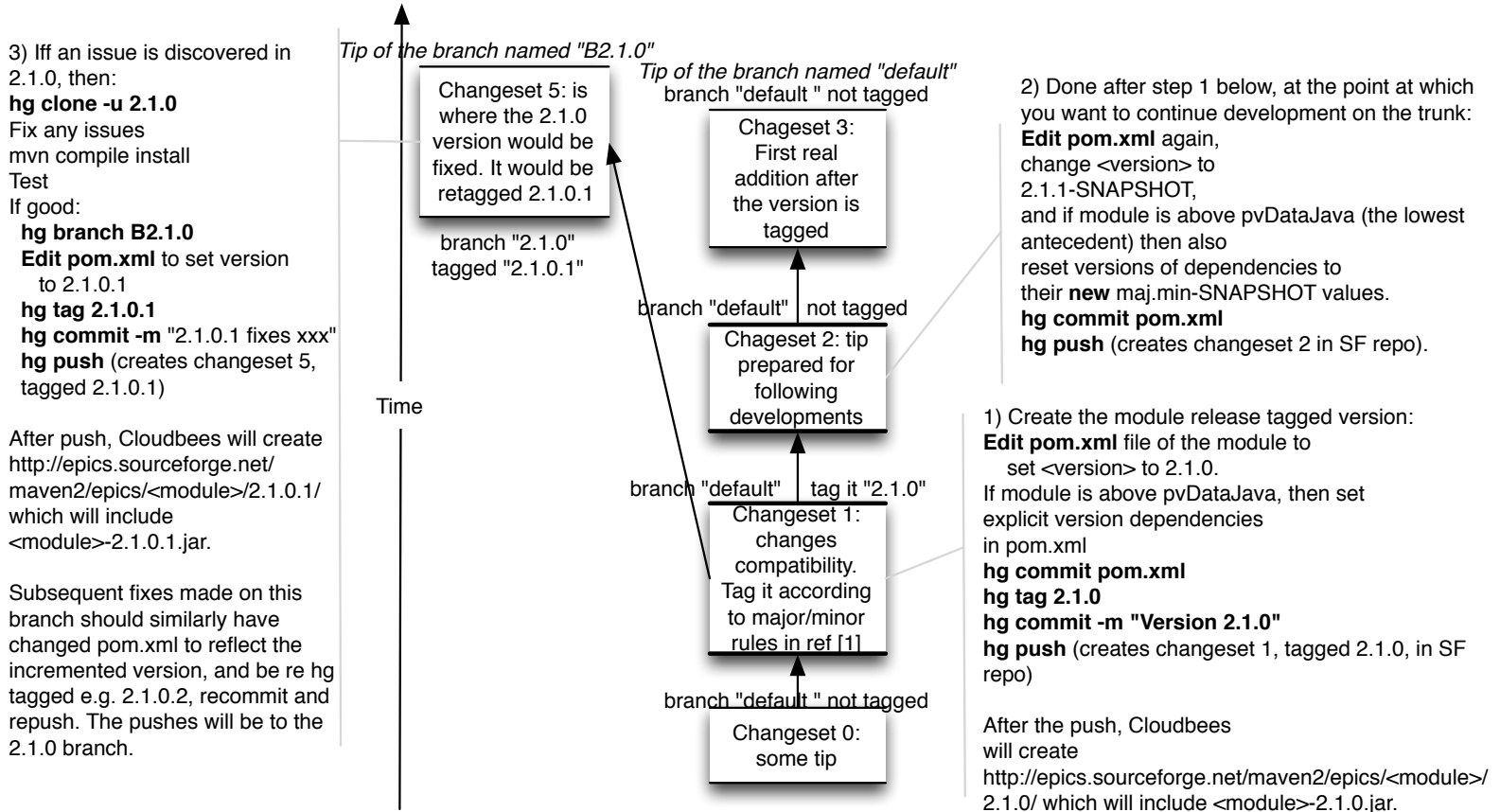# Process related to creating the release version of a Java module of EPICS V4, mercurial and maven commands

Tip of the branch named "B2.1.0"

3) Iff an issue is discovered in 2.1.0, then:
hg clone -u 2.1.0
Fix any issues
mvn compile install
Test
If good:
  hg branch B2.1.0
  Edit pom.xml to set version
    to 2.1.0.1
  hg tag 2.1.0.1
  hg commit -m "2.1.0.1 fixes xxx"
  hg push (creates changeset 5, tagged 2.1.0.1)

After push, Cloudbees will create http://epics.sourceforge.net/maven2/epics/<module>/2.1.0.1/ which will include <module>-2.1.0.1.jar.

Subsequent fixes made on this branch should similarly have changed pom.xml to reflect the incremented version, and be re hg tagged e.g. 2.1.0.2, recommit and repush. The pushes will be to the 2.1.0 branch.

Changeset 5: is where the 2.1.0 version would be fixed. It would be retagged 2.1.0.1

branch "2.1.0" tagged "2.1.0.1"

Tip of the branch named "default"
branch "default " not tagged

Chageset 3: First real addition after the version is tagged

branch "default" not tagged

Chageset 2: tip prepared for following developments

Time

branch "default"   tag it "2.1.0"

Changeset 1: changes compatibility. Tag it according to major/minor rules in ref [1]

branch "default " not tagged

Changeset 0: some tip

2) Done after step 1 below, at the point at which you want to continue development on the trunk:
Edit pom.xml again,
change <version> to 2.1.1-SNAPSHOT,
and if module is above pvDataJava (the lowest antecedent) then also reset versions of dependencies to their **new** maj.min-SNAPSHOT values.
hg commit pom.xml
hg push (creates changeset 2 in SF repo).

1) Create the module release tagged version:
Edit pom.xml file of the module to
  set <version> to 2.1.0.
If module is above pvDataJava, then set explicit version dependencies in pom.xml
hg commit pom.xml
hg tag 2.1.0
hg commit -m "Version 2.1.0"
hg push (creates changeset 1, tagged 2.1.0, in SF repo)

After the push, Cloudbees will create http://epics.sourceforge.net/maven2/epics/<module>/2.1.0/ which will include <module>-2.1.0.jar.

Changeset 0 is the changeset directly before the changeset you want to "be" your release. Changeset 1 is the one you want to be your release.
Only creating changesets 1 and 2 comprise the process of a "release". If the release needs a fix, the branch will be created
(at the time of the fix implementation), subsequent fixes to that version will be on that branch also.

Steps 1. and 2. can be done in one step using the "Maven Release Plugin" (http://maven.apache.org/maven-release/maven-release-plugin/index.html)

[1] EPICS V4 Release Process, http://epics-pvdata.sourceforge.net/release.html