D3.1 – Report on Application Software Readiness

Due Date	Month 12
Delivery	September 30th, 2016
Lead Partner	UCL
Dissemination Level	Public
Status	Final
Approved	Executive Board yes
Version	V0.9

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 671564.

DOCUMENT INFO

Date and version number	Author	Comments
02.09.2016 v0.1	James Suter	First version
05.09.2016 v0.2	Alfons Hoekstra, Saad Allowayd, Pavel Zun, Andrew Svitenkov	RBC / Blood rheology / ISR added.
14.09.2016 v0.3	Olivier Hoenen, Onnie Luk, David Coster	Fusion section added
15.09.2016 v0.4	James Suter	Version with integrated Partner contributions.
19.09.2016 v0.5	Onnie Luk	Modification to Fusion section
21.09.2016 v0.6	Robin Richardson	Modification to HemeLB section
21.09.2016 v0.7	James Suter	Changes to integrate with Deliverable 2.1.
28.09.2016 v0.8	Keeran Brabazon	Internal Review
28.09.2016 v0.9	Tomasz Pointek	Internal Review

CONTRIBUTORS

Contributor	Role
James Suter	Author of this deliverable
Olivier Hoenen	Fusion application contributor
Saad Alowayyed	RBC / Blood rheology contributor
Alfons Hoekstra	WP2/WP3 contributor, Project Coordinator
Robin Richardson	Aneurysm Flow Dynamics contributor
Derek Groen	Aneurysm Flow Dynamics contributor
Ulf Schiller	WP3 contributor
Peter Coveney	WP3 contributor
Simon Zwart	Astrophysics contributor
Pavel Zun, Andrew Svitenkov	ISR contributor

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

LIST OF ABBREVIATIONS

AMUSE	Astrophysical Multipurpose Software Environment
CFL	Cell-Free Layer
CG	Coarse-Grained
CTG	Generic Task Graph
CUDA	Compute Unified Device Architecture
RC	Replica Computing [Pattern]
DMC	Distributed Multiscale Computing
ECOL	Energy Consumption Optimization Library
ECOS	Energy Consumption Optimization Service
EEE	Experimental Execution Environment [Project Testbed]
ES	Extreme Scaling [Pattern]
FEM	Finite Element Methods
GPGPU	General-Purpose Graphics Processing Units
HMC	Heterogeneous Multiscale Computing [Pattern]
HMM	Heterogeneous Multiscale Manager
HPC	High Performance Computing
HPMC	High Performance Multiscale Computing
IBLB	Immersed Boundary Lattice Boltzmann
ISR	In-Stent Restenosis
LBL	Load Balancing Library
MAPPER	Multiscale Applications on European e-Infrastructures [Project]
MC	Multiscale Computing
MC	Multiscale Computing
MCP	Multiscale Computing Pattern
MML	Multiscale Description Language
MMSF	Multiscale Modelling & Simulation Framework
MPI	Message Passing Interface
MTO	Multisite Transport Overlay
MUSCLE	Multiscale Coupling Library and Environment
QCG	Quality in Cloud and Grid
RBC	Red Blood Cells
RVE	Representative Volume Element
SMC	Smooth Muscle Cell

1 Executive summary

The objective of Work Package 3 "concerns the instantiation of multiscale computing patterns with the selected grand challenges". This includes "enabling application software to interface with high-level tools and middleware services". The aim of this document is to report on the readiness of application software to be instantiated as one or more multiscale computing patterns using the ComPat technology components.

Each of the multiscale simulation scenarios for the applications can be expressed as one, or as a combination of the three patterns defined in the project: Extreme Scaling (ES), Replica Computing (RC) and Heterogeneous Multiscale Computing (HMC). We have distinguished between fast and deep-track applications. Fast-track applications will be deployed in the first phase of the project, while the group of the deep-track applications will be deployed subsequently to support more sophisticated multiscale computing scenarios. The fast-track applications have been identified as those where the single-scale models are well optimised and scale-bridging components (to couple different single-scale models) are already developed. As such, these applications will be ideal tests for instantiating computing patterns using the current ComPat technology stack (WP 5) on the Experimental Execution Environment (WP 6) and for developing tools in WP 2 that convert xMML descriptions of the applications into task graphs and subsequently execution recipes. We have identified three applications as fast-track, with one fast-track application for each Multiscale Computing Pattern (MCP); these applications are currently in the process of being installed and tested on the EEE. The fast-track applications are: calculating ligand-protein binding affinities, global turbulence simulations and Milky-Way Galaxy simulations. Additionally, the In-Stent Restenosis application is of an advanced state and may also be considered as fast-track. In each case, the applications currently interface with ComPat components that control flow within coupling environments (FabSim, MUSCLE, AMUSE). Ultimately, these control flow tools will be directed by the execution recipes developed by software developed in WP 2, starting from an xMML description of the application.

The deep-track applications require improvements in scale-bridging methods or single-scale models to tackle our grand challenges; these developments are described in this document and are expected to be finalised in the next reporting period, allowing the deep-track track applications to be instantiated as all components of the ComPat technology stack and tools using generic task graphs from WP 2 become available. For each of the applications, in this document we describe its readiness to be instantiated as a computing pattern, both in terms of scientific developments, and operability with the ComPat high-level tools and middleware services. We also describe the components we expect the

applications will be operating with during the next reporting period. Therefore, this document also presents an initial , schedule for Task 3.2, “Instantiation and validation of multiscale computing patterns”.

2 Application software readiness

The main objective of ComPat is to develop generic and reusable High Performance Multiscale Computing (HPMC) algorithms that will enable us to tackle scientific grand challenges, as defined in Section 2.2. The algorithms will provide scalability, robustness, resiliency, and efficiency of multiscale applications with extreme data requirements. ComPat is an application driven project, with a variety of scientific grand challenges at its core. There are nine grand challenge applications from four scientific domains to drive these developments. In this deliverable, we will report on the specific requirements of each application, including the work that has been achieved so far and future plans for developing the applications as tailor-made instantiations of one or more computing patterns in the remainder of the project. This document fulfils Milestone 3 “Application Software Readiness Established”.

The main aim of the construction of the computing patterns (so called Multiscale Computing Patterns - MCP) is to simplify the implementation of the applications. In effect, this will reduce the “time to market” as well as to enhance the execution of common multiscale application scenarios. From the applications’ point of view, the proposed patterns determine the ordering and composition of single scale models that are coupled (via scale bridging methods) within a multiscale application. In order to orchestrate the execution of the pattern-based applications on HPC resources, as part of WP5 and WP6, a comprehensive technology stack has been developed – see Deliverable 5.1 for details. As part of WP6, an Experimental Execution Environment (EEE) has been created, with a functioning ComPat stack, as described in Deliverable 6.1. The EEE currently consists of the following machines: SuperMUC (LRZ), STFC machines (NCD, BGAS, BlueWonder), Inula (PSNC) and Eagle (PSNC).

For efficient testing of the EEE, and to enable applications to start interfacing with the ComPat stack as soon as possible, we have decided to distinguish between fast-track and deep-track applications. The applications assigned to the former are required to be deployed in the first phase of the project, with limited functionality, while the group of the deep-track applications will be deployed subsequently to support more sophisticated scenarios. In conjunction with this, there are corresponding deep and fast-track components of the ComPat stack. For more details on the ComPat technology stack, and the division between fast and deep-track components, please see Deliverable 5.1. WP 3 will be working closely with WP 2 during the next reporting period to develop the software which will go from an xMML description of each application to a task graph description and will

ultimately, via generic MCP task graphs, produce the input / configuration files for the application using ComPat tools. The fast-track application will provide the first examples of each MCP (see section 2.1 and for greater detail see Deliverable 2.1). The deep-track applications will use the multiscale computing patterns code generator which use generic task graphs as they emerge from WP2.

This deliverable reports on the specific requirements of each application to be instantiated as a computing pattern. For the fast-track applications, where much of the multiscale functionality is already in place, this includes examining how to get the application software to operate with the fast-track ComPat tools and middleware services. This information, combined with the knowledge of the MCP of the application, will be used by the software developed in WP 2. Ultimately, the ComPat stack will allow these multiscale applications to achieve greater scalability, robustness, resiliency, and energy efficiency than is currently possible, allowing more demanding scientific challenges to be investigated. In the following, we indicate the current status of integration with the EEE and ComPat technology stack, a timetable for producing high-fidelity scientific data, instantiating the application as a MCP, and finally, the challenges this involves. In Table 1 and in Section 2.2 we list the current and future integration of ComPat components for each application (a detailed description of each component in the ComPat technology stack listed in Table 1 and their capabilities is given in Deliverable 5.1).

For applications in the deeper track, more complicated and sophisticated scenarios are desired. For deep-track applications, in a similar manner to the fast-track applications, we describe the scenarios we plan to instantiate as computing patterns, how these will operate with the ComPat tools and the scientific developments required. To achieve our grand challenge objectives, scientific developments are often required. For example, to couple different levels in a multiscale simulation where the characteristic time and length scales are not easily separated, or where significant pre- or post-processing is required to create initial or boundary conditions for simulations on a different time or length scale. This coupling is referred to as a scale-bridging method, corresponding to the arrow between single scale models in a Scale Separation Map. It is clear that scale-bridging methods can entail a complicated hand-shake, requiring multiple steps and computations. However, this is often where scientific developments are required to realise grand challenge objectives and therefore a significant amount of work has been spent in this area during this reporting period. Similarly, there has been a considerable amount of effort in improving the performance of the single-scale models, using the profilers and debuggers of WP 4.

The ComPat tools allow us to think creatively about how multiscale scenarios, aligned with the MCP patterns, can extend our current multiscale simulations to tackle these grand challenges. The MCP

patterns will effectively shorten the time to implementation for such complicated multiscale simulations, while hiding the complexity of dealing with fault tolerance and energy-efficiency from the user.

2.1 Multiscale computing patterns

We have identified three computing patterns that, based on our previous experience with multiscale modelling, we believe to be most relevant for high performance multiscale computing, namely: Extreme Scaling (ES), Replica Computing (RC) and Heterogeneous Multiscale Computing (HMC). Each of the multiscale simulation scenarios for the applications described in the rest of the document can be expressed as one, or as a combination of these patterns. A major design decision for ComPat is the realisation that the Multiscale Computing Patterns will be expressed on the level of a task graph, generated from xMML. This is then taken together with execution recipes specific for a pattern and performance models or data for the single scale models and scale bridging algorithms, to determine the actual execution, which will then be specified in configuration files for the ComPat middleware. The formulation will provide enough information to determine runtime aspects, such as number of cores needed, average amount of time used and/or energy consumed. Equipped with this formal specification, the high-level tools will be able to provide estimates for the runtime behaviour of a model on different resources and produce runtime configuration files. In Year 2, WP 3 will be working closely with WP 2 and WP 5 to achieve this. Initially, we will define the xMML specification for each application and produce performance data for single scale models and scale bridging methods (using WP 4 tools or performance models generated in WP2). Software developed in WP 2 will use this information to create task graphs to generate configuration files and execution scripts for the ComPat middleware for that application. In the upcoming period, for the fast-track applications, we will be integrating the single scale models and bridging methods with the ComPat tools and middleware. The set of actions required to perform these multiscale simulations on the EEE will feed back into work conducted in WP 2, such that by Month 18, application specific execution recipes matching to their corresponding MCP (or combination of patterns) will be generated by the WP 2 software. For more details, see Deliverable 2.1.

It should be noted that the use of MCPs is flexible - some of our applications will actually require hybrids of the multiscale computing patterns. As WP 2 produces the tools to firstly translate task graphs from xMML, embed that application task graph in generic task graphs, and then translate the generic task graphs to execution recipes and configuration files (ultimately including functionality for fault tolerance, load balancing and energy awareness), we will begin to automate the implementation of these multiscale computing instances using these tools.

2.2 Grand challenge areas

ComPat is a science driven project. The urgent need to push the science forward and stay world leading in simulation driven science and engineering is our primary motivation. The challenges in each of the domains represented in ComPat are of enormous intellectual, societal, economic and industrial concern. The first is in biomedicine, where we need to obtain deeper understanding of pathophysiology of cardio-vascular disease, as well as to provide personalised models of the vasculature in near to or real time for the purpose of supporting clinical decision-making, and to rapidly and accurately calculate ligand-protein binding free energies in drug discovery and personalized medicine.

In nuclear fusion, our ability to understand interactions between turbulence at very small scales and the large scale plasma behaviour holds the key to control its magnetic confinement in order to produce clean and carbon free energy for the indefinite future. In astrophysics, we aim to understand the formation processes of stars in their clustered environment, as well as the origin and propagation of structure in the stellar disk of the Milky Way Galaxy. A fourth grand-challenge aims to attain the Holy Grail of materials science, namely the ability to predict the materials properties of macroscopic samples of matter based on the specification of the atoms and molecules comprising it.
In Table 1, we show which parts of the ComPat technology stack will be used by each application. We have indicated the fast-track applications and components.

2.2.1 Biomedicine

2.2.1.1 Aneurysm flow dynamics (HemeLB) – deep-track

Grand challenge: Simulate the vasculature in the brain for the purpose of supporting of clinical decision-making.
MCP pattern: Extreme Scaling / Heterogeneous Multiscale Computing
Scientific Readiness: There has been significant software development work concerning HemeLB's code and data model to allow scaling to higher core counts on HPC facilities than was previously possible, as required for the Extreme Scaling pattern. This has mainly been achieved through reducing HemeLB's memory consumption per node during the initialisation and decomposition stages. Further work of this sort was carried out on HemeLB at an Extreme Scaling workshop held at LRZ, Munich (29th Feb - 3rd Mar 2016). There is also an ongoing collaboration with Allinea to use their performance profiling tools to identify computational bottlenecks (WP4).

In terms of scientific research, coupling of HemeLB to a 1D Computational Fluid Dynamics solver (PyNS) [1] has allowed inlet velocity profiles to be determined *in silico*. This is an example of an ES scaling pattern with the 1D solver as the auxillary model and HemeLB as the primary model. We have

also performed large multiscale simulations of the entire Circle of Willis, with the aim of exploring the effects of thrombosis in the Basilar Artery, and Left and Right Internal Carotid Arteries. We have implemented the coupling using FabSim. Improved velocity inlet weighting algorithms have been developed which give a more physically accurate inlet profile, important for the intended clinical applications of this simulation code. Furthermore, there has been research concerning the feasibility of velocity and resolution downscaling to reduce the computational cost of clinical applications of Lattice-Boltzmann simulations. The simulations (see Figure 1 below) were run for both the Circle of Willis and the Middle Cerebral Artery. This work has revealed the strong effect changes in peak velocity have on the resulting flow patterns, and the likely need for ensemble simulations to measure uncertainty in the results. Additionally, code for the simulation of colloidal particles has been integrated to HemeLB for the purpose of modelling magnetic drug targeting in the brain, with an intended eventual application to the region of flow diverters.

Future work on HemeLB will focus on the introduction of flow diverters within blood vessels to explore the long range effects of their introduction, such as likely candidate sites for new aneurysm formation. This will involve the running of many replicas to deal with uncertainties in medical input data, and to span the physiological range of heart rates in the patient post-intervention. This work would be difficult to schedule efficiently without the ComPat tools being developed for the RC pattern, which will greatly simplify the workflow, allowing use of all available computing resources with no added complexity. Medical applications must run and complete within clinical timescales (a couple of hours), and therefore the performance models in the ComPat tool chain will also be very useful for this case. Once completed successfully, we plan to build on this work to investigate the effects of the actual deployment of a stent within a vessel. This would be achieved by using the ES pattern, with HemeLB as the primary model simulating the stresses of the blood flow on the inserted stent, coupled to the ‘stent builder’ (developed during this project) as the auxiliary model, modifying the shape of the stent. The efficient running of a stent deployment simulation greatly benefits from use of the ComPat tools, particularly due to the very different computational requirements of the primary and auxiliary models in this case, and naturally fits into the ComPat paradigm. A stent deployment simulation corresponds to the ES pattern.

Readiness to operate with ComPat high-level tools and middleware services:
HemeLB is already integrated with FabSim for deployment on remote HPC resources. We will be shortly installing HemeLB on the EEE.

Timescales for ComPat Component Integration:

- **Current Components**: FabSim, Single-Scale Debuggers and Profilers, MPI.
- **Components to added in M12-M24**: QCG-Client Tools, MUSCLE / MUSCLE-HPC, Job and Advance reservation manager, Scheduler and resource allocator.

- Other deep-track components (listed in Table 1) to be added when they become available.

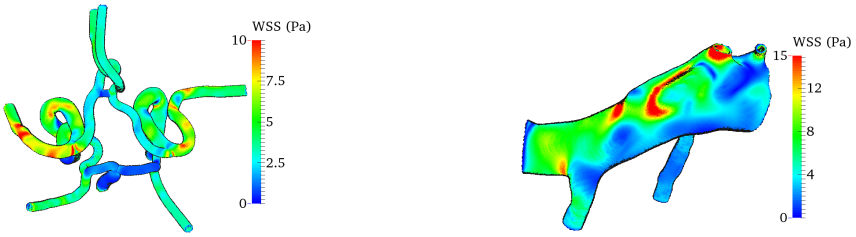

Figure 1: The wall shear stress (WSS) induced at the peak of a systole in the Circle of Willis (Left), and the Middle Cerebral Artery (Right) using HemeLB.

2.2.1.2 Red Blood Cells (RBCs) and platelet transport – deep-track

MCP pattern: Extreme Scaling

Grand challenge: Modelling of blood as a suspension

Scientific Readiness: During the first year of the project, the main focus has been on preparing our red blood cell based blood flow simulation for developing the Extreme Scaling (ES) computing pattern, in collaboration with WP2. This pattern exemplifies a class of multiscale applications in which a primary model allocates all the computational power and there is a requirement for a number of auxiliary models. Using the ComPat tools with this pattern will resolve the performance drawback originating from the huge difference in computational requirements between the primary and the auxiliary models. More details on development of the pattern are reported in Deliverable 2.1.

Currently, we are utilising an Immersed Boundary Lattice Boltzmann (IB-LBM) suspension simulation framework (Ficsion [2]), as a primary model. Palabos [3], a fully parallelised open-source LBM-based fluid solver, acts as auxiliary model. Our effort in WP3 in the first year of the project has been on optimisation and scalability of the Ficsion code (Fig. 2), and setting up the required scale bridging framework to enable the final multiscale model, coupling the continuous flow models to the suspension model.

In cell based blood flow simulations, Red Blood Cells (RBC's) migrate toward the middle of a channel which causes a reduction of RBC's near the walls, known as the Cell-Free Layer (CFL) [4]. This results in a higher level of haematocrit in the middle of the channel. For a fully resolved blood flow simulation framework like Ficsion, running on a large number of processors, the inhomogeneous distribution of RBCs among processors may become a major source of loss of parallel efficiency. We

calculated the fractional load imbalance overhead [5] using an earlier performance prediction model [6]. From a breakdown on the fractional communication and load imbalance overhead for the channel flow using 3-D domain decompositions, we notice that the load imbalance due to the external walls is observed when the number of cores discretising the y-direction increases. To circumvent this bottleneck, which will become much more severe when scaling up the number of processors up to $O(10^4)$ or more, we are developing a Load Balancing Library (LBL). This will reduce the load imbalance in such a situation and therefore increase parallel efficiency. This library discretises the fluid domain into small blocks. These blocks contain a variable number of Lagrangian points (that are used to discretize the RBCs) that determine the computational load per block. The number of Lagrangian points shared with neighbouring blocks contributes to communication overheads. Shared points are listed per block. Taking this information as an input to the METIS library [7] produces a block affiliation array, makes decisions regarding the distribution of blocks to processes in order to balance computational work. This library is integrated in both Ficsion and Palabos. Moreover, decreasing the load imbalance of the computation will reduce the load imbalance in communication as well. We are currently in the phase of setting up large test runs, and we expect to deliver the load balancing library by the end of 2016. We will also investigate the use of this library with other codes in the ComPat project.

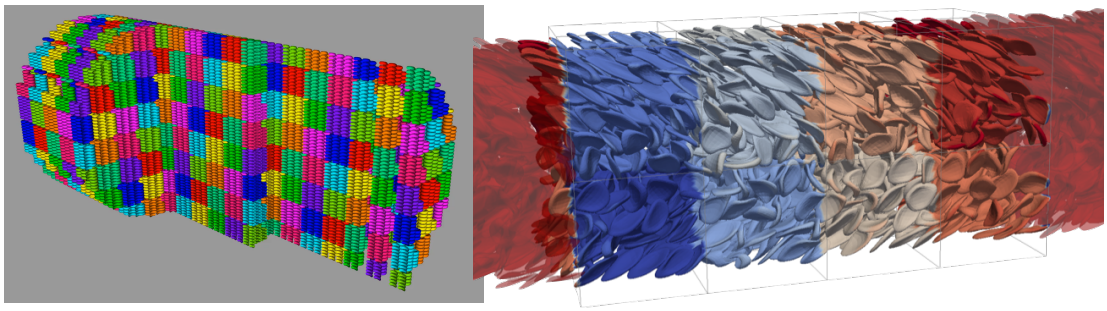

Figure 2: Snapshot from Ficsion. Right, initial condition for a run with 45000 RBCs on 1024 processors, and left a snapshot for an equilibrated system of 8000 RBCs on 16 processors.

We profiled the code using Allinea tools (see Deliverable 4.1 for some results of this work) and found that a significant amount of runtime is spent in one of the small computational-cost functions, which means that there is likely an issue with cache usage. We created data structures where cache locality is ensured. Furthermore, we reduced cache misses by avoiding unnecessary structure packing (and unpacking), which had been used in the original framework. These optimisations increased the overall performance by 30%.

When scaling up a cell based blood flow simulation to hundreds of thousands of cells defining the appropriate initial positions and alignments are quickly becoming an issue. Before the start of the project, we used a simple packing algorithm that positioned the cells along a regular grid using their bounding boxes. Defining the gap between the bounding-boxes also defines the hematocrit level. This

initialisation method however, has some disadvantages: it results in a very low entropy state far from the equilibrium. The necessary amount of iterations for the cells to spread out, mix and then finally fill the domain takes up a significant amount of iterations and therefore simulation time. To circumvent this problem, a more advanced initialization procedure has been developed. A small kinetic simulation is carried out where the cells are represented as rigid homogeneous ellipsoids with a shrunken volume. These small ellipsoids start with a random position and alignment within the domain. Afterwards, they are grown to match the actual size of the desired cell types (RBCs, platelets, etc.). During this process they can collide and push or rotate one another. This method is computationally much less expensive than computing irregular, deformable object collisions. On a single node the usual computation is on the scale of several minutes for a hundred thousand particles. The final state of this simulation allows for a randomised initial condition for the cells that is closer to a steady state. In the general case of flow inside of a straight tube this saves on average of 30-35% of the total computation time. The initialisation for a big run is shown in Figure 3.

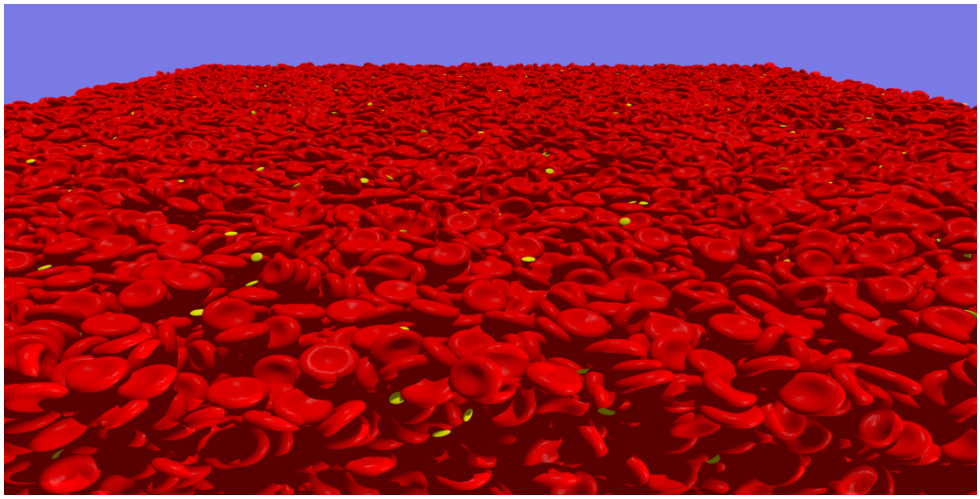

Figure 3: The initial setup for a simulation run with 70,000 Red Blood Cells

Considering all reported optimisations, which have led to a factor 10 improvement in computational speed, we are now ready to use Ficsion as an application for the ES computing pattern.

Readiness to operate with ComPat high-level tools and middleware services:

We already implemented a MUSCLE based sandbox application to couple Palabos with Ficsion. This coupling is shown in Figure 4, demonstrating that the continuous fluid field represented in LBM is coupled to the in- and outlet of the dense suspensions fluid. However, the actual scale bridging model is still missing in this work, and will be completed during a later stage of the project. Nevertheless, while the scale bridging algorithms are being developed, we can already use the sandbox implementation to test all issues with respect to the ES pattern. In close collaboration with WP2 we will, in the next 6 months, develop a first realization of the ES pattern, using a Palabos/Ficsion application to drive this development, executed on the EEE.

Timescales for ComPat Component Integration:

- **Currently integrated components**: MUSCLE, Single-Scale Debuggers and Profilers, MPI.
- **Components to added in M12-M24**: QCG-Client Tools, Job and Advance reservation manager, Scheduler and resource allocator, Performance Prediction Models (from WP2).
- Other Deep-track components to be added when they become available.

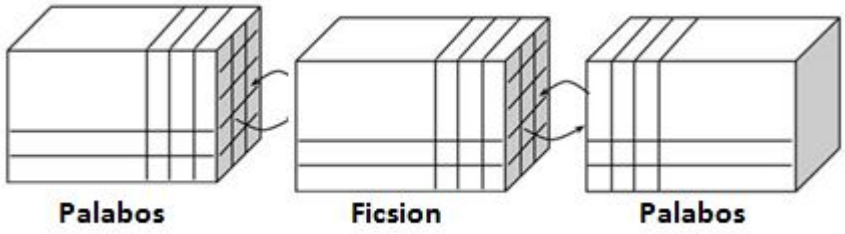

Figure 4: An example of an ES application, a blood suspension model (Ficsion, the primary model) coupled to two continuous blood flow models (auxiliary models, Palabos) that provide the in- and outflow conditions for the suspension model.

2.2.1.3 In-Stent Restenosis (ISR) – deep-track

Grand Challenge: to obtain deeper understanding of pathophysiology of vascular disease

MCP pattern: Extreme Scaling / Replica Computing

Scientific Readiness: The *3D in-stent restenosis* (ISR3D) application is intended for simulation of the restenosis process, a potential complication after coronary artery stenting. ISR3D is a fully coupled 3D multiscale model, which includes several single-scale submodels as well as utility modules which facilitate communication between the submodels. The submodel structure is similar to the one used by Caiazzo et al. in [8] and is based on our earlier two-dimensional studies [9–13]. The submodels are described in this section.

Mechanical model of Small Muscle Cells (SMCs). The agent-based model is used for modelling of the mechanical response of walls of a coronary artery. Individual cells of the vessel tissue as well as the stent struts are presented as an array of point-wise particles with no mass, and their interactions are provided by repulsive and adhesive forces. The effective radii of particles represent the radii of corresponding cells and change during growth. Mathematically the problem is formulated as a Cauchy problem for a system of ordinary differential equations. The coordinates of agents after each step of the biological model or stent deployment model are taken as initial conditions.

The typical distance of interaction of particles is 20-30 μm, and $1 \cdot 10^6$ to $5 \cdot 10^6$ cells are used in the model, depending on the size of the vessel. The integration timestep for numerical solution of the system of ODEs is $\sim 10^{-3}$ s, while the typical time of one mechanical simulation is from 1 s up to 2 min (stent deployment). This model is implemented in parallel on a shared memory system.

Hydrodynamic model. Blood flow in the stented vessel is driven by a stationary Newton hydrodynamic model, which provides relevant range of shear stresses on the vessel walls. The simulation of blood flow in the vessel is provided by the lattice Boltzmann method (the D3Q19 L-BGK variant). The space step of the mesh is equal to the average size of SMC-agent (15 μm) while the simulation domain is the whole stented region of the vessel, i.e. up to 15 mm in length. The array of states of all cells of the mesh in simulation domain (void/wall/stent) in the current simulation time of biological model provides an input for hydrodynamic solver. The timestep of simulation is 10^{-4} s and the time of getting a stationary flow is around 10^{-2} s. The field of flow velocities in each lattice cell at the next timestep, and shear stress values in the cells near the wall of the vessel are the output of the solver.

Utility mapping modules. To produce the input data for the hydrodynamic solver the array of agents has to be presented as a surface of the vessel wall. Each cell of the hydrodynamic solver mesh where an agent is present is marked as solid, and then the obtained configuration is smoothed. In the same way the values of shear stresses and drug concentration in the vessel wall are mapped to the corresponding agent-cells.

Biological model of SMC. A cell cycle model is used for modelling the cell dynamics. The cell lifecycle is a sequence of growth, replication and division of the cell; at the end of the lifecycle the cell divides into two daughter cells. The processes that influence the cell lifecycle take place in the 30 μm neighbourhood around the cell, and the time scale of one cycle is around 24-48 hours.

The growth of separate cells is modelled by a finite-state automaton. For each cell it can be in a state of growth (G1), synthesis/repeat growth/mitosis (S/G2/M), or be idle (G0). Cells move from one state to the next, and stop or die under the influence of external factors such as mechanical stresses (from mechanic model of SMC), the concentration of nitric oxide (calculated from the shear stresses passed by the hydrodynamic model), and the concentration of growth suppressing drugs (from diffusion model). The biological model provides new radii, states and coordinates of the cells as its output. Growth of the neointima (scar tissue) takes several dozens of cell cycles and stops several weeks after the stenting procedure. Growth can progress up to the full vessel occlusion in some special cases.

Drug diffusion model. This model is implemented by the anisotropic diffusion equation on the same mesh as the hydrodynamic model. The model has the same data as hydrodynamic model at the input and provides the drug concentration in each mesh cells as its output.

Data transmission between the submodels. Figure 1 shows the inter-block communication scheme, where SMC is the smooth muscle cell model (mechanical and biological); Blood Flow is the lattice Boltzmann solver; Drug diffusion is the anisotropic diffusion model; Blob is the thrombus model (currently unused); Voxelizer, Thrombus mapper, Distributor and Collector are the utility modules. The SMC block transmits an array of agent coordinates and radii to a utility module which defines the state of the lattice cells (lumen/wall/stent) and passes the array with this data (Domain) further. For the flow solver this array is modified, and all “stent” and “wall” cells are marked as solid. Then, the flow

solver passes the array of wall shear stresses, and the diffusion solver passes the array of drug concentrations to another utility module which maps these values to the SMCs and sends them to the SMC model.

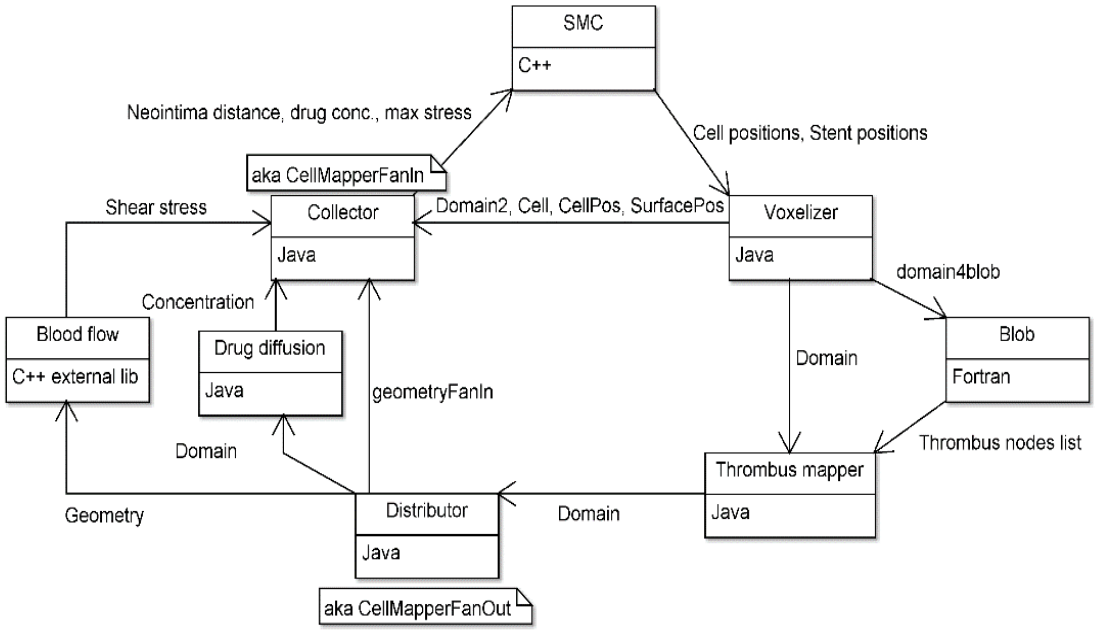

Figure 5: ISR3D data communication scheme showing the initialisation and first two cycles in the ISR3D model.

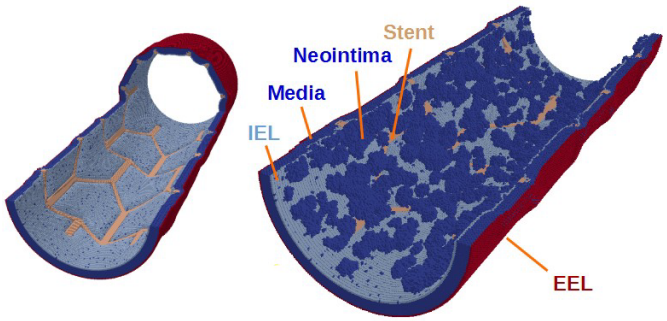

Figure 6: Snapshot of an ISR3D simulation

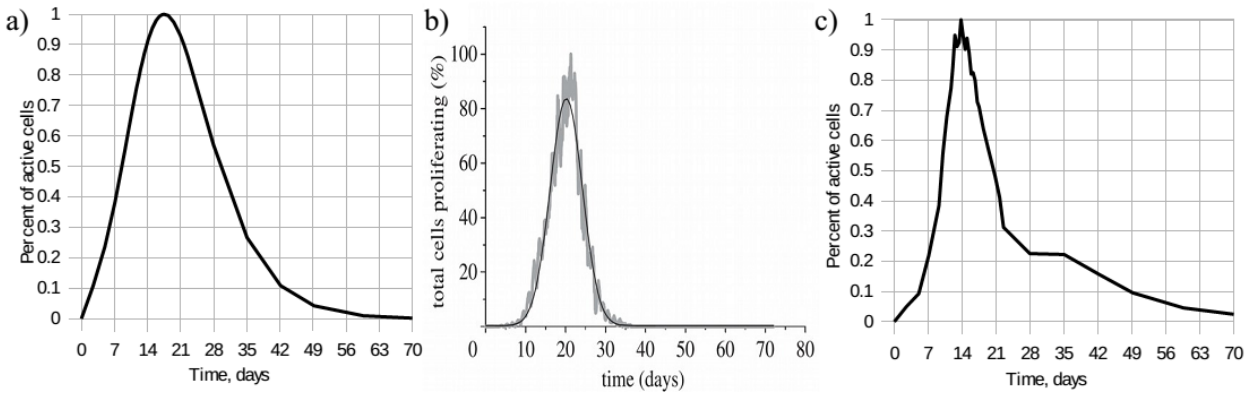

Figure 7: Normalized peak absolute growth fraction (NPAGF) for (a) function fitted to experimental data [14], (b) earlier 2D ISR computational model [11], (c) current 3D ISR model.

Preliminary results obtained using the ISR3D model

Figure 6 shows a snapshot of a simulation (not showing the bloodflow). We have considered several stent deployment and reendothelization scenarios with different deployment depths and rates of endothelium function recovery. Figure 7 shows the normalized peak absolute growth fraction (NPAGF) for (a) a function fitted to experimental data [14], (b) for the earlier 2D ISR computational model [11], and (c) for the current 3D model. Unlike the in-vivo based function (a), the 2D function (b) shows a late start of growth, and its late dynamics is symmetrical to early dynamics. The peak proliferation time in (b) happens a few days later than in (a). Despite that, the 2D model was able to capture the general neointima growth dynamics.

The 3D model (c) is based on the same ruleset as the 2D model (b). The biological rules stayed unchanged, while the physical rules were adapted to account for the 3rd dimension and the higher average number of neighbours for each cell. The transition from 2D to cylindrical 3D geometry has qualitatively affected the growth dynamics. Unlike the 2D case, in 3D vessel growth starts immediately after stenting, which agrees with the data (a). Also, the late proliferation does not stop abruptly, but continues for an extended period of time. This also agrees better with the experimental data. However, the late growth is not in complete agreement with the NPAGF in (a). This can be attributed to shortcomings of the 3D model as well as to the shape of the function used to produce the results in (a): Schwartz *et al.* use a function that produces good results for early growth, but doesn't capture, for example, late vessel remodelling [14].

The results of the simulation qualitatively agree with the histological data, although quantitatively the simulated growth is significantly lower than experimental (figure 3). This can be attributed to the absence of explicit extracellular matrix (ECM) in the model. According to some experimental studies, ECM can make up more than 50% of the neointima volume [15–17], and in our model it is assumed that the ECM passively fills the gaps between cells and does not actively push them apart. Also, the results described here correspond to fast endothelial recovery. Slower recovery scenarios may lead to more pronounced neointima growth.

We have also simulated different endothelium recovery scenarios in our model and found that the recovery speed severely affects the degree of restenosis, which agrees with the experimental data [11].

Readiness to operate with ComPat high-level tools and middleware services:

The ISR3D application is fully compliant with the Multiscale Modelling Simulation Framework (MMSF). The workflow of this application has already been described in the language used in the MMSF. For a description of the MMSF and a discussion of the ISR task graph, see Deliverable 2.1. The control flow and coupling in the application is currently handled by MUSCLE2, part of the ComPat technology stack.

Timescales for ComPat Component Integration:

As the ISR3D application is fully compliant with the MMSF, it provides an ideal test-bed for the developments made in Work Package 2, such as using single scale performance models to create input / configuration files to be executed on the EEE (by ComPat tools such as MUSCLE or QCG tools). As a first step, the ISR3D will be installed on the EEE in the next 3 months.

2.2.1.4 Binding affinities: rapid and accurate calculation of ligand-protein binding free energies – fast-track

Grand Challenge: Rapid and accurate determination of drug binding efficiency

MCP pattern: Replica Computing

Scientific Readiness: Our theme of current research is to predict the strength of macromolecular binding free energies using computationally based molecular modelling. Rapid and accurate calculation of binding free energies is of major concern in drug discovery and personalized medicine. To perform modelling and calculation with optimal efficiency, we have developed the Binding Affinity Calculator (BAC), an automated molecular simulation based free energy calculation workflow tool. Over the past few years, we have uncovered and developed two new ways of calculating the free energy of binding of ligands to proteins [18,19].

The underlying computational method is based on classical molecular dynamics (MD). These MD simulations are coupled to the molecular mechanics Poisson−Boltzmann surface area (MMPBSA) method to calculate the binding free energies. For purposes of reliability, ensembles of replica MD calculations are performed for each method, and we have found that *ca* 25 of these are required per MD simulation in order to guarantee reproducibility of predictions. This is due to the intrinsic sensitivity of MD to the initial conditions, since the dynamics are chaotic. Therefore, the BAC is an ideal example of the replica computing pattern and, as such, it is one of the first applications that will be integrated into the ComPat technology. This application was not originally part of the ComPat Description of Action. However, due to recent successful large-scale simulation runs using the BAC (detailed below) and the BAC's fit with the Replica Computing pattern, combined with the potential for powerful, more complicated replica scenarios using the ComPat technology and computing patterns, means it will be an ideal fast-track application.

Readiness to operate with ComPat high-level tools and middleware services:

We have recently calculated the free energy of binding of an unprecedented number of ligands to proteins by utilizing both Phase 1 and 2 of the SuperMUC machine at LRZ (250,000 cores in total). To achieve this feat, we required a highly specific set of FabSim scripts and commands, combined with human interaction, to ensure successful completion of a workflow composed of a large ensemble of interdependent replica simulations. In the next reporting period, we will be integrating FabSim with the ComPat middleware services (QCG client tools) and the Jobs and Advanced Reservation Manager.

The creation of replicas will be handled by Pilot Jobs (a form of multilevel scheduling – see Deliverable 5.1 for details on all components of the ComPat technology stack).
When Deep-Track components such as the performance prediction models and the Replica Computing code generator become available, we expect the QCG tools will be able to schedule and execute our free energy of binding simulations in the most efficient way (including energy efficiency), with the minimum of human interaction. The ComPat architecture will therefore make the pre-execution and execution of our simulations easier and automated, as well as more robust, reusable and with shorter turnaround times. It will also allow us to straightforwardly extend this methodology to a range of new use cases and create more complicated workflows (for example, re-simulation and modification of promising drug candidates, which will require multiscale coupling to quantum simulations for ligand parameterization).

Timescales for ComPat Component Integration:

- **Currently Integrated Components**: FabSim, MPI, profiling tools (there is ongoing work to ensure WP4 tools work with the AmberTools post-processing programmes used to calculate free binding energies). The BAC is currently being installed on the EEE.
- **Components to added in M12-M24**: QCG-Client Tools, Job and Advance reservation manager, Scheduler and resource allocator, Pilot Tools (for job spawning).
- Other deep-track components to be added when they become available.

2.2.2 Materials science – Deep-track

Grand Challenge: Predicting the materials properties of macroscopic samples of matter based on the atoms and molecules composing it.
MCP pattern: Heterogeneous Multiscale Computing
Scientific Readiness: In the materials science application, we use systematic coarse-graining where the macro-scale (coarse-grained) simulation employs interactions (force-fields) whose parameters are determined from one or more micro-scale (atomistic or quantum-mechanical) simulations. This represents a specific instance of a heterogeneous multiscale model. In this reporting period, we have been developing methods under the HMC pattern that support automation of systematic coarse-graining procedures. We have been developing scientific tools to automatically deploy micro-scale simulations to determine accurate force fields, given an initial state extracted from the macro-scale coarse-grained simulations. These tools will be connected to the HMM "on-the-fly" database, and will be executed when the database does not contain the parameters required for the macro-scale simulation. More specifically, these scale-bridging tools include a Monte-Carlo building tool for creating atomistic simulations with realistic geometries, a reverse-mapping tool for building atomistic

simulation input from coarse-grained simulations and a workflow tool for creating coarse-grained interaction parameters from these atomistic simulations.

In our application area of nanocomposites, we have used these methods to describe the mechanism that leads to full exfoliation and dispersion of organophilic clays when mixed with molten hydrophilic polymers (Figure 8). This process is of fundamental importance for the production of clay–polymer nanocomposites with enhanced materials properties [20]

In future work, we will be developing methods to couple coarse-grained molecular dynamics with Finite Element Methods (FEM), using the HMC pattern to include an HMM database containing the response of nanocomposite Representative Volume Elements (RVE) computed at the atomistic or coarse-grained scale; similarly, micro-scale simulations will be automatically deployed if required.

Readiness to operate with ComPat high-level tools and middleware services:

The parameterisation routines for creating coarse-grained interaction parameters are already integrated with FabSim. The LAMMPS molecular dynamics code will be installed on the EEE and tested using QCG Tools.

Timescales for ComPat Component Integration:

- **Current Components**: FabSim, MPI
- **Components to added in M12-M24**: QCG-Client Tools, Job and Advance reservation manager, Scheduler and resource allocator.

Other Deep-track components to be added when they become available: the HMM manager and the on-the-fly database, to be developed in WP2.

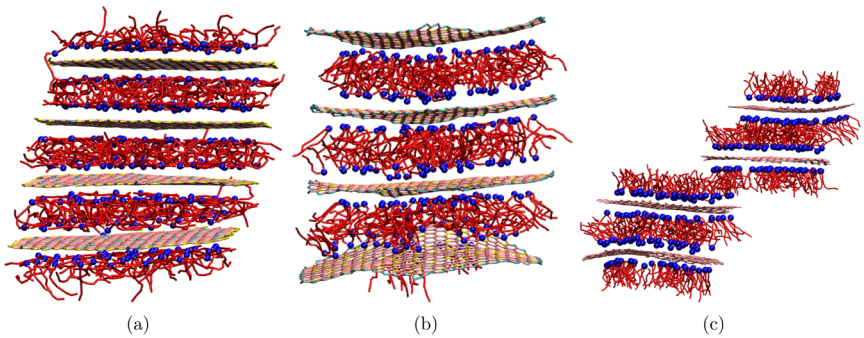

Figure 8: Snapshots from coarse-grained molecular dynamics simulations of a selected alkyl ammonium treated clay tactoid immersed in a melt of PEG polymer, illustrating the exfoliation of the clay tactoid with sufficient chemical treatment of the surface.

2.2.3 Fusion – Global Turbulence Simulation (fast-track) / Flux-Tube Chain (deep-track)

Grand Challenge: Predicting the performance of future fusion devices such as ITER (International Thermonuclear Experimental Reactor) and DEMO (DEMOnstration Power Station) based on finding plasma states that are consistent with respect to transport, turbulence and magnetic equilibrium.

MCP pattern: Extreme Scaling (fast-track and deep-track) / Heterogeneous Multiscale Computing (deep-track)

Scientific Readiness: The fusion application scenario developed by IPP implies that a coupled simulation can be expressed in the form of a workflow, where each of the three submodels involved (1D transport, 2D equilibrium and 5D turbulence) uses a standardized interface made of Consistent Physical Objects (CPO). Development work has been done towards the addition of CPO interface for all submodels in our Extreme Scaling scenario. At least one instance of each type of submodel has been completed (ETS for transport, CHEASE for equilibrium, GEM and dFEFI for turbulence), and we will use these codes for the fast-track version of our ES application, where the focus is on bridging time scales between turbulence and transport. Work is ongoing for the interface of ORB5, a global gyrokinetic code for turbulence that serves as our primary model for the deep-track ES application (Figure 9). Besides finalizing the interface implementation, we need to find the right prescription for calculating transport and understanding how the profiles can be updated in a global code. We are also developing several toy turbulence models which will have the same characteristics as dFEFI and ORB5 in terms of noise. As these toy models are a lot cheaper computationally, we can test a large variety of scale bridging scenarios without requiring huge computing allocations.

For the deep-track application based on Heterogeneous Multiscale Computing pattern, the dFEFI submodel needs to be extended so each fluxtube can run independently. In this HMC application, we will consider scale bridging in space as well as in time: we will study especially how the positions of fluxtubes in space can affect averaging parameters for the noisy turbulence results.

Readiness to operate with ComPat high-level tools and middleware services:

We have revived the coupled application developed in the MAPPER project (http://www.mapper-project.eu), which utilises MUSCLE, a less expensive 3D gyrofluid submodel (GEM), and simplified 1D submodels (BOHMGB, GEM0) for turbulence. In addition, we updated, from the last version available, the physics codes, the generic data-structure along with its associated tools and libraries, and MUSCLE2 (fixes the Fortran API). The updated software is installed on the EEE, on the following machines: SuperMUC (LRZ), Eagle and Inula (both PSNC). We also implemented a preliminary MUSCLE2 kernel for the 5D gyrokinetic fluxtube code (dFEFI), and are in the process of preparing initial test scenario for this fast-track ES application.

Work is also ongoing for profiling and debugging dFEFI and ORB5 codes using Allinea tools (WP4). Scalability of ORB5 has been verified on a CYCLONE equilibrium test case, in which CYCLONE is a project created and operated from 1996 – 2000 by the U.S. Department of Energy to benchmark fusion codes against various experiments. Strong scaling from 1,000 to 8,000 cores shows a parallel efficiency of 90%.

Timescales for ComPat Component Integration:

- **Current Components**: MUSCLE2 (requires latest version from SVN), debuggers and profilers, MPI.
- **Components to be added in M12-M24**: QCG-Client Tools (for submission on different systems on the EEE), Job and Advance reservation manager, Scheduler and resource allocator, Performance Prediction Models (from WP2).
- Other Deep-track components to be added when they become available: the HMM manager and the on-the-fly database, to be developed in WP2, for the HMC applications.

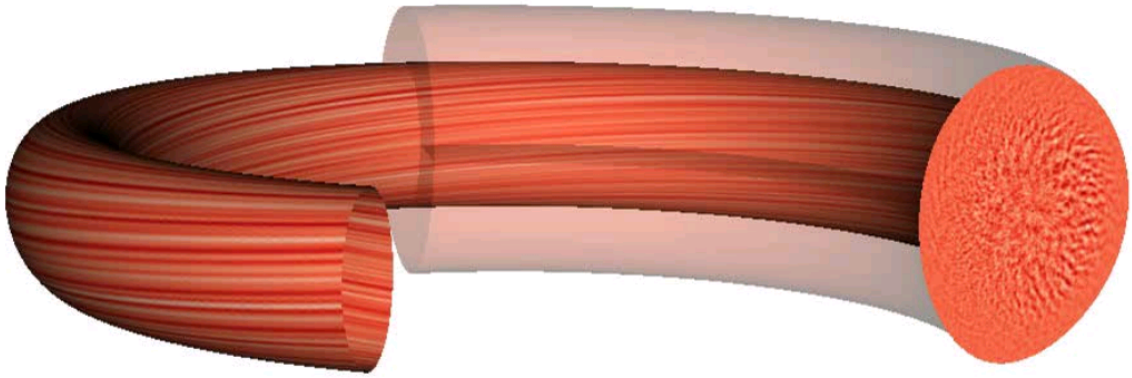

Figure 9: Snapshot of the perturbed electrostatic potential from a ORB5 simulation of ion-scale plasma turbulence for the ASDEX Upgrade tokamak, including kinetic electrons. A cut-away view of a single flux surface visualizes the inherent anisotropy of the tokamak.

2.2.4 Astrophysics – Milky-Way Galaxy simulation fast-track

Grand Challenge: Understanding the formation processes of stars in their clustered environment, as well as the origin and propagation of structure in the stellar disk of the Milky Way Galaxy.

HPMC pattern: Extreme Scaling (fast-track) / Heterogeneous Multiscale Computing (deep-track)

Scientific Readiness: Leiden is working on three prongs for multi-scale applications. In one project we are attempting to simulate the formation and early evolution of a star cluster, from the giant molecular cloud, via the star and planet formation process all the way to the first supernova.

Within this project we are in the process of coupling seven astrophysical models together into one solvable numerical environment. We have addressed the extreme multi-scale and multi physics problem of star and planet formation. The range of numerical solvers we propose to use has never been combined before, and such a wide range of temporal and spatial scales covered in the simulation have also not been considered.

In Figure 10, we present the first results of this coupled simulation, which is coupled using AMUSE (the Astrophysical Multipurpose Software Environment). The bright objects are stars, rendered with a Hubble Space Telescope point spread function. The dark material is interstellar dust, and the reflective nebulosity is hydrogen gas in the simulation. The planets are not visible, but their evolution is taken into account self consistently. At this stage the cluster is about 1.6Myr old, and by the time we took this snapshot it contained about 2000 stars and about 800 planets, varying in mass from the mass of the Earth to about 100 times the mass of Jupiter.
The extreme scale in this simulation is addressed with the Nemesis module in AMUSE [21]. This module solves the gravity for each planetary system with a separate symplectic N-body integrator, whereas the global dynamics of the cluster is addressed with a direct N-body solver. The hydrodynamics is addressed using a Smoothed Particles Hydrodynamics code, and stellar evolution is taken care of using the SSE parametrized stellar evolution code [22]. The coupling between the hydrodynamics and the gravitational N-body dynamics is realized with the 2nd order bridge [23].

We are currently preparing several papers describing these advances:

- Pelupessy, I., *et. al,* "The Oceanographic Multipurpose Software Environment (OMUSE)"
- Pelupessy, I., *et. al,* MNRAS, "Self-consistent High-order Symplectic coupling between gravitational systems"
- Portegies Zwart, S.*, et. al,* MNRAS, "Multi-scale and multi-physics simulation of the formation of planetary systems"
- Wall, J.*, et. al,* "Young Star Clusters Starting with Gas"
- Portegies Zwart, S. & Jeroen Bedorf, J., IEEE Software, "Fitting the Universe in a computer"
- van Elteren, A., *et. al*, "the Multipurpose software Environment"
- Fujii, M., *et. Al.,* "The dynamics of stellar disks in live dark-matter halo"

Timescales for ComPat Component Integration:

- **Current Components**: AMUSE, MPI / MPI_Comm_spawn. In the foreseeable future, we will be installing the coupled astrophysics simulations we describe above on the EEE as a fast-track application.

- **Components to added in M12-M24**: MUSCLE2 / MUSCLE-HPC (Fortran API), QCG-Client Tools (for submission on different systems on the EEE), Job and Advance reservation manager.
- Other Deep-track components to be added when they become available.

Figure 10: Snapshot from the coupled astrophysics simulation, performed by coupling 7 astrophysical models together into one solvable numerical environment.

3 Conclusions

The software readiness of the applications in the ComPat project has been addressed in this document. The division into fast-track and deep-track applications allows us to develop and refine not only the ComPat infrastructure and technology efficiently (using the fast-track applications), it also allows us to consider new and novel multiscale scenarios for grand challenge applications (for deep-track applications) and develop the scientific advances required.

The ComPat architecture (discussed in detail in Deliverable 5.1) is currently being implemented, deployed and verified. As specified in Deliverable 5.1, using in-depth analysis, supported by questionnaires completed by the relevant Work Packages, the architecture is designed for the multiscale simulation scenarios envisaged for tackling our grand challenges. In the next phase of the project, therefore, the multiscale applications will be instantiated using the ComPat high-level tools, computing patterns, middleware and execution environment. At the same time, in WP 2, we will be developing software to instantiate the applications from an xMML description and information on the

performance of the single scale models and scale bridging methods comprising the multiscale application.

This document sets out which parts of the ComPat architecture each application will be using and gives a timetable for their integration, with the task of producing high fidelity scientific data (for the fast-track applications) by the end of M24. In WP3, we will therefore be working in close collaboration with work packages WP5, WP6 and WP2. With WP2, we will firstly be devising xMML specifications for each multiscale computing scenario. By the end of the M18, we will execute the fast-track applications from the application xMML on the EEE by using the tools developed in WP2. When available, we will integrate the applications with software developed to create configuration files and execution scripts produced from generic task graphs.

It is clear that this division into fast and deep-track applications (and components) will allow the project to be more agile; we will be able to proceed in the next phase with applications that use the extreme scaling and replica computing patterns, while at the same time we can develop the HMC pattern (the HMM manager and the on-the-fly database). It will provide example applications to develop the software in WP 2, test the ComPat technology stack in WP 5 and the EEE (WP 6). At the same time, we will develop deep-track applications to use the software developed for generic MCPs. As part of the annual project meeting, there will be a week-long workshop (10-14th October 2016), where all members of the project will be working towards installing the applications on the EEE, integrating them with the ComPat stack (as described in Section 2.2) and developing application xMML and task graphs (see Deliverable 2.1). We also plan to have numerous face-to-face meetings between members of WP 2, WP 3 and WP 5 during the next reporting period to continue these collaborations.

Plans for Year 2:

1. Produce a full xMML specification for each application (with WP 2).
2. With WP 2, decide how we map the applications onto generic task graphs.
3. For fast-track applications, ensure execution on EEE using the ComPat technology stack (with WP 5 and WP 6). WP 2 can use the execution scripts as examples for designing software to transfer between task graphs and configuration files / execution scripts.
4. Perform performance analysis for single scale models and scale bridging methods (using tools from WP 4) on each HPC resource to profile the runtime behaviour of each part of the multiscale model on different resources.
5. Using tools from WP 2, test the fast-track applications using execution recipes from application specific task graphs (Month 18).

6. For deep-track applications, test single scale models and scale bridging methods on the EEE as development is finished, including performance models.
7. Test the software developed in WP 2 that uses generic task graphs to generate input / configuration for each application as they become available.

4 References

[1] S. Manini, *et al.* "PyNS: An open-source framework for 0d haemodynamic modelling." *Annals of biomedical engineering* 43.6 (2015) 1461-1473.

[2] L. Mountrakis, E. Lorenz, O. Malaspinas, S. Alowayyed, B. Chopard, A. G. Hoekstra, "Parallel performance of an IB-LBM suspension simulation framework", *Journal of Computational Science* 9 (2015) 45–50.

[3] www.palabos.org, last accessed 18 Jul 2016.

[4] D. a. Fedosov, B. Caswell, A. S. Popel, G. E. M. Karniadakis, "Blood Flow and Cell-Free Layer in Microvessels", *Microcirculation* 17 (8) (2010) 615–628.

[5] S. Alowayyed, L. Mountrakis , E. Lorenz , A. G. Hoekstra, "Load Balancing of Parallel Cell-based Blood Flow Simulations", *Computer Physics Communications* (submitted), 2016.

[6] L. Axner, J. Bernsdorf, T. Zeiser, P. Lammers, J. Linxweiler, A. G. Hoekstra, "Performance evaluation of a parallel sparse lattice Boltzmann solver", *Journal of Computational Physics* 227 (10) (2008) 4895–4911.

[7] G. Karypis, and K. Vipin, "A fast and high quality multilevel scheme for partitioning irregular graphs." *SIAM Journal on scientific Computing* 20.1 (1998) 359-392.

[8] A. Caiazzo, *et al.* "A Complex Automata approach for in-stent restenosis: Two-dimensional multiscale modelling and simulations". *Journal of Computational Science* 2 (2011) 9–17.

[9] H. Tahir, C. Bona-Casas, A. G. Hoekstra, "Modelling the effect of a functional endothelium on the development of in-stent restenosis." *PLoS One* 8 (2013) e66138.

[10] H. Tahir, C. Bona-Casas, A. J. Narracott, J. Iqbal, J. P. Gunn, P. V. Lawford, "Endothelial repair process and its relevance to longitudinal neointimal tissue patterns: comparing histology with i*n silico* modelling", *Journal of the Royal Society Interface* 11 (94) (2014) 20140022.

[11] H. Tahir, A. G. Hoekstra, E. Lorenz, P. V. Lawford, D. R. Hose, J. P. Gunn, "Multi-scale simulations of the dynamics of in-stent restenosis: impact of stent deployment and design". I*nterface Focus* 1 (2011) 365–73.

[12] C. M. Amatruda, C. Bona-Casas, B. K Keller, H. Tahir, G. Dubini, A. G. Hoekstra, "From histology and imaging data to models for in-stent restenosis" *International Journal of Artificial Organs* 37 (2014) 786–800.

[13] H. Tahir, I. Niculescu, C. Bona-Casas, R. M. H. Merks, A. G. Hoekstra, "An in silico study on the role of smooth muscle cell migration in neointimal formation after coronary stenting" *Journal of the Royal Society Interface* 12 (2015) 20150358.

[14] R. S. Schwartz, A. Chu, W. D. Edwards, S. S. Srivatsa, R. D. Simar, J. M. Isner. "A proliferation analysis of arterial neointimal hyperplasia: Lessons for antiproliferative restenosis therapies" *International Journal of Cardiology* 53 (1996) 71–80.

[15] R. S. Schwartz RS, Holmes DR, Topol EJ. The restenosis paradigm revisited: An alternative proposal for cellular mechanisms. *Journal of the American College of Cardiology* 20 (1992) 1284–93.

[16] R. D. Kenagy, J. W. Fischer, S. Lara, J. D. Sandy, A. W. Clowes, T. N. Wight, "Accumulation and Loss of Extracellular Matrix During Shear Stress-mediated Intimal Growth and Regression in Baboon Vascular Grafts". *Journal of Histochemistry & Cytochemistry* 53 (2005) 131–40.

[17] A. Farb, F. D. Kolodgie, J. Y. Hwang, A. P. Burke, K. Tefera, D. K. Weber "Extracellular matrix changes in stented human coronary arteries" *Circulation* 110 (2004) 940–947.

[18] D. Wright, B. Hall, O. Kenway, S. Jha, P. V. Coveney, “Computing clinically relevant binding free energies of HIV-1 protease inhibitors” *Journal of Chemical Theory and Computation* 10 (2014) 1228-1241

[19] S. Wan, B. Knapp, D. Wright, C. Deane, P. V. Coveney, “Rapid, Precise, and Reproducible Prediction of Peptide−MHC Binding Affinities from Molecular Dynamics That Correlate Well with Experiment” *Journal of Chemical Theory and Computation* 11 (2015) 3346-3356

[20] J. L. Suter, D. Groen, P. V. Coveney, “Mechanism of exfoliation and prediction of materials properties of clay-polymer nanocomposites from multiscale modelling” *Nano Letters* 15 (2015) 8108–8113

[21] F. L. Pelupessy, *et al.* "The astrophysical multipurpose software environment" *Astronomy & Astrophysics* 557 (2013) A84.

[22] J. R. Hurley, O. R. Pols, C. A. Tout, “Comprehensive analytic formulae for stellar evolution as a function of mass and metallicity” *Monthly Notices of the Royal Astronomical Society* 315 (2000) 543

[23] N. de Vries, S. P. Zwart, and J. Figueira. “The evolution of triples with a Roche lobe filling outer star.” *Monthly Notices of the Royal Astronomical Society* (2014) stt1688.

		Nanomaterials	Biomedicine					Fusion		Astrophysics
		"on-the-fly" coarse-graining (HMC) (RC)	Aneurysm flow dynamics (ES) (HMC)	In-stent restenosis (ES)(RC)	**Binding affinities (RC)**	RBC and platelet transport (ES)	Blood rheology (HMC)	**Global turbulence simulation (ES)**	Flux-tube chain (HMC)	**Milky-Way Galaxy simulation (ES)** (HMC)
ComPat high-level tools	**QCG client tools**	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö	
	FabSim	Ö	Ö		Ö					
	Monitoring					Ö	Ö	Ö	Ö	Ö
	HPMC Pattern Code Generators	Ö	Ö		Ö		Ö	(Ö)	(Ö)	Ö
	Debuggers						Ö	Ö	Ö	
	Profilers	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö
	Performance Prediction Models	Ö	Ö	Ö	Ö	Ö	Ö	(Ö)	Ö	Ö
Application libraries and parallel programming	**ADIOS (Parallel I/O)**					Ö			Ö	Ö
	MPI	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö
	CPU Parallel	Ö	Ö							Ö

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No 671564.

toolkits	**Library**	**Ö**	**Ö**							**Ö**
	GPGPU Parallel Library	**Ö**								**Ö**
	ECOL		**Ö**			**Ö**			**(Ö)**	
Patterns' libraries	**MUSCLE**	**Ö**	**(Ö)**	**Ö**		**Ö**	**Ö**	**Ö**	**Ö**	
	MPWide	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**		**(Ö)**	**Ö**
	AMUSE									**Ö**
Patterns' services	**Pilot jobs**				**Ö**					
	HMM Manager	**Ö**					**Ö**		**(Ö)**	**Ö**
	On-the-fly database	**Ö**		**Ö**					**(Ö)**	
Middleware Services	**Energy-aware Scheduler /Resource coordinator**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**		**Ö**	**Ö**
	Jobs and advance Reservation Manager	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**
	Workflow	**Ö**		**Ö**	**Ö**		**Ö**	**(Ö)**	**(Ö)**	

	Manager	**Ö**		**Ö**	**Ö**		**Ö**	**(Ö)**	**(Ö)**	
	Data Transfer (grid FTP/ Globus)	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**Ö**	**(Ö)**	**Ö**	
	ECOS	**Ö**	**Ö**	**Ö**		**Ö**	**Ö**		**Ö**	
	MTO		**(Ö)**	**Ö**		**Ö**	**Ö**		**Ö**	

Table 1: The required components of the ComPat architecture for each application. The fast-track applications are in red and the fast-track ComPat technology components are in Blue. Bold ComPat technology components are deep-track. The ticks in brackets indicate that the use of the component depends on the development of the application. Each of the components is described in detail in Deliverable 5.1.