Making a release

Don't do these steps from memory.

- Make sure the last run of the autobuilder was successful.
- Check trac to make sure there are no important pending tickets.
- Update `doc/share/` with `cd doc; make share-up`.
- Make sure `doc/NEWS.txt` is up-to-date. (Mention important known bugs!)
- Make sure `doc/README.txt` is up-to-date.
- Make sure `doc/FAQ.txt` is up-to-date.
- Make sure AUTHORS is up-to-date.

Make sure your system has up-to-date tools (Autotools, Swig, Doxygen, ...) before continuing.

- Bump the version number in `configure.ac`.
- Run `bootstrap`.
- Write the `ChangeLog` entry for all the above changes (But don't commit it before `distcheck`.)
- Run `make distcheck`.
- Commit all changes on success. Commit suicide otherwise.
- Tag the repository for the release.
- Append a `a` to the version number in `configure.ac` and commit this new change so that the next run of the autobuilder won't create a release.
- Copy the files created by `distcheck` to `/lrde/dload/vaucanson/`
- Create the release page on the LRDE wiki
- Update the Vaucanson page to point to it
- Send an announcement to vaucanson@lrde.epita.fr. The text of the announcement should explain what Vaucanson is (so we can forward the mail to another mailing list) and should include the list of major improvements since the last version (i.e., the top of `NEWS`). Do not assume that people will follow links to get details.
- If the release is a beta release, or an intermediate release before a major release, make it clear in the announcement and on the wiki.
- Install any new major release on `vcsn.enst.fr`.
- Complete and detail this list with what was missing (whatever will help the next guy doing the release).

Template arguments naming convention

Template arguments:

- A : Automaton structure.
- AI : Automaton implementation.
- S : Series.
- SI : Series implementation.
- W : a Word.

In the case where multiple possibilities could be used, suffix the template argument with the appropriate numbering. For example, to enable the use of two different automaton implementations for each argument of an algorithm:

```
template <typename A, typename AI1, typename AI2>
Element<A, AI1>
algorithm(const Element<A, AI1>& a1, const Element<A, AI2>& a2>);
```

Macros to handle with care

The VCSN_GRAPH_IMPL macro must only appear in three locations:

- include/vaucanson/context
- include/vaucanson/automata/generic_contexts
- include/vaucanson/misc/usual_macros.hh (be careful when defining new macros using it)

Any other use is irrelevant and may be very harmful. Moreover this macro must never be used in a file with guards.