Template proposal for computing time applications

*This template is based on the Gauss Center for Computing (GSC) calls and the PRACE project access calls. The actual kind of information requested for the different calls might very.*

0 General project information 1

1 Introduction 1

2 Preliminary Work 1

3 Project Details 1

4 Description of Methods and Algorithms 2

5 Work Schedule 2

6 Code performance 2

7 Estimated computing resources needed on <supercomputer> 3

8 References 5

# 0 General Project Information

* New project proposal / Project extension for project <project name>
* Period: mm yyyy – mm yyyy
* Project title: <project title>
* Principal investigator/project leader: <name>
	+ Job title
	+ Organisation name
	+ For commercial companies:
		- Is the head office of the organization in Europe?
		- % of R&D activity in Europe as compared to total R&D activity
	+ Department
	+ Group
	+ Address
	+ Country
* Project contributors/collaborators

# 1 Introduction

* Give a short outline of the scientific background of your research, including references. (About 1 page)

# 2 Preliminary Work

* Provide a brief summary of your preliminary work in connection with the proposed project, including references. (About 1 to 2 pages)

# 3 Project Details

* Describe your proposed project in detail, structured in sub-projects and/or steps, if applicable.
* Please, include information about:
	+ Scientific questions you want to address
	+ Scientific goals you want to reach
	+ Approach to reach these goals
	+ Expected impact on the research area
	+ Novelty and timeliness of the proposal
* About 1 to 2 pages per sub-project/step

# 4 Description of Methods and Algorithms

* Describe the scientific and numerical methods and algorithms you are planning to use/improve/develop.
* Codes, packages and libraries needed to undertake the project, how will these enable the research to be achieved
* About 1 to 2 pages

# 5 Work Schedule

* Provide a short work schedule for each sub-project/step.
* Include a table and/or Gantt chart.
* About 0.5 pages per sub-project
* Example for a Gantt chart:

Figure 1 Work schedule for the project (Gantt chart)



# 6 Code performance

* Elaborate why <supercomputer> is suitable for your proposed project and why. Include information about:
	+ Which code(s) will be used
	+ How is the code parallelized (pure MPI, mixed MPI/OpenMP or Pthreads)
	+ Amount of memory necessary (per core/node and in total)
	+ Scaling plots and table with speedup results for runs with typical parameter sets and problem sizes of the planned project, including I/O
	+ Describe architecture, machine/system name, and problem size used for the scaling plots (about 1 to 2 pages)
* Describe your and your team’s experience with the requested kind of architecture.
* An example table and a scaling plot:

Table Scaling behavior of <code> on <architecture> and <system> at <location>. This test was performed with 5\*106 particles; absolute timings per time step (s) and relative speedup normalized to 1024 cores are given.

|  |  |  |
| --- | --- | --- |
| #Cores | Absolute timing (s) | Speedup |
| 1024 | 189.6 | 1.0000 |
| 2048 | 99.0 | 1.9154 |
| 4096 | 55.6 | 3.4088 |
| 8192 | 30.8 | 6.1376 |

Figure 2 Scaling behavior of <code> on <architecture> and <system> at <location>. This test was performed with 5\*106 particles.



* Hints for BlueGene/Q architectures:
	+ In order to use this architecture efficiently, pure MPI codes must use at least 2 tasks per core and hybrid codes must use at least 2 threads per core
	+ Scaling data up to at least 8192 cores needs to be presented; within this efficient scaling up to 4096 cores must be demonstrated

# 7 Estimated computing resources needed on <supercomputer>

* Outline the amount of resources you request for the current granting period and describe in detail:
	+ Planned runs
		- Problem size for planned runs
		- Number of runs planned per sub-project and code
		- Number of jobs that can run simultaneously, i.e. jobs that do not depend on each other
	+ Are you able to do check pointing?
		- If yes: Maximum time between two checkpoints (in hours)
	+ Expected job size (minimum, average and maximum):
		- Estimated computing time per run: wall clock time of a typical job execution in appropriate units, preferred is core hours (core-h)
		- Number of CPU cores/nodes per run, sub-project and code
		- Job memory (total memory usage over all cores of jobs)
	+ Total amount of requested computing time
	+ Computing time for pre- and/or post processing
	+ Storage
		- Total amount of storage required at a time
			* For scratch files during simulation, log files and checkpoints (scratch)
			* Results and large input files (work)
			* Source code and scripts (home)
			* Archive storage (archive)
		- Number of files to be stored (scratch, work, home and archive)
	+ Data transfer: total amount of data to be transferred to/from the production system (in GB/TB)
	+ I/O
		- Describe the I/O strategy of the code (for example usage of I/O libraries, MPI I/O, netcdf or HDF5)
		- Be aware that I/O has to be adequately managed for highly parallel applications, especially in case of I/O intensive applications.
		- In general, serial I/O is not suitable for large amounts of data.
		- Example:
			* I/O data traffic (read and write) per hour for a typical production job (in GB)
			* Number of files generated per hour for typical production jobs
* Examples for estimating resources:

Table 2 Example 1 for estimating resources (with sub-projects)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sub-project | Size (particles) | Single $Δt$ 16384 cores | $Δt$ per run | Core-h per run | # Runs | Core-h / sub-project |
| SP 1 | 8\*106 | 40s | 500 | 91,00 | 50 | 4.50 m |
| SP 2 | 2\*106 | 20s | 1000 | 94,00 | 100 | 9.35 m |
| … |  |  |  |  |  |  |
| Total requested time on <supercomputer>: **n** core-h |

Table 3 Example 2 for estimating resources (with steps per run)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Run type | # Runs | # Steps/ run | Wall time/ step | # CPU cores | Total core-h/ run type |
|  | r1 | s1 | w1 | p1 | r1\*s1\*w1\*p1 |
|  | r2 | s2 | w2 | p2 | r2\*s2\*w2\*p2 |
|  | r3 | s3 | w3 | p3 | r3\*s3\*w3\*p3 |
| … |  |  |  |  |  |
|  |  |  |  |  | *Total* |

# 8 References

<list>