ACS produces PreReleases and Releases aligned with ALMA delivery life cycle. This means that every 2 months a new release is scheduled for ALMA with name COMMON-<YYYY><MMM> (COMMON-2019JUN, COMMON-2019AUG, etc.), the development of a specific release starts around 2 months before the release. The "Requirements Deadline" is reached around 1 month before the end of development and after this last month ends, the "Pull Requests for Verification" should have been created. At this point the verification phase starts which lasts roughly 18 days (3 working weeks). In the case of COMMON, there's no validation phase, although some tickets may require validation, which should be carried out along an ONLINE-<YYYY><MMM> release validation phase. During this life-cycle, pre-releases and releases are created.

The dates for each of these phases is coordinated by the Release Manager and the information is persisted in Release Planning confluence page.

ACS gets built using a dedicated Jenkins build service and an on-demand docker infrastructure. Each release has 4 tasks configured:

- ExtProd-<OS>-<RELEASE>: Builds the external products and prepares an artifact with the results. Triggers ACS-<OS>-<RELEASE>.
- ACS-<OS>-<RELEASE>: Retrieves ExtProd-<OS>-<RELEASE> artifact and builds ACS, preparing an artifact as a result. Triggers ACS-test-<OS>-<RELEASE>.
- ACS-test-<OS>-<RELEASE>: Retrieves ExtProd-<OS>-<RELEASE> and ACS-<OS>-<RELEASE> artifacts and executes the test modules on ACS.
- Publish-COMMON-<RELEASE>: A parameter for either PreRelease or Release is chosen, which defines the destination directory in WebDav. Retrieves ExtProd-<OS>-<RELEASE> and ACS-<OS>-<RELEASE> artifacts and copies them to WebDav according to the given parameter.

ACS Pre-releases are intermediate roll-outs of the COMMON software, which are used by other developers to anticipate changes and to prepare their environments for development of their upcoming releases. In principle, it would be possible to have 1 ACS pre-release at the end of the development phase, along with preparing all the "Pull Requests for Verification". Usually, at least one additional ACS pre-release is prepared before, so developers can anticipate their work and have configured their building and testing environments beforehand. There are situations that also trigger additional ACS pre-releases, like finding important bugs during verification or development time.

Specially with sensitive changes (C++11, Java11, Python 3, etc.) that may last longer than 1 release, we try to roll-out more ACS pre-releases, in order to be updating what is available to other subsystems to current ACS/COMMON developments.

The ACS Release should be one per release at the end of the verification phase. If the verification phase went smoothly, it could very well contain exactly the same as the last ACS pre-release. In cases of bugs that required to be fixed during the verification phase, it would also include these bug-fixes into the release, even if there were no new ACS pre-releases including them at that point. There may be more than one ACS release, but only in the scenario where an important ticket is found after verification and needs to be added to the release.

All pre-releases and releases are stored in webdav ('http://webdav.sco.alma.cl/restricted/PreReleases/' and 'http://webdav.sco.alma.cl/restricted/Releases/', respectively) in a directory with the release name. Inside the release directory, there will be directories for each OS used to build the specific release (for instance 'http://webdav.sco.alma.cl/restricted/Releases/COMMON-2019APR/RH7.6/'). Inside this directory, there will be files both for ExtProd (External Products) and ACS itself with the following structure:

<Type>-<SHA>-<build-id>-<OS>-<Arch>.tar.bz2

- <Type>: Either ExtProd or ACS
  - ExtProd: These are ACS' external products, such as ACE/TAO, JacORB, OmniORBpy, RTI DDS, Python, Tcl/Tk, etc.
  - ACS: This is the core of ACS; everything in 'ACS/LGPL' and 'ACS/NO-LGPL' directories
- <SHA>: Last Git commit id that got in the build for the release/pre-release
- <build-id>: This is a per release unique incremental id used to identify a build of a release. This is obtained from the building system (Jenkins at this moment), and helps to relate a Jenkins task to a build, and to easily know which is the latest offered release (This could also be obtained from the Git commit id, but that is more work).
- <OS>: The operating system used to prepare the build for release
- <Arch>: The operating system architecture used to prepare the build for the release