System Tests

Several Steps:

0) Construct configure script (done automatically by RM)

1) Gleam – generate mc.root, digi.root and recon.root. (batch farm)

2) RootAnalysis scripts – generates set of root histograms. (batch farm)

3) Calculates metadata – means, rms, number of entries, KS test result etc. (done on the machine that submitted the job)

4) User/Web interface.

DB Tables

release_tests – package name and version of the software,

system_tests – test_name, script, cpu time, memory, logfile, datafiles
standard_testid, test_result (failed, pending or suceeded)

metadata – systest_id, info, info_value (store an arbitrary amount of metadata
for each histogram in the test corresponding to systest_id).

These tables are very much centered on the concept of systests as a
test of a code release.

Systests on Data

We should not try to make a single system that both verifies the code and checks data quality
 .... it cannot be both a floor wax **and** a desert topping

That is not to say that we should not borrow many elements from the current system, and that we could not pull something usable together quite quickly.

Systests and Data/pipeline

The simplest thing to do is to rewrite the systests scripts so that they accept run_id as an argument, and know how to query the pipeline DB to find the location of the data.

May also need to alter the DB schema somewhat.

I do not believe that KS tests of distributions against standards will work well with data – there are likely to be too many false positives to make an analogue of the current system work (we could check this with existing data though)

Instead, we could expand the metadata table to contain much more information about each run. Much of the information provided in the SVAC reports could be generated (and stored in a DB) by the systests.

Systests and Data pipeline

Switch to using the systest package written by Tracy to make the histograms (or at least a package based on this)

Make as many histograms as we like, but we will need to carefully pick a small subset to be perused by a human (as we can no longer rely on the KS test to pick an interesting sample).

- We could display this subset of plots in some sort of report. There should also be access to a “standard” plot and a description of what it should look like.
- It would be convenient to allow a choice of overlay for all plots, (MC, standard data run etc).
- Histos will need to be normalised/rescaled in some way (by duration)

It might be worth starting by trying to recreate something like the SVAC reports from the systest data.

I think that it will be relatively easy to generate the data/plots, but that the challenging part will be coming up with an interface/report generator that really displays the information well, without overwhelming the user.

For January, we might want to try a system that is very similar to the current systests.

0) Construct configure script (done automatically by pipeline)

1) EngineeringModel – generate merit.root, digi.root and recon.root.
   - no longer needed as is run in the pipeline anyway?

2) RootAnalysis scripts – generates set of root histograms.
   - Switch to Tracy's code. I have used this occasionally in the current systests and nobody has complained (or noticed...). This will make it easier to make histos from many runs since chains are nicely handled.

3) Calculate metadata – means, rms, number of entries, KS test result etc.
   - This script has to be rewritten and expanded for data.

Do we have an equivalent of the release_tests table anywhere else in the system? system_tests table needs to be replaced with something run oriented