

Atlas Offline SW Release Building Strategy

Frederick Luehring

Indiana University

December 8, 2005

SIT Parallel Session

Atlas Software Workshop at LAL

Introduction

- I am worried about the way that we have been closing, building, and validating the 11.0.X releases.
 - The process seems not as organized as it should be.
- What this talk is and is not:
 - This talk is not about the build tools.
 - I believe that we have the tools that we need to manage building releases with the geographically dispersed Atlas developer community.
 - Some work remains on Project Builds, Testing, and Subversion
 - This talk is about the way we manage building a release.
 - This talk is not trying to assign fault.
 - This talk is an attempt to start discussion that will lead to improvements.

Introduction (continued)

- This discussion must include everyone:
 - The subdetector SW developer community
 - The physics community
 - The SIT community
 - The commissioning community
 - The production community

Observations on Recent Builds

- When asked how close they are to being ready, developers give very optimistic time estimates.
- Developers who are not fully aware of the release schedule request tags through the build start.
- Developers who are fully aware of the release schedule put tags in at the very last minute.
 - All 11.0.X had several tags that were not tested in a nightly.
 - No matter how many times the release is delayed, many tags come in the at the last minute.
- The full tag approval process rarely rejects any tags.
- There is no full-time, dedicated release coordinator.
 - DavidQ has been serving as the release coordinator.
 - Coordinators have been identified for each project.

Observations (continued)

- As soon as the tag collector is locked to start a build, there is immediate pressure to open the next release.
- The validation and testing packages often are not ready when the release is built.
- The release patching process is not well publicized.
 - Sometimes packages are patched in place without a new version being created.
- Bugs are not systematically reported in one place.
- Show stopper bugs are found after the release is out.
- Some developers clearly use Y.0.X as their main development path instead of Y.X.0.

Consequences

- The release schedule slips - sometimes badly.
- There is little discipline about deadlines.
- People lose track of the schedule.
- Planning is difficult.
- The release builds converge very slowly.
- There is little or no time to validate the release.
- The kit must be changed after it has been announced.
- The lack of structure makes mistakes more likely.

Fixes - Build Management

- That when a release is being built someone post a daily status message about the progress of the build to atlas-sw@cern.ch.
- That we all agree to stop putting new functionality into production releases (Y.0.0, Y.0.1, ...)
 - Originally there was the intention to only allow bug fixes for the production releases but discipline has slipped.
 - It does seem reasonable that a high-level (SPMB?) decision could be made to allow specific new functionality in Y.0.X releases for specific well-discussed cases.
- That we again have a dedicated release coordinator.
- That starting two days before a Y.0.X build written requests are required for new tags.
 - Ideally there should only be bug fix tags during this period.

Fixes - Build Validation

- That starting from two days before the release someone should validate the release each day.
 - The validation could simply be running Kit Validation however it must start before the build and not after.
 - The validation process should lead to requests for bug fixes.
 - Validation should include monitoring of memory leaks.
- That the validation of the build and kit be a two stage process with quick tests followed by longer tests that continue after the kit is released.
- That we not open the next release in a branch until the current kit is validated as working.
 - Of course there will still be cases where hidden problems are discovered after the kit is built.
 - Not clear how this plays out with project builds.

Fixes - Release/Kit Patching

- In my tenure as release coordinator, patching caused the most controversy.
- That we should minimize the amount of patching packages in the release in place.
 - There is a trade-off between the delay in creating a new version and the confusion caused by patching in place.
 - Patching should only take place before the release is announced.
 - Ideally there should be no patches in place after the release is announced and patch should cause a new release version.
- That the kit have its patch revision number changed every time that kit is changed in anyway once the kit has been announced to the community.
- That all changes after the first full build of a release be announced by an email to atlas-sw@cern.ch.

Conclusions

- We need to strike the delicate balance between having a too little control of the build process and having too little flexibility to react to the inevitable problems that develop during the building process.
 - I do realize that there will always be this tension.
- Some of the possible fixes suggested may not be practical or may be unacceptable to the developers.
- However I do ask that we open a dialog on how make the releases converge to code usable for production.
 - We need to think about how to organize the new project-based approach to building the releases.
 - It seems unlikely that the current loosely structured approach will work when we have the fixed deadlines imposed by the LHC schedule for data taking.