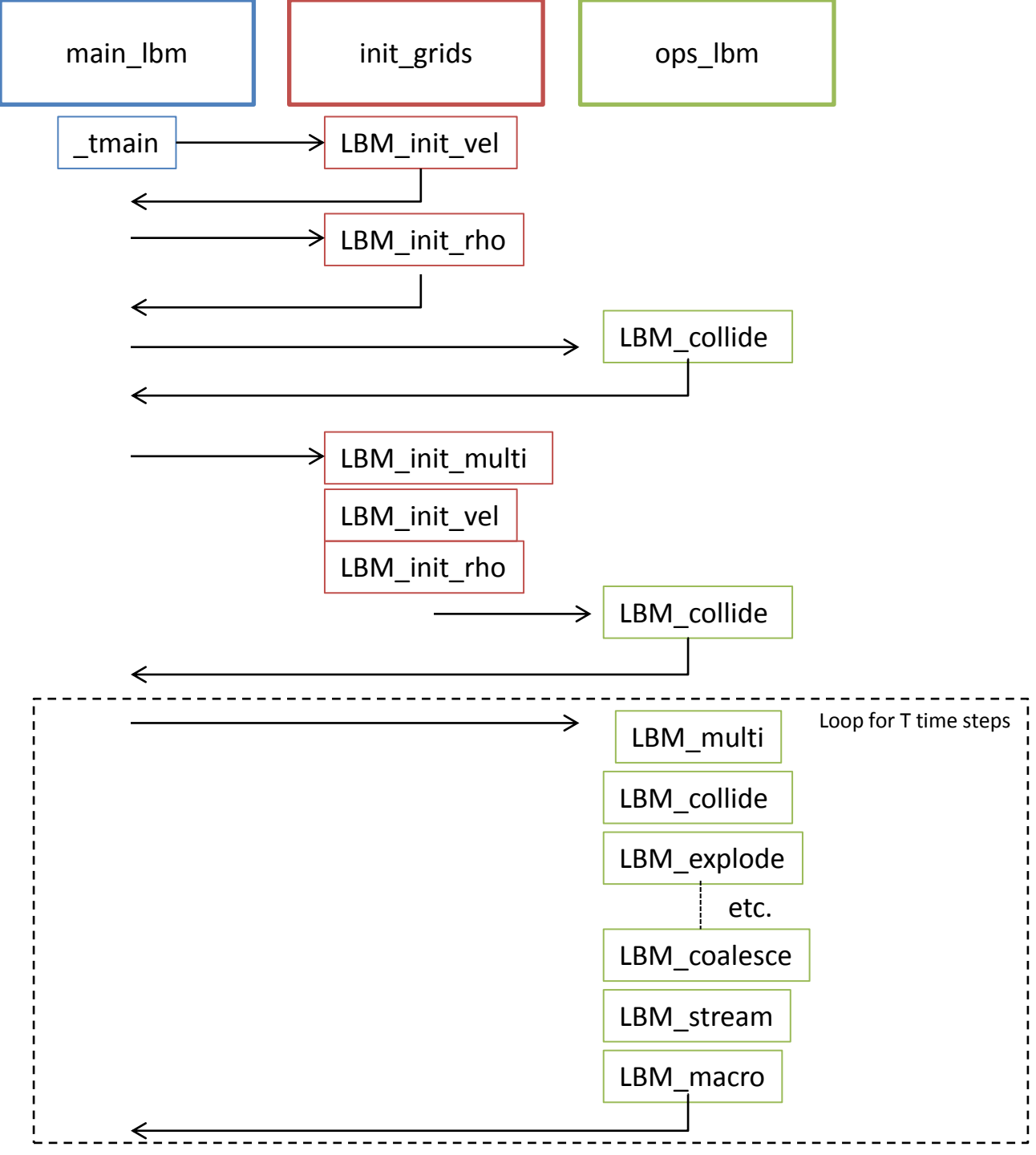

Description

- Initialise Level 0 grid
- Initialise L0 velocity
- Initialise L0 density

- Initialise L0 f to feq

- Initialise refined grids
- Initialise Lr velocity
- Initialise Lr density
- Initialise Lr f to feq

- Enter main algorithm
  - Collide
  - Explode
  - etc.

  - Coalesce
  - Stream
  - Update Macroscopic

`struct` GridData

```
std::vector<int> XInd;
std::vector<int> YInd;
std::vector<int> ZInd;

std::vector<double> XPos;
std::vector<double> YPos;
std::vector<double> ZPos;

std::vector<double> f;
std::vector<double> feq;
std::vector<double> u;
std::vector<double> rho;

std::vector<int> LatTyp;

double omega;
double dx;
double dy;
double dz;
double dt;
```

This structure (public class in C++ speak) is initialised for every grid. At the moment only one grid at each refinement level is allowed. Code stores each instance of this structure in an array called `Grids[ ]`.

Vectors of indices of the lattice sites on the grid. On L0 these vectors identify which of the sites are refined.

Vectors of positions in a global reference frame of the lattice sites on the grid

Arrays of populations, and macroscopic quantities for every lattice site on the grid

Array of labels assigned to each lattice site upon initialisation which identifies whether it is to be operated on or passed over by operating subroutines.

Relaxation time and lattice site spacing for grid.