THE TAG COLLECTOR – A TOOL FOR ATLAS CODE RELEASE MANAGEMENT.

Solveig ALBRAND, Johann COLLOT, Jerôme FULACHIER, Fabian LAMBERT
Laboratoire de Physique Subatomique et de Cosmologie,
CNRS - IN2P3 / Université Joseph Fourier, 53 ave des Martyrs,
38026 GRENOBLE, France

INTRODUCTION

The Tag Collector is a web interfaced database application for release management. It was designed and implemented during the summer of 2001, for ATLAS software in response to a near crisis situation. Until this time the ATLAS librarian constructed a build of the software release after a cascade of e-mails from developers; communicating the correct CVS code repository version tag of their respective packages. This was subject to all sorts of human errors, and inefficient in our multi-time zone environment. In addition, it was difficult to manage the contents of a release. It was all too easy for a prolific developer to introduce a well-intentioned change in his package just before a build, often with unsuspected border effects. Developers were also asking for regular, and frequent builds for integration testing.

The tool is interfaced with CVS, and also with CMT, the configuration management tool [1]. Developers can interactively select the CVS tags to be included in a build, and the complete build commands are produced automatically. Other features are provided such as verification of package CMT requirements files, and direct links to the package documentation, making it a useful tool for all ATLAS users.

The tool has proved extremely successful, and features that are outside the scope of the original design have been requested. It was decided to treat the initial software as a prototype, and follow a classic software development cycle, resulting in a completely new design of the application. The new version will be more flexible and easier to maintain, and will include a large number of new features.

This article lists some Tag Collector functions and how they are used for release management within ATLAS. The new features, which will be implemented in Tag Collector II, are described. Tag Collector II will be based on the AMI generic database management software also developed at the LPSC Grenoble [2]. Finally we discuss some other possible applications of the Tag Collector application.

SOME RELEASE MANAGEMENT CONCEPTS USED BY ATLAS

Release and release cycle:

A release is a set of components that have been built together.

ATLAS Software Development utilizes a 4-tier hierarchy of releases:

- Nightly Releases
- Developer Releases
- Production Releases
- Bug-fix (or Patch) Releases

Management sets goals [3] for production releases, and determines the steps to be taken to attain them. In general, there is a linear progression from one release to the next, and several developer releases are needed to get from one to the next. However, in some cases it has been found advantageous to build a branch release for bug fixes. Typically when a release contains a few bugs in a reduced number of packages. The corrected branch release will become the recommended release for production, bur all developers can continue to work on the main release development implementing the new features required.

Package:

A package is a set of software components (such as applications, libraries, tools etc.) that are to be used for producing a system. Several persons may participate in the development, and one of them is responsible for coordinating the complete package.

Package membership and structure may be determined physically, for example by the directory hierarchy, or logically. ATLAS software currently uses uniquely the physical location of a package to determine package membership.

Container or Group Package:

This is a type of package that may contain some other packages like leaf or other container packages.

Leaf Package:

This is a type of packages that may not contain other ones. These packages contain the source code.

The complete package name of a leaf package includes its package group name.

Requirements File:

The configuration management tool constructs the make file for the component libraries of the system. So it needs to know which other packages are "used" by the code, and are needed for the compilation, and which sub packages make up the package. The mechanism used by CMT is a text file called the "requirements file" contained in each package. For a container package the file contains the name and the exact tag of contained packages. For a leaf package the file contains the complete package name and the version tag for packages that are needed by the leaf package for compilation.

DEVELOPER INTERACTION WITH TAG COLLECTOR

Developers check their software into the repository and give it a tag, following the policy laid down by ATLAS management. If the newly tagged software package is to be included in an Atlas release, then it must also be declared to the Tag Collector.

Using the web interface the developer selects the function "Add a New Tag". Tag Collector obtains the list of available tags for the package from CVS, presenting the most recently found tag by default. It is not possible to declare a tag which does not exist.

When a developer enters a new tag for a leaf package, Tag Collector marks the container package as needing a new tag. The package manager must gather the tags of the leaf packages under his responsibility. This list must be placed in the CMT requirements file. Tag Collector can suggest the correct requirements file for the container package. The manager can copy and paste this into the file and retag it in the repository.

THE ROLE OF THE RELEASE MANAGER

Release policy is determined by ATLAS software management, and enforced by the release manager.

Since 2001 6 release managers have successively occupied the function, each one working for 6 months. It is an arduous task, and requires juggling the many conflicting priorities and personalities of a large project; on the other hand it has greatly facilitated dissemination of knowledge of ATLAS software. The rapid turnover of release managers has been very productive for Tag Collector development, because each new manager has brought fresh ideas for new features.

The release manager can control the locking and unlocking of parts of the release using the functions of Tag Collector. Locking means that developers cannot add new tags to the release without prior approval from the manager. In particular, core software is required to be stable for a certain time during the run up to a production release. This gives developers of non-core software time to integrate and test the new features of core software, using the developer releases.

Selective locking and unlocking is also useful for controlling bug-fix releases. Only those packages with bugs are unlocked, so this prevents other developers from "sneaking" in changes to their packages without proper time for testing for those packages that may be affected by the change.

THE ROLE OF THE LIBRARIAN

The librarian has a technical role in the building of releases. Tag Collector helps by providing a list of all the packages which have been declared in the correct format for the configuration management tool. They can simply be copied and pasted into the configuration file. The librarian and the release manager have the same set of privileges for Tag Collector, in fact they are often the same person.

BUILDING A PRODUCTION OR DEVELOPER RELEASE

These releases of ATLAS software are built in a hierarchic manner. Tag Collector provides a list of all the top container packages and versions to be included in the release. The person building the release, usually the librarian, pastes this file, into the requirements file for the special package called AtlasRelease, and then checked into CVS. After logging on to a build machine, the librarian invokes the configuration management tool to check out the complete set of files needed to build the release by a recursion in the requirements file for the release.

BUILDING A NIGHTLY RELEASE

ATLAS uses a system called NICOS [4] (NIghtly COntrol System), a tool that automates nightly builds of large software projects on UNIX-like platforms. Several releases, with different options, are constructed on a regular nightly basis. They have a validity of one week. One of the purposes of nightly releases is to allow integration testing and migration. Thus is not normally expected that nightly releases will be fully functional.

Tag Collector was an essential step for the success of using regular nightly builds within ATLAS because it provides a text file to NICOS containing a list of the most recent tags submitted to Tag Collector. Thus, nightly releases ignore the package hierarchy. In this way a developer can verify that his package builds correctly without waiting for the package manager to update the top package files.

IMPLEMENTATION OF TAG COLLECTOR

The first version of Tag Collector was implemented in PHP, and uses a mySQL database. It is tightly coupled to CVS and to CMT.

Tag Collector has become an essential part of ATLAS software development. Since 2001, many extensions have been requested. Although the use of PHP allowed very rapid development, the resulting code is difficult to maintain. Therefore it was decided to rewrite Tag Collector using the AMI generic database management software[2]. The new version of Tag Collector will benefit from the complete set of generic software in AMI such as the Web service interface.

It is worth noting that the development of Tag Collector has enriched AMI because some of the new Tag Collector features have been implemented in a generic way. A good example is the AMI user management described in [2]

TAG COLLECTOR WEB INTERFACE

To access the Tag Collector as a guest, logon to http://atlasbkk1.in2p3.fr/athena/ with user name = "demo", and no password.

Figures 1 to 4 show some screen shots of the Tag Collector web interface.

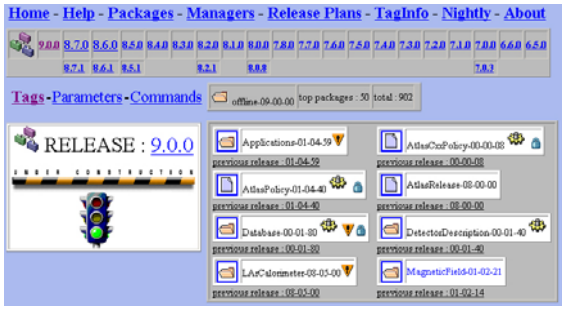

Figure 1 The top page for release 9.0.0, open for new user tags.

Figure 2 The details of the Database package.

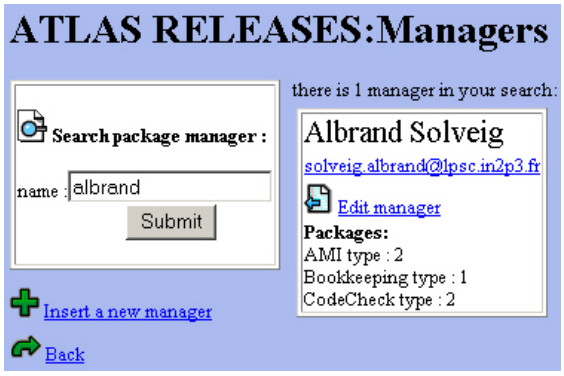

Figure 3 Searching for a package manager.

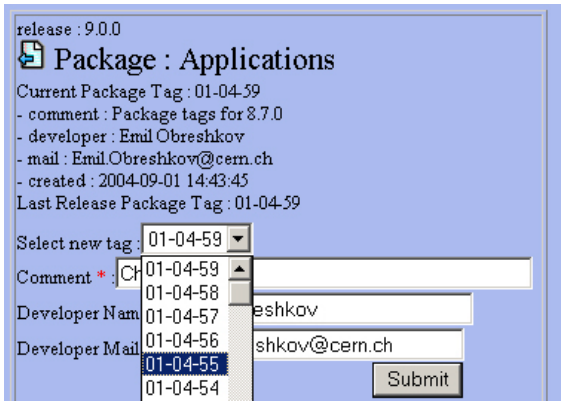

Figure 4 Selecting a tag from those found in CVS

NEW TAG COLLECTOR FEATURES

Tag Collector II followed a classic SW development process. We first opened a requirements gathering phase, during which the many users of Tag Collector could propose new features. The resulting requirements review was reviewed in December 2003 [5]. Tag Collector was then completely redesigned.

The first version of Tag Collector II will be released in November 2004. It will have many new features:

- The tightly coupled links to CVS and CMT have been replaced by "plug-in" modules, so that other tools can be used by Tag Collector, so that use of the tool is not restricted to the choices made by ATLAS.
- It will be possible to define package groups in a logical manner, as well as by their physical position.
- It will be possible to break the main release into sub-groups. So, instead of each release implying a complete make of all the software, parts of the system can be built and frozen independently. Development on these froze parts can continue, with developers benefiting from the Nightly release. Other packages

would continue to build against the frozen sub-release.

- There will be improved support for branch releases.
- There will be a higher resolution of user roles.
- The Release Manager functions will be more flexible.

CONCLUSION

Tag Collector has become an essential part of ATLAS software. It is a central to release management, and is also used by developers as a source of information on package structure as it is easier to navigate within Tag Collector than within the repository.

The first version of Tag Collector is very ATLAS specific, but the new version will be adaptable to other experiment's configurations.

Tag Collector could be easily adapted for applications other than release management. Many evolving systems require coordination between independently developed parts. We are at present considering the best way to continue our development and support of Tag Collector in a wider context.

REFERENCES

[1] http://www.cmtsite.org/

[2] ATLAS Metadata Interfaces (AMI) and ATLAS metadata catalogs, Solveig Albrand, Jerôme Fulachier. This conference.

[3] http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/Release/Policies/main.html

[4] http://www.usatlas.bnl.gov/computing/software/nicos/

[5] http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/Development/qa/Reviews/Reports/TagCollector