CORE® 5

CORE Data Migration Guide

Vitech

Vitech Corporation

2070 Chain Bridge Road, Suite 100
Vienna, Virginia 22182-2536
703.883.2270 FAX: 703.883.1860
Customer Support: support@vitechcorp.com
www.vitechcorp.com

Publication Date: September 2007

Table of Contents

Preface....................iv
Overview of CORE 5 Schemas....................1
  Base Schema v50 vs. Base Schema v40 1
CORE 4 to 5 Data Migration....................3
  Data Migration Overview 3
  CORE 4 to 5 Data Migration Support 3
  Step-by-Step Instructions for Data Migration 4
  Manual Migration Steps for Custom Simulation Scripts 6
  Migrating Data that Use Project-Specific Schema Extensions 6

Preface

This guide describes the new schemas provided with CORE 5 and describes how to migrate projects using the CORE 4.0 schema (either base or C4ISR) to the 5.0 schema (base, DoDAF, or DoDAF v1.5) using a special purpose schema and associated set of scripts. After reviewing the information describing the new schema and various migration considerations, contact Vitech Customer Support (support@vitechcorp.com) to obtain the latest scripts for application to your project.

For those project teams seeking additional assistance to fully leverage CORE 5, Vitech's professional services team has developed a package including training and customized migration support services. Please contact your account executive or Vitech Customer Support (support@vitechcorp.com) for additional information.

Overview of CORE 5 Schemas

CORE 5.0 and CORE 5.1 support four different project schemas: Base Schema v50, DoDAF Schema v50, Base Schema v40, and C4ISR Schema v40. An additional schema, DoDAF v15 Schema v50, was released with CORE 5.1.5. The Base Schema v40 and C4ISR Schema v40 are identical to the respective CORE Release 4.0 schemas. Although the Base Schema v50 is substantially changed from the Base Schema v40, it still fully supports the model-based systems engineering paradigm, verification planning and tracking, and program management (as did the Release 4.0 base schema). DoDAF Schema v50 and DoDAF v15 Schema v50 are built upon the Base Schema v50 and includes additional classes, attributes, and relations beyond the C4ISR Schema v40 in order to better support the approved DoD Architecture Framework[1]. A summary of the changes to the base schema is provided in the following section.

Base Schema v50 vs. Base Schema v40

When compared to the v40 base schema, the v50 base schema includes changes to class, attribute, and relation definitions. Following is a summary by schema entity of the changes made to the v40 base schema to develop the v50 base schema.

Classes

The abstract classes were restructured to better control the inheritance of attributes and relations. There are now more abstract classes both in depth and breadth. This restructuring eliminated the unnecessary inheritance of some attributes and relations.

Some non-abstract classes were consolidated in order to simplify the schema. Other classes were renamed. The Annotation and Engineer classes were removed. Table 1 shows the relationship between the consolidated or renamed v50 classes and their counterparts in the v40 base schema. Note that where several classes were consolidated into one, such as **Requirement**, an attribute was added to reflect the original class.

Table 1 Comparison of Classes for CORE 5 and 4

Base Schema v50 Class	Base Schema v40 Class
Component	**Component** **System**
DefinedTerm	**Glossary**
Exit	**CompletionCriterion**
ExternalFile	**ExternalGraphic** **ExternalText**
Organization	**Leader** **ResponsibleOrganization**
ProgramActivity	**Activity**
ProgramElement	**Program** **Project** **WorkPackage** **WorkUnit (alias Task)**
Requirement	**Constraint** **OriginatingRequirement** **PerformanceIndex**
N/A	**Annotation** **Engineer**

[1] Department of Defense Architecture Framework (DoDAF).

Attributes

In addition to the removal of unnecessary attributes through the restructuring of the abstract classes, the majority of the v50 base schema attribute definition changes consisted of the adoption of simplified terminology. For example, **VerificationEvent** eventDuration became duration and **Risk** mitigationStatus became status. In v40, there was a proliferation of "type" attribute names which included a reference to the class, such as riskType. These have been changed to use the name type wherever possible. (This not only simplifies scripting, but allows the inclusion of a generic Type Sort Block useable across many classes.) Note that where an attribute's internal name changed, in most cases the alias remained unchanged.

Relations

Through a combination of restructuring the abstract classes and trimming of target lists, many unnecessary or non-meaningful class relations were removed, such as **Risk** *caused by/causes* **Category**, **Document**, **DomainSet**, etc.

The number of parent-child relations was minimized in v50. Specifically, the ten parent-child relations used in v40 were consolidated into four parent-child relations in the v50 base schema, as shown in Table 2.

Table 2   Base Schema v50 Parent-Child Relations

Parent-Child Relation	Class
built from / built in	**Component**
decomposed by / decomposes	**Function** **Item** **Product** **ProgramActivity**
Includes / included in	**Category** **Organization** **ProgramElement** **State/Mode** **VerificationEvent**
refined by / refines	**Document** **Requirement**

Note: To learn more about using the Base Schema v50, refer to the CORE System Definition Guide available on the CORE installation CD in the Docs folder, via the CORE Help menu under documentation, or via download from our web site.

CORE 4 to 5 Data Migration

Data Migration Overview

Migration from CORE 4 to CORE 5 is a two-part process. The first part is to upgrade your CORE software from release 4 to release 5. (Note: Refer to the CORE Installation Guide available on the CORE CD for detailed information on upgrading your software and receiving a CORE 5 license.) The second part is to migrate project data from the 4.0 schema to the 5.0 schema which represents the next step in the evolution of the CORE system definition language. While Vitech Corporation recommends that users update their software to CORE 5 as soon as possible, there is no requirement to migrate project data immediately or at all. CORE 5 supports both the new 5.0 schema as well as the 4.0 schema. While Vitech recommends that active projects migrate to the 5.0 schema, they should do so only after careful preparation. Projects in the latter stage of their lifecycle may be best served by continuing to use the 4.0 schema indefinitely.

Preparation for Data Migration

The single most important part of the process in migrating to the new 5.0 base/DoDAF schema is for members of the project team to familiarize themselves with the new system definition language before migrating to the 5.0 schema.

Migrating to the 5.0 schema involves more than migrating the data from the CORE 4.0 schema. If your project has made extensive use of CORE's flexibility to extend the schema or develop custom reports, you might decide not to migrate to the new schema at this time. Prior to deciding to migrate to the new 5.0 schema, the project team needs to evaluate how project-specific CORE customization that are dependent upon the schema definition language will impact the migration, including:

1. Schema extensions
2. Additional or customized scripts
3. Filters, sort blocks, and hierarchy definitions.

CORE 4 to 5 Data Migration Support

The database migration process consists of several activities designed to optimally convert your project data from the 4.0 schema to the 5.0 schema:

- ***Importing project data into a hybrid schema.*** A custom schema representing the 4.0 schema plus corresponding pieces of the 5.0 schema has been created to support project data migration. Though not appropriate for general use, this schema makes it possible to transform data from the old schema to the new.
- ***Utilizing the migration capabilities of the schema extender.*** CORE's schema extender includes powerful migration capabilities to support extensions and changes to the base schema. During the migration process, a conversion schema is loaded to handle renaming classes, attributes, and relations in the most efficient way possible.
- ***Caching and restoring modification stamps.*** While the database migration transforms numerous classes, attributes, and relationships, it does not alter the system engineering content of the database. It simply adjusts the terminology of the underlying schema. However, in the process of completing the transformation, the modification stamp on the affected elements is updated. The migration scripts include the option of caching the modification stamps at the beginning of the process and then restoring them at the end so that the modification stamp reflects the last engineering content change rather than the schema migration.
- ***Removing the ownership relationship.*** Previous versions of CORE automatically created the owned by relationship whenever an element was created. This information was redundant with the creator attribute and has been removed from the 5.0 schema. Those teams who wish to maintain information pertaining to the ownership relationship have the option of exporting this data to a comma-delimited file during the migration process.

- ***Generating a "debris" file.*** Significant effort has been put into streamlining the 5.0 schema to eliminate base attributes, relationships, and target classes that are unused when following the fundamental approach of model-based systems engineering with CORE as taught in the introductory course. This streamlining simplifies use and improves communication within the team. To ensure that no data is lost during this process, the migration scripts scan the database for these attributes, relationships, and target classes, outputting any occurrences found to a "debris" file. This process of identifying, outputting, and removing this information is controlled by an external configuration file which can be tuned as desired.
- ***Handling basic relationship transformations.*** In streamlining the schema, a number of relations have been changed to simplify and improve the system definition language. In some cases, a single relation has been separated into multiple relations (based upon the target class) to better separate the semantic meaning. Where semantics align, multiple relations have been consolidated into a single relation. The migration scripts read an external control file and then traverse the database implementing these conversions.
- ***Applying advanced transformation rules.*** Some transformation rules are more complex (e.g., creating a new element and populating its attributes and relationships if specific conditions are met). Custom scripts run as part of the main migration script handle these advanced transformation rules.

The database migration process must traverse the database several times to implement the transformation rules. Though the migration can be completed in CORE Enterprise, Vitech recommends that projects complete the migration in CORE Workstation and then import the resulting data file into CORE Enterprise. Vitech will provide a temporary license of CORE Workstation to those Enterprise teams who do not have access to Workstation as needed. Please contact your account executive for further details.

Step-by-Step Instructions for Data Migration

All steps of the database migration should be completed by a team member with administrator permissions for the project database.

Note: Vitech has developed a custom schema and script package available to help assist with the migration to the 5.0 schema. **This migration package can only be used with CORE databases that have not implemented any extensions to the base 4.0 schema**. See the Section "Migrating Data that Use Project-Specific Extensions" for additional guidance when migrating projects using schema extensions.

1. Contact Vitech Customer Support (support@vitechcorp.com) to obtain the latest set of migration scripts and special migration base schema **4.0 to 5.0 Conversion Schema**. **Note: This migration package supports projects that have not implemented any Project-Specific Schema Extensions to the base 4.0 schema**. Review the following sections for more information: Manual Migration Steps for Custom Simulation Scripts and Migrating Data that Use Project-Specific Schema Extensions and
2. Unzip the migration package and copy the folder contents to your **CORE Workstation** directory.
3. Export your project data from CORE 4.0 schema as an rdt file. Archive this file as a record of your project prior to database migration.
4. From the Home page of the Project Explorer, select New Project and create a new CORE 5.0 project using the special migration base schema **4.0 to 5.0 Conversion Schema**. This schema is a hybrid 4.0/5.0 schema which will facilitate the migration. This specialty schema should only be used for the migration.
5. Select File → Import to import the project data file from step 1 into the new project.

6. Select File → Import to import the schema file **40 to 50 Transformation.sch** from the 40 to 50 Migration folder (installed in the CORE 50 Workstation folder).
7. From the Project menu, select Shift to Database Mode.
8. Select Tools → Run Script and select the **4.0 to 5.0 Database Migrator** located in the Utility script folder.
9. Select whether or not to maintain the current modification stamps during the translation. Vitech recommends that you click **Yes** so that the modification stamps reflect your last engineering changes rather than the last change made during the migration process.
10. Select whether or not to remove the referenceList attribute for ExternalFile elements. This information is not used in current scripts. Vitech recommends that you click **Yes** to remove this information from the database.
11. Select whether or not to generate a file containing the *owned by* relationship for all elements.
12. Select the configuration file to control the basic relationship transformations. The default configuration file suitable for the 4.0 base and C4ISR schema is **Relationship Conversions.csv** in the 40 to 50 Migration folder (installed in the CORE Workstation folder).
13. Select the configuration file to control the removal of classes, attributes, and relationships no longer used in the 5.0 schema. The default configuration file suitable for the 4.0 base and C4ISR schema is **Debris Specification.csv** in the 40 to 50 Migration folder (installed in the CORE Workstation folder).
14. If you asked to generate a file containing the *owned by* relationship for all elements, specify an output file (.csv format suitable for review in Microsoft Excel).
15. Specify a debris file (.csv format) for the material removed from the database during the migration.
16. The migration scripts will now traverse the database completing the following steps:

    Step 1: Caching modification stamps (if requested)
    Step 2: Removing ownership relationships
    Step 3: Removing debris
    Step 4: Converting relationships
    Step 5: Handling advanced transformation rules
    Step 6: Restoring modification stamps (if requested)

    The time required to complete these steps is a function of the size and nature of your project database. Several schema transformation rules apply to OriginatingRequirements, Constraints, and PerformanceIndices so requirements-intensive databases will require more time to migrate.
17. After the migration script completes, export the database using the File → Export command. Only the database and folders should be exported (not schema). Select the CORE 4.x format and export the Project Database.
18. Import the transformed database into CORE 5. Review and archive the ownership and output files as appropriate.
19. To clean up files after you have finished migrating data to the 5.0 schema, you would want to delete the following file and folders from the CORE directory (the default directory for installation is C:\Program Files\Vitech Corporation\CORE Workstation):

    file: 40 to 50 Conversion Schema.bsh
    folder: 40 to 50 Migration
    folder: Reports\40 Reports\Utility\40 to 50 Database Translator

Manual Migration Steps for Custom Simulation Scripts

The database migration process does not traverse any custom simulation scripts looking for impacted elements. If your simulation models make use of the NamedElement Construct (particularly for selecting exit paths), you should review your simulation scripts to ensure that the constructs point to the appropriate elements.

Migrating Data that Use Project-Specific Schema Extensions

Projects teams who have extended the CORE schema may need to tailor the migration process to incorporate their extensions. Built upon the base 4.0 schema, the migration scripts and files can be extended to support schema extensions used by your project. The first step in migrating databases with custom extensions is to assess the impact of those extensions and how the extensions will be represented in the 5.0 schema. Many schema extensions will be seamless requiring no modification of the migration process or scripts. Other extensions may overlap the new 5.0 schema and require modification.

Schema extensions fall into three general categories:

- Simple attribute additions. These can frequently be handled by importing the current schema extension into the newly created project immediately after step 4 of the migration process and creating an equivalent attribute in the 5.0 schema.
- Relation extensions. Basic relation changes can often be handled by extending the conversion specification in **Relationship Conversions.csv**. Complex changes or those that conflict with the 5.0 schema may require custom transformation rules.
- Class extensions. Simple class extensions unrelated to transformed classes can be handled by importing the current schema extension into the newly created project immediately after step 4 of the migration process. Class extensions which conflict with class names in the 5.0 schema as well as class extensions which subclass transformed classes may require custom transformation rule.

Project teams are encouraged to contact Vitech Customer Support (support@vitechcorp.com) for assistance in migrating custom extensions. The support team is available to review your schema extensions and provide a high-level recommendation of the nature and degree of tailoring required. Your team can then review this recommendation and determine how best to proceed with the migration. Customized migration support services are available from the Vitech Professional Services team, as needed.

THIS PAGE LEFT INTENTIONALLY BLANK.

Vitech Corporation
2070 Chain Bridge Road, Suite 100
Vienna, Virginia 22182-2536
703.883.2270 FAX: 703.883.1860
Customer Support: support@vitechcorp.com
www.vitechcorp.com