Easy profiling using macros

- The include header util_profiling.h contains utilities for profiling your code. For instance during a simulation loop one wants to time a certain computation or monitor say the kinetic energy, then one writes
- #include <util_profiling.h>
- while(true)
- {
  - START_TIMER(“MY_TIMER”);
  - …. do the computation….
  - STOP_TIMER(“MY_TIMER”);
  - RECORD(“ENERGY”, my_value);
- }
- Notice that all that is needed are the macros START_TIMER, STOP_TIMER and RECORD

Writing Profiling Data

- Once simulation is over one would want to write the profiled data. This is done by obtaining the corresponding monitor and getting the values stored during simulation. Here is the outline of the code for doing this:
  - std::ofstream file;
  - ……
  - Profiling::Monitor * E = Profiling::get_monitor("MY_ENERGY");
  - file << "E = " << util::matlab_write_vector(E->get_values()) << ";" << std::endl;
  - Profiling::TimerMonitor * T = Profiling::get_timer_monitor("MY_TIMER");
  - file << "T = " << util::matlab_write_vector(T->get_values()) << ";" << std::endl;
- Notice that string values are used to uniquely identify both Monitors and TimerMonitors. The application programmer can name these as he/she pleases. In the example code above we use the matlab write vector utility function from the header file util_matlab_write_vector.h file