

RCP Rules

Marc Paterno
CD/CPD/APS, FNAL

October 9, 2002

Abstract

This document describes the “business rules” which the RCP system is designed to follow.

1 Background

This is an extremely concise introduction to the RCP system. See the document **Run Control Parameters at DØ** for a more detailed introduction. *RCP-Manager* uses a series of disjoint databases while looking for parameter sets:

- a set of readonly databases, which can’t be added to by the user,
- a single writable database, which can be added to by the user

The databases are always searched in the same order: readonly databases, in order, then writable database. What databases to read, and their ordering, is determined by the environment variable `RCP_DATABASE_PATH`. This contains the names of the databases (not the names of files, or any such thing). It also contains the information about what “flavor” of database is to be used; right now, that can be either *FileSystemDB* or *RCPDatabaseInMemory*.

Scripts are looked for under the directories listed in the colon-separated list of directories specified by the environment variable `RCP_SCRIPT_BASE`¹. These directories are searched in the order they are specified. The *FileFinder* class expects to see the directory structure that would be generated by a series of `addpkg` commands (directory for each package, `rcp` directory under each package directory, RCP files in the `rcp` directory). Parameter sets are associated with a single RCPID. Parameter sets can be given many names, but each name can map to only one parameter set. An *RCPName* is a combination of *databaseName*, *packageName*, *objectName*, and *versionName*. The *versionName* may either be assigned by the release mechanism (for entries into the official database), or generated by the database itself (for entries into personal databases). Rules for other databases are up to the folks who define those databases.

¹For versions of **RCP** prior to v00-09-00, `RCP_SCRIPT_BASE` was a single directory

2 Cases

This section of the document lists the “cases” to which the RCP system must respond, and what it must do in each circumstance. This is the place to check in order to determine what effect a proposed addition to the RCP rules would have upon the system, even before an attempt is made to implement the change in rules. The RCP system *must* have a defined behavior in each circumstance listed in these rules. What happens when a user calls `RCPManager::extract(“package”, “object”)`

1. System looks for a script: a local file containing a parameter set.
 - (1) If a script is found, continue with 2.
 - (2) If no script is found, continue with 3.
2. If a script is found:
 - (1) System parses script to make an incomplete parameter set. An incomplete parameter set does not contain a valid RCPID. The incomplete parameter set is presented to the readonly databases, in order, for completion. This stops when the first database is found that contains a matching set of parameters, or when all readonly databases have been checked unsuccessfully.
 1. If a match has been found, it is checked to see if the given name (here, “package, object”) matches any of the names in the matching parameter set.
 1. If no matching name is found, it is an error. An exception is thrown (and may be caught) that indicates the requested parameter set (matching the script) is already known by a different name; the user can get it by asking for that name.
 2. If a matching name is found, then the user is returned an RCP containing the parameter set, and with the RCPID associated with that parameter set in the database.
 2. If no match has been found, the parameter set is presented to the writable database.
 1. If this database has a matching parameter set, with the same name, the user is returned an RCP containing the parameter set, and with the RCPID associated with that parameter set in the database.
 2. If the database has a matching parameter set, but does not have the same name, this name is added to the database, and the user is returned an RCP containing the parameter set, and with the RCPID associated with that parameter set in the database.
 3. If no matching parameter set is found, a new RCPID is issued by the database, the parameter set is entered into the

database, associated with the new RCPID, and the user is returned an RCP object containing the parameter set, and with the RCPID newly associated with that parameter set in the database.

3. If no script is found:
 - (1) The system looks up the `versionName` to be associated with items in the (current) database – this is done independently for each database. Note that a database may be configured to have a distinct *current version*, or to just use the most recently timestamped version of each parameter set. When a request (by name) for a parameter set is made of a database that is configured to have a current version, then that version tag is inserted in each *RCPName* presented to that database.
 - (2) The readonly databases are searched in order for a parameter set matching the given name. A parameter set matches if the `packageName` and `objectName` (and, if present, if the `versionName` and `databaseName`) are the same.
 1. When the first database containing a match is found (it may contain more than one match), the best match of the candidates is selected. The best match is the one with the most recently timestamped matching name. The user is returned an RCP containing the associated parameter set, and with the RCPID associated with that parameter set in the database.
 - (3) If no match is found in a readonly database, the writable database is checked.
 1. Resolution is the same as for the readonly databases.
 2. Nothing will be written to the writable database in this case.
 - (4) If no match is found in the writable database, an exception is thrown (and may be caught) that indicates no parameter set known by the given name could be found.