

## HPC Software Development Call 2010/11

### Adaptive Multi-Resolution Massively-Multicore Hybrid Dynamics

Adrian Mulholland, Christopher Woods, Simon McIntosh-Smith, University of Bristol

#### Part I. Previous Track Record

**Adrian Mulholland (AJM)** is an EPSRC Leadership Fellow and Professor of Chemistry, and a founder member of the Centre for Computational Chemistry, University of Bristol. He is the current Chair of CCPB (the collaborative computational project for biomolecular simulation), was Chair of the Molecular Graphics and Modelling Society (2004-2008) and is a member of the Editorial Advisory Board of Biochemistry (ACS). He was a member of the UK High End Computing Strategic Framework Working Group (2005-6) that wrote the current UK strategic framework for HPC. His research focuses on the investigation of biomolecular structure, function and dynamics by computer simulation. A central theme in his research is the development and application of combined quantum mechanics/molecular mechanics (QM/MM) techniques, in particular for the investigation of enzyme-catalysed reactions. He is internationally recognised in this field (22 invited conference lectures and 19 invited seminars in the last 5 years). He has a strong interest in the application of high performance computing, particularly for biomolecular simulations. His research group consists of 3 PDRAs and 6 PhD students, with current funding from EPSRC, BBSRC, NIH(US), the Thai Government and Pfizer. He has extensive experience in the planning and management of collaborative research projects. He actively collaborates with a number of experimental biochemistry research groups in the UK and abroad. His research has been highlighted in Science, The Economist, The Guardian, Chemical and Engineering News, Chemistry World and elsewhere.

**Simon McIntosh-Smith (SMS)** is a world-expert in heterogeneous many-core computing, with expertise in the design of energy-efficient, parallel computer architectures and in massively parallel, fault tolerant software development. He is the current Chair of the International Many-core and Reconfigurable Supercomputing Conference, MRSC. He is an invited member of the International Exascale Software Project, IESP, which is a global collaboration between major high performance computing users in the US, Europe and Asia. He has been a member of the IEEE, ACM SuperComputing programme committee since 2010. He is internationally respected in his field, being a regularly invited keynote speaker on heterogeneous many-core computing and energy efficient HPC.

SMS is the inventor on eight patents in novel parallel hardware and software. Prior to joining the University of Bristol in 2009, he was co-founder of ClearSpeed Technology in 2002, where, as Vice President of Architecture and Applications, he led the development of the world's first many-core, high performance computing accelerator, the CSX-line of parallel processors. The CSX600 (2005) and CSX700 (2008) processor designs held world records in energy efficiency upon their introduction.

While at ClearSpeed SMS invented the heterogeneous, multi-/many-core math library, CSXL, the world's first BLAS and LAPACK library that could execute simultaneously on a many-core parallel processor and a multi-core host. CSXL enabled the first modern accelerated supercomputer, Tokyo Tech's TSUBAME, which as a result of CSXL acceleration and ClearSpeed CSX600 processors became the fastest supercomputer in Asia in 2006 and a top ten system world-wide, reaching 56 TFLOPS. While at ClearSpeed he was responsible for two of the first ports of Molecular Dynamics (MD) codes to accelerators (GROMACS and Amber), and for one of the first accelerated, MD-based drug docking codes (BUDE). He also supervised the first port of a quantum mechanics code to massively parallel accelerators (Molpro). Prior to founding ClearSpeed, SMS co-developed the world's first General-Purpose Graphical Processing Unit (GPGPU): Pixelfusion's 1,536-core F150 GPU, developed in 2000.

In total SMS has fifteen years of industrial experience in high performance and parallel computing working for companies including Inmos, STMicroelectronics, Pixelfusion and ClearSpeed (co-founder). During this time in industry he gained considerable experience developing commercial-grade software, leading software teams of up to 25 engineers, managing multi-million pound software development budgets, and successfully delivering high-quality software to customers for more than a decade. This level of success depended on a mix of innovation and the rigorous application of software engineering industry best practise.

SMS has designed a new High Performance Computing unit for undergraduate Computer Science students, starting in Oct. 2010, one of the first academic programmes in the world to teach the new OpenCL parallel computing language.

**Christopher Woods (CJW)** is an experienced developer of scientific software. During his PhD and PDRA with Prof. J. W. Essex (University of Southampton) he developed novel methods that significantly improved the reliability and accuracy of calculations of relative free energies from atomistic simulations, and developed new Monte Carlo simulation software (ProtoMS, <http://protoms.org>) that is now used by the pharmaceutical industry and several research groups worldwide. During postdoctoral research with AJM he has developed new simulation methods and software to allow efficient Monte Carlo sampling of molecules represented using QM/MM methods. CJW has an extensive

background and experience of molecular simulation methodology. He has developed novel hybrid dynamics algorithms and software, and has written an invited Royal Society of Chemistry (RSC) review of multiscale simulation methods (“Multiscale modelling of biological systems”, *RSC Special Periodicals Review*, 2008) CJW is also an accomplished programmer. He has written a range of computational chemistry software, using a variety of C++, Fortran, Perl and Python, and has experience of writing HPC programs that utilise OpenMP and MPI. CJW has been employed on a previous successful EPSRC software development project; EP/F010516/1 in which he ported the DFT component of the Molpro quantum chemistry program to numerical accelerators (ClearSpeed) and multi-core processors (via OpenMP). In addition, CJW is the developer of an advanced molecular simulation software framework (Sire, <http://www.siremol.org>). This new framework provides libraries of molecular simulation building blocks, written in C++, which are wrapped up and exposed to a user as Python classes. These components provide a foundation that allows for the rapid development of novel HPC molecular simulation software. CJW has been awarded EPSRC funding (EP/G042853/1) and national (HPCx) and international (DEISA) computing time during which he ported the software to a range of HPC resources. These ports are maintained and available from the publically accessible website and source code repository. At Bristol CJW has developed self-guided, web-based workshops in Perl, Python and OpenMP, and with the CCPB he has developed and run workshops on QM/MM ([http://www.ccpb.ac.uk/workshop/qm\\_mm](http://www.ccpb.ac.uk/workshop/qm_mm)) and free energy methods ([http://www.ccpb.ac.uk/workshop/free\\_energy](http://www.ccpb.ac.uk/workshop/free_energy)), the second of which was based around his ProtoMS simulator software. In addition, CJW developed and taught a self-guided, web-based workshop teaching simulation algorithms as part of a European Molecular Biology Organization (EMBO) practical course in biomolecular simulation (<http://cwp.embo.org/pc10-24/>). CJW’s research focus is developing novel methodologies for use in the pharmaceutical industry, with which he has close contacts (e.g. presenting invited talks at UCB, GSK, Sanofi-Aventis and Evotech, and having regular six-monthly meetings with UCB). These close contacts give CJW deep knowledge of the methods and software that are needed to ensure computational chemistry remains relevant to rational drug design. Related to this proposal, CJW has recently developed a method that allows absolute protein-ligand binding free energies to be calculated from first principles in a single, distributed ensemble simulation (Woods, Tyka, Sessions and Mulholland, *J. Chem. Phys.*, 2010, submitted). This method would allow the free energy cost of protein conformational change to be accounted for directly when calculating absolute protein-ligand binding affinities, *if* the sampling of protein conformational change could be sped up by several orders-of-magnitude.

**University of Bristol (UOB).** The research will be conducted as a joint collaboration between the School of Chemistry and the Department of Computer Science. The School of Chemistry was ranked 4<sup>th</sup> in the UK in the most recent RAE, and boasts a dynamic Centre for Computational Chemistry (CCC). The Centre currently consists of 7 members of academic staff, a Computer Support Officer, 6 research staff and 15 PhD research students, in purpose-designed space. It will provide an outstanding environment for the proposed research, excellent computational resources, and broad expertise in computational chemistry. The School of Chemistry continues to attract substantial investment from a variety of sources, prominently including the Bristol ChemLabS Centre for Excellence in Teaching and Learning, funded by a HEFCE grant of £4.5m, supplemented by £16.3m capital investment in the project by the University. The Department of Computer Science at the University of Bristol is home to research groups specialising in High Performance Computing, Intelligent Systems, Computer Vision & Image Processing, Cryptography, Personal Systems, and Systems Architecture & Design. Funding for the department’s research comes from a variety of sources including EPSRC, ESRC, AHRC, SSRC, MRC, EU, DTI and industry sponsors, and its currently active research grant portfolio is valued at over £11m. The department was ranked fourth in the country in the recent Complete University Guide and currently has 36 academic staff, 25 research staff and 73 PhD students.

Bristol is also home to Blue Crystal, a bespoke 37-teraflop high-performance computing facility coupled to a 1 PetaByte storage system, shared by scientists and engineers across the University. Blue Crystal is managed by the Advanced Computing Research Centre (ACRC), a dedicated team of four staff who design and support Bristol's World class HPC systems, and vigorously promote the use of HPC across the university's academic community. As a result Bristol boasts a vibrant and rapidly growing multi-disciplinary HPC community, with 158 active HPC research projects involving 309 researchers, representing a research portfolio of over £11m and rising. ACRC has a track record of keeping Bristol at the forefront of novel HPC technologies, including introducing the first heterogeneous many-core accelerators from ClearSpeed in 2006, and more recently GPU clusters. The latter are likely to feed into the design of the next version of Blue Crystal, resulting in Petascale performance in just a few years' time.

## **Part II: Proposed Research and its Content**

We propose to develop highly scalable software that will exploit next generation, heterogeneous, massively parallel architectures to deliver orders-of-magnitude performance increases for conformational sampling in molecular simulations. The software will scale, as it will use a **distributed, adaptive, fault-tolerant and energy-aware, work packet queuing system**. This, coupled with a **multi-resolution hybrid dynamics algorithm**, will allow for the sampling of conformational changes that had hereto been inaccessible to current molecular simulation codes. The software will be applied initially to accelerate the sampling of protein conformational change within the types of simulations used in the pharmaceutical industry, however, it will also be generally applicable to simulations of any condensed phase molecular system, e.g. to model the separation of carbon nanotubes in solutions of amphiphilic surfactants.

### **A: Background**

*Timeliness: Why is this needed now?*

Future applications of rational drug design will depend critically on the ability to model protein conformational change and protein flexibility. Previous successful applications of computational methods in rational drug design targeted proteins that had small, well-defined binding pockets, in proteins that were either relatively rigid, or changed little upon drug binding. Increasingly, medicinally interesting protein targets have open and flexible binding sites. To understand binding, computational models have to be able to predict how these sites will change shape upon association with the drug. Coupled to this, a new generation of drugs are being developed that target the interactions between protein surfaces (protein-protein interaction inhibitors). In these cases, the binding site is extremely flexible, as it is formed between two (or more) proteins. Existing molecular modelling algorithms and software are incapable of stepping up to the challenge of modelling highly flexible proteins. New software and new algorithms are needed urgently to ensure that computational science continues to play an important role in the pharmaceutical industry.

*Timeliness: What makes this problem solvable now?*

Three recent major developments mean that we are now in a position to solve the challenge of efficiently accounting for protein flexibility when predicting binding affinities, and when modelling protein conformational change;

- (1) Numerical accelerators, which provide teraflops of computational power via many-core processors, have matured and are now widely and cheaply available. Standardised methods of programming these processors have been developed (e.g. OpenCL). We are amongst the earliest pioneers of many-core processors, designing and programming architectures from ClearSpeed, NVIDIA and AMD since 2002. Our experience has shown us that we can obtain orders of magnitude real-world speed-ups of simulations if we develop new algorithms and new software that specifically target these many-core architectures.
- (2) We have developed a new molecular simulation framework, called Sire[1], which provides software building blocks that allow for the rapid prototyping and development of new molecular modelling programs, and uses a distributed work packet queuing system that enables the development of adaptive, fault-tolerant software. Sire provides libraries of robust classes that provide an easy environment in which to write distributed multi-core modelling programs that can take full advantage of HPC supercomputer clusters.
- (3) We have designed a new **multi-resolution hybrid dynamics algorithm** that will allow for the simulation of molecular dynamics to be broken into two parts: a near-field, atomistic part, and a far-field, coarse grain part. The near-field part is used to model the interactions between neighbouring molecules, using traditional atomistic potentials, and uses Monte Carlo (MC)[2] to model the local dynamics of individual atoms. The far-field part models the remaining molecular interactions using a coarse-grain (beaded) potential, and uses hybrid rigid-body dynamics[3,4] to model global dynamics (e.g. large-scale protein conformational change). This multi-resolution split of both the dynamics and the molecular interactions makes the algorithm ideally suited to heterogeneous computing platforms such as HPC clusters equipped with numerical accelerators.

### **B: Research Hypothesis**

*Uniqueness: How is the proposed software different?*

The proposed multi-resolution hybrid dynamics program will be adaptive, fault-tolerant, energy-aware and massively parallel. The software will break the simulation into packets of work, which can be distributed across all available resources via a **distributed, adaptive, fault tolerant and energy-aware work packet queuing system**. For example, if there were many accelerators available, the work packet queuing system would dynamically choose to process all of the coarse-grain, far-field interactions using those accelerators. Alternatively, if few accelerators are available, or they are busy with other parts of the simulation, then the work packet queuing system would automatically re-route these packets for slower computation using traditional multi-core processors. The software and algorithms will be designed to

be adaptive and fault-tolerant. This is because they will be based on MC, and also because, like Sire[1], if the queuing system detects any failure in a work packet, it can cancel and resubmit that work packet to an alternative resource. In addition, the work packet queuing system will be developed to include a “health-score” for each work packet, which could be measured and used as a means of both recovering viable results from a potentially “ill” work packet, and for logging nodes that are failing and that should be avoided. The work packet system will also be developed to be energy-aware, as the historic energy cost of processing a work packet will be measured, and then used by the distributed work queues to decide to which resource the work packet is allocated. For example, if the results of the simulation were not needed immediately, then the work packet could be diverted from an accelerator, and instead run using low-power processors (e.g. clusters of Intel Atoms), or deferred until the CO<sub>2</sub> cost of energy production was low. This would give the simulator the choice of minimising the total simulation runtime or minimising the total energy and CO<sub>2</sub> cost.

*Outputs: What software will this project deliver?*

- (1) An open source multi-resolution hybrid dynamics program, used to accelerate sampling of molecular systems.
- (2) A distributed, adaptive, energy-aware, fault tolerant work packet queuing system, which will be fully documented and released as a separate open source library for use by other HPC software.
- (3) A fully documented, open source library of software building blocks for accelerated dynamics and hybrid dynamics, and multi-resolution modelling, which can be adapted by other molecular simulation programs.

*Research Idea: Why multi-resolution hybrid dynamics, and why adaptive, energy-aware, fault tolerant work queues?*

This proposed software to be developed is founded on two key ideas;

- (1) A multi-resolution, hybrid dynamics algorithm that uses Monte Carlo to connect large numbers of dynamics trajectories together into a single, rigorously correct thermodynamic ensemble.
- (2) A distributed, adaptive, energy-aware, fault tolerant work packet queuing system, which allows a single simulation to be distributed over a heterogeneous, massively parallel supercomputer.

These two ideas were developed in direct response to three long-standing trends in hardware development:

#### Hardware trend 1: increasing parallelism replaces increasing clock speeds

The decade from 2000-2010 has seen one of the most radical changes in the history of computing. Moore’s Law[5], an observation that the number of transistors integrated on a silicon chip grows exponentially, roughly doubling every two years, has continued unabated. Until 2003 this rapid rise in available transistors was turned into performance increases primarily by finding ways to run a processor’s clock speed at higher and higher rates, through deeper pipelines, larger caches, sophisticated branch and cache prediction schemes and so on. But as processor clock rates continued to climb, so too did processor power consumption, also growing exponentially. In 2002 Intel’s CTO Pat Gelsinger made a famous prediction that if things continued the way they were, processors would be using over 1000 watts by 2010, a distinctly unpalatable proposition. Instead, the industry and the market chose to cap processor power consumption at about 100 watts for desktop and server CPUs. Given that a typical processor is of the order of 1cm<sup>2</sup>, even at 100W the thermals of dissipating the same waste heat as a 100W light bulb from such a small area cause tremendous problems in heat dissipation and device reliability. In around 2003 processors hit the 100W cap and at this point clock speeds had to be reigned in. Processor designers still had ever increasing numbers of transistors at their disposal, but increasing clock speeds had to be dropped as the main method increasing performance.

The obvious way forward was parallelism. Ever since 2003, when CPUs were predominantly single core devices, the number of cores in processors has steadily increased; indeed it has roughly followed the exponential Moore’s Law, doubling every two years or so, and in 2010 we are already seeing mainstream CPUs with between 12 and 16 cores in a single chip, representing between three and four doublings in 7 to 8 years, in-line with the predicted trend.

One of the most successful types of architecture for turning transistors into more parallel performance has been the Graphical Processing Unit, or GPU. These many-core processors were originally developed for 3D graphics, a task which contains large degrees of parallelism (millions of polygons in a scene, millions of pixels displayed every frame) and so naturally these processors went parallel very rapidly. In addition, 3D graphics became increasingly general purpose, and so GPUs have eventually become massively parallel general-purpose processors. In 2010 the latest GPUs have hundreds or even thousands of cores running in parallel, with core counts tending to double every generation.

The next natural evolution is for *heterogeneous* processors, which will integrate several heavyweight, general purpose cores alongside a larger number of lightweight, GPU-like cores. Indeed AMD has already announced its Fusion product line, due to be launched in late 2010 – it represents the first integrated x86 CPU+GPU, a model we believe will become the predominant architecture over the next decade.

## Hardware trend 2: fault-tolerance will have to be increasingly handled throughout the software stack

With increasing transistor counts, core counts, and shrinking silicon feature sizes, faults due to cosmic rays, electrical noise, manufacturing defects, and other transitory ‘glitches’ are set to become an increasing problem[6]. For HPC systems that are predicted to scale to  $O(10^6)$  cores and  $O(10^9)$  parallel threads in order to deliver ExaFlops of performance by 2018, the cost of trying to handle all of these challenges purely in the hardware itself is prohibitive[6]. And yet without comprehensive mechanisms and techniques for dealing with transient errors, the mean time between failure (MTBF) of such HPC systems could be measured in minutes, a disastrous situation. Instead, it is expected that the responsibility for coping with transient errors will be shared across the hardware and software stacks. This means that for the first time it will be routine for applications themselves to be concerned with fault-tolerance.

This greater exposure of applications to the issues of fault tolerance will generate renewed interest in algorithmic approaches which naturally lend themselves to fault tolerance. One promising class of algorithms are those that rely on generating many samples in order to approximate some mathematical function. MC[2] is one such algorithm, as many moves may be processed in parallel, leading to excellent scaling characteristics on massively parallel systems. Additionally, more moves can be generated than actually needed, allowing some of the results to be ignored if they fail to return, return to late, or fail some consistency check. In this way a massively parallel computation using one of these algorithmic approaches may continue to operate correctly and at near full-speed, even as the hardware on which it is running develops faults or suffers transient errors.

## Hardware trend 3: energy efficiency becomes the limiting factor on performance

Power consumption, or inversely *energy efficiency*, became the ultimate limiting factor on performance once we reached the ceiling on thermal density. Once processors were dissipating as much heat as was possible from their surface area there was no-where else to go – in essence, *performance per watt*, rather than outright performance, is key when there is a finite limit on the watts available.

While hardware designers have been hard at work maximising the energy efficiency of the hardware, the same cannot yet be said of software designers. To date, software has almost completely ignored the issue of energy efficiency. In this proposal we aim to create one of the world’s first *energy aware* software applications. We will use design techniques that will optimise the energy efficiency of our software, such as focusing on minimising data movement, choosing arithmetic operations over memory accesses where there is a choice, trimming bloat from code and data sizes to reduce active memory footprints, and using event-driven process models in preference to polled architectures.

In addition to general energy efficiency, large-scale HPC applications will become increasingly carbon-aware, as tax regimes change to encourage reductions in CO<sub>2</sub> emissions. The carbon intensity of energy is not uniform in time; instead, the CO<sub>2</sub> emitted during the generation of energy changes depending on the instantaneous mix of energy generation sources such as coal, gas, nuclear, wind, tidal etc. This mix changes depending on the time of day, the current weather conditions, other loads on the national grid and any local energy generation. In addition to making the software itself as energy-efficient as possible, we will design the software to be aware of its own carbon emissions, something we believe is potentially another first in HPC. We propose to develop the load balancer and job scheduler in the work packet queuing system so that it will be able to take real-time information on energy carbon intensity from sources such as the iDEaS project[7] at the University of Southampton and make decisions to minimise its own CO<sub>2</sub> emissions; for example by choosing to run in periods when the carbon intensity is lower, or by favouring lower power processors and longer running times over minimising execution time.

## **C: Programme and Methodology**

*Methodology: What are the details of the algorithm?*

The multi-resolution hybrid dynamics program will be based on a hybrid Monte Carlo (HMC) algorithm[3]. HMC is a well-established method of connecting together a series of molecular dynamics trajectories into a single thermodynamic ensemble of structures. An HMC simulation is composed from a series of HMC moves, with each move advancing the conformation of the molecular system from state  $i$  to state  $j$  using the following algorithm:

- (1) Momenta for each particle in the system are randomly generated based on the simulation temperature.
- (2) The total energy (potential+kinetic) of state  $i$  is calculated and saved.
- (3) The coordinates of the particles are advanced using a time-reversible, energy conserving dynamics integrator.
- (4) The final configuration of the dynamics trajectory is state  $j$ . The difference in total energies between state  $i$  and state  $j$  is used in a MC test to decide whether to accept the move to state  $j$ , or instead to return to state  $i$ .

We will make novel use of rigid body HMC[4]. The atoms in the molecular system will be grouped together into rigid beads. The coordinates of the atoms in the beads will be advanced using a rigid body dynamics integrator. This will advance the dynamics of the atoms' parent beads. For example, the three atoms in a water molecule could be grouped into a single bead, and the three atoms in the water then moved by rotating and translating the bead that represents that water. Rigid body dynamics allows the atomic-level detail of the molecular system to be retained, while performing dynamics using the reduced dimensional phase space of the beads. This bead-level dynamics will be complimented with atomistic MC moves to include atomic-level sampling. Multiple resolutions of dynamics (atomic and bead) can thus be combined together into a single trajectory. Furthermore, this algorithm is trivially parallelisable, as states  $i$  and  $j$  can each represent ensembles of configurations, and thus large numbers of MC moves can be attempted in parallel. MC is also naturally fault-tolerant, as moves lost due to hardware failure can be easily repeated, recovered or discarded. MC is also adaptive, as it is possible to tailor the balance of MC moves according to the availability of the hardware necessary to perform the move, or the historic energy cost of performing each type of move. HMC will allow for the generation of large ensembles of diverse molecular configurations, which will enable the accurate calculation of thermodynamic properties such as binding free energies and conformational entropies. Because HMC uses MC, kinetic properties cannot be calculated easily. However, for most molecular simulations, it is the thermodynamic properties, rather than the kinetic properties, that are of primary interest.

In addition to using multiple resolutions of dynamics, we also propose to use multiple resolutions of molecular model. We propose to use, simultaneously, both an atomistic model of the molecules, and a coarse grain model of each of the beads. The choice of which model to use between a pair of molecules or residues will be based on distance, with molecules that are close together interacting using the atomic-resolution model, while molecules that are further apart using a coarse-grain bead-resolution model. A switching function would, at the changeover distance, smoothly switch between the two models, ensuring consistency and also eliminating discontinuities in the force. A multi-resolution model will allow the software to take full advantage of heterogeneous hardware platforms, such as CPU/GPU clusters or integrated CPU/GPU processors (such as AMD's Fusion). The beaded model could be accelerated using GPUs if they were available, while in parallel, the atomic model would be evaluated using the host CPUs.

*Methodology: How will the software be developed? How will the programming team be trained?*

The two key areas of software development give rise to the need to hire two experienced programmers (CJW plus a PDRA). The two programmers will work together; one taking the lead developing the adaptive energy-aware, fault-tolerant work packet queuing system, while the other takes the lead developing the GPU-accelerated multi-resolution hybrid dynamics code. Development will be rapid, as it will make heavy use of the existing molecular simulation building blocks and the basic work packet queuing system available in Sire[1]. CJW and the PDRA will work closely with SMS, whose expertise in heterogeneous many-core computing, energy efficient computing and building massively parallel fault-tolerant software will be critical in designing the code.

Training will be provided by CJW (a highly experienced C++ and Python programmer) and SMS (who developed one of the first HPC undergraduate courses to teach OpenCL). This will be supplemented by AMD, who will provide training in OpenCL in the context of the limits of current and upcoming GPU and integrated CPU/GPU platforms (e.g. AMD's Fusion). In addition, NVIDIA will provide training to the development team in the use of their OpenCL profiling tools, thereby providing the expertise to profile and optimise the code at each stage of development.

*Programme: How will the software be developed to be sustainable? What are the milestones?*

To ensure that the software is developed to be sustainable, best practice that has already been used by CJW and SMS in prior software will be carried forward, and will be applied during this project. The software will be developed in public, on a publically accessible, version-controlled repository. The software will be developed as a set of open source, robust, self-contained, documented, unit-tested building blocks, which may be combined together arbitrarily to create new software. These will be written in C++ for the CPU, and OpenCL for the GPU, and these building blocks will be wrapped up using boost.python[8] so that they may be used directly in user-level Python scripts. The software will be written using a portable subset of C++ and OpenCL, making use of standard, widely available and portable libraries (e.g. BLAS, boost[8], Qt[9]). The build system will, like Sire, be based on cmake[10], with the ctest component of cmake used to run unit tests and performance regression tests for each component. Every building block will be fully commented, with these comments automatically processed to generate the publically available API documentation.

CJW and SMS have discovered that a very successful method of software development is a thrice-write model. In this model, three iterations of the software are written:

- (1) Proof-of-concept: Quickly written version of the software used to test the viability of the underlying ideas.
- (2) Prototype: Rewritten version of the proof-of-concept, which implements the software design in full.

- (3) Production: Rewritten version of the prototype, using experience gained writing and using the prototype to solve real problems in order to create a refined design, correct poor design decisions, and make the APIs of (perhaps unexpectedly) related components consistent. This version focuses on robustness and completeness.

This model allows the design of the software to evolve as it is written. Performing a full re-write from prototype to production prevents sub-optimal design decisions at the beginning of the project leading to kludges or unnecessary limitations in the final production product. The thrice-write model also provides three natural milestones; the proof-of-concept version will be complete within 6 months, the prototype will be complete within 14 months, while the production version will be complete within 22 months. The prototype will be applied to real problems by our partners, which will help identify any usage issues, which will be addressed in the production version of the code.

To ensure sustainable development, we also propose to collaborate with the Software Sustainability Institute (SSI)[10], who have committed to providing 20 days of consultation with the following aims:

- (1) Collaboration to create the processes necessary to generate high-level documentation for the software that would complement the processes we already have in place to write component-level documentation.
- (2) In stage 1, to help design a virtualised unit and system testing framework that would allow the unit tests and performance regression tests to be automatically run nightly on a variety of hardware and operating system platforms, and also as code is checked into the source code repository.
- (3) In stage 2, to look at how this unit and system testing framework could be deployed using institutional or national level resources.

The creation of the unit and system testing framework is important, as in our experience we know that this will help ensure that bugs in the software are found and fixed quickly, across all supported hardware and operating system platforms. In addition, the framework will also ensure that ports of the software to rare or older platforms can be maintained even if the primary developers lose direct access to those systems.

#### *Programme: What would be developed in Stage 2?*

In stage 2 we propose to collaborate with other groups to integrate the open source library of accelerated multi-resolution hybrid dynamics building blocks developed during stage 1 into established molecular simulation frameworks (e.g. OpenMM[12]). We also propose to pro-actively engage with developers of existing dynamics and sampling software (e.g. Folding@home[13]) to help port their code to use our building blocks to accelerate their applications. Additionally, in stage 2 we plan to apply for time on HECToR, both to port and optimise the code, and also to perform a grand-challenge style simulation on either a nanomolecular or biomolecular system (e.g. predict pocket formation on ligand binding between two protein surfaces, or to predict carbon nanotube aggregation and separation in the presence of amphiphilic surfactants). We also plan to use our contacts within the international HPC community to gain access to computer time at one of the US national laboratories in order to optimise our software components on multi-petascale class systems. Finally, we plan in stage 2 to collaborate with iDEaS project[7] at the University of Southampton to allow our work packet scheduling system to use real-time CO<sub>2</sub> data to decide when and how to schedule work packets for computation. We also plan to integrate our scheduler with existing cluster scheduling software, such as Torque. These two developments would then allow our work packet queuing system to work across multiple supercomputer clusters (in potentially different countries), thereby allowing HPC software to be written so that the workload could be moved geographically to take advantage of regions of low CO<sub>2</sub> energy production (e.g. if the wind picked up in Ireland, then work packets could be rescheduled and sent to an Irish supercomputer, to take advantage of low CO<sub>2</sub> wind power). This would provide a framework that would allow HPC software to adapt to a future when energy availability, not CPU availability, is the limiting factor, and when supercomputers may end up spinning up and spinning down based on the availability of cheap, low CO<sub>2</sub> energy. Fulfilling these plans would require more manpower, so in stage 2 we would request funding for an additional two developers to join the programming team.

#### *Programme: Management*

To ensure that the stage 1 and stage 2 projects deliver on their potential, **AJM** will assume the role of manager, and will chair bi-monthly progress and review meetings during which progress will be discussed, short-term goals set, and any risks that may complicate development identified. Plans would then be made to mitigate or work around their impact. **AJM** has extensive experience of managing large projects, and his expertise in identifying and managing risk will be vital to ensure that development does not become bogged down or derailed by technical issues.

## D: Impact

### *Community: Who will use the software?*

We have already formed a community of partners who are keen to use the software once it is ready:

- (1) Dr. R. Sessions at the University of Bristol plans to use the software to predict protein-ligand binding affinities of potential drug molecules to the flexible binding sites of proteins targeted for the treatment of Alzheimer's.
- (2) Dr. A. Sneddon at the University of Bristol plans to use the software to simulate separation of carbon nanotubes, and their interactions with amphiphilic surfactant molecules.
- (3) Dr. J. Michel at the University of Edinburgh plans to use the software to accelerate protein sampling within binding calculations and commits time to test the software and perform large-scale validation studies.
- (4) Dr. R. Laws at Evotech (a €250m European pharmaceutical company) plans to use the software to help design protein-protein interaction inhibitors and will provide industrial problems on which the software can be tested.

These initial partners will provide example problems on which the software will be tested throughout its development (e.g. proof-of-concept, prototype and production), and they will provide guidance to ensure that the documentation of the software is complete, and that ease-of-use of the software is high.

We propose to increase this user community by holding a workshop at the end of stage 1. This workshop, based along the same lines as the CCPB workshop CJW ran for his ProtoMS software[14], will teach hybrid dynamics as a means of accelerating protein dynamics. This workshop will be open to members of industry and academia, and would additionally advertise and teach use of the hybrid dynamics software. This workshop will expand the community of users, who will be engaged to help guide development of the software during stage 2. The community will be organised via a central website, which will contain forums, documentation, tutorials, news, downloads and links to the source code repository. With help from the SSI, we will make this community self-sufficient during the course of stage 2, thus ensuring that the software will continue to be developed, documented and used over the next 15-20 years.

### *Academic and Industrial Impact: Who will benefit?*

The immediate beneficiaries of this software will be industrial and academic scientists interested in accounting accurately for protein conformational change within thermodynamic simulations, such as those used to predict protein-drug or protein-protein binding. This will be of great benefit to anyone who is studying highly flexible protein-drug systems, studying protein-protein interactions, or who is designing drugs that are targeting protein-protein interfaces. Furthermore, the algorithms and software developed will be applicable to any condensed phase system, so could be used to study nanomaterials (e.g. to study interactions between amphiphiles and carbon nanotubes) or polymer melts. The software itself, being built on top of Sire, will inherit its portability and open source license. Just like Sire, this new software will be written as a collection of reusable, robust building blocks, which may themselves be taken and re-used to create other novel programs. This will provide a framework that will allow others to rapidly prototype molecular modelling software that utilises both traditional processors and numerical accelerators. In addition the adaptive, energy-aware, fault tolerant queuing system will be developed independently and released as a separate open source library. This will be of benefit to anyone writing new fault-tolerant, energy aware HPC software. Finally, we plan in stage two to port the software building blocks developed in stage 1 to other, widely used molecular simulation frameworks, e.g. OpenMM[12], and to offer the software as a means of accelerating existing applications such as Folding@home[13]. This broadens the immediate impact and user-base of the software, and will ensure that the code and ideas developed here will immediately be useful to, and used by, the wider community of molecular modellers.

- (1) <http://siremol.org>
- (2) N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, *J. Chem. Phys.*, **21**, 1087 (1953)
- (3) S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, *Phys. Lett. B*, **195**, 216 (1987)
- (4) N. Matubayasia and M. Nakahara, *J. Chem. Phys.*, **110**, 3291 (1999)
- (5) G. E. Moore, *Electronics*, **38**, 114 (1965)
- (6) B. Camp, IESP workshop, <http://www.exascale.org/mediawiki/images/c/ce/CampIESP.pdf>, (2009)
- (7) <http://www.ideasproject.info>
- (8) <http://www.boost.org>
- (9) <http://qt.nokia.com>
- (10) <http://www.cmake.org> and
- (11) <http://software.ac.uk>
- (12) <https://simtk.org/home/openmm>
- (13) <http://folding.stanford.edu>
- (14) [http://www.ccpb.ac.uk/workshop/free\\_energy](http://www.ccpb.ac.uk/workshop/free_energy)