Curriculum Vitae for Ola Skavhaug

Personal information

Address:	Aslakveien 31, 0753 Oslo	E-mail:	ola@xal.no
		Phone:	+47 926 12 490
Born:	10.04.1974	Nationality:	Norwegian

Summary

I am a capable software developer, researcher, and project leader with 14 years professional experience. My career in Norway's highest ranked ICT research institution, Simula Research Laboratory and its subsidiaries Kalkulo and Simula Innovation has given me a broad technical and managerial background. My main areas of technical expertise are mathematical and numerical software development, algorithm development, advanced scripting with modern scripting languages, parallel programming, software testing and deployment, library design, and scientific visualization.

Through my work, I have obtained an extensive set of skills that allows me to understand and solve challenges in collaboration with other experts. Today, industrial challenges are often multidisciplinary, and involve competences from several fields at once. Hence, to successfully deliver results, communication is key. Modern, agile software development methods facilitate this, and through my work the last ten years, this has been my modus operandi.

Technical Skills

Languages	C, C++, Java, Fortran, Python, Javascript, Perl, PHP, Bash, Tcl/tk, Matlab, Sql, LaTeX, HTML, XML
Frameworks	Numpy, SciPy, Matplotlib, MPI, BSP, Swig, Boost, Stl, VTK, FEniCS, PETSc, SLEPc, Diffpack
Tools	Subversion, Mercurial, Git, cvs, Make, CMake, Scons, GCC, Autoconf, Linux, css, MySQL

Education

2004	Dr. Scient in Computer Science, The Faculty of Mathematics and Natural Sciences, University of Oslo. Thesis' title: "Numerical Methods and Software with Applications in Computational Finance".
1998	Cand. Scient in Computer Science, Department of Informatics, University of Oslo

Professional Experience

2013 –	Consultant, Expert Analytics
2011 – 2013	Innovation manager at Simula Innovation
2010 – 2011	Senior Scientific Programmer at Kalkulo AS
2007 – 2010	Research Scientist and head of the computational middleware software activity at the Centre of Biomedical Computing (CBC) at Simula Research Laboratory
2005 – 2007	Research Scientist and head of the project Software for PDEs"at Simula Research Laboratory
2004 – 2005	IT-manager, Simula Research Laboratory
2004	System Administrator, Simula Research Laboratory
2004 – 2010	20% Associate Professor, Department of Informatics, University of Oslo
2001 – 2004	Ph.D. student at the Simula Research Laboratory
2000 – 2004	20% Teaching Position at the Department of Informatics, University of Oslo
2000 – 2001	Ph.D. student at the Department of Informatics, University of Oslo

Other Experience

2009 – 2013	Employee representative in the board of directors, Simula Research Laboratory
2005 – 2006	Board member, Øraker Barnehage AS

Languages

Norwegian	Mother tongue
English	Fluent
German	Basic

Personal Skills

Management	Motivate and lead experts and PhD students, define and implement ned projects, facilitate communication in informal surroundings to break up the work day.
Applied mathematics	Analyze, develop and implement complex algorithms in applied sciences, while balancing constraints like flexibility and efficiency. Short, agile development cycles with discussions and feedback from problem owners.

Some interests and hobbies

Physical	Telemark skiing, running, biking, climbing.
Gastronomical	Beer brewing, sausage making.
Other	Reading, traveling, trekking, expeditions.

Extended descriptions of select projects

Activity mCASH backend development
Role Senior Python Developer
Staffing 12-15 Python developers
Description In this project, I am working on most parts of the backend of a new mobile payment system. This includes financial transaction handling, the internal bank implementation, messages emitted through various protocols based on recipients, OpenID Connect scopes implementation and payment for these, web handlers for endpoints, and Datastore transaction in the Google app engine, all in Python. The development is test driven, with tests covering close to 100 percent of the code base, and follows the Scrum agile method.
Tools Python, Google app engine, Git, buildout, nose tests, webapp2, Jinja2, OAuthLib, JSON, html, javascript, jQuery, Pusher

---

Activity Software for PDEs, Simula Research Laboratory
Role Leader, scientist and software developer
Staffing 6-8 scientists, developers and PhD students for two years
Description Under my responsibility the project defined and developed novel software frameworks for advanced computer simulation and visualization and delivered excellent scientific results. Simula Research Laboratory applied and was awarded a Centre of Excellence by the Research Council of Norway in 2007, where the activity of the Software for PDEsbecame a central component.

Activity Python Computing Components
Role Main developer
Description PyCC is a modern and efficient scripting framework that is used to solve differential equations modelling the electrical activity in the human heart – the so called bidomain equations. The complexity of the problem, and the use of the tool to conduct research, required both flexibility and efficiency. To meet these needs we implemented a high level scripting interface in Python for flexibility, and migrated bottlenecks to low-level extension modules implemented in C/C++ and Fortran. Central activities were library design, interface building strategies, cross language techniques, code generation, and third party software integration.
Tools C/C++, Fortran, Python, Swig, MPI, Subversion, Scons, PETSc, FEniCS, Hypre, BoomerAMG, Diffpack, GNU Compiler Collection and Debugger, Valgrind

---

Activity Viper
Role Main developer
Description Viper is a lightweight runtime visualization framework for scientific data and results. It grants the underlying visualization library, VTK, direct access to the simulation result, thereby minimizing memory copies for efficiency. Viper can visualize both scalar and vector data, as well as wireframe geometries (meshes).
Tools C/C++, VTK, Python

Activity	Gotran
Role	Main developer
Description	Systems of ordinary differential equations are often complex, and implementing these in a numerically efficient way is both time consuming and error prone. To remedy this, I have implemented Gotran – a compiler that takes ODEs described in a high level DSL (domain specific language) and generates highly specific and numerically efficient C/C++ code. By building on top of another software project I have implemented, Swiginac, Gotran utilizes symbolic manipulation during several of the code transformations to reduce the number of floating point operations needed to evaluate the ODE systems during simulation.
Tools	C/C++, Python, Swiginac, Swig

Activity	Swiginac - extending Python with symbolic mathematics
Role	Main developer
Staffing	Open source project with several contributors
Description	Swiginac is a symbolic mathematics module for Python. It is built by exposing GiNaC, a symbolic manipulation library written in C++ to Python with Swig. The efficiency of the underlying C++ library makes Swiginac one of the fastest technologies in its class in Python, and the possibilities of writing expressions in various ways makes Swiginac well suited for code generation purposes. Swiginac was developed as a side project during my PhD, in order to make a system for automatic code verification of numerical simulators.
Tools	C++, Stl, Python, Swig, Distutils, Subversion, Make

Activity	Department of Informatics, University of Oslo
Role	Associate Professor
Description	Over a period of ten years, I have given lectures in two popular courses at the university, teaching students how to apply high-level computer languages for advanced problem solving.
Tools	Python, Perl, Bash/Sh, Tcl/tk, CGI

Activity	Famms - automatic code verification for PDEs
Role	Main developer
Description	Standard PDE problems can be formulated as $F(u) = 0$, where F is a possibly non–linear system of differential equations. The task is then to find the unknown u. By selecting a manufactured, analytical solution instead, called v, we can compute $b = F(v)$. Then by defining $G = F(v) - b$, we again obtain a standard problem on the form $G(v) = 0$. Forgetting that we know v, we can try to solve the last equation to see if the numerical simulator is working as expected. The method above is commonly referred to as the method of manufactures solutions, and Famms, the software system I implemented, automates this process by calculating both b and the perturbed problem G with minimal effort.
Tools	C/C++, Python, Swig, Swiginac, Diffpack, FEniCS, PyCC.

Activity	Biomedical computing
Role	Developer and project leader
Staffing	Two developers
Description	In this project, a California based software company in biomedicine wanted to incorporate some of the technology I had developed into their commercial code to strengthen the finite element analysis and visualization capabilities of their software. Over a period of six months, we successfully integrated the components into their code, such that they could us PyCC and Viper.

Activity	Symphonical
Role	Main developer
Staffing	Project leader and two developers
Description	Symphonical is a web-based collaboration tool. Initially it was conceived as a tool for running agile software development projects based on the metaphor of post-it notes on a virtual wall, a scope that since has been widened. We created the first prototype of the system, in 2005, before it was spun out as a company and developed further by others.
Tools	PHP, Mysql