Concurrent Assembly Mock-Up Best Practices

CADDS® 5 15.0

DOC40203-003

Parametric Technology Corporation

Parametric Technology Corporation, 140 Kendrick Street, Needham, MA 02494 USA

Table of Contents

Preface

Implementing CAMU

Managing Assemblies

Models

Assembly Structure

Tips and Troubleshooting

Positioning and Constraints

Preface

Concurrent Assembly Mock-Up Best Practices describes various techniques for implementing, managing, and using CAMU.

Related Documents

The following documents may be helpful as you use *Concurrent Assembly Mock-Up Best Practices*:

- *Concurrent Assembly Mock-Up User Guide and Menu Reference*
- *EPD.Connect User Guide*
- *Managing CADDS 5*
- *Installing CADDS 5*
- *Installing and Configuring EPD.Connect*

Book Conventions

The following table illustrates and explains conventions used in writing about CADDS applications.

Convention	Example	Explanation
Menu selections and options	List Section option, Specify Layer field	Indicates a selection you must make from a menu or property sheet or a text field that you must fill in.
User-selected graphic location	X, d_1 or P1	Marks a location or entity selection in graphic examples.

User input in CADDS text fields and on any command line	`cvaec.hd.data.param`
	
	`tar -xvf /dev/rst0`
		Enter the text in a CADDS text field or on any command line.

System output	`Binary transfer complete.`	Indicates system responses in the CADDS text window or on any command line.

Convention	Example	Explanation
Variable in user input	`tar -cvf /dev/rst0` filename	Replace the variable with an appropriate substitute; for example, replace filename with an actual file name.
Variable in text	tagname	Indicates a variable that requires an appropriate substitute when used in a real operation; for example, replace tagname with an actual tag name.
CADDS commands and modifiers	INSERT LINE TANTO	Shows CADDS commands and modifiers as they appear in the command line interface.
Text string	`"SRFGROUPA"` or `'SRFGROUPA'`	Shows text strings. You must enclose text string with single or double quotation marks.
Integer	n	Supply an integer for the *n*.
Real number	x	Supply a real number for the *x*.
#	`# mkdir /cdrom`	Indicates the root (superuser) prompt on command lines.
%	`% rlogin` remote_system_name `-l root`	Indicates the C shell prompt on command lines.
$	$ `rlogin` remote_system_name `-l root`	Indicates the Bourne shell prompt on command lines.

Window Managers and the User Interface

According to the window manager that you use, the look and feel of the user interface in CADDS can change. Refer to the following table:

Look and Feel of User Interface Elements

User Interface Element	Common Desktop Environment (CDE) on Solaris and HP	Window Manager Other Than CDE on Solaris, HP, and Windows
Option button	ON — Round, filled in the center OFF — Round, empty	ON — Diamond, filled OFF — Diamond, empty
Toggle key	ON — Square with a check mark OFF — Square, empty	ON — Square, filled OFF — Square, empty

Online User Documentation

Online documentation for each book is provided in HTML if the documentation CD-ROM is installed. You can view the online documentation in the following ways:

- From an HTML browser
- From the Information Access button on the CADDS desktop or the Local Data Manager (LDM)

Please note: The LDM is valid only for standalone CADDS.

You can also view the online documentation directly from the CD-ROM without installing it.

From an HTML Browser:

1. Navigate to the directory where the documents are installed. For example,

   `/usr/apl/cadds/data/html/htmldoc/` (UNIX)

   Drive:`\usr\apl\cadds\data\html\htmldoc\` (Windows)
2. Click `mainmenu.html`. A list of available CADDS documentation appears.
3. Click the book title you want to view.

From the Information Access Button on the CADDS Desktop or LDM:

1. Start CADDS.
2. Choose Information Access, the i button, in the top-left corner of the CADDS desktop or the LDM.
3. Choose DOCUMENTATION. A list of available CADDS documentation appears.
4. Click the book title you want to view.

From the Documentation CD-ROM:

1. Mount the documentation CD-ROM.
2. Point your browser to:

   CDROM_mount_point`/htmldoc/mainmenu.html` (UNIX)

   CDROM_Drive:`\htmldoc\mainmenu.html` (Windows)

Online Command Help

You can view the online command help directly from the CADDS desktop in the following ways:

- From the Information Access button on the CADDS desktop or the LDM
- From the command line

From the Information Access Button on the CADDS Desktop or LDM:

1. Start CADDS.
2. Choose Information Access, the i button, in the top-left corner of the CADDS desktop or the LDM.
3. Choose COMMAND HELP. The Command Help property sheet opens displaying a list of verb-noun combinations of commands.

From the Command Line: Type the exclamation mark (!) to display online documentation before typing the verb-noun combination as follows:

```
#01#!INSERT LINE
```

Printing Documentation

A PDF (Portable Document Format) file is included on the CD-ROM for each online book. See the first page of each online book for the document number referenced in the PDF file name. Check with your system administrator if you need more information.

You must have Acrobat Reader installed to view and print PDF files.

The default documentation directories are:

- `/usr/apl/cadds/data/html/pdf/`doc_number`.pdf` (UNIX)
- CDROM_Drive`:\usr\apl\cadds\data\html\pdf\`doc_number`.pdf` (Windows)

Resources and Services

For resources and services to help you with PTC (Parametric Technology Corporation) software products, see the *PTC Customer Service Guide*. It includes instructions for using the World Wide Web or fax transmissions for customer support.

Documentation Comments

PTC welcomes your suggestions and comments. You can send feedback electronically to `doc-webhelp@ptc.com`.

Chapter 1

Implementing CAMU

This chapter describes the Concurrent Assembly Mock-Up (CAMU) and its implementation requirements for use with Electronic Product Definition (EPD).

- Overview
- Determining Requirements
- Devising a Strategy
- Prototyping Solutions
- Running a Pilot Program
- Implementing CAMU for Production

Overview

Concurrent Assembly Mock-Up (CAMU) is used to design an overall product structure. You can use the product structure to manage the overall development of the product. You can design individual part models in the context of the overall assembly with several users working concurrently.

CAMU provides a better way of managing the assemblies that you design. You can see and work on both a logical representation of the product structure and on the geometry of the assembly.

Data redundancy is eliminated by associating part models to components of an assembly. An assembly can have several instances of the same component, each associated with a single part. Any changes you make to that one part are reflected in all component instances associated with that part and within all assemblies that contain the same component instance.

Assemblies created using CAMU can be made up of any combination of new or existing piece parts and subassemblies or both. Piece parts or subassemblies can be designed in both the Parametric and Explicit environments - either within or outside CAMU.

This document describes generic, best practices for implementing and using CAMU and EPD within an organization.

Implementing CAMU and EPD

The driving force for implementing CAMU is usually the need to reduce lead time, reduce rework, and improve the quality of information at the earliest possible stage of product development.

CAMU is an integral part of Electronic Product Definition (EPD), which is the former name of the combination of products - including CADDS and Optegra - people, and processes to provide an overall solution to improve the engineering product development process. Today EPD is more commonly referred to as flexible engineering and Collaborative Product Commerce (CPC), a pivotal element of which is Windchill.

Deciding to implement CAMU and EPD is a strategic decision that will have a profound and far-reaching effect on the way your organization operates. Although EPD focuses on the product development process, its impact is much wider. Its implementation involves more than technology. Often changes need to be made in the processes and people and their interaction.

Before implementing CAMU, you must determine how you want to use it within your organization. Successful implementation involves the study of the people, processes, and technology - and quite possibly adjusting the dynamics of all three.

Developing a Basic Implementation Process

Before implementing CAMU within your organization, develop the basic process that is required for any successful system implementation:

- Determine requirements for implementation.
- Devise an implementation strategy.
- Prototype solutions.
- Execute a pilot program.
- Implement a production program.
- Test and revise the implementation as required.

Determining Requirements

Successful implementation of CAMU often requires that your existing processes be modified to better use new technological capabilities. Considering the people and process issues is central to the success of the technology. You can address the people issues through training, support or consultancy, and general involvement. You can address the process issues in one or more of the following ways:

- Performing your own requirements analysis
- Completing an eFINDER exercise
- Performing a Quality Functional Deployment (QFD)

Creating the Requirements Document

Perform your own requirements analysis and create a complete requirements document. It must contain clear, unambiguous, and agreed upon functional and general requirements. In some cases, the output from a QFD or eFINDER exercise can yield the same degree of clarity as a carefully drafted requirements document.

The requirements document must have some degree of elasticity because the following situations can occur during any new technology and process implementation:

- You will learn what is really needed and what is possible to implement using the technology.
- Some items that seemed clear in the early stages of requirements definition can become ambiguous and require redefining.
- Some items that seemed unimportant can become priorities and vice versa.
- Some items may have been overlooked and will require adding to the implementation plan, while others may be removed from the plan.

There should be an agreed upon framework in which to proceed. This framework provides guidelines to scope and measure progress against.

- Estimates of the work involved
- Budget requirements
- Proposed milestones and time lines

Performing a Consultative Analysis

It is recommended that you ask our Global Services consultants to perform an eFINDER exercise to provide a business context in which to develop a framework for implementing EPD within your organization. The eFINDER method focuses on identifying the best practices methodology that will give a specific organization the greatest return on its investment.

The objective of the eFINDER is:

- Define and prioritize your business objectives
- Define and prioritize best practices
- Estimate potential benefits

Devising a Strategy

A typical strategy for fully implementing CAMU is to build a complete product database. Each user must have access to all aspects of the product based on their user authority through a single user interface, such as EPD.Connect, Optegra, and Windchill Workflow.

The path to build a complete product database is from a component-centered approach, through an assembly-centered approach, to a complete product-centered approach.

Gaining skills in model geometry definition and digital product assembly is your first step. This guide is designed to help you gain those skills by suggesting various best practices. If you have not already done so, you can obtain general CAMU instruction and information from the *Concurrent Assembly Mock-Up User Guide and Menu Reference* and the various modeling and related documentation.

The next step after gaining skills is to manage the assembly, and its components, within the framework of a project. Management of the project involves incorporating an integrated product structure, data, and workflow management system. PTC provides this solution in EPD.Connect, Optegra, and Windchill used in combination with CAMU. This combination of products requires a fit between the digital assembly and other engineering-related databases, such as revision control processes, scheduling systems, and others. It also requires that various nongeometric information (weight, cost, release dates) be accessible and tracked.

Prototyping Solutions

When there is consensus on the requirements for the implementation of your system, the best way of proceeding is to prototype a solution or solutions. Prototyping can mean producing dummy systems, that is, systems that look like the proposed solution but that do not actually work. Using a prototype is a productive way of testing the proposed usage of the system with a small group of potential users. This testing gathers their opinions, gets them involved, and determines ways of displaying functions that are appropriate and intuitive in the context of their work or task environment.

Other forms of prototyping may involve configuring specific pieces of off-the-shelf functionality tested with some type of dummy representative data, such as sample parts in a small scale assembly. This prototyping provides some experience in the application of particular types of functions to help shape the way in which the final solution needs to be constructed.

The results of the prototyping need to be reviewed, and the way to move forward agreed on between all parties. As the project unfolds, further periods of prototyping will again be appropriate and useful for new functions that are to be implemented later in the project.

Running a Pilot Program

Focus your initial implementation of CAMU on specific project teams in a pilot program rather than on widespread production. When you apply the lessons learned from the pilot program to a new project, you cannot simply streamline processes to improve productivity. You can begin a reengineering process that gives you the lead time, quality, and business improvements that you expect.

To run a discrete pilot program, use some or all of the following specifications:

- Number of users
- Range and scope of the data
- Total volume of data
- Breadth of functionality offered by the system
- Degree of integration with other systems

Pilot programs require that participants are appropriately briefed and trained beforehand and have ample resources available during the trial to help them resolve issues as they arise.

The experiences and the feedback gained from the participants in the pilot program are invaluable in determining issues that require resolution.

Implementing CAMU for Production

Set the expectations within the user community about what the system will and will not offer in its initial implementation for production. Use small group seminars to disseminate this information. The seminars must:

- Introduce the system verbally and in demo format.
- Explain why and how the system is commissioned.
- Present an operational overview.
- Communicate the training plan, if there is one.
- Contain a question-and-answer forum.
- Provide documentation that users can read at their leisure.

Encourage users to offer their criticisms, suggestions for improvement, and prioritization of enhancements, and so forth. A successful implementation depends largely on user support.

Designate a focal person within your organization as the coordinator of all activities involved in implementation. This person can serve as the liaison between PTC and your organization. Your liaison can provide a consistent channel of communication on all matters to do with the implementation.

Adding System Enhancements

Use a consolidation period of several months between the first implementation and any attempt to add further system enhancements. Aim for a moderate amount of improvements and determine an effective process.

Determining Effective Processes

Use small teams to devise effective working practices and processes based on your system requirements. You can use the eFINDER to produce easy-to-understand process maps. These process maps provide a starting point for identifying the opportunities for improvement. The eFINDER also helps you to identify the priorities and benefits of process improvements.

Some of the questions about determining effective processes that you need to answer are:

- To what level of detail do I build the assembly structures?
- Which components or assemblies do I concentrate on first?

- How do we handle product derivatives?
- How do we use CAMU and EPD.Connect to manage data and work processes using Optegra Gateway and Windchill?
- Who is responsible for the assemblies and at what stage?
- How do I know the assembly structure is complete and up-to-date?
- How do we manage revision control, sign off, and so forth?

Chapter 2

Managing Assemblies

This chapter presents the followings topics:

- Managing Data
- Archiving the Assembly
- Recovering from System Problems
- Understanding Multiuser and Single-User Mode
- Understanding the ODB_SERVER Process
- Understanding the _db File

Managing Data

You can use a data management tool, such as Vault, to perform check-in, check-out, and revision control on the assembly and all its constituent components.

Please note: The best way to use CAMU and Vault together is from within EPD.Connect.

Tracking Assembly Database Items

Use the database management system to track the assembly directory. The assembly directory is same as the Name entered on the Activate New Assembly or Activate Old Assembly property sheet.

The assembly contains the following items:

- The assembly database file `_db`
- All Adrawings (CADDS parts subordinate to the `_db` used to view, position, and document the assembly geometry) within the assembly directory, such as `default`
- The `root` model (if there is one) within the assembly directory
- All models associated with components within the assembly structure (`_db`)

Including Model Items in the Assembly Database

The database management system must include each model associated with its components within the assembly structure. The model directory, that is the same as the name entered on the Activate Model property sheet (or outside CAMU, on the Activate Part property sheet), contains some or all of the following items that must be managed:

- The part database itself (`_pd` and `_fd`)
- The explicit graphics file (`_gr`) containing the drawing and model graphics data along with the tessellation data
- The `vp_links` directory. This directory must exist to house the files generated by the VIEW COMPONENT command. When you choose not to view the component, or if you exit the assembly, these files are removed. In case of an abrupt termination, you must manually delete these files.

  You can delete these files using scripts or external utilities such as `slay` or `cleanup`. You can also use the `odb_admin` utility, but be careful when you use it in multiuser mode. The `slay` and `cleanup` scripts are available from the

User Default environment that can be created using the install tools in (Software Loading and Installation Command) SLIC. You can also find these scripts at the path:

`/usr/apl/cadds/scripts`

Please note: Make sure that no user is working on the assembly of which you want to delete the `vp_links` directory.

Locking Models

When locking models, do not use the Model option from the LOCK COMPONENT command unless you want to prevent other users from activating the associated model. When you use the Model option, CAMU creates a `LOCK` file and a `TEMP` file in the associated model directory. If system stops responding, you must delete these `LOCK` and `TEMP` files - a potentially tedious operation. Closing any suspended models after you file them will save your time in the long run. You can use scripts to delete the `LOCK` and `TEMP` files, as explained in the previous section.

Using the File Assembly Property Sheet

When you file the assembly using the FILE ASSEMBLY command, all Adrawings, models, and assembly structures in your active assembly, whether suspended or not, are filed in the database. This process can be time-consuming. Instead, consider using FILE > FILE ASSEMBLY W/OPTIONS. The File Assembly property sheet appears. You can select the following filing options:

- Assembly Structure — to update the `_db` file, the `default` Adrawing, and the root node model.
- All Adrawings — to file the active Adrawing and active model. You can also specify the Adrawings and models to be filed and updated from runtime lists in the property sheet.
- All Models — to file all or selective active and suspended models from a runtime list.

The view state information of an Adrawing, such as that of an assembly, is stored in the assembly database `_db` file. Therefore, you must file the assembly structure with the assembly Adrawing at once in order to save the view states of the Adrawing.

Maintaining an Assembly Database

You must establish consistent maintenance practices when working with CADDS assembly databases, because dependencies on external references can compound problems. Some of the issues you may encounter are:

- After running `validate_db` or `ckCAD` on any of the components in your assembly (on the `_pd` file), always delete the `_gr` file and regenerate the file. After using the database maintenance tool on a part database file, the associated `_gr` file can become out of sync with the parts `_pd` file.
- If you want to copy an assembly from unknown sources to your active assembly, first check the integrity of the assembly structure tree in a new assembly. When satisfied with the assembly's integrity, insert the external assembly into your current assembly. Check that the inserted assembly does not create a nested-nested reference assembly. A nested-nested reference assembly is reference components or assemblies one level above the current reference. The assembly tree structure must contain only the components and reference assemblies that are in your part list.

Archiving the Assembly

The following items must be archived for any given assembly:

- The assembly database (`assembly_name/_db`)
- Adrawings of the assembly (`assembly_name/adraw_name/_pd and assembly_name/adraw_name/_fd`)
- The DDRAW of the Adrawing named default (`assembly_name/default/_gr`)
- Models associated with component classes in the assembly and product structure (`model_name/_pd` and `model_name/_fd`)
- The explicit graphics file (`_gr`) for all models associated with component classes in the assembly or product structure (`model_name/_gr`), which contains the drawing and model graphics data along with the tessellation data.
- The `vp_links` subdirectory of each model (`model_name/vp_links`) (See "vp_links Directory" on page 2-7)
- Explicit figures used by or within models or Adrawings (`figure_name/_nfig` and `figure_name/_gr`)
- Procedure and execute files
- Applicable feature databases
- The NC toolpaths (`model_name/*.jcf`)
- Model-specific CVMAC routines

CADDS Drawing Files

All the CADDS drawing and model graphics information is stored in the explicit graphics file (`_gr`). The `_gr` file is subordinate to the CADDS part (model) that was active at the time you issued the ACTIVATE DRAWING command. You must consider archiving the `_gr` file of each model in your assembly when you archive the assembly and its models so that you can view the component in CAMU.

Please note: Whenever you activate and save a part, the size of the explicit graphics file (`_gr`) file may increase or decrease by up to 1%. The information in the file is restructured or reordered every time you save the part. This is an expected behavior when compressed data is read from and written to a file.

If the `_gr` file of your model is missing, the VIEW COMPONENT command dynamically generates the graphics data for that model by reading the `_pd` file of that model. However, this can adversely impact the performance of the VIEW COMPONENT command. Additionally, to view the model in any of the shade or HLR rendering modes, the `_gr` file should contain the tessellated data for the solid

or surface geometry in that part. To create the tessellated data in the `_gr` file, activate the part in the Explicit environment, run the Render View or Change Renderview command with any of the shade or HLR rendering options, and file the part without specifying the NOSHADE or NOTVF option.

vp_links Directory

The `vp_links` directory contains the files that are created by the VIEW COMPONENT command. These files exist only for the duration of time that the associated model geometry is viewed. They are deleted when you choose to exit or not to view the assembly. This directory is required for viewing the model in CAMU. For this reason, you must archive the `vp_links` directory of each model in your assembly when you archive the assembly and its models. Do not archive the contents of a `vp_links` directory.

A `vp_links` subdirectory is created when you activate a part using the ACTIVATE PART or the ACTIVATE MODEL command, provided you have the following line enabled in your `.caddsrc` or `.caddsrc-local` file:

```
setenv CADDSENABVP 'yes'
```

If you do not have the above settings in your `.caddsrc` or `.caddsrc_local` file, you can use the `caddsenabvp` script to create the `vp_links` subdirectory. Using the `caddsenabvp` script you can create a `vp_links` directory for every CADDS part in a specified directory. The script is at the following location:

```
/usr/apl/cadds/scripts/caddsenabvp
```

Recovering from System Problems

If the system stops responding or fails when you or another user is active in the assembly, follow this crash recovery sequence:

1. Run the `slay` script to stop your CADDS processes.
2. Remove the `LOCK` and `TEMP` files from the following items:
   - All suspended models
   - All suspended Adrawings
   - The active part (Adrawing or model)
   - The active assembly database
   - All models that you have locked by specifying the MODEL option with the LOCK COMPONENT command.
3. Remove all files within each applicable `vp_links` directory. When the system fails, junk files accumulate in the `vp_links` subdirectory of each model that was viewed using the VIEW COMPONENT command. Delete the files, but not the `vp_links` directory.

Please note: Make sure that no user is working on the assembly of which you want to delete the `vp_links` directory.

4. Use the `odb_admin` utility to remove users with a crashed assembly session.

Cautions for the ODB_SERVER Process

The ODB_SERVER process provides multiuser concurrency control on an assembly. The `/usr/apl/cadds/scripts/odb_admin` tool allows individual users to clean up their own session from the ODB_SERVER without interrupting or stopping other activities.

Note the following cautions for ending the ODB_SERVER process:

- Do not use the UNIX `kill` command to end ODB_SERVER process if any user is still working in the assembly.
- Never use the command `kill` with the `-9` option. This stops the ODB_SERVER process and leaves the ODB_DAEMON process incomplete. You may not be able to activate the same assembly again if the ODB_DAEMON process is terminated early.

ODB_SERVER Process in a Multiuser Environment

The UNIX utility `odb_admin`, located at `/apl/cadds/scripts`, removes the users who have abnormally exited the assembly from the `ODB_SERVER` list. If you use this utility, you avoid having to restart the `ODB_SERVER` for everyone. The `odb_admin` utility displays the following message and then clears the users that are not active from the `ODB_SERVER` list.

```
## Enter the full path assembly name
```

The use of the `odb_admin` utility is well documented in the *Concurrent Assembly Mock-Up User Guide and Menu Reference*.

Understanding Multiuser and Single-User Mode

To use CAMU effectively, you must understand the difference between the multiuser and the single-user modes of CAMU. The following sections explain the modes in detail.

Multiuser CAMU

Multiuser CAMU is one server providing the `ODB_DAEMON` process to a number of clients. The clients are set up to connect to this machine through the `DB_DAEMON_HOST` variable set in the `.caddsrc` file. The assembly database should reside on the same machine that has the `ODB_DAEMON` process running.

All the current active users can view the changes made to the assembly by refreshing the assembly.

You can use the following commands or status window to identify the status of an object in multiuser CAMU environment:

- VERIFY OBJECT displays whether a component is active or suspended, and if active, which user is working on it.
- LIST ADRAWINGS displays the details of Adrawings in the assembly, including which user has write permissions to a particular Adrawing.
- The CAMU Status window states the name of the selected component, its associated model name, the name of the user who has locked the component, and its status (inactive, active, or suspended).

Single-User CAMU

Do not confuse the single-user CAMU with Parametric Multi-part Design (PMD). The single-user CAMU is an individual machine with its own `ODB_DAEMON` process. The single-user CAMU offers other benefits over PMD besides concurrent operation, such as:

- Supporting reference assemblies
- Zooming of assembly structure tree
- Compressing and expanding the assembly structure

The single-user CAMU setup is very common, but you cannot attain true concurrent engineering with this kind of setup.

Note the following points before you decide on multiuser or single-user CAMU mode:

- If many users share the single-user CAMU and activate the same assembly, naming conflicts can occur. This problem does not occur with multiuser CAMU, because there is only a single server maintaining the assemblies.
- With single-user CAMU, the site has to maintain multiple servers.
- Single-user CAMU is easier to set up, but you must be able to manage the assemblies properly.
- Multiuser CAMU may need EPD implementation and customization, for example, zoning, Product Structure analysis, Read Only Areas (ROA), and PDM.

Understanding the ODB_SERVER Process

The CAMU server for the multiuser has an ODB_SERVER process for each assembly. Multiple users can be active in any assembly. The SERVER process has the name of the assembly. For example, in a UNIX shell, if you give the following command:

```
#ps –ef |grep ODB
```

the following result is displayed:

```
/usr/apl/cadds/bin/ODB_DAEMON
/usr/apl/cadds/bin/ODB_SERVER=USR2.CAMU.PARTS.ASSY1&DB
```

Interrogate the assembly using the `odb_admin` tool to determine the number of active users in each assembly.

If you require more information, refer to the Help pages.

Understanding the _db File

The _db file represents the assembly and is a binary file. It contains information that relates to the nodes of the parts, relative position, and global or instance attributes of the components of the assembly. The file also contains the Adrawing names and view states for each Adrawing of the assembly.

Since CADDS 5 Revision 6, the manner in which the view states are stored for each Adrawing in the _db file has changed, and the size of the CAMU _db file has decreased significantly. For CADDS 5 Revision 5 assembly, it is necessary to activate each existing Adrawing in order to update the format of the view state information to the new format and file the assembly. You can then see a significant decrease in the size of the _db file.

Editing the _db File

You can convert the _db file to an ASCII file called _ps (product structure). You can edit this ASCII file and restore it as a _db file. The mechanism for this procedure is mainly within EPD.Connect.

Since CADDS 5 Revision 9, the problems of compatibility were solved by introducing CABagent. EPD.Connect is sufficient for interrogating assemblies.

Controlling the Size of the _db File

The amount of real data stored is dependant on the size of the assembly. This _db file can grow to a large size. For an assembly with 2,000 components, anything up to 5 MB is acceptable when not active. You must compress the structure frequently. The size of the _db file grows while the assembly is in use, and the file size is compressed when the last user of the assembly exits the assembly normally.

Controlling the Size of the _db File

Minimize the size of the _db file as follows:

- File the assembly using the Structure option on the File Assembly/w Options menu.
- File the assembly using the File option on the File and Exit Assembly menu.

Using the previous two options compresses the _db file when exiting from the assembly. Using these options can be time-consuming, but they prevent the corruption of assembly database files in the future.

- Use the Reference option, instead of the Copy option on the Add Assembly menu to reference an external assembly. Any changes to the reference assembly are instantly reflected in all of its references.

  If you use the Copy option on the Add Assembly menu, the objects from an external assembly are inserted to the main assembly as a local object, causing an increase in the size of the _db file of the assembly.

Chapter 3

Models

This chapter describes best practices for the following items:

- Guidelines for Creating Model Geometry
- Guidelines for Shading Model Geometry
- Detailing Models from within CAMU

Guidelines for Creating Model Geometry

Use the following guidelines to create model geometry:

- Use solid or surfaced double-precision or C5 models. Surfaced models can be either ISD (Interactive Surface Design) or NURBS (Non-Uniform Rational B-Splines) trimmed surfaces.
  - Represent surface models with finished, trimmed NURBS surfaces, not with boundary or profile curves overlaid or projected onto untrimmed surfaces patches.
  - Ensure that models within the assembly are of the same precision.

    You can set up a global configuration part in your own home directory. A configuration part is configured as a template with defined construction defaults. This template of defaults is used to create any future part.
  - Associate 3D wireframe models (if they exist) to the relevant component in the assembly structure. Rebuild them as 3D solids or surfaces as time permits.
  - Generate solid models as explicit or parametric models. The model should be detailed in an explicit drawing and should contain a current `_gr` file. This file should contain the tessellation data that is synchronous with the model.
- Do not create model geometry using NURBS. However, trimming must yield NURBS surfaces.
- Ensure that all surface models have a mesh suitable to the shape of the surface (that is, 2x2) before filing, in order to identify the surface models while viewing large models and assemblies.

Please note: A general rule for maintaining visual clarity within the assembly is to have surface graphics meshes no larger than 2x2.

- Blank all model geometry not required for viewing by the rest of the team before filing the model, for example, construction geometry, layers, and others.
- Ensure that all components have entities to assist with their location in an assembly, such as, lines down the center of bores and shafts where they mate with other components and lines along the x-, y-, and z-axis of the part datum (model space origin of the model).

Blanking Model Construction Geometry

Filling the assembly with the blanked model construction geometry provides two advantages:

- Prevents the construction geometry from appearing within the assembly when the model associated with the component is viewed.
- Displays all layers regardless of the layers that were echoed when the model was filed.

Controlling Model Space Origin of a New Model

Specifying the origin (Model Space Origin) of a model you are about to create is of key importance. If you do not specify the MSO before creating the model using the ACTIVATE MODEL command, the MSO of the model will coincide with the origin of the active assembly. You can create piece parts that are properly positioned within the active assembly using this default location of MSO, but you cannot use these models in other assemblies. It is very difficult to position the piece parts if you use these models in other assemblies.

You must specify the model space origin for any model in your assembly before you create the model.

The *Concurrent Assembly Mock-Up User Guide and Menu Reference* explains the procedure to create models in the Parametric or Explicit environment using both the default and the user-specified location in regard to the model origin.

You can create a sample part which can be used as the AD_HANDLE_PART referenced in the `.caddsrc` file. Ideally, this part should contain three lines at x0y0z0, one for each axis and each a different length. After it is created, this sample part can be accessed when any component that is not associated with a model is chosen with the TRANSLATE, ROTATE, and ORIENT COMPONENT commands within the Assembly Design environment.

Understanding Viewed Geometry

Viewed geometry is visible but does not reside within the active part database. You can see it, but you cannot edit it. See also “Creating Assembly Engineering Drawings” on page 5-3. The characteristics of viewed geometry follow:

- If an entity is blanked within the viewed model, that entity does not appear when you view the model in your assembly using the VIEW COMPONENT command.
- If you have applied fonts on an entity using the Allviews option within the viewed model, then the applied fonts appear when you view the model in your assembly. If the Allviews option is not used, you do not see the fonts when viewing the model.
- Any command that creates geometry, such as Explicit Booleans, dimension insertion, sectioning, and other entity insertion commands, creates the geometry in the active part. To see the name of the active part, view the Status bar using the Information icon in the upper left of the display.

Guidelines for Shading Model Geometry

You can shade the active assembly in the explicit graphics window using one of the following shading commands:

- RENDER VIEW
- CHANGE RENDERVIEW

When you start shading the assembly, CAMU uses the tessellation data that is stored in the `_gr` file of each model in the assembly. This tessellation data is in sync with the model if you had used the RENDER VIEW or CHANGE RENDERVIEW command at least once for that model.

In this case, the tessellation data stored in the `_gr` file is updated when you file the assembly.

The quality of the shading is affected by the shading smoothness factor used when the tessellation data is created for that model. You can set the smoothness factor using any of the following command:

```
SELECT GRAPHSHADE <smoothness>
```

The greater the value of the smoothness factor, the finer the tessellation. Additionally, the `_gr` file created on filing the part will have a larger file size.

Please note: To see viewed models using the VIEW COMPONENT command with hidden lines removed or shaded in the CADDS graphics window, each model requires a current `_gr` file containing the drawing and model graphics data along with the tessellation data.

Detailing Models from within CAMU

The MDRAW drawing facilitates visually seamless movement as you move between active models. You can activate another drawing (not named MDRAW) when active in the model by using the following menu sequence:

Adrawing > Drawing > Activate New or Activate Old

When you are in an active drawing, define views as you normally do when creating explicit detail drawings. For example, define a Front view and fold a Top and Right view from it. Insert associative dimensions.

Please note: All models that were visible in the previously active part are also visible in the MDRAW. To display only the model geometry that resides within the active part, use the BLANK COMPONENT command.

To detail a model from within CAMU, activate it and then activate any other new or existing drawing within the model (Adrawing > Drawing > Activate New or Adrawing > Drawing > Activate Old). Do not detail the model when you are active in the MDRAW.

Please note: Do not use a model's MDRAW drawing as an engineering drawing. Views that you define on the MDRAW are not saved.

Chapter 4

Assembly Structure

This chapter presents the following topics:

- Determining How to Structure Your Product
- Creating the Assembly Structure
- Establishing Naming Conventions
- Using Special Assembly Sheets
- Using the Reference Assembly Approach
- Accelerating Assembly Activation
- Activating a New Component

Determining How to Structure Your Product

One of the first steps in creating an assembly is to determine how to structure the product. The second step is to determine the best approach.

Structuring Approaches

You can design your CAMU assembly structure according to functional groupings. The three most prevalent scenarios are as follows:

- Create a master assembly containing the complete product assembly. Specify the users who can manipulate, view, or query the master assembly.
- Create a separate assembly for each major functional grouping, or create main subassembly branches. For example, a vehicle would contain a body, chassis, electrical, and other assemblies.
  - The project manager can assemble the entire product by adding each of these functional assemblies, that is, body, chassis, electrical, and others to the master assembly database as reference assemblies.
  - The users within each of these subassemblies can see the entire assembly or specific areas by adding each of these top-level assemblies into their own assembly as a reference assembly.
  - This structure type can prevent nesting and looping of reference assemblies.
- Create trees according to geographic area, and within these areas, break down the trees by functional groups. One technique is:
  a. Divide the product, a car, ship, airplane, and others, into a series of zones based on factors such as complexity, people involved, or geographic locations for design and manufacture.
  b. Create zone trees to group all the models together within the geographic zone.
  c. Use a module-type approach to construct trees from the functional areas but keep them divided by geography.

  For example, a car manufacturer can have a number of trees in the assembly structure for the brakes module. The number of trees depends upon the brakes module and how many zones it contains.

Determining the Best Approach

Each approach to structuring a product has its own advantages and disadvantages. Determining the best approach depends on your design environment, systems, and procedures.

Please note: You may find that much of your subassembly division already exists through functional groupings. If so, use these functional groupings as the basis for your CAMU assembly structures.

- Creating a master assembly containing the complete product assembly, as described in "Structuring Approaches" on page 4-2, is the best approach because it is true concurrent design. However, factors such as logistics and network considerations may preclude this option in your design environment.

  Guidelines for this method are described in the section "Creating the Assembly Structure" on page 4-4.

- Creating a separate assembly for each major functional grouping, described in "Structuring Approaches" on page 4-2, is relatively easy, because each function has a tree consisting of the structure and geometry required. However, this approach does not promote cross-functional design. This approach often involves duplication of effort with regard to adding and removing reference assemblies.

  A scenario depicting this approach is presented in "Using the Reference Assembly Approach" on page 4-11.

- Creating trees according to geographic area, described in "Structuring Approaches" on page 4-2, has a high number of variants or installed systems, and procedures can make this more complex than it seems at first.

Creating the Assembly Structure

Guidelines for creating a master assembly containing the product assembly, described in “Determining How to Structure Your Product” on page 4-2, follow.

Assembly Structure

- Build a single master assembly in which all users can work concurrently. Each functional group requires its own subassembly branch within the master assembly.
- Structure the assembly based on functional groupings. For example, a chassis subassembly includes the brake system subassemblies.
- Assign an owner to each major subassembly branch within the master assembly. The owner generates, populates, and maintains the subassembly branch. A single branch containing components from several designers requires a single owner.
- Include all high priority parts in the assembly structure. The group or project team decides the priority as package-critical or weight-sensitive, or they develop other criteria.
- Use a standardized naming convention for component names within the assembly structure and within your CADDS models. If you change a component class or a model name, tell your group. The master assembly root node as well as the top-level node of each subassembly must also conform to the naming conventions.

Associated Models

- Ensure that the associated models in your assemblies have at least read access to their `vp_links` directory.
- Associate CADDS models to leaf nodes only and not to the subassembly nodes.

  To blank the assembly structure, select the root node and use the Blank option on the Display Structure property sheet.
- Ensure that all models associated to components within a branch belong to the same functional group.
- Use the No Model option on the Change Component property sheet for each component class that does not have an associated model, such as the subassembly components.

- If the subassembly branch contains models from two or more component areas, determine the ownership of the branch.
- If the model geometry has an alternate design, the alternate design must be shown within the assembly structure along with the existing design.
- Where subassembly branches consist of derivative models, important derivatives based on worst-case conditions must be shown in the assembly structure.

Establishing Naming Conventions

Establish and adhere to a standardized naming convention for each of the following objects:

- Models
- Drawings
- Adrawings
- Assembly Structure Components

Naming Conventions for Models

Specify a unique model name and part name. Make sure that the name you specify does not relate to the model or the part.

Model Name

The model name must match the actual part number it represents. This ensures that there is no duplication of data for parts in the production stage. It also ensures that links, which allow other people to view data during design, are not broken.

Derivative Parts

Use a model as the basis upon which to build another model for the purposes of manufacturing, packaging, or creating envelopes outside the models, for issues such as clash detection.

The current practice is to copy these parts to a new name and use them within the assembly. Copying the parts breaks any link or relationship with the parent part that could possibly cause problems downstream, including data duplication.

You can use the Insert Ppart functionality to establish a link between the derived part and the parent part. This helps you keep the imported part in sync with the parent part. Using the Insert Ppart functionality you can insert the complete parent part or the required entities of the parent part into the target part.

Create a solid model if a part exists in wireframe and is needed for packaging or testing. This solid model can be a representative part rather than the full-detail geometry. For example, the packing engineer cannot modify the released part. A copy of the part can be made with the -`CAMU` extension appended to the part name.

For example

```
Original release partname = XYZ.123456
Modified part for CAMU = XYZ.123456-CAMU
Part representing envelope = XYZ.123456-PACK
```

Although you now have three different parts, they can be controlled easily in CAMU. Use the revision mechanism to control the level of detail of the part that you want to see.

The tree looks the same - one node representing each part. You can use revision control to swap in or out of various versions of the part that you want to see.

For example, create a node on the tree using the ADD COMPONENT command. Use the CHANGE COMPONENT command to associate the tree node to the part `XYZ.123456`.

View this part in the assembly using the VIEW COMPONENT command. Use the Revision option on the Change Component menu to swap in and out of the different versions of the geometry. If you display the part `XYZ.123456`, which is the original production part name, and you want to see or work on the outline version of the component, select the relevant node on the tree and open the Change Component menu. The displayed part automatically switches to the CAMU model, that is, the modified copy of the release part, when you add CAMU to the revision control. The released version of the part is displayed to replace the CAMU version when you choose the NO REVISION option.

Naming Conventions for Drawings

Drawing names must adhere to the existing naming conventions, for example, `sheet1of3` or some other convention. When you activate a model from within CAMU, a drawing named MDRAW is activated. Do not perform any tasks, such as defining a view, dimensioning, or sectioning, because MDRAW cannot store views. The MDRAW is designed to facilitate seamless visual transition when switching nodes.

Naming Conventions for Assembly Drawings

An assembly drawing, that is, an Adrawing, enables you to see and visually manipulate the model geometry that comprises your assembly or product. When you activate an Adrawing, both a CADDS part and a drawing named DDRAW are activated. To activate an Adrawing, use Adrawing > Drawing > Activate New or Adrawing > Drawing > Activate Old.

You can have multiple Adrawings within an assembly. Use a descriptive name for Adrawings, for example `Exploded-All`, `Design`, `Section AA`, `Sheet1of2`, `Sheet2of2`.

Please note: Each Adrawing has a standard view state, that is, location and orientation of model instances. Each Adrawing can also have an exploded or alternative view state, that is, location and orientation of model instances that are modified.

You can activate additional drawings within an Adrawing. Establish a naming convention and adhere to it.

Please note: Unless otherwise specified, an Adrawing named `default` is automatically activated when you activate an assembly.

Naming Conventions for Assemblies

A sample naming convention for an assembly is explained in the next table. You can choose a convention that is simple or complex. In the example in the table, the assembly name contains the following items:

- Assembly type abbreviation
- Project name or code
- Four-digit functional group code
- Brief descriptor (optional)

The assembly name `ASM.VBD8050.5110.ABS` consists of following items:

Code Value	Description
ASM	Assembly name or code
VBD8050	Project name or code (project name matches the Vault project name)
5110	Functional group number
ABS	Brief Descriptor (optional)

Major product variants can be named at a higher level in the tree. For example, you can use the names Mid-engine, Rear engine, and others. Detail level variants, such as the inclusion or exclusion of a particular vehicle feature, are shown at the functional group level. The detailed level variants are shown in parentheses. For example:

```
ASM.VBD8050.5110.ABS (brake system with deluxe brake features)
ASM.VBD8050.5110.NONABS (brake system with simple brake features)
```

An assembly may not exist for every variant. Several variants can be used in a single assembly. Both of the previous top-level nodes can be subassemblies or branches within the same assembly. The `ASM.VBD8050.5110` assembly could contain the `Brake-abs` and `Brake-noabs` subassembly nodes and their applicable branches.

Naming Conventions for Components

A general guideline for naming components is to use descriptors as component class names and part numbers for the associated model names. Component names must not reflect the associated model part names. Use a standard descriptor, such as chassis, rim, or body. Using this descriptor you can use many different part configurations within the same assembly tree. Using this descriptor is ideal during the design and prototyping stages. At the conceptual stage, generic trees can be used for product configurations to avoid duplicate part names in different products.

Using Special Assembly Sheets

This section describes the three assembly sheets used by CAMU.

ADRAW

An ADRAW is a part of an assembly that stores the details of viewed parts, layered offsets, exploded positions, and other viewed details of the assembly. You can have multiple ADRAWs in an assembly, but in a multiuser CAMU mode, each user must have a unique ADRAW in the working assembly. Characteristics of an ADRAW are:

- When you create a new assembly, an ADRAW is created with the name `default`. You can then change it to another name.
- Each ADRAW has an associated DDRAW.

DDRAW

A DDRAW is a part of an assembly that stores the informations related to the views and layers while an ADRAW is active in the assembly.

- Do not save the drawing named DDRAW if it contains CADDS entities.
- Do not create any geometry or perform a hidden lines removal operation in a DDRAW.

MDRAW

An MDRAW is a CADDS drawing file that stores the view state, or orientation, scale, number of views, and viewed model instances, of the assembly when moving from one active model to another. The MDRAW maintains the view state of the assembly only until the ADRAW is active.

- You cannot activate an explicit MDRAW.
- You cannot use all viewing-related and drawing-related commands on an MDRAW.

Using the Reference Assembly Approach

Reference assemblies contain the basic building blocks of the functional groupings and are used to build trees required above this level.

Use CAMU from the conceptual to the engineering development stage. In the early stages, the packaging engineer builds the tree with input from the component engineer. This tree may not have a highly detailed structure. The tree can evolve as the design progress from simple to complex.

For example, in the case of seat assembly named SSM, the packaging engineer knows that he needs front seats. He does not know if it will come from a supplier or will be designed in-house. The only information he has on his tree is a seat node or seat reference assembly. The geometry may not have a correct part name and may be a space envelope for the seat assembly. It may be that an old assembly or envelope is being used. At a particular point in the design process, the engineering of the seat is transferred to the interior design engineers who start the detailed design of the seats. The design engineer now has the responsibility to generate and maintain the trees for the seat assembly. The design may be created in-house and the seat assembly, SSM, may break down into two more levels.

The seat node is created as a reference assembly, ASSY.proj.SSM3000, where 3000 is the functional group for seats.

```
Seats
     Front_seat
             Seat_Chassis
             Seat_Headrest
             Seat_Frame
```

The tree structure is as follows:

```
             3000    Seats
             3100    Front_seat
             3001    Chassis
             3002    Headrest
             3003    Frame
```

The packaging engineer never has to worry about what is happening to the seat design or the tree structure. The reference assembly is 3000. When this assembly or the parts are updated, the packaging engineer sees the new version concurrently. The component engineer builds trees in-line, and the packaging engineer uses these trees to complete the vehicle.

In the early stages of development, if the component engineer has not constructed the trees to specifications and is working with single parts, the packaging engineer can add his own nodes to include the basic geometry being developed. The component engineer must remove the individual nodes and replace them with the reference assembly when the tree has been developed. The packaging engineer must not modify the geometry.

In the early stages of the design process, use the correct part name. Name the preliminary model and use the same name or append a suffix such as pack to the name for revision control. By using the suffix pack for the revision, the packaging engineer can use the same part name on the tree. He can then remove the revision on the menu and swap it for the new part when the real part arrives.

Accelerating Assembly Activation

Use the following methods to accelerate the assembly activation:

- Always activate an Adrawing named `default` or some other name that contains no viewed components or geometry.
- Visually blank the assembly structure from the root node down by selecting the root node and using the Blank option on the Display Structure property sheet before you file and exit the assembly. Visually blanking the assembly structure accelerates the process of activating the existing assembly.
- From CADDS 5i Release 11 onward, use the Level option of the ACTIVATE ASSEMBLY command to control the level of nested reference assemblies that can be opened. This option has been integrated into the ACTIVATE ASSEMBLY command with two other commands, OPEN REFERENCE and CLOSE REFERENCE. Use the Level option to improve performance by activating the assembly with level 1 and using the OPEN REFERENCE command to open only the first level of a specific reference assembly.

Activating a New Component

Note the following points before activating a new component of an assembly:

- Check the orientation and the location of the component origin.
- Create a part with three lines of different lengths, representing the three axes. Use this created part as the handlepart. You can use this handlepart to locate and orient the other models associated with the assembly components.
- Change the location and orientation. You can do this at any time, but it is best to do it first.
- Select the ADRAW before activating a part if you want to work in a part with an exploded assembly.

Chapter 5

Tips and Troubleshooting

This chapter describes the typical issues you may face while using CAMU and offers tips and troubleshooting.

- Tips and Reminders
- Encountering Network Problems with CAMU

Tips and Reminders

This section contains useful tips and reminders for using CAMU.

Using Optegra Vault

Inserting and retrieving an assembly tree into and from the Vault is currently a two-stage operation:

1. Select EDM > Access mode > PARTS and use the PUT, PUT NEW, or UPDATE command on the assembly ADRAWING, for example, `ASM.proj.3000.DEFAULT`
2. Select EDM > Access mode > FILES and use the PUT, PUT NEW, or UPDATE command on the assembly `&db` file, for example, `ASM.proj.3000.&db`

Please note: Inserting the assembly tree to Vault does not put all the models into the Vault as well. You must insert the models individually.

Filing Selectively

The assembly database consists of the `default` Adrawing, the `root` model, or the suspended part's Adrawings and models. If you want to file only the active model and not the complete assembly database, use the File > File Assembly with Options menu sequence. You can select the specific databases you want to file and leave the suspended Adrawings and model database inactive.

Please note: Do not use the File > File Assembly menu sequence.

When an Adrawing is selectively filed, you must also file the assembly structure in order to save the view states of the Adrawing.

Creating Assembly Engineering Drawings

CAMU supports the associative dimensioning of viewed geometry (geometry that does not physically reside in the active part) and hidden line removal (HLR). It also supports associative removal of hidden lines.

Do not detail the assembly within the Adrawing named `default`. Use `default` as a springboard to the assembly. Detail the assembly from within an appropriately named Adrawing.

Saving and Restoring Viewstates

CAMU enables you to save the viewstates of the components of an assembly either to the active part or the default Adrawing directory. It also allows you to restore the viewstate information from a saved file.

The viewstate file stores both the component instance name and the CAMU-id to enable EPD.Connect to also work with viewstates. An error message is displayed when the component instance name is not found.

For more information on how to save and restore viewstates, refer section Saving and Restoring Viewstates in the *Concurrent Assembly Mock-Up User Guide and Menu Reference.*

Activating a Model

When modifying geometry in the Parametric environment, switch to the Explicit environment regularly to update the `_gr` file. Make sure that you file the latest version of the part so that it is available to all the required users. To avoid regenerating the graphics data when using the VIEW COMPONENT command, the `_gr` file must be up-to-date.

Quitting a Command

Type `Control + E` if you encounter problems when switching environments or if the following message appears:

```
Command not valid at this time
```

Error Messages

The following table describes the various error messages that appear when using CAMU, as well as their origin and their solutions.

Error Message	Origin	Solution
`Error in XSORTGCF`	SORT DBA in CAMU	Does not cause a problem. Ignore the message.
Tessellation errors Segmentation violation (in CADDShade only)	CADDShade	Try increasing the size value. Often a problem with tcyls and plane surfaces. Try rotating the view. You may have to recreate the surface.
No graphics Result is error code 67	CAMU	Exit the assembly and reactivate it.
Assembly drawing is read-only `ERROR: The object specified is already locked by another user`	Locking of the CAMU tree after a crash even though the assembly has been reset	Exit CADDS, use the `slay` script, log out and log in, or restart.
`Unable to lock model DMP.123456`	Locking of the CAMU tree. Not a local model	Not a serious problem. Unlock the component, then lock it again using the Subassembly option. Do not use the Model option.
Result is error code 22	No ODB_DAEMON running	Run `/usr/apl/cadds/scripts/db_daemon` and restart CADDS.
There is a problem activating the assembly tree. You will have to reissue the ACTIVATE ASSEMBLY command.	CAMU	The `_db.LOCKW` and `_db.LOCK` files are left in the assembly because CAMU was improperly exited. Remove these temporary files.
`ERROR: Result is error code 9017`	VIEW COMPONENT command	You may be trying to view a non-CADDS 5 model into a parametric assembly. Convert the part into a CADDS 5 part and reactivate the assembly.
`ERROR: *** Hidden-line removal could not be completed ***` `Exit Code 26241 was generated.`	HIDE OBJECT command in CAMU	Your assembly may contain one or more bad solids. View smaller portions of your assembly and hide the viewed portions again. Continue viewing components until the error is reproduced and try to locate the bad solid. Once located, run the `validate_db` command on the problem part or parts. This may or may not fix the problem. If running validate_db does not fix the solid, you may need to recreate the solid.

Error Message	Origin	Solution
`ERROR: Result is error code 70003`	VIEW COMPONENT CINAME command	The part may have a zero length `_fd` file, which is either corrupt or is not filed yet. If the part is new, file it. There is no resolution for a corrupt zero length `_fd` file.
`ERROR: Error Code 70005`	CAMU	Quit your CADDS session and restart.
`Error Code 70006`	Transferring a model to a new directory and viewing it in CAMU	The error relates to the vp_links file. Ask your system administrator to delete the vp_links directory of the affected model and recreate it. Reactivate the model via the CAMU assembly.
`DR51 Error`	HIDE OBJECT command	If you are in the Parametric environment and are using CADDS 5 Release 12 or later, try to repair the part using `validate_db`. If validate_db reports any dex corruptions, run `validate_db -dex`. Perform the following steps after running `validate_db` and filing the part through `validate_db`. 1. Activate the part in the Parametric environment. If the dex table was deleted using `validate_db -dex`, the system regenerates the dex table. 2. Enter the Explicit environment. 3. Activate a drawing. 4. Change Extents Least Update. 5. File Part. The Hide Object command may work now. If using CADDS 5 Release 13 or later, perform the following steps: 1. Enter the Parametric environment. 2. Regen History. 3. Enter the Explicit environment. The system regenerates the dex table. 4. Check Dbase. 5. File Part. The Hide object command may work now.

Error Message	Origin	Solution
`Error Code 109`	CAMU	A component you are trying to view has exceeded the maximum limit for vp_link files. Close the assembly, remove all the vp_link files from the affected database, and reactivate the assembly.

Encountering Network Problems with CAMU

You can relate the system problems encountered when using CAMU to a poor Network File System (NFS) and in some cases, poor Network Information Service (NIS) setup. The NFS controls the remote disks in a UNIX network and is central to using CAMU.

Mounts

Follow the guidelines for mounts in *Installing CADDS 5* when using CAMU. For remote NFS mounts, the local mount for each user to a common assembly directory on a remote system must be the same on all systems.

If the assembly directory on the server is accessed from different mount points on the network, the absolute path name will be different, even if the assembly database is accessible through different mount points through NFS. As a result, each user activates different assemblies, even though they have NFS access to the same assembly. For example, User1 has mounted his parts directory from the server onto his client as:

```
#mount -F hsfs -o rw server:/usr/camu/parts
/usr/user1/parts(Client1)
```

User2 mounts the same directory on his machine as follows:

```
#mount -F hsfs -o rw server:/usr/camu/parts
/usr/user2/parts(Client2)
```

Even if these two clients share the same disk through NFS, this path is seen as two separate path names.

Please note: It is very important that all clients mount the paths with the same path name.

Permissions

CAMU supports only the `umask 2` permission. The user and the group have Read/Write access, while others have read-only access. If you change permissions, CAMU can fail for a number of reasons, and the following error message is displayed:

```
error code 82
```

Parts Data Area

You can encounter problems when filing the assembly due to lack of space on the server partition or disk. If the disk is 95% full, then there will be performance issues related to filing the assembly. The problem can be exacerbated by the number of users using the parts data area.

It is a best practice to keep the assembly database on the same machine that has the ODB_DAEMON process running. For many users, operating in single-user mode, this machine can be the local machine with the assembly filed locally.

However, for large installations where the parts are often located on a server or servers it may be necessary to locate the assembly database and the ODB_DAEMON process running on another server to minimize network and cpu loads. Many customers use high performance networks running at 100 MB/sec and special network interfaces such as ATM and FDDI.

NFS Setup

When setting up separate ODB_DAEMON and parts servers, increase the number of NFS daemons to 16 or 32, depending on the number of users.

The NFS daemons are usually set up at startup. On Solaris, they are in the `rc1.d` or `rc3.d` directory. For example, the file `S15nfs.server` contains an entry as follows:

```
If grep –s nfs /etc/dfs/sharetab >/dev/null ; then
/usr/lib/nfs/nfsd –a 4
/usr/lib/nfs/mountd
fi
```

Increase the `nfsd` number from 4 to 16. This number relates to the number of daemons running after startup. If extra daemons are required, the system must spawn them. Increasing this `nfsd` number improves performance, because the machine does not have to spawn the extra processes.

Check for `.nfs` files as follows:

```
#du –a |grep –i .nfs
```

The existence of `.nfsxxxx` files is a good indication of network performance problems. These files appear randomly only under peak load conditions, such as a number of users accessing or filing large assemblies simultaneously. Generally the network shows good performance characteristics, but it is a best practice to check over a reasonable period to identify the cause of the network slowing down.

NIS Setup

If the network is slow to activate assemblies, consider bringing the `/etc/group` file onto each client locally. In NIS, the groups file is served by the ypserver. Using the file locally improves performance, saves time, and helps the ODB_DAEMON process resolve ownership and group issues.

Examining the Network

The best methods to check for potential issues are provided by the UNIX operating system. Use commands such as `vmstat`, `iostat`, `top glance` (on HP UNIX) to check the performance of the ODB_DAEMON server. For example, the UNIX command `vmstat` reports the virtual memory statistics as follows.

```
#vmstat 5
```

```
procs      memory            page            disk          faults      cpu
r b w   swap  free  re  mf pi po fr de sr f0 s0 s6 --   in   sy   cs us sy id
0 0 0 666992  7824   0  22 326 0 430 0 57  0  0  0  0  649 2829  775  4  2 94
0 0 0 665688  7896   0  44 172 0 448 0 57  0  3  0  0  622 2131  657  3  3 94
0 0 0 666016  8536   0  34 32  6 166 0 22  0  1  0  0  502 1082  384  0  1 99
0 0 0 664680  7920   0   0  0  0  0  0  0  0  0  0  0  427  894  335  0  1 99
```

The listing shows a typical output of the `vmstat` command. The important feature is the CPU column. If the server is slowing down, the `us` (user) field will be approximately 70%, `sy` (system) will be approximately 30%, and `id` (idle) will be between –5 and +5%. In the `faults` column, `cs` will be approximately 4 digits.

You can also use the UNIX command `iostat` to retrieve the CPU information in the same manner. This command splits the partitions and shows the swap space.

Monitor the servers during peak load time, such as at the end of a shift, at the start of lunch hour, or at any other appropriate time. The following UNIX command detects collisions on the network. A 5% collision is considered high.

```
#netstat –i
```

Check if the CAMU server is being used by the other CADDS users. Depending on the number of users, it is recommended that you dedicate a server specifically for CAMU. Do not use PCs in the network of machines used for CAMU if possible.

Examining an Assembly

The best method to check the structure of an assembly is to use EPD.Connect and activate the assembly as a CAMU file or open it as a product structure. EPD.Connect will fail to activate the assembly if there is a significant problem. If you are successful, run a part list report and confirm that the parts do exist as expected.

The process is the same as for CADDS, in which a process is spawned from the ODB_DAEMON process and EPD.Connect creates a ODB_SERVER process for the assembly.

Chapter 6

Positioning and Constraints

This chapter discusses the best practices for the positioning and orientation of components in CAMU.

- Positioning and Orienting the Components
- Creating the Layout

Positioning and Orienting the Components

Use CAMU to parametrically locate, orient, and change components within the assembly. Create a common layout template, consisting of construction planes (Cplanes) and simple geometry, which is present in each part. The copies of the layout in each part are constrained, giving you the ability to locate, orient, and change the components of the assembly.

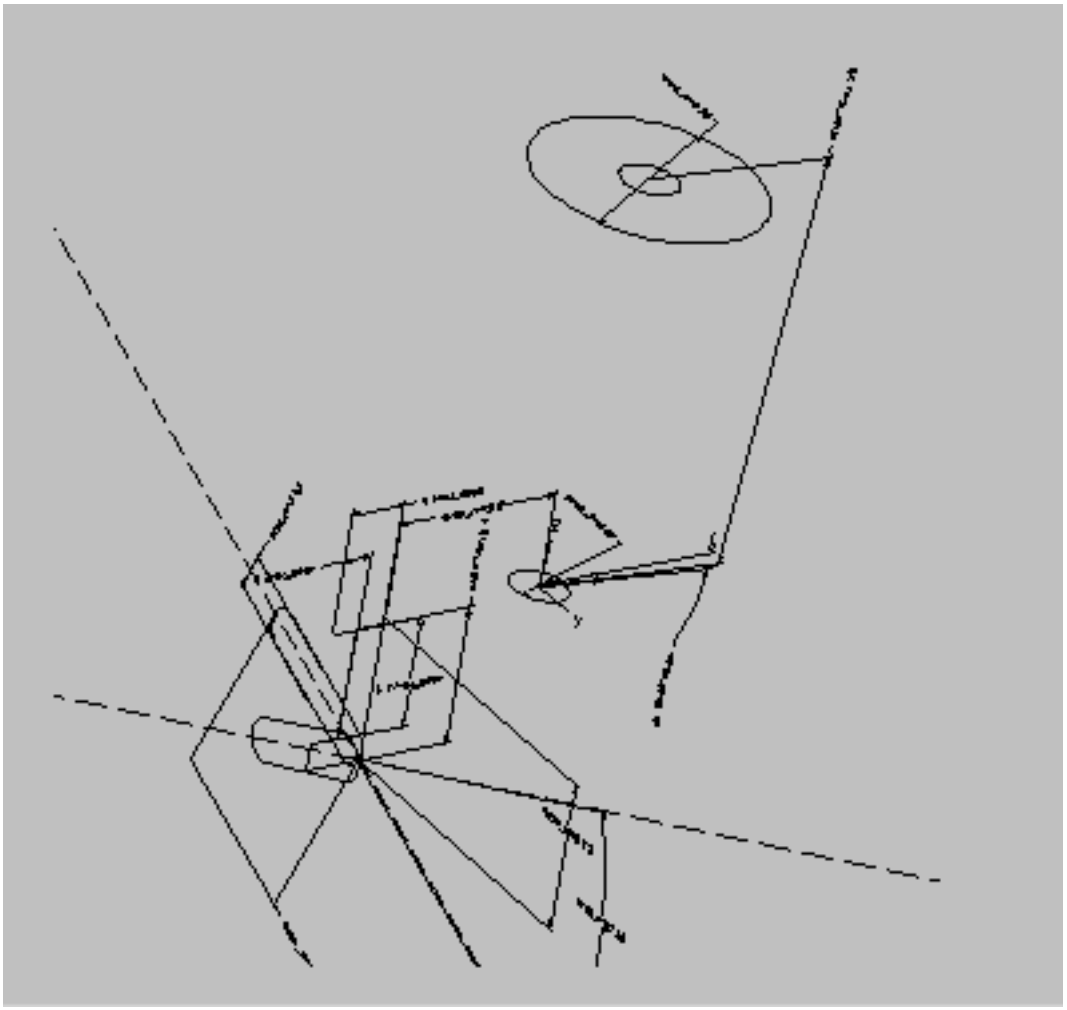
In CAMU there is no duplication of component data. This technique places the layout information in the first part of the history of every component in the assembly. This is a technique that is typically applied to small assemblies that are developed in single-user mode, and is helpful in solving motion and kinematics problems.

Creating the Layout

The master layout is very important. The geometry created for the master layout must represent the main functions of the assembly. For example, if you are laying out an automobile, you should create the geometry to locate and orient the wheel base, engine, gearbox, and other main components. The basic guidelines for creating a parametric assembly layout follow:

1. Activate an assembly.
2. Activate the root node. The layout is created on the root node.
3. Create the layout.
   - Define Cplanes, variation geometry, centerlines, and construction profiles that describe the function of the design from the default Top Cplane.
   - Define the layout with flexibility because each component uses these parametric relationships, as shown in Figure 6-1.

Figure 6-1 Wireframe Assembly Layout



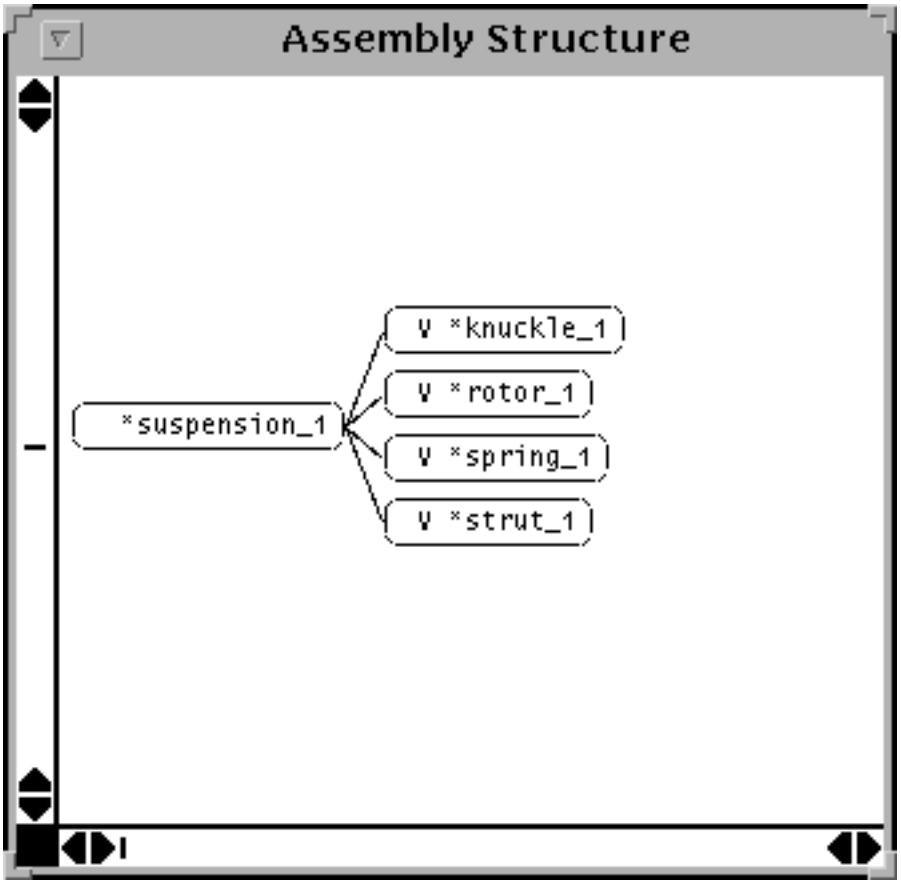
4. Make sure that the layout changes parametrically as required.
5. Export the variables that control the base relationships of the layout.
6. Activate an Adrawing and file the assembly.

Creating Components from the Layout

After the master layout has been created and has passed the test for potential design changes, the layout is propagated from the master layout to each of the components in the assembly.

- The COPY PART command is used in the Assembly environment to copy the layout to the individual components when they are created. Because the layout is only a collection of Cplanes and wireframe geometry, less duplicate data is created than when using other commands.
- The new parts are included in the assembly through the ADD COMPONENT command. The parts are viewed on the TOP Cplane at 0,0,0. See Figure 6-2.

Figure 6-2 Assembly Structure



At this point, each component has the layout copied to the beginning of its history. You can now change the component design, attaching the models to the input layout history.

Propagation of Layout Changes to the Entire Assembly

If the original product design changes, you must change the layout. Modifications to the layout are propagated through the assembly by using the following procedure:

1. Activate the root node layout.
2. Lock and import the affected components with the IMPORT COMPONENT command. (Remember to set your `CADDS_MAX_PART` environment variable in the `.caddsrc-local` file to a sufficient number. The number must be equal to the sum of the number of imported components, the root node, and the number of suspended Adrawings.)
3. Clear the values of the exported variables of the imported components by selecting the OFF option of the FIX COMPONENT command. To automate these last two steps, create a command file.
4. Write equations for each of the layout parameters and the imported component parameters. You may have to define many equations, even in a simple assembly. You can automate this step by creating an equation file outside of CADDS.
5. Change the parameter of the root node layout to reflect the changes required in the layout. This automatically solves the root node constraints and changes the imported parameters of the assembly.
6. Solve the imported components with the SOLVE COMPONENT command. This propagates only the new parameter values to the constraint system of the imported component. If other parameters of the component are related to the exported variable, they also change values according to the equation.
7. Regenerate the imported components with the REGENERATE COMPONENT command. You can see the geometry, positioning, and orientation changes.
8. File the assembly changes.

Appendix A

CADDS 5 Version-Specific Enhancements to CAMU

This appendix lists the enhancements in CAMU for the various versions of CADDS 5.

Table A-1 CADDS 5 Revision 6

Functionality Enhanced	Details
Check Database	Validates and checks the part database.
CAMU Programming Interface	Create the TOOLKIT executable file through CV-DORS using the `tkit_load` script.
Intersect Component	Section a set of assembly components with multiple planes.
Assembly Machining	Store the geometry and NC data in separate files with full associativity. Reference multiple parts with the NC tool path. Create model fixtures and setups.

Please note: CAMU for CADDS 5 Revision 7 had no enhancements.

Table A-2 CADDS 5 Revision 8

Functionality Enhanced	Details
Parametric Positioning	Position the components using the mate, align, orient, and fix positioning constraints. Modify the parts that are parametrically positioned. Create constraints between both the parametric and explicit parts. Please Note: The constraints are only accessible in the Parametric Assembly environment.
CAMU Status Window	Identify the user who has locked a component. Identify the status (that is, activated, suspended, or model does not exist) of the model associated with the component.
List Adrawing	Identify the name of the user who has write permission to a particular Adrawing.
Verify Object	Identify the model status.

Table A-2 CADDS 5 Revision 8

Functionality Enhanced	Details
Export Assembly	Export the active assembly to create a new assembly with a different name. Export a subassembly to create an instance of a new reference assembly.
Activate Model in Reference Assembly	Activate and modify the model associated with the reference assembly. Iterate modifications on several reference trees before validation decisions. Deactivate an active reference assembly using the QUIT REFERENCE command. The assembly becomes read-only.
List Assembly	List all the active or nonactive reference assemblies sourced within the active assembly.
Associative Dimensioning to Viewed Parts	Create the associated dimensions at the part or assembly level. (Only in the Explicit environment). Updates the dimensions automatically with the changes in the assembly.
Associative Dimensioning to HLR Entity	Preserve the dimensions defined to the HLR limbs and surrogates when you modify the model or the assembly component. Preserve the previously created dimensions by using the UPDATE HLRIMAGE command. Preserve the dimensions defined to the limbs and surrogates of the entities with hidden lines removed, using the HIDE OBJECT command.
Associative Dimensioning to Sections	Preserve the associative dimensions applied to a section view when you modify the sectionee or the sectioner. Preserve the previously created section dimensions using the REGENERATE SECTION command.
Associative Section to Crosshatches	Create associative crosshatching using the DEFINE SECTION command. Verify that the sections through multiple solids or assemblies have opposite crosshatch angles for adjacent solids. Preserve the changes made to crosshatch parameters using the REGENERATE SECTION command.

Table A-3 CADDS 5 Revision 8.1

Functionality Enhanced	Details
CALLBACK_SERVER Process	Starts the CALLBACK_SERVER process that is similar to the ODB_SERVER process. It includes the path of the active assembly and communicates with the CAMU status window.
CALLBACK_AD Process	Starts when the CAMU session is started. This process is associated with every Assembly Design (AD) client session. The ODB_SERVER process communicates the session information to the CALLBACK_SERVER process and later broadcasts this information to all the client CALLBACK_AD processes. Each CALLBACK_AD process communicates this information to its associated AD process. The AD process then updates its CAMU status window with this information.
Reference Assembly Locking	Locks the reference assemblies to provide good control over maintaining assembly revision. You can modify the reference assembly.
Part Mirroring	Creates a mirror copy of the source component. The mirrored parts associated with the created mirror copy of the component have an associative relationship with the parts of the original component.

Table A-4 CADDS 5 Revision 9.x

Functionality Enhanced	Details
Tool for Checking Corrupt Component Node	The ckCAD utility allows you to check for the corrupt model part associated with a component. An F character is displayed on the component node if the component is corrupt.
Identifying Root Node Lock of a Reference Assembly	The root node of a reference assembly appears with an ** indicator to identify the lock status of the root node of the reference assembly in the assembly structure.
Graphics-only Mirror Copying	Creates the graphics-only mirror copies of the components, subassemblies, and the reference assemblies about a specified plane. Here the orientations of the mirrored component are changed so that it appears mirrored, though the associated part is the same as that of the source component.

Table A-5 CADDS 5 Release 10

Functionality Enhanced	Details
Associative Topology Bus (ATB)	The ATB allows you to share the part and assembly data across CADDS 5 and Pro/ENGINEER. You can access Pro/ENGINEER parts or assemblies in the Assembly environment. You can also share part geometry, topology, and assembly data in CADDS 5 and Pro/ENGINEER in an associative manner.
Verifying Object	Obtain details regarding the mirroring plane and mirror status. Retrieve information about the source part of a model associated with a component.
Activating Model with Inserted Ppart	When you activate a part that contains a part inserted in it using the INSERT PPART command, if the part image is not synchronized with its source part, a popup menu appears allowing you to update the part.
Disabling the CAMU Status Window	Use the environment variable USE_CAMU_STATUS_WINDOW to set the CAMU Status window. Setting the USE_CAMU_STATUS_WINDOW variable to yes enables you to use the CAMU status window. For better performance during locking and unlocking of components, set this variable to no.

Table A-6 CADDS 5 Release 11

Functionality Enhanced	Details
COPY VIEW Command	Use the COPY VIEW command with the Appearance option for assembly drawings.
WIndows NT Support for CAMU	Activate assemblies irrespective of whether they are on a UNIX or Windows NT platform. You can map different hard disks on the same or different Windows NT machines as different drives on the same Windows NT machine. This enables you as an Windows NT user to access remote disks. You can also map drives of UNIX machines to Windows NT machines. Windows NT users can activate assemblies on either a UNIX or Windows NT server, using a UNIX ODB_SERVER. Windows NT ODB_SERVER for mixed UNIX and Windows NT clients is not supported
Control Opening Level of Nested Reference Assemblies	When activating an existing assembly, you can read and display the specified level of nested reference assemblies in the assembly tree window. You can control the level of reference assemblies to be read during assembly activation. This improves the performance of assembly activation time. The REFRESH ASSEMBLY command is limited to opened assemblies. Ability to further modify open and closed reference assemblies Equivalent functionality in EPD.Connect — CAMU mode.
Enhanced Change Component Command	Change the model associated with the component in an assembly. Maintain associativity with any drawing entity after changing the model name.

Table A-6 CADDS 5 Release 11

Functionality Enhanced	Details
CAMU Performance Improvement	Improvement in the performance of the REFRESH ASSEMBLY and FILE ASSEMBLY commands on a large assembly with large numbers of viewed components. During the node switch from one model to another, or a context switch from the Explicit environment to the Parametric environment, and vice versa, the annoying flicker is not visible. This indirectly improves the performance of CAMU to some extent. In the Parametric environment, the performance of viewing the second instance of an already viewed component has improved.

Table A-7 CADDS 5 Release 12

Functionality Enhanced	Details
Exporting Assembly with Associated Adrawings and View States	Choose to transfer the Adrawings and the view states associated with the main assembly to the exported assembly when exporting an assembly. Export an assembly with its Adrawings. The filed view states are transferred to the exported assembly.
Plotting Assembly Structure	Plot the Assembly Structure to a plotter device or save the output to a file using the PLOT TREE and PLOT ASSYTREE commands. The output is in vector format.
Filling Model in History Replay Mode	Save the intermediate state of history replay when you are in history replay mode of an active model and activate another Adrawing or model.
Performance Improvement in View Part	In Assembly Explicit environment, when switching nodes from an Adrawing to a model or between models, only those part instances activated and suspended are viewed off and on. All parts are not viewed off and on. Parts that are modified and filed before switching nodes are also viewed off and on.