openQA Project - action #68684

Port os-autoinst's build system to CMake

2020-07-06 16:52 - mkittler

Status:	Resolved	**Start date:**	2020-07-06
Priority:	Normal	**Due date:**	
Assignee:	mkittler	**% Done:**	0%
Category:	Feature requests	**Estimated time:**	0.00 hour
Target version:	Ready		
Difficulty:	medium		

Description

This idea is not new and there are already existing changes one can use as a starting point:
https://github.com/os-autoinst/os-autoinst/pull/1369

- AC0: The old build system still remains in place until the CMake-based build system has been tested sufficiently.
- AC1: Everything is built using CMake. There are no nested build systems. (That is actually the most annoying part of the current build system. The auto-tools based scripts invoke another Makefile generator to build the OpenCV bindings. This is responsible for a lot of issues within the current build system.)
- AC2: Support out-of-source-tree builds to avoid polluting the Git checkout and ease . (Likely we nevertheless need to create symlinks for certain binaries so the Perl runtime can find them without further hacks.)
- AC3: Don't hard code any compiler/linker flags which are better set from the outside. (E.g. the old build system hardcodes -g3 in some place. In the CMake port we should avoid doing that and rely on the standard way of the build system to configure a debug build.)
- AC4: The dependency lookup is done using CMake's find modules or pkg-config (instead of hard coding library names and paths).

History

#1 - 2020-07-15 14:31 - mkittler

- Status changed from New to In Progress

PR: https://github.com/os-autoinst/os-autoinst/pull/1470

This PR does not include everything. However, it is a good start and ready to be merged (since it does not include any messy WIP code and leaves the existing build system intact).

Building, installing and generating the documentation works already. Out-of-source tree builds are now finally possible. To use such builds for development (without installation) one has to create symlinks of the binaries within the source directory (via make/ninja symlinks).

Still missing:

- Targets for installing openvswitch
  - This should be directly supported by the CMake build script because it should make use of variables such as CMAKE_INSTALL_PREFIX to achieve a consistent installation.
- Targets for tidying C++ and Perl files
  - It is not hard to invoke the scripts by hand but targets would be nice to have.
- Targets for invoking the Perl testsuite and computing coverage
  - I'm not sure whether it makes sense to have these within CMake.
    - pros: These targets rely on a successful built so it would of course be nice if that dependency would be tracked within CMake. It is also a bit weird to use a different Makefile for testing. CTest also has some nice features, e.g. it allows to include/exclude tests conveniently using a regex.
    - cons: There are a lot of targets to consider and it might take a while until everything is in place.

#2 - 2020-07-15 16:43 - cdywan

mkittler wrote:

> Still missing:
>
> - Targets for installing openvswitch
>   - This should be directly supported by the CMake build script because it should make use of variables such as CMAKE_INSTALL_PREFIX to achieve a consistent installation.
> - Targets for tidying C++ and Perl files
>   - It is not hard to invoke the scripts by hand but targets would be nice to have.
> - Targets for invoking the Perl testsuite and computing coverage

    - I'm not sure whether it makes sense to have these within CMake.
        - pros: These targets rely on a successful built so it would of course be nice if that dependency would be tracked within CMake. It is also a bit weird to use a different Makefile for testing. CTest also has some nice features, e.g. it allows to include/exclude tests conveniently using a regex.
        - cons: There are a lot of targets to consider and it might take a while until everything is in place.

Having consistent dependency handling would definitiely be nice to have.

Adding these gradually seems fine to me anyway. We're also leaving the "old" ones in place for now, right? My concern if any would be that we're once again diverging from the commands used in openQA and this causes needless, sublte confusion all the time (like TESTS= assuming a different folder, different verbosity defaults etc.)

#3 - 2020-07-16 14:13 - mkittler

> We're also leaving the "old" ones in place for now, right?

Right.

> My concern if any would be that we're once again diverging from the commands used in openQA

CMake has of course a different usage than automake so breaking everyones build script is unavoidable when we finally remove the automake build script at some point. But we're switching from one standard build system to another and removing a lot of rough edges by the way so I guess that's ok.

---

target_link_options must not be used yet, see PR description https://github.com/os-autoinst/os-autoinst/pull/1473

#4 - 2020-07-16 16:43 - mkittler

Install target for openvswitch files and fix for target_link_options: https://github.com/os-autoinst/os-autoinst/pull/1475

#5 - 2020-07-22 10:28 - mkittler

PR for running tests and documentation has been merged: https://github.com/os-autoinst/os-autoinst/pull/1479

Only details like targets for tidying C++ and Perl files are missing. But before tweaking such details it would likely be more interesting to test the CMake build system within Travis and OBS.

#6 - 2020-07-23 16:26 - mkittler

PR for using CMake within the CI and OBS: https://github.com/os-autoinst/os-autoinst/pull/1482

#7 - 2020-07-24 10:57 - okurz

I saw that you added build instructions to README but the README also references INSTALL.asciidoc which explains the autotools way. Also you have removed the call of autotools based build so it is not tested anymore. As I would prefer to not remove autotools support right away but give it a bit of time I suggest you add back a (very simple) build-check of the autotools way and merge the build instructions from README with INSTALL.asciidoc

#8 - 2020-07-24 13:46 - mkittler

I've missed INSTALL.asciidoc indeed. Seems like this file hasn't been updated for a while, e.g. it suggests to install the openQA package for installing dependencies (although os-autoinst-devel would be the right package now). So likely the file deserves a general update and parts with are redundant to the main README.md or the openQA documentation should be removed.

> As I would prefer to not remove autotools support right away but give it a bit of time I suggest you add back a (very simple) build-check of the autotools way

Right, a simple build check within our CI would still be nice.

I've also created a PR to move openQA to CMake: https://github.com/os-autoinst/openQA/pull/3283

However, we could actually *not* merge it and use is as an occasional check for the autotools build system.

I've tested the CMake-based RPM on openqaworker11 now (with https://github.com/os-autoinst/os-autoinst/pull/1484) and it basically works. I couldn't execute a full test run because something seems wrong with worker setup.

#9 - 2020-07-28 09:14 - okurz

mkittler wrote:

> I've missed INSTALL.asciidoc indeed. Seems like this file hasn't been updated for a while, e.g. it suggests to install the openQA package for installing dependencies (although os-autoinst-devel would be the right package now). So likely the file deserves a general update and parts with are redundant to the main README.md or the openQA documentation should be removed.
>
>> As I would prefer to not remove autotools support right away but give it a bit of time I suggest you add back a (very simple) build-check of the autotools way
>
> Right, a simple build check within our CI would still be nice.
>
> I've also created a PR to move openQA to CMake: https://github.com/os-autoinst/openQA/pull/3283
>
> However, we could actually *not* merge it and use is as an occasional check for the autotools build system.

I think we should use cmake build based environments in openQA as well to benefit from that. However as long as we keep automake we should have a test for that. As suggested in the meeting 2020-07-28 I suggest as the first step take a look into the packages in https://build.opensuse.org/package/show/devel:openQA/os-autoinst that currently do not build and try to fix them, maybe even with using automake in these cases which gives us automatic test coverage, e.g. SLE_12_SP5 finds no "ninja". As an alternative add an automake based in travis CI as parallel job

#10 - 2020-07-29 09:39 - mkittler

PR for using/testing autotools within SLE12 builds (merged, x86_64 build succeeds on devel:openQA):
https://github.com/os-autoinst/os-autoinst/pull/1488
PR for documentation: https://github.com/os-autoinst/os-autoinst/pull/1489

#11 - 2020-07-30 12:47 - mkittler

What's remaining is using CMake within openQA's CI. That will be a little bit tricky:
https://github.com/os-autoinst/openQA/pull/3283#issuecomment-666318629

#12 - 2020-07-31 13:12 - tinita

Is it intentional that os-autoinst travis-ci is now running tests twice? Once without coverage and then with?

#13 - 2020-08-04 11:55 - mkittler

tinita That's a bug. This PR should fix it as well as some of your other comments from the chat: https://github.com/os-autoinst/os-autoinst/pull/1498

#14 - 2020-08-06 12:28 - mkittler

- Status changed from In Progress to Resolved

I implemented all of my initial ACs and also added documentation, added further fixes/tweaks according to feedback and enabled the build within os-autoinst's and openQA's CI.

The only thing left is the target for C++ code formatting but so far it is not actively used anyways so I guess I can consider this issue resolved.

Nevertheless, there's a pending PR for the tidy-cpp target: https://github.com/os-autoinst/os-autoinst/pull/1502