Remove all cyclic dependencies

11/14/2015 11:37 AM - David van der Spoel

<table>
<thead>
<tr>
<th>Status:</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td></td>
</tr>
<tr>
<td>Category:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
</tbody>
</table>

Description

In order to make the code truly modular we would have to get rid of all cyclic dependencies. We check for those already during building but as a point of reference I thought it would be good to write down a hierarchy. In the long run every level in the hierarchy should only depend on the levels above. Before we reach this goal we have to remove legacyheaders (#1415). Trying to sketch the hierarchy I got stuck at level 3 already, where topology and fileio are cyclically dependent. So let us work on this and improve the modularity further.

0) external libraries, including system, thread_mpi, MPI etc.
1) utility
2) math
3) mdtypes, topology, fileio

Associated revisions

Revision e8e13a0e - 11/15/2015 10:21 AM - David van der Spoel
Merged legacyheaders/types/state.h into mdtypes/state.h
Also merged swap/enum.h into legacyheaders/types/enums.h
to be moved later out of legacyheaders.
Part of #1415, #1854.

Change-id: le53f190d6798b81c531895de80ef25c060836479

History

#1 - 11/14/2015 01:09 PM - Gerrit Code Review Bot
Gerrit received a related patchset '1' for Issue #1854.
Uploader: David van der Spoel (davidvanderspoel@gmail.com)
Change-id: le53f190d6798b81c531895de80ef25c060836479
Gerrit URL: https://gerrit.gromacs.org/5338

#2 - 11/14/2015 04:41 PM - Teemu Murtola
Not sure if you know it, but we already have a visualization script for the dependencies, that also lays out the modules in a hierarchy. And it highlights all suppressed cycle edges in red, and ignored those in the layout. This is described in docs/dev-manual/gmxtree.rst. This probably provides a good starting point for thinking about the dependencies. And at least for the suppressions that I've put in docs/doxygen/cycle-suppressions.txt, I've put some thought into thinking about the dependency that I want to suppress, so working to remove those marked dependencies would provide one starting point.

#3 - 11/14/2015 05:13 PM - David van der Spoel
Impressive stuff. Also scary. Looks like it will be a while before we make real inroads towards the goal of modularity.
We have funding though Horizon 2020 project BioExcel for the next 3 years that talks about “GROMACS will be turned into a state-of-the-art module-based C++ library.” There’s no specific requirements for us to fulfill, but it’s going to get a slice of at least my time :-) Also as part of that project, CPMD wants to re-write its QM/MM implementation to call GROMACS force routines rather than GROMOS (for both performance and licensing reasons). Making that less impossible will be one of my personal foci, because the associated cleanup will surely help us in the long term.

(Also, Michael Shirts, Peter Kasson and I have an application under review that includes the objective of implementing a GROMACS API, which would also be thought to be very welcome for lots of folks.)