

Understanding Nested UCM Project VOB Environments

Will Frontiero

March 17, 2009

INTRODUCTION3

OVERVIEW4

STANDARD SINGLE UCM PROJECT VOB DEVELOPMENT5

CREATING A NESTED PROJECT VOB STRUCTURE7

 USING UCMUTIL LINK_PVOB.....9

 MULTISITE ENVIRONMENTS AND COMMON ISSUES THAT CAN OCCUR 14

SUMMARY16

REFERENCES16

Introduction

IBM® Rational® ClearCase® Unified Change Management (UCM) makes extension use of an administrative project VOB (PVOB) association. This white paper provides details and best practices for environments that implement an administrative VOB structure that incorporates the use of multiple PVOBs (nested PVOBs). This paper also discusses additional considerations for ClearCase MultiSite environments. It does not contain all possible configurations involving UCM PVOBs in a UCM environment.

This document is designed to be read by ClearCase UCM administrators who are responsible for their organizations UCM configuration.

Before proceeding you should have an understanding of the general UCM concepts covered in the [Understanding UCM](#) section of *IBM Rational ClearCase Managing Software Projects*.

Overview

Nested project VOB structures can add an additional layer of scalability and functionality to a UCM environment that allows administrators to share metadata across multiple UCM project Structures. This nested configuration can help environments distribute metadata across multiple administrative VOBs avoiding large VOB databases. It can also be beneficial in helping to improve performance by offloading newly created projects to a separate project VOB structure. This paper should help avoid many of the problems that could be encountered when linking project VOBs.

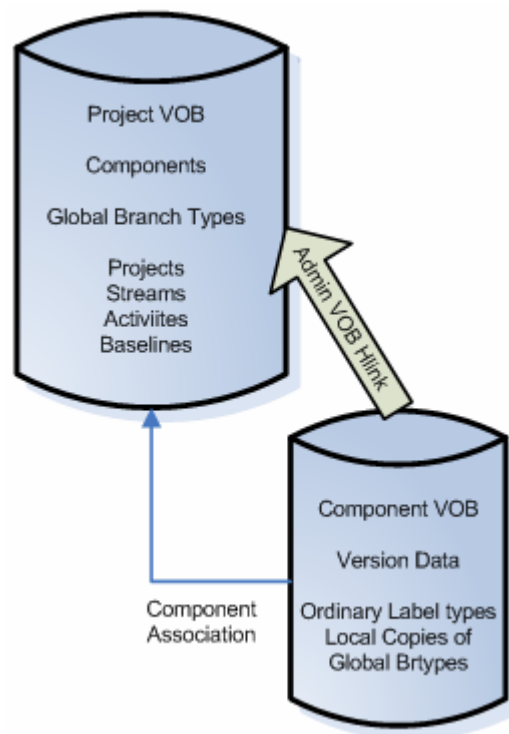
The following topics are covered in further detail below:

- Standard Single UCM project VOB development
- Creating a Nested project VOB Structure
 - a. Using an Existing project VOB as an administrative VOB for child project VOB.
 - b. Using a common administrative VOB for multiple project VOBs
- Using ucmutil link_pvob
- MultiSite environments and common issues that can occur

Standard Single UCM Project VOB Development

Standard UCM development requires a connection between a project VOB and a component VOB. The component VOB can contain one or more component root storage locations depending on its configuration. Standard development practices depend greatly on the integrity of the links between these VOBs. Each VOB contains a variety of hyperlinks to ensure data is protected and catalogued in the necessary locations. Creating data in a VOB that is a member of an administrative VOB structure implies shared data. The shared data is created in either the administrative VOB or the child VOB depending on the data type.

Here is a diagram of a standard UCM project VOB and component VOB association. The diagram shows some of the key data generated in both VOBs discussed in more detail below.



Standard UCM development implies that file and directory element data will be generated in the component VOB. Each version generated in the component VOB is associated with a UCM Activity. Every version generated must be associated with a ClearCase branch type.

The project VOB houses a global copy of every branch type needed during development. When a version is created, a local copy of the global branch is generated in the component VOB to be used for future version creation on that branch. Each branch is associated with a UCM stream housed in the project VOB.

The stream is used to allow or prohibit component VOBs from creating versions on the associated global branch type.

Projects are created in the project VOB and are used to organize and apply policy to a group of streams. When development reaches a key point in the projects lifecycle, a UCM baseline is created as a stamp in time.

The UCM baseline is an object created in the project VOB. It is associated with an ordinary label type generated in the component VOB. The label type is applied to a single latest version of every element in the selected configuration. Generally, this configuration is determined by the stream where the baseline is being created.

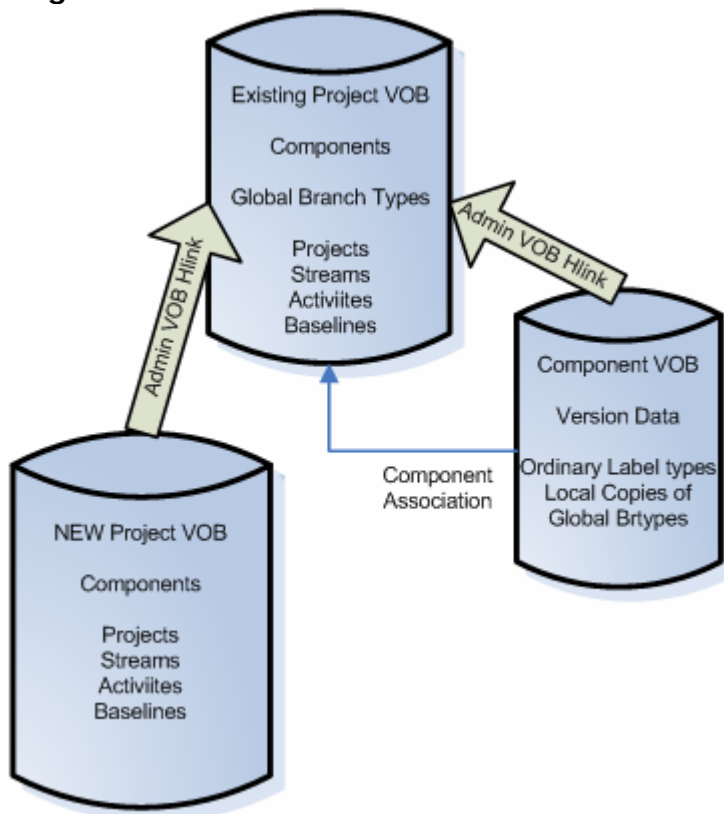
Creating a Nested Project VOB Structure

This section provides information about adding a new project VOB to an existing UCM development environment. A new project VOB means it contains no existing projects or components. The purpose of the project VOB is association with an existing UCM environment. When two or more project VOBs are associated to a common administrative VOB hierarchy, it is called a nested project VOB structure. In a nested structure there should be only one common administrative VOB. All VOBs in the administrative VOB hierarchy should point to a single top level VOB. The top level administrative VOB contains the objects needed to enable reliable distributed UCM development.

At this point, there are two ways to accomplish adding a new project VOB to an existing UCM environment.

- a.) Add a new project VOB and choose either project VOB (existing or new) as the common administrative VOB. However, this option is not very scalable. This configuration runs the risk of over populating a single project VOB. It also implies a higher importance of the existing project VOB.

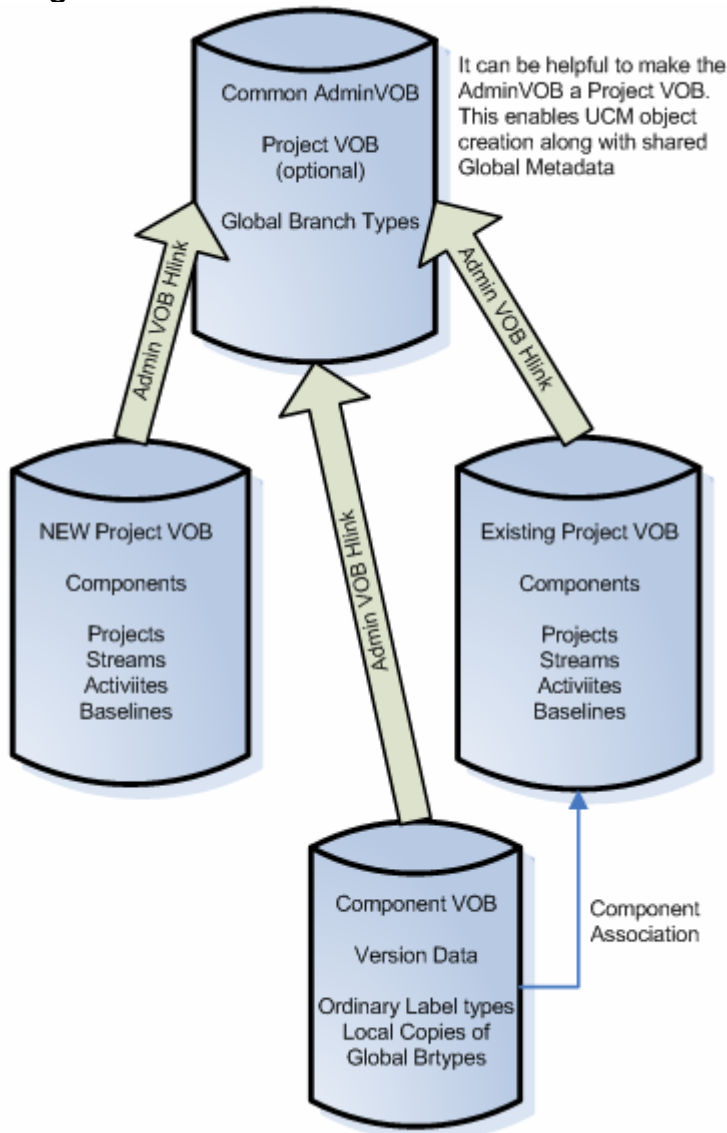
Diagram A.1:



Add a new project VOB and a new common administrative VOB. This configuration becomes scalable and implies equal importance of both project

VOBs. It helps to keep project VOB specific objects separate from shared global metadata such as branch types. It will also keep metadata organized in a common administrative VOB. This configuration will require the use of a utility bundled with ClearCase. The utility is called "ucmutil" and is discussed in greater detail below.

Diagram A.2:



Using ucmutil link_pvob

In the above configuration the use of the “ucmutil” utility was needed. The utility has a sub command called “link_pvob”. This sub command allows administrators to link an existing project VOB to a top level administrative VOB. The tool requires that the existing project VOB not yet be associated with any other administrative VOB. The tools purpose is to perform the following operations.

- a.) Push all existing global branch type definitions in the existing project VOB to the target administrative VOB
- b.) Re-associate all existing streams with the global branch type definitions moved to the target administrative VOB
- c.) Redirect the administrative VOB link of any associated component VOBs to the new target administrative VOB. As you can see in the diagram above, the component VOB now points to the common administrative VOB which is the result of running “ucmutil link_pvob”. You should also note that the component association remains with the original project VOB.

The “ucmutil” utility is found in the following locations:

- Windows: %CLEARCASE_HOME%\etc\utils\ucmutil.exe
- Linux/Unix: /opt/rational/clearcase/etc/utils/ucmutil

Refer to technote 1237620 [About ucmutil](#) for further information about this utility.

Example usage:

```
M:\adminView>ucmutil link_pvob -verbose -from vob:\pvobing -to vob:\commonAdminVOB
Checking branch type ccr_c_deliver.
Checking branch type composite_Integration.
Checking branch type deliver_Integration.
Checking branch type william_child_deliver.
Checking branch type william_child2_deliver.
Checking branch type william_deliver.
```

```
Total of 6 global branch types in VOB \pvobing.
Total of 6 UCM branch types checked.
```

```
M:\adminView>
```

You can see in the example above that the link_pvob subcommand will locate all global branch type definitions for the child project VOB. This example is run in a –verbose (preview) mode. It is always recommended to run this tool in preview mode before actually performing the operation. It is also helpful to redirect the output of the command to a log file for future reference. This command should be considered irreversible. Once the command is run with “-fix” flag, the administrative VOB hyperlink is drawn and all metadata is restructured to

represent the hierarchical change. There are some issues that could occur during the operation, therefore, the tool looks to ensure the VOBs specified in the command are locked with the "-nuser" switch. Only the user running the command should be able to access the VOBs when running the "ucmutil link_pvob" operation. For this purpose you should lock the VOBs excluding the user running the operation with the "-nuser" switch.

Example of locking the project VOBs with the -nuser switch:

```
-----
M:\adminView>ucmutil link_pvob -fix -verbose -from vob:\pvobing -to vob:\commonAdminVOB
ucmutil: Error: VOB \pvobing should be locked with "-nuser" before running this tool.
```

```
M:\adminView>cleartool lock -nuser DOMAIN1\william vob:\pvobing
Locked versioned object base "\pvobing".
```

```
M:\adminView>ucmutil link_pvob -fix -verbose -from vob:\pvobing -to vob:\commonAdminVOB
ucmutil: Error: VOB \commonAdminVOB should be locked with "-nuser" before running this tool.
```

```
M:\adminView>cleartool lock -nuser DOMAIN1\william vob:\commonAdminVOB
Locked versioned object base "\commonAdminVOB".
-----
```

In the above example you can see that the operation will check to ensure both project VOBs are locked. It will then recommend that you lock the VOB with the -nuser switch.

To recap, before running the ucmutil link_pvob with the -fix option, you should perform the following operations:

- a. Lock the project VOBs with the -nuser switch
- b. Run ucmutil link_pvob -verbose and redirect the output to a log file for historical purposes. (It can also be helpful for troubleshooting)
- c. Ensure all branch types listed in preview mode are mastered locally at the site running the operation. (This topic will be covered in the MultiSite environment section later in this paper.)
- d. Ensure all component VOBs associated with the child project VOB are NOT locked. Locking of the component VOBs will restrict the utility from accessing and changing necessary data (You can lock -nuser the component VOBs when locking the project VOBs to avoid the issue).

Below is an example of running the utility to link project VOB "\pvobing" to a common administrative VOB "\commonAdminVOB":

Here is a list of metadata before the operation:

- a) Listing of global branch types in "\commonAdminVOB"

```
M:\adminView>cleartool lstype -s -kind brtype -invob \commonAdminVOB
main
```

- b) Listing of global branch types in "\pvobing"

```
M:\adminView>cleartool lstype -s -kind brtype -invob \pvobing
ccrc_deliver
composite_Integration
deliver_Integration
main
william_child_deliver
william_child2_deliver
william_deliver
```

- c) Listing of administrative VOB hyperlinks for associated component VOBs in "\pvobing"

```
M:\adminView>cleartool describe -l -ahlink -all vob:\pvobing
Hyperlinks:
  AdminVOB@42@\brokeComp <- vob:\brokeComp
  AdminVOB@42@\cvobing <- vob:\cvobing
```

- d) Here is an example of a UCM streams association to a global branch type before Linking the two project VOBs

```
M:\adminView>cleartool describe -l -ahlink -all stream:deliver_Integration@\pvobing
deliver_Integration
Hyperlinks:
  IndependentGuard@306@\pvobing -> brtype:deliver_Integration@\pvobing
  UseBaseline@114@\pvobing -> baseline:cvobing_INITIAL@\pvobing
```

- e) Here is the administrative VOB hlink pointing from the associated component VOB

```
M:\adminView>cleartool describe -l -ahlink -all vob:\cvobing
\cvobing
Hyperlinks:
  AdminVOB@42@\cvobing -> vob:\pvobing
```

Here is an example of running the commands to link the two VOBs

- a.) Run the tool in preview mode with the `-verbose` flag and redirect the output to a log file:

```
M:\adminView>ucmutil link_pvob -verbose -from vob:\pvobing -to vob:\commonAdminVOB >
c:\link_pvob_log.txt
```

- b.) Run the tool to Create the link between the VOBs with the `-fix` flag:

```
M:\adminView>ucmutil link_pvob -fix -verbose -from vob:\pvobing -to vob:\commonAdminVOB
Checking branch type ccrc_deliver.
Moving branch type ccrc_deliver.
Checking branch type composite_Integration.
Moving branch type composite_Integration.
```

```

Checking branch type deliver_Integration.
Moving branch type deliver_Integration.
Checking branch type william_child_deliver.
Moving branch type william_child_deliver.
Checking branch type william_child2_deliver.
Moving branch type william_child2_deliver.
Checking branch type william_deliver.
Moving branch type william_deliver.
Reconnecting the moved global branch types.
There is no global type in source VOB. Do you want to redirect the AdminVOB hyperlink? [no]
yes
Redirecting AdminVOB hyperlink.

```

```

Total of 6 global branch types in VOB \pvobing.
Total of 6 UCM branch types checked.
Total of 6 UCM branch types moved.
Total of 6 Stream hyperlinks reconnected.
Total of 2 VOBs has been reconnected to new Admin VOB.

```

```
M:\adminView>
```

Here is a list of migrated and changed metadata as a result of the operation:

a) Listing the global branch types that now exist in “\commonAdminVOB”

```

M:\adminView>cleartool lstype -s -kind brtype -invob \commonAdminVOB
ccrc_deliver
composite_Integration
deliver_Integration
main
william_child_deliver
william_child2_deliver
william_deliver

```

(Notice all the global branch types that previously existed in “\pvobing” have been migrated to “\commonAdminVOB”)

b) Listing of Admin VOB hyperlinks in “\pvobing” after running ucutil link_pvob

```

M:\adminView>cleartool desc -l -ahlink -all vob:\pvobing
\pvobing
Hyperlinks:
AdminVOB@310@\pvobing -> vob:\commonAdminVOB

```

(Notice the Admin VOB hyperlinks for associated component VOBs are no longer pointing to the project VOB. The only Admin VOB hyperlink is the link created by the ucutil link_pvob utility)

c) Here is an example of a UCM streams association to a global branch type after linking the two project VOBs:

```

M:\adminView>cleartool desc -l -ahlink -all stream:deliver_Integration@\pvobing
deliver_Integration
Hyperlinks:
UseBaseline@114@\pvobing -> baseline:cvobing_INITIAL@\pvobing
IndependentGuard@306@\pvobing -> brtype:deliver_Integration@\commonAdminVOB

```

(Notice the IndependentGuard hyperlink now points to the global branch type migrated to the commonadministrativeVOB)

- d) Listing of Admin VOB hyperlinks in “commonAdminVOB” resulting from the ucutil tool

```
M:\adminView>cleartool desc -l -ahlink -all vob:\commonAdminVOB
\commonAdminVOB
Hyperlinks:
  AdminVOB@43@\brokeComp <- vob:\brokeComp
  AdminVOB@310@\pvobing <- vob:\pvobing
  AdminVOB@225@\cvobing <- vob:\cvobing
```

(Notice the Admin VOB hyperlinks previously pointing to “\pvobing” have now been recreated to point to “\commonAdminVOB” which contains all global branch types.)

- e) Here is the altered Admin VOB hyperlink pointing from the component VOB associated with “\pvobing”

```
M:\adminView>cleartool desc -l -ahlink -all vob:\cvobing
\cvobing
Hyperlinks:
  AdminVOB@225@\cvobing -> vob:\commonAdminVOB
```

(Notice that even though this VOB contains a component Data associated with “\pvobing”, it now points to “\commonAdminVOB to obtain any global type definitions.)

MultiSite Environments and Common issues that can occur

At this point we have covered running the `ucmutil link_pvob` utility to create nested project VOB structures, however, we have not discussed what can happen when the VOB is replicated in a ClearCase MultiSite environment.

When replicating a VOB, you must consider how the VOB is configured at your local site along with replicas in its family. The first and most important consideration is what must be replicated. As you can now understand, the VOBs share required data to operate without error. A nested project VOB structure must be replicated as a single structure for this exact reason. When replicating one VOB in the structure, you must replicate them all. This is also true for backup and restore. When backing up one of the VOBs, you should backup all for optimal recovery purposes.

The following is a list of issues that could occur if only some of the VOBs in the structure are replicated to alternate sites. We will use the structure created above for our examples. If both the project VOB and associated component VOB were replicated without the common administrative VOB:

- a.) The replica site would not have access to the necessary global branch types associated with the UCM streams
- b.) The replica site would most likely need to create streams. The streams would not have access to the common administrative VOB where the associated global branch types must exist.
- c.) The project VOB and component VOB would have broken administrative VOB hyperlinks pointing to a vob that does not exist.

IMPORTANT: This is very dangerous. If a `"cleartool checkvob -ucm or -hlinks"` is run against the VOBs, it will see the hyperlinks as broken. The `checkvob` will actually remove the hyperlinks as a repair. This operation will replicate to the originating site. This will destroy the administrative VOB structure at the originating site and cause disaster in the production environments. This error will not occur if all VOBs in the structure are replicated. An alternative, if needed, is to lock the AdminVOB hyperlink type in the VOBs with `-nuser`. This will not allow users to create or delete AdminVOB hyperlinks.

Example:

```
cleartool lock -nuser domain\user hlink:AdminVOB@\pvobing
```

Next we address the possible issues that could occur with existing UCM environments. It is possible that an existing project VOB and associated component VOBs are already replicated to multiple sites. Below is a list of considerations when planning to link existing MultiSited UCM VOBs.

Before running `ucmutil link_pvob` with the `"-fix"` flag in a MultiSited environment, you should consider the following.

- a) Ensure that all operable metadata is mastered at the site running the ucutil command.
- b) Run the tool with the "-verbose" flag first and gather a list of branches involved. Ensure all streams associated with the branch types are mastered where the utility is run. Refer to the "[cleartool chmaster](#)" man page for instructions on changing mastership of a stream and all of its objects.
- c) Review the AdminVOB hyperlinks associated with the existing project VOB to determine what VOBs need to be mastered where the tool is run. The Admin VOB hyperlinks associated with each component VOB will be recreated to point to the newly linked common administrative VOB.
- d) When running the ucutil link_pvob with the "-fix" flag, save the output so it can be used to troubleshoot any errors that occurred. To redirect output when running the command, you can not be in the ucutil prompt. You must type the command from a ClearCase view context. You should capture standard error along with standard out. The example below only shows standard out. Standard error will be shown in the command window.

Example:

```
M:\adminView>ucutil link_pvob -fix -from vob:\pvobing -to vob:\commonAdminVOB >
c:\linkPvob_runLog.txt
```

Summary

In addition to the added scalability and functionality offered by using a nested UCM project VOB structure, there is also a level of complexity with this configuration that requires special attention. This paper has outlined some best practices to follow when linking UCM project VOBs that should help you avoid many of the problems that could be encountered when linking project VOBs.

References

[IBM Rational ClearCase Managing Software Projects manual](#)
Under the topic of [Using multiple PVOBs](#)