEJ'11] M. Wermelinger, Y. Yu, A. Lozano, and A. Capiluppi (2011). Assessing architectural evolution: a case study. Empirical Software Engineering, 16(5), pp. 623–666.

[ICSE'09] M. Wermelinger, Y. Yu, and M. Strohmaier (2009). Using formal concept analysis to construct and visualise hierarchies of socio-technical relations. Proceedings of the 31st International Conference on Software Engineering, companion volume, pp.327-330, IEEE.

Subject: Subject Group 04: Engineering, technology, instrumentation, materials science, experimental fluid dynamics / Computer engineering (including software)

Present Research:	Dr Wermelinger's research centres on software evolution and maintenance. He's interested in getting a deeper understanding of how software evolves and in developing techniques and tools to help managers and developers analyse their project's evolution and avoid future maintenance problems. Methodologically, his research empirically explores, by mining open source software repositories [GTSE'11], the links between the structure, evolution, and quality of software and its developers, to uncover factors impacting on software evolution and maintenance. Research questions include: do badly structured names reflect poor design quality [ICSM'11a]? How to support visual analytics of socio-technical relations between software structure and team structure [ICSM'11b, ICSE'09]? Are design principles relevant for evolving successful architectures [ESEJ'11]? Answers help point to practices (e.g. naming conventions and team organization) for sustainable evolution of long-lived software systems.
Present Position:	Senior Lecturer in Computing
Present Employer:	The Open University
Present Department:	Computing Department
Present Position Start Date:	01/09/2004
PhD Awarded Date:	16/12/1999
PhD Expected Date:	

Co-Applicant Personal Details

Title: Dr

First Name: Miguel

Surname: Goulão

Preferred Name:

Address Line 1: Departamento de Informatica

Address Line 2: Faculdade de Ciencias e Tecnologia

Address Line 3: Universidade Nova de Lisboa

Address Line 4:

Address Line 5:

Town: Caparica

Postcode: 2829-516 CAPARICA

Country: Portugal

Nationality:

Nationality
Portuguese

Email Address: miguel.goulao@di.fct.unl.pt

Telephone (home):

Telephone (mobile):

Telephone (work): +351 212948536

Fax: +351 212948541

Co-Applicant Career Summary

Statement of qualifications and career:

Qualification	Date
Teaching Assistant at FCT/UNL	2000-2008
Assistant Professor at Universidade Nova de Lisboa	Since 2008
Member of the QUATIC Conference Series Steering Committe	Since 2001
Member of the Experimental Software Engineering Research NETwork(ESERNET)	2003
Lecturer at the Domain-Specific Modeling - Theory and Practice International Summer School	2010-2011
Member of the Portuguese Committee for the Quality in Information and Communications Technology	Since 2001
Program Co-Chair of the QUATIC Conference	2004 and 2010
Co-Supervisor of 1 PhD Student (Ongoing), Supervisor of 7 MSc Students (3 finished, 2 waiting for public trials, 2 ongoing)	Since 2009

Field of Specialisation: Software Engineering

Publications:	[PLATEAU2011] Ankica Barisic, Vasco Amaral, Miguel Goulão and Bruno Barroca. "Quality in Use of Domain Specific Language: a Case Study", Proceedings of the Workshop on Evaluation and Usability of Programming Languages and Tools (PLATEAU 2011), held at Splash 2011, Portland, Oregon, USA, October, 2011. [CIbSE2010] Pedro Gabriel, Miguel Goulão and Vasco Amaral. "Do Software Languages Engineers Evaluate their Languages?". Proceedings of the XIII Congreso Iberoamericano en "Software Engineering" (CIbSE'2010), Universidad del Azuay, ISBN-978-9978-325-10-0, Cuenca, Ecuador, April 2010. [SQM2007]Miguel Goulão and Fernando Brito e Abreu, "An overview of metrics-based approaches to support software components reusability assessment", Book Chapter in Software Quality Measurement: Concepts and Approaches, ICFAI Books, 2007 [QUATIC2007] Miguel Goulão and Fernando Brito e Abreu. "Modeling the Experimental Software Engineering Process". 6th International Conference on the Quality of Information and Communications Technology (QUATIC 2007), IEEE Computer Society, Lisbon, Portugal, 2007. [QoSA2005] Miguel Goulão and Fernando Brito e Abreu. "Formal Definition of Metrics Upon the CORBA Component Model". In Quality of Software Architectures and Software Quality, First International Conference on the Quality of Software Architectures, QoSA'2005 and Second International Workshop on Software Quality, SOQUA 2005, LNCS 3712, Springer, Net.Object Days 2005, Erfurt, Germany, September, 2005.
Subject:	Subject Group 04: Engineering, technology, instrumentation, materials science, experimental fluid dynamics / Computer engineering (including software)
Present Research:	Miguel Goulão's research focuses on the development and application of Experimental Software Engineering techniques to the evidence-based validation of Software Engineering claims. Recently, Miguel has been researching on the impact of software language properties (particularly with Aspect-Oriented and Domain-Specific Languages) in the productivity of software developers and the quality of the outcome of their work. He is particularly interested in how such evaluation can guide the evolution of those languages, from the perspective of a languages engineer. Another research stream in which Miguel is working is on software evolution, through mining of software repositories, in order to find evindences concerning the past evolution of software products and their corresponding processes (e.g. concerning the evolution of change requests), as well as the construction and validation of prediction models of product and process properties, based on the information mined from those repositories.
Present Position:	Assistant Professor of Informatics Member of the CITI Research Center
Present Employer:	Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
Present Department:	Department of Informatics
Present Position Start Date:	17/12/2008
PhD Awarded Date:	17/12/2008
PhD Expected Date:	

Proposal

Subject:	Subject Group 04: Engineering, technology, instrumentation, materials science, experimental fluid dynamics / Computer engineering (including software)
Project Title:	Quo vadis, software project?

Research Aims: The scientific aim of this project is to further our understanding of how software projects evolve. In particular, we wish to measure and visualize a project's evolution so that we can detect or predict trends that might lead to problems, e.g. if the software is growing too much too fast or if the rate at which defect reports are handled is decreasing and becoming unsustainable. Such alerts could help project managers and developers take corrective action in advance, avoiding escalating development costs. The novelty of this research project is the combination of radically different views on the same project (including changes to code, developer turnover, and defect report handling) with the aim of obtaining a rich answer to the question where the project is heading to: success or trouble?

Start Date: 01/03/2012

End Date: 28/02/2013

Research proposal:

There has been much research into software evolution since the seminal work by Manny Lehman in the 1970s. However, existing work, including our own, is often fragmented into specific problems and approaches, which are tried out on different software projects. An approach to a multi-faceted look at the evolution of the same project is missing, and that is the nature of this project.

The purpose of the project is to see how the partners' previous software evolution analysis approaches might be best extended and combined in order to get added value, i.e. to obtain a richer understanding of the past evolution and its trends and a better indication of potential problems than just by applying each approach separately.

More specifically, we will apply the formal concept analysis and visualization techniques of [ICSE'09] and the metrics and clustering techniques of [ESEJ'11] to different socio-technical relations, e.g. to establish who has been working together on the same parts of the software and whether there are unconnected 'islands' of sub-teams. Using two different techniques to analyse the same data for the same purpose allows us on the one hand to compare the pros and cons of each technique and on the other to get multiple perspectives on the same phenomenon (collaboration within the team) for a better understanding of it. As a further example of how our previous work can be extended and applied in new ways, we plan to add information about who developed which part of the software to the naming analysis of [ICSM'11a]. This allows us to see whether deviant naming conventions, which can make code understanding and communication among developers harder, are specific to certain developers, who thus could benefit from mentoring, or to certain parts of the code, which should be looked into or documented as special cases.

Taking measurements from all these analyses (e.g. the percentage of identifier names not conforming to standard patterns, the size of clusters of developers, the average time to handle a defect report) over several releases of the same software system, and applying machine learning techniques to the obtained time-series [??ref needed], will allow us to detect any trends and extrapolate them into the near future, thereby aiming to answer the question in the title of this research project. By then repeating the whole process for several software projects we can gain confidence whether the socio-technical relations we are extracting and the metrics we are calculating from the software project's repository are indeed good trend indicators.

Modern software projects use sophisticated repositories that keep track of who added which part of the code when, who reported or handled which defect when, etc. Therefore all the analysis and visualization techniques to be used in this research can be fully automated; otherwise the project would not be feasible within the proposed time frame. The repository mining and analysis tools we built in our previous work were developed with particular research questions and particular artefacts (e.g. only defect reports) in mind. The mutual visits of this project are needed to work together on transforming our heterogeneous set of independent tools into a cohesive interoperable toolset. This requires explaining to each other in depth how our tools work and their built-in assumptions, and then discussing and designing together the overall tool chain architecture and input/output data formats, and packaging and documenting the resulting toolset so that other researchers can use it. The toolset will be built incrementally and iteratively, with our tools being integrated one by one, tried out on a software repository, and the results feeding into the next integration step. To best fit this plan, we will have several short visits along the year.

Resources required:	Dr Yijun Yu, Senior Lecturer in Computing at the Open University, has been a research collaborator of Dr Wermelinger for the past years. He has wide experience in automated software engineering, a sub-field which seeks to improve the engineering of software via automated tool support. For this project, his input will be essential to make the tools of both project partners inter-operate.
Participants:	Simon Butler is a PhD student supervised by Drs Wermelinger and Yu. He is working on vocabulary usage in source code (i.e. in the software's textual instructions as written by programmers), in particular whether there are any inconsistencies in such usage with respect to the software design's structure and if such inconsistencies could pose problems in understanding and maintaining the code. Mr Butler's expertise will be useful to assess e.g. whether a project's vocabulary changes over time and differs among developers, as this might point to potential communication and coordination problems between sub-teams.
Comply with Policy on use of Animals:	Not applicable
Benefits to individuals/institutions:	Both partners will directly benefit from mutual knowledge transfer and from the assets (publications, tools and datasets) resulting from this project. The collaboration and the assets not only set the foundation for future common bids, we also plan to incorporate the assets and the obtained insights about software evolution into our teaching at each institution. The international collaboration will also be a good experience for our PhD student.
Benefits to UK:	Studies state that software maintenance (i.e. modifying existing software) accounts for the majority of all software development costs. This research, by developing techniques to monitor and forecast a software project’s trends, can have economic impact. We will deposit the outputs on the Open University’s open access repository, one of the UK’s largest and most visited, for the benefit of academia and industry. Project results will be presented at practitioners’ conferences, like BCS SPA.
Benefits to Overseas Country:	Like the UK and any other modern society, Portugal is reliant on software to provide services across all domains: health, communications, finances, etc. Given that the way software is developed is not country-specific, the same benefits as for the UK apply.
Lay Report:	

From smartphones to airplanes, from washing machines to shopping websites, there is hardly any device or service in modern society that doesn't include or depend on one or more computers. Health care, finance, transport, communication and other essential activities would collapse if all computers stopped working. While that is an unlikely, isolated IT failures that lead to all sorts of problems abound (see www.risks.org).

Simply put, computers are made of hardware, the electronics, and software, the instructions that do calculations, process inputs, output text and images, etc. Software is among the most fascinating artefacts invented by humankind: it's executable 'thought stuff'. It can't be touched (only the medium on which software is stored has physical embodiment) and yet, being a text written by programmers to give instructions to the computer, it has a structure. Software systems are also among the most complex systems built by humans. Operating systems like Windows and financial software in banks, to give two examples, comprise millions of lines of code, written over many years, sometimes decades, by hundreds or thousands of programmers. Given the effort required, it's understandable that most software today is not written from scratch but by evolving existing software, i.e. by adding new functionality, correcting existing defects, and adapting to new technology.

Studying how software evolves, and developing techniques that help programmers understand existing software they have not written, and make changes to it without leading to new defects, is therefore a scientifically and practically relevant, challenging, and interesting research area, which I work in. Most large software projects use specialized tools and repositories to keep track of the various versions of the software, who added which lines of code when, who reported or fixed which defect when. Hence, the work done in this research area is akin to archaeology, namely finding in the repositories those traces of past activity that can point to evolution problems.

It's often difficult for managers and programmers to keep an overview of their project, because the system being developed is large, because people come and go over time, and because communication is difficult among a large team, often distributed over several locations. The aim of the research we are proposing is to uncover potentially problematic trends and present them, so that managers and programmers can take corrective action. Examples of problematic trends are: Is the software system growing too much too fast, and becoming too complex? Is the structure of the software diverging from the structure of the team (making work allocation harder)? Is there a lot of programmer turnover (meaning that past experience gets lost)? Are programmers using uncommon or abbreviated words in their code (making it harder to understand for others)?

To present such trends, we will draw on techniques and tools we have developed previously and combine them to get a richer perspective on the software system's past, and then apply machine-learning techniques that detect trends and forecast their future, if nothing is done to the contrary. The goal is to learn from history to answer early enough the question: Where are you heading to, software project, success or failure?

Financial Details

Financial Details:

Year	Payment type	Justification	Amount Requested
Year 1	Travel International	Drs Wermelinger & Yu, April 2012, Lisbon, £400; Dr Wermelinger, July 2012, Lisbon, £200; Dr Goulão, September 2012, Milton Keynes, £200; Dr Wermelinger & Mr Butler, December 2012, Lisbon, £400; Dr Goulão, February 2013, Milton Keynes, £200	1,400.00
Year 1	Subsistence	Drs Wermelinger & Yu, April 2012, Lisbon, £1044; Dr Wermelinger, July 2012, Lisbon, £522; Dr Goulão, September 2012, Milton Keynes, £1200; Dr Wermelinger & Mr Butler, December 2012, Lisbon, £1044; Dr Goulão, February 2013, Milton Keynes, £600	4,410.00
Year 1	Research Costs	NA	0.00
Total			5,810.00

Sum requested from the Royal Society: 5810.00

Grant Tenure: 1 year