**Intro**

This is a project plan for continuing migrating unstructured grid editing functionality from Fortran to C++. The goal of the first phase was to prove the efficiency of the C++ implementation when performing mesh orthogonalization, which is the most complex grid operation available in interactor (Fortran).

The efficiency was proved to be comparable to the Fortran one, and even better in multithread settings. Moreover, the code interfaces better with the C#/C++ GUIs, has a better structure, is unit-testable and implements the algorithms in fewer lines of code.

In this project plan, a possible way forward is proposed.

**The plan**

In this phase medium/small size functions (denoted with letters M and S) will be ported, unit tested and connected to the Grid Editor. The proposed steps are the following:

1. Complete orthogonalization with snap to land boundary option (M, coding finding land boundaries algorithms, Dijkstra’s algorithm).
2. Implement snap to land boundary (S, after the item above, coding this part is straightforward).
3. Implement splines equations (S).
4. Implement curvilinear to unstructured grid converter (S).
5. Implement generate curvilinear mesh from splines (M, orthogonal version from Sander, with the desired interactivity in the GUI to stop the algorithm).
6. Implement generate curvilinear mesh from splines (M, transfinite version from Herman, interactivity as above).
7. Add a third-party triangulation library (M, could be triangle.c from R. Shewchuk or the Sepran library from TU Delft).
8. Implement triangular mesh generation within a polygon (S).
9. Implement polygon edges refinement. This functionality is needed to produce triangulations in polygons with user-defined resolutions (M).
10. Implement the algorithm to convert mesh boundaries to polygon (M).

The number of functionalities that can be coded in the second phase is larger than phase one because many of the algorithms rely on data structures filled during the administration phase, which has been already coded.

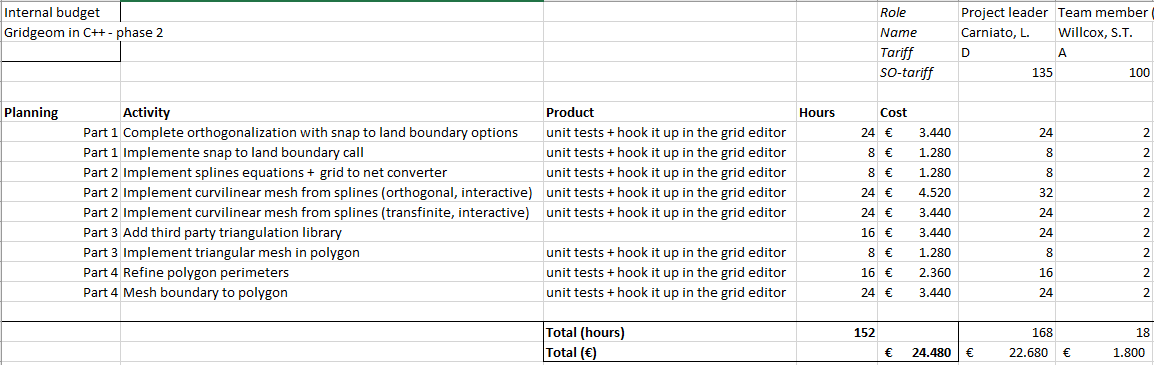
After implementing the points above, the remaining functions to replace completely the Fortran dll with the gridgeom++ requires the implementation of:

* Grid refinement from samples
* Making grids from a table of parameters (e.g. grid in squares with defined dx and dy)
* Edge/nodes manipulations (merge nodes, move nodes, delete edges).

The coding of these additional operations will be postponed in a third phase. In this phase of the project Scott Willcox is proposed as C++ code reviewer.

**Time / budget / planning / team**

See the table below, which is also available as a spreadsheet:



The requested budget for the project is € 24.480,--. At the end of the project, the results with the new library will be demonstrated in the Grid Editor.

**Risks and measures**

Based on the previous experience with phase one, not much risk is foreseen in re-writing the algorithms in terms of reverse engineering current Fortran code.

|  |  |
| --- | --- |
| Risk | Using Sepran triangulation in Gridgeom++ requires too much re-design. |
| Measure(s) | Triangle.c will be used instead |

|  |
| --- |
| Copy for  To be defined. |
|  |