# [DISCUSS] Code re-organization

> This page contains topics supporting ongoing discussion at dev@syncope.apache.org.

With SYNCOPE-620, the source tree was restructured in order to ease maintenance, allow easier extendability and obtain more modularity.



## Source tree <= 1.2.X

Up to 1.2.X, the effective source tree (e.g. excluding utility modules providing the standalone distribution, .deb artifacts, installer, ...) is organized into four main modules:

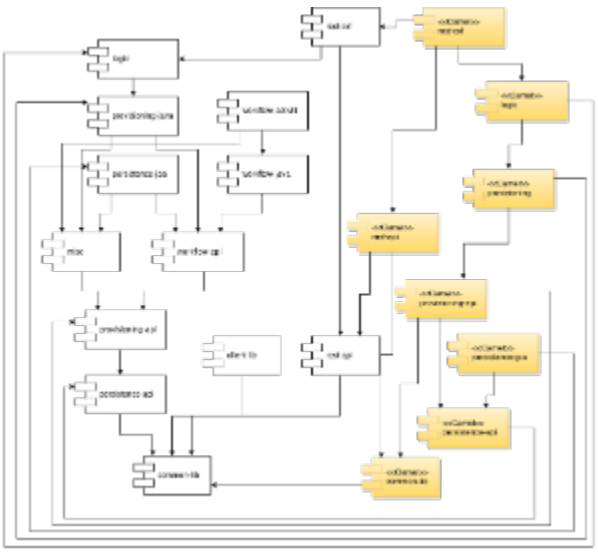
1. common
   JAR library including
    a. JAX-RS 2.0 REST services definition
    b. Java transfer objects (TO) to be used with REST services
2. client
   JAR library with utilities for invoking REST services
3. core
   WAR web application exposing the REST interface and implementing the whole business logic including provisioning, workflow management and persistence to an internal RDBMS via JPA.
4. console
   WAR web application providing rich GUI for interacting with core

When creating a new project from archetype, the `core` and `console` sub-modules generated are actually web applications empowering the WAR overlays feature.

The nice part about this is that generated projects can easily override any single file from the official Syncope Maven artifacts; the major drawbacks are that the whole Maven artifacts need to be downloaded, and that any single feature to be implemented must be thought as a "deviation" from the standard behavior; moreover, even if some features are not required in the specific project (Activiti workflow adapter, Camel integration, ...) the related dependencies are still to be carried over, because they are part of the official artifacts.

## Source tree >= 2_0_X

Starting with 2.0.0, the modules were re-organized according to the following package diagram (where packages are Maven modules, actually):



The general approach taken with this refactoring was to split, wherever possible and meaningful, the existing code into API and implementation, and to introduce new modules for each relevant feature.

Moreover, a proper mechanism for handling extensions was introduced, for which the reference implementation for Camel integration was also provided.

> Even if not likely at the moment, one can think to provide an alternative, LDAP or MyBatis based, implementation for syncope-core-persistence-api.

## The upgraded archetype

The upgraded archetype for 2.0.X is still generating the `core` and `console` sub-modules as web applications; differently from the past, however, such applications enlist as dependencies one or more Syncope JAR artifacts (no more WAR overlays).

### Core

The following dependencies are set:

1. syncope-core-rest-cxf
   Provides a web-fragment with REST interface; depends on syncope-core-logic (general business logic) which in turn relies on syncope-provisioning-java (default provisioning management engine)
2. syncope-core-workflow-java
   Bare workflow adapter implementation;  also includes common features for workflow management
3. syncope-core-workflow-activiti
   Activiti-based workflow adapter implementation
4. syncope-core-persistence-jpa
   OpenJPA-based internal persistence module

From the list above, any generated project can remove syncope-core-workflow-activiti to avoid embedding Activiti dependencies or provide its own workflow adapter implementing the API defined in syncope-core-workflow-api.

Moreover, if integration with Camel is desired, the following dependencies can be added:

1. syncope-ext-camel-rest-cxf
   Provides implementation of additional REST interface methods
2. syncope-ext-camel-provisioning
   Camel-based implementation of syncope-provisioning-api
3. syncope-ext-camel-persistence-jpa
   OpenJPA extensions for managing additional persistence table

### Console

The following dependency is set:

1. syncope-client-console
   Provides the whole Wicket-based GUI, again as web-fragment; depends on syncope-client-lib for the actual communication with core

If integration with Camel is desired, the following dependency can be added:

1. syncope-ext-camel-client-console
   This will enable an additional tab under Configuration > Extensions with facilities for handling Camel integration