A Guide for the Use of Untracked Parameters in RCPs at DØ.

Marc Paterno
Fermi National Accelerator Laboratory
Heidi Schellmann
Northwestern University

October 9, 2002

Abstract

This document spells out the policy of the DØ experiment, regarding the use of untracked parameters in *RCP*s.

1 Background

The **RCP** system exists primarily for the purpose of tracking those parameters used to configure a production executable, while allowing users to modify those parameters as easily as possible. However, some parameters used by such an executable do not need to be tracked. In the absence of another mechanism to enter parameters into a running program, the **RCP** system was expanded to include the concept of *untracked parameters*.

Untracked parameters are, by their definition, never tracked in any **RCP** database. Thus, any *RCP* object retrieved from an **RCP** database will never contain any untracked parameters. This, unfortunately, interacts poorly with another feature of the **RCP** system, *nested RCPs*.

A nested *RCP* is an *RCP* object that was obtained from within *another RCP* object. The mechanism through which this nesting is done is that the *RCPID* of the nested *RCP* object is contained within the carrier *RCP* object. When the user calls the function `RCP::getRCP()` to obtain the nested *RCP*, the system makes a call to the *RCPManager* to extract the nested *RCP* from a database, using the *RCPID* of the nested *RCP* to identify the object to be returned. Since all nested *RCP*s are thus obtained from an **RCP** database, they never contain any untracked parameters.

2 Purpose of Untracked Parameters

Because untracked parameters are never stored in any **RCP** database, their use for configuration of algorithm objects and framework modules in any manner

that will affect the results of reconstruction is expressly forbidden by DØ policy.

Untracked parameters may only be used to set parameters that do *not* affect reconstruction output. Two of the primary examples of the allowable use of untracked parameters is the setting of debug flags and the setting of verbosity levels in diagnostic output. Since neither of these affect the results of reconstruction, they are allowed.

3 Recommendations For Use

3.1 Primary Recommendation

In order for any untracked parameters to be available from an *RCP* object, that object must have been created by reading an RCP script, and not extracted from a database. Since the top-level framework *RCP* object is always made by reading a local script, it is recommended that all untracked parameters to be used by a given program reside directly in this top-level framework RCP.

3.2 Secondary Recommendation

A second method for obtaining *RCP* objects containing untracked parameters is to directly call `RCPManager::extract(const string& package, const string& object)`. Using this method directly, rather than relying on nested *RCP*s, ensures that the system will read any local script of the correct name *before* referring to the **RCP** databases; if this local script contains untracked parameters, the *RCP* object which is returned will contain those untracked parameters. Please see § 4 for a warning against the use of environment variables in this context.

4 Caution Against Environment Variables

Users are cautioned *against* the use of environment variables in programs to carry the "names" of *RCP*s in software that will become a part of production executables. While this mechanism may be convenient for the individual user, it presents a problem for the operators of the production system. Widespread use of environment variables to control the production executables would impose a large burden during setup of the production environment, which is not viewed as acceptable at this time.

5 Examples

Suppose one is working on the framework package *HiggsReco* from the package "higgs_reco". In the constructor of *HiggsReco*, one wants a debug flag, that controls some diagnostic output. In the framework RCP script, one would put:

```
... (other framework RCP parameters may come before this)
untracked bool debug = true;
... (other framework RCP parameters may come after this)
```

In the constructor for *HiggsReco*, one would put:

```
HiggsReco::HiggsReco(fwk::Context* pc) :
  Package(pc),
  Process(pc),
  // ... possibly other interfaces follow
  _debugFlag(frameworkRCP().getUntrackedBool("debug", false)),
  // ... possibly other data members follow
{
  // Ideally, the body of the constructor will be empty,
  // because all initialization is done in the colon-
  // initialization list.
}
```

Note that the default value for the untracked parameter named "debug" is false, because one does not want the reconstruction farm (which will use default values for all untracked parameters) to be producing reams of tracing output.

Also note the use of the *Package* member function `frameworkRCP()`, which returns a reference to the framework *RCP* object.

Finally, please note that the initialization of all the data members of *HiggsReco* is performed in the initialization list, which is a good practice to follow whenever possible.