Gromacs - Task #1430

FindGROMACS.cmake version detection

02/02/2014 09:00 PM - Teemu Murtola

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>Teemu Murtola</td>
</tr>
<tr>
<td>Category:</td>
<td>build system</td>
</tr>
<tr>
<td>Target version</td>
<td>5.1</td>
</tr>
</tbody>
</table>

**Description**

Currently, share/template/cmake/FindGROMACS.cmake hardcodes several function names, one for each version of Gromacs that it can identify, and uses a series of check_library_exists() with these to detect the version. This requires adding such a function for each new version, which requires extra maintenance and can become more and more complex if the code moves to C++.

Additionally, the current logic is confusing (although correct) in that it uses init_domdec_vsites: this function exists already in Gromacs 4.6, but in libmd, which isn't linked in while doing the checks (unless the user overrides the libraries).

Since 4.6, gromacs/version.h has existed, and would provide a simpler way for detecting the exact version. The header is automatically updated by the build system, at least until we reach a level of API stability that we may actually confidently say that we can keep the API version fixed between some releases, so this would hopefully reduce maintenance. An approach similar to what FindBoost.cmake does with boost/version.hpp should work relatively well here as well.

**Associated revisions**

Revision c6a675bf - 08/13/2014 11:38 PM - Teemu Murtola

Use native CMake mechanism for find_package(GROMACS)

Instead of various detection stuff that only worked well in the presence of pkg-config, use native CMake mechanisms: package configuration files and automatic export of library import definitions. Include directories and preprocessor macros influencing the installed headers are still propagated separately. The new mechanism also works for arbitrary suffixes, and is relocatable (as long as external libraries that GROMACS links against are not moved). A simple FindGROMACS.cmake is still there to hide some of the complexity to support multiple suffixes, but it is not strictly necessary if the using code wants to do those slightly more complex find_package() invocations directly.

Generalize the machinery that populated libgromacs.pc to use it also for this purpose.

TODO for later (requires changes that are better done outside this patch):
- Improve the versioning logic

The machinery and its current limitations are documented in the Doxygen page on using GROMACS as a library. Some of these could possibly be improved with additional effort, but the current approach is hopefully a reasonable compromise between usability, robustness, and complexity.

Closes #1430, #1554

Change-Id: l49c50375a5abebfe8614704c175e6ed22c9daa56

11/25/2015 1/2
We should at least replace init_domdec_vsites by something correct.

Teemu, do we have a good C function, which was added with 5.0?

For the purpose, anything from a new feature that is always built would do - e.g. an ljpm function like ewald_function_lj.

Gerrit received a related patchset `1` for Issue #1430.
Uploader: Teemu Murtola (teemu.murtola@gmail.com)
Change-Id: i49c50375a5abebfe8614704c175e6ed22c9daa56
Gerrit URL: https://gerrit.gromacs.org/3762

The CMake package configuration file patch solves this in a different way.

Target version changed from 5.x to 5.1

There's ongoing issues related to API stability, but we must deal with those later, and so can do so in a separate issue