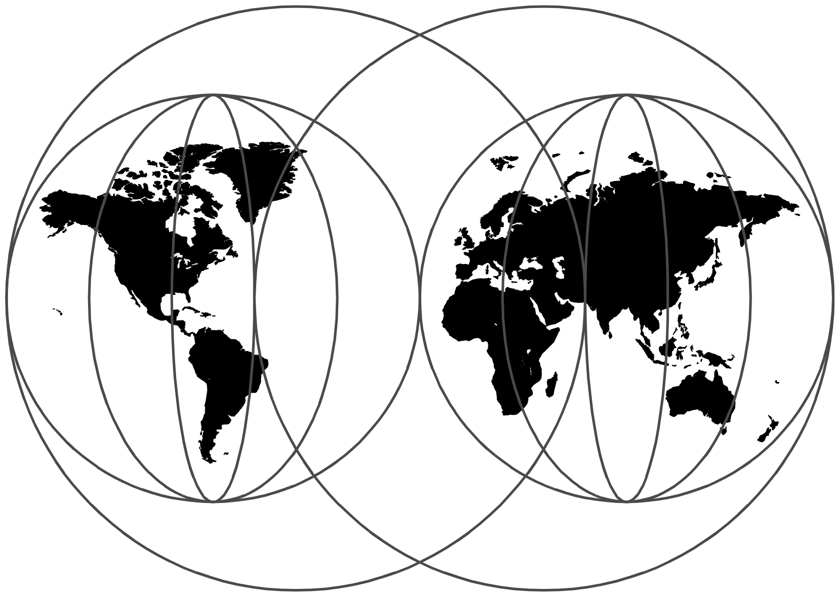
VisualAge TeamConnection Enterprise Server From CMVC to TeamConnection Version 3

Jean Gedeon, Lee Perlov, Kevin Postreich, Angel Rivera



International Technical Support Organization

http://www.redbooks.ibm.com

SG24-2226-00

SG24-2226-00

International Technical Support Organization

VisualAge TeamConnection Enterprise Server From CMVC to TeamConnection Version 3

December 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix H, “Special Notices” on page 303.

First Edition (December 1998)

This edition applies to VisualAge TeamConnection Enterprise Server Version 3, Release Number 01, Program Number 04L3809 (a)-2954 for use with the following Operating Systems:

- VisualAge TeamConnection client (OS/2, Windows 95, Windows NT, HP-UX, AIX, Sun Solaris)
- VisualAge TeamConnection server (OS/2, Windows NT, HP-UX, AIX, Sun Solaris)
- VisualAge TeamConnection build server (OS/2, Windows NT, HP-UX, AIX, Sun Solaris, OS/390, and Windows 95)

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. QXXE Building 80-E2
650 Harry Road
San Jose, California 95120-6099

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

This redbook is intended as a guide through the process of migration from either CMVC V2.3 to VisualAge TeamConnection, or from previous releases of TeamConnection to the current level of the product

The objective is to provide the reader with the information needed to permit making the right choice among multiple possibilities, in order to best meet business requirements.

Part 1 contains a general introduction to VisualAge TeamConnection and its concepts. It will be specially useful for the newcomers to TeamConnection. Chapter 3 puts emphasis on network needs and give some hints to those unfamiliar with TCP/IP concepts.

Part 2 covers the subtleties of UDB installation. Those who are using OS/2 or Windows platforms and have installed their environment using the TeamConnection installation wizards can skip it. Others may find some benefit to reading the chapters that pertain to their specific environment.

Part 3 covers migration to VisualAge TeamConnection Enterprise Server version 3.1. It is intended as a reference document to help the reader succeed in this task. This will be especially useful for Team Leaders, Project Managers, and anyone else involved in the development and distribution or maintenance of software applications.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization San Jose Center.

Jean Gedeon is a Software Program Manager at the International Technical Support Organization, San Jose Center.He writes extensively and teaches IBM classes worldwide on all areas of software change and configuration management.

Before joining ITSO one year ago, Jean Gedeon worked in IBM Europe Middle East & Africa (EMEA) Application Development/Object Technology Center of Competence as a software program manager

Lee Perlov Mr. Perlov is an Advisory Software Engineer in the VisualAge TeamConnection/CMVC development team. He started working for IBM in 1985 in Gaithersburg, Md, working in the Federal Systems Division on various projects for the United States intelligence community. He then moved to RTP to work on library development and support.

Mr. Perlov received a B.S. degree in Accounting from the University of Florida in 1983. He also completed two years of graduate work in the Department of Computer Science at the University of Florida.

Mr. Perlov is currently a member of the VisualAge TeamConnection Services team, as well as family, system and web administrator for the services TeamConnection family.

Kevin Postreich Mr. Postreich is a Software Engineer with the VisualAge TeamConnection development group. He joined IBM in 1980 as an electronic engineer in Charlotte, North Carolina. He relocated to RTP as an MVS systems programmer.

Mr. Postreich is currently a member of the TeamConnection development/test team, and a member of the first defense team for customer support.

Angel Rivera Mr. Rivera is an Advisory Software Engineer with the VisualAge TeamConnection/CMVC development team. He joined IBM in 1989 and since then has worked in the development and support of library systems.

Mr. Rivera has an M.S. in Electrical Engineering from the University of Texas at Austin, and a B.S. in Electronic Systems Engineering from the Instituto Tecnologico y de Estudios Superiores de Monterrey, Mexico.

Mr. Rivera is currently the team lead for the development of CMVC 2.3 and provides service support for customers of VisualAge TeamConnection.

Thanks to the following people for their invaluable contributions to this project:

Mr. Mark Dunn and Mr. Tim Orlovski, Advisory Software Engineers with the VisualAge TeamConnection development group, Mr. Suchy, Software Engineer and documentation team lead for the TeamConnection product, and last but not least, Mr. Ruby, Senior Software Engineer and the lead architect of TeamConnection

Many thanks also to Laymond Pon, ITSO San Jose Center AD Manager and Mary Vinopal, Enterprise Service Development Manager, Raleigh laboratory for getting this project started. Special thanks to everyone at the ITSO San Jose Center and Raleigh TeamConnection development laboratory, in particular Elsa Barron and Mary Comianos, for their continuous support.

We are grateful to Shirley Hentzell for her unfailing editorial review. Thanks also to Eric Valade, from EMEA AD/OT CoC, Jim Palistrant from the TeamConnection Worldwide Marketing support for all their continuous support and technical hints throughout the process of writing this book.

Jean Gedeon
International Technical Support Organization, San Jose Center

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in "ITSO Redbook Evaluation" on page 331 to the fax number shown on the form.
- Use the electronic evaluation form found on the Redbooks Web sites:

  For Internet users`http://www.redbooks.ibm.com`
  For IBM Intranet users`http://w3.itso.ibm.com`
- Send us a note at the following address:

  `redbook@us.ibm.com`

Part 1. TeamConnection Bells and Whistles

Chapter 1. Today's Application Development Challenges

1.1 What the Development Challenges Are

Application development teams today face tremendous challenges. While both applications and application development environments are becoming more and more complex, development teams are challenged to continuously increase code quality through adherence to process standards and audit trail management. The application development tools available today to help with these issues often come up short, addressing only specific problems and not providing sufficient tool integration to deliver fully integrated software development environments.

Applications under development are made more complex by the need to support advanced graphical user interfaces (GUI), object-oriented and client/server technologies, and heterogeneous distributed computing.

Client/server applications are moving past decision support into the third generation of mission-critical, high-volume transaction processing, which brings with it additional complexity in application design.

The team's development environment is also becoming more complex in an effort to meet these challenges. Members of the development team are distributed on a local area network (LAN), yet they have the same needs for coordinated team programming support that is available in the centralized host environment. Object technology, with an increasing focus on support for reuse, and tool integration at the data level are requirements that add to the complexity of the development environment.

Successful software development organizations are measured by Software Engineering Institute (SEI) maturity levels and by compliance with ISO 9000 standards. These maturity levels and standards present new challenges to development teams that are being asked to deliver software products to market at faster and faster rates.

1.2 How TeamConnection Can Help

TeamConnection helps address application development challenges by:

- Enabling you to organize your components for reuse
- Enabling you to manage the changes to your software more efficiently

- Helping you to rebuild your applications more efficiently after they have been modified
- Improving team communication and deployment through e-mail notification on actions required, application changes, and project status
- Packaging your applications for delivery so you can get them to your customers more quickly and with greater reliability
- Providing a development model that can help raise your SEI maturity level and improve your application quality through reliable, efficient, and repeatable processes
- Providing extensive problem tracking and change control for both fine-grained development objects (for example, data elements) and coarse-grained application development objects (files)
- Providing extensive reporting that can be used for tasks such as impact analysis, project status, project management, quality analysis, and TeamConnection: workload balancing
- Providing a secure repository for your software assets and the information about them, with role-based access
- Providing an application development information model along with repository services that provide the basis for tool integration
- Supporting the nondisruptive evolution of the information model. End users can maintain their investment in current tools and their existing skill base working with those tools and still allow for introduction of new tools into the development environment.
- Supporting data exchange with other platforms and modeling tools, including migration of legacy data, through IBM Exchange for OS/2, an optional feature expected to be available in a later release
- Providing backup and recovery facilities

1.3 What TeamConnection Does

TeamConnection integrates software configuration management services and object-oriented repository services on a semantic model for tool integration to support application development in client/server team programming environment. Applications can be targeted for any platform and can be developed as stand-alone, client/server, or distributed applications.

TeamConnection delivers the functions your application development teams need to manage development data, application versions, and application configurations. The integrated problem tracking and change control system ensures that, while your application developers become more productive, your project leaders can still effectively manage the development process and track progress.

TeamConnection automates and streamlines your application build process and integrates it with the version control and change control processes. The build process is tied to release management and extended to provide a framework for delivery of software.

An open, extensible information model provides the vehicle for data sharing between a set of integrated tools using TeamConnection. This object-oriented information model enables an extensible architecture, thus ensuring continued support for new versions of existing tools, as well as new tools that are brought into the active repository environment.

TeamConnection supports application development customers who employ model-driven development to:

- Support application development by teams charged with the responsibility of developing complex applications quickly and maintaining those applications as responsively modeled objects.
- Facilitate data sharing, where all members of the development team, whatever their roles, have a common understanding of an application's data (that is, semantic consistency of data is enforced).
- Exploit a LAN topology for the application development environment.
- Use distributed application development data, which further exploits a network topology in the management of application development resources.

TeamConnection is built on the DB2 universal database (UDB), IBM’s object-oriented database, which allows TeamConnection's repository to store fine-grained application-development data. Its software configuration management services work equally well with fine-grained modeled objects and conventional coarse-grained objects (files). The integration of these higher-level services with more traditional repository services such as constraint checking, version management, concurrent and distributed access, and data exchange, gives TeamConnection unique strengths in the industry.

TeamConnection provides the following services:

- Configuration management
- Release management
- Version control

- Integrated and repeatable builds
- Packaging and distribution support
- Problem tracking and change control
- Reporting
- Backup and recovery
- Object-oriented repository
- Information model
- Data constraints
- Information model schema evolution

TeamConnection marries repository technology with software configuration management services in a team environment. Thus you can store all of your development data in one place regardless of form and provide a consistent set of services and processes for your entire development team, not just programmers.

TeamConnection helps you manage your development process by organizing your data, controlling changes to it, and providing reporting capability.

TeamConnection increases your team's productivity by automating common development tasks, notifying team members when action is required, and providing a vehicle for tools to share data.

TeamConnection helps improve the quality of your products by providing repeatable, reliable processes that ensure proper authorization for changes.

TeamConnection provides an open, customizable, and scalable environment that can evolve and grow with your development team.

Chapter 2. Meeting VisualAge TeamConnection

This book introduces you to the world of VisualAge TeamConnection. TeamConnection objects, as well as their relationships, are smoothly and simply presented without formal definitions, while remaining firmly linked to the real development world of software applications.

To reach this objective, we first discuss the challenge of managing software application at the turn of our century, then propose a simple software development model synthesized from the numerous experiences of IBM customers. Then we will plug this model into TeamConnection and play with it.

2.1 Managing Software Change: The Power of TeamConnection

If you expect to upgrade a legacy application to deal with Year-2000 implications, or to develop new Java code to provide Internet access to your enterprise database, keep reading. The next few ounces of paper could be your life preserver.

Look at Figure 1 on page 8, and imagine driving a car with elliptical wheels of different sizes, because someone forgot to propagate a code change at the right time. Thanks to VisualAge TeamConnection, such bumpy rides are easier to prevent and you won't have to fix them afterward.

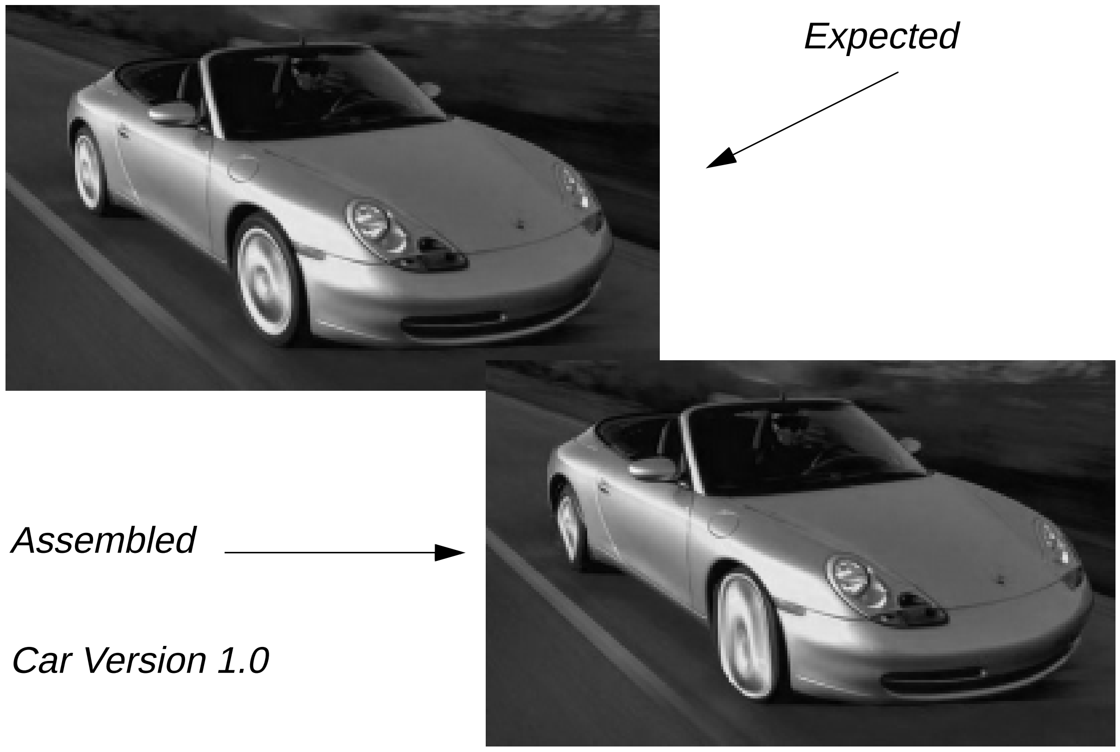


Figure 1. What Goes Wrong when Coherence Is Lost

2.1.1 Objectives

Customers fund you in order to get a defect-free software application that matches their needs. They have no interest in how many versions of the thousand files are built in it. From the point of view of software, requirements and defects are both changes, and one change can generate up to a thousand file versions. This is why this section is entitled “Managing Software Change” and not “Managing Software Configuration.”

Figure 2 on page 9 illustrates two techniques available to produce a car. You can either:

- Select your car parts from a parts catalog used to build all possible models, or
- Order a predefined model (or version) from the catalog of car models.

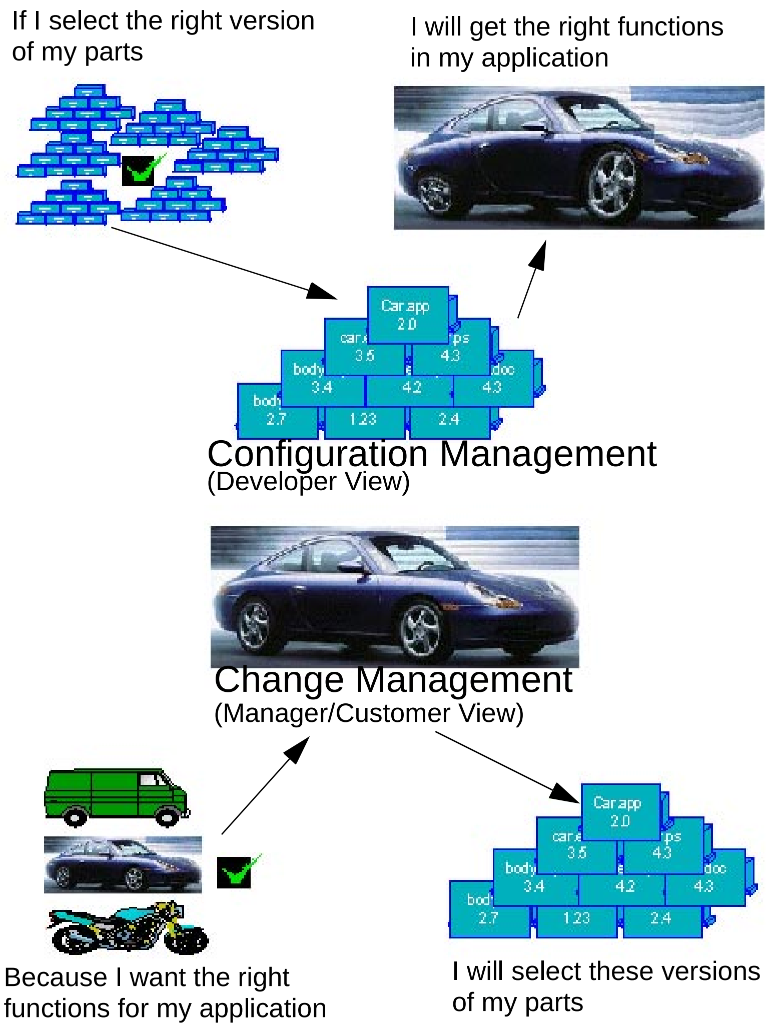

If I select the right version of my parts
I will get the right functions in my application
Car.app 2.0
car.. 3.5
..ps 4.3
body.. 3.4
4.2
..doc 4.3
bod.. 2.7
1.23
2.4
Configuration Management
(Developer View)
Change Management
(Manager/Customer View)
Because I want the right functions for my application
I will select these versions of my parts

Figure 2. Change Management versus Configuration Management

By choosing Option 2, you are certain to get the expected model. In other words, if you try to configure an application, you may not get all the correct changes, but if you manage the changes you will.

In the software development field, developers tend to focus on file versions, while project managers and customers necessarily focus on changes. We can call software configuration management a bottom-up approach, while change management is top-down, since an application version can always be associated with a list of changes.

VisualAge TeamConnection allows you to manage your application changes. Should you want a specific car model, provide the list of required options, and TeamConnection gives you exactly the car you expect. In other words, TeamConnection manages both version and configuration control for you.

At this stage, we introduce some concepts of TeamConnection language:

- Functions, evolution, and options are *features* in TeamConnection.
- Bugs and defects are *defects* in TeamConnection.
- Data, source files, and documentation are TeamConnection *parts*
- Products, applications, versions of applications, and composite objects—any groups of related data—are TeamConnection *releases*. They are also called *configurations.* A configuration could be represented by a group of parts.
- The relationship between parts and a specific build process is known as a TeamConnection *build tree* (in fact a graph, if parts are common to more than one configuration).
- A lists of changes (defect or feature) for a given application is a TeamConnection *driver*.

2.2 Types of Problems Encountered

First of all, a software application can change over time. The best known example is the Year-2000 problem. For example, it can require you to:

- Upgrade a legacy application to handle the four-digit dates required after 1999 for finance and accounting, while COBOL 74 is based on two-digit dates.
- Integrate a previously developed code into a Year-2000-compliant application.
- Reorganize a legacy code to take advantage of today's programming techniques.

If you take up the challenge, you are now likely to find yourself struggling with three application versions living in parallel:

1. The legacy application, initial version (Version 1.0). This version is used to handle previously generated and archived data.
2. The new application, which is Year-2000 compliant, but does not know how to handle the new European Currency Unit (ECU) due by year-end 1999, an omission that will affect your accounting data in fifteen countries. This version (Version 2.0) is installed in most of your international sites.

3. The future application, which is both Year-2000 and ECU-compliant, but is still under development (the future Version 3.0).

Moreover, each of these versions can have built-in defects. Figure 3 on page 11 shows the impacts of defects found in the first two versions. When customer satisfaction has top priority, you first fix the bugs in the defective version where they were found. You then need to propagate the fix to all other impacted versions. Unfortunately, the same files are likely to have been modified in each version to fix different defects. This means that you must merge the changes to ensure the consistency of your entire application. Handling all that looks quite complex.

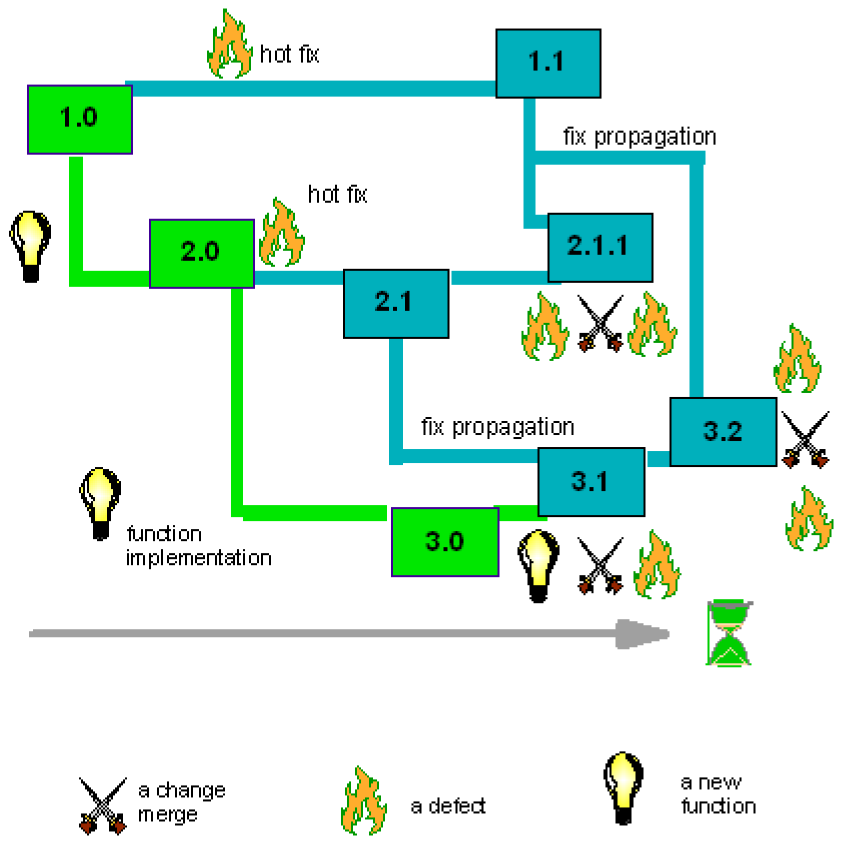


Figure 3. Version Control Is A Fact

These are the problems you might encounter, and their implications. We now proceed to a more detailed analysis.

2.3 Place Your Application Under Control

A single instance (or version) of an application was once considered enough. From now on, however, your application is likely to change often, maybe in the next few months. For example, customers may demand that it run on various operating systems, be accessible from a Web browser, and comply with National Language Support requirements. To adapt your application to new technologies, whether to improve or to maintain it, you need first to know it throughout, then know how to modify it, and finally, find out what you need to change.

2.4 Studying the Anatomy of Your Application

It is useful to dissect your application in order to find out its design and dependencies and understand its build process.

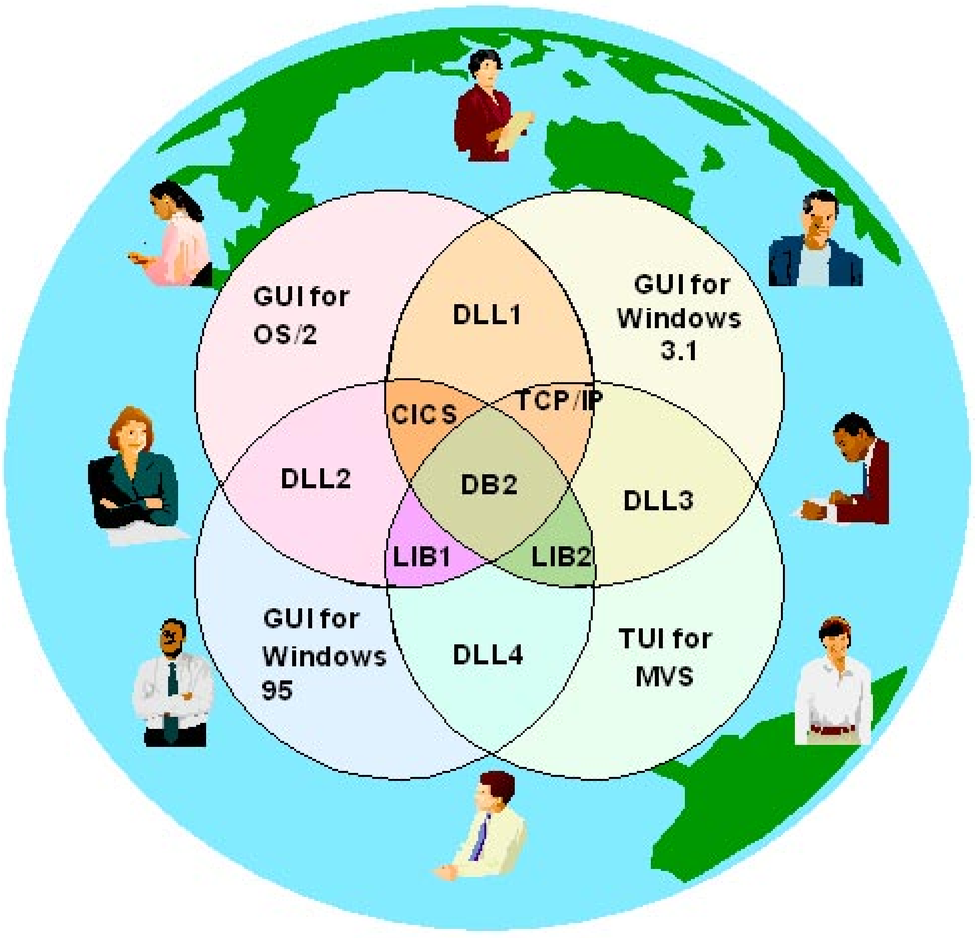


Figure 4. Anatomy of a Legacy Application

Most legacy applications are so patched and altered they seem like labyrinths. Maintaining code in such a situation can be torture, as those who developed it initially and made the early changes are likely to have retired or moved on. Figure 4 on page 13 shows an application that offers realistic challenges. If you want to control the development process, you need to find answers to many questions. To get the right answers, it is important to ask the right questions. Here is a sample of questions that could be used to assess your application:

- Can my application be split into independent subsystems? If the answer is yes, what are the interfaces between the subsystems? (If the answer is no, expect a highly difficult job.)
- Does my application have prerequisites or corequisites? What operating systems or external applications does it require? Does it need to access a database through a transaction manager over a network? What protocols are to be used?
- What are the data cross-dependencies, and how can I check them? (For example, how do I figure out the class hierarchy for an application that was developed with an object-oriented language? If a 3GL is used, what are the cross-references?)
- Does my application share data with other applications or various versions of the same application? And when? Is it:
    - At run time (execution), such as dynamic libraries, distributed objects?
    - At build time (compilation, link edition), such as include files, classes, static libraries, database connection?
    - At creation time (automatic generation), such as document templates, existing frameworks?
- Who is responsible for a given group of files, or a subsystem of the application? Who is authorized to do what? Are different groups, departments, or companies involved in the development of the application? How is the data access control handled?
- Can I split the build process of my application into small steps? If I use a make program, can I explode the make file into smaller ones?
- Are there different build processes depending on the development life cycle?
    - Coding phase: Debug options are used for compilation and link edition. Optimization is set aside. If the application needs to connect to a database, a fake or a partial one is usually used.
    - Test phase: To run faster, we optimize the build process. If the application needs to connect to a database, a test database is used to test the performances. The source files are recompiled with a cross-compiler, linked together, and then mapped to the hardware system.
    - Production: We no longer need to debug information. The real database is used.
- What is the pace of changes on the identified items?

- What are the deliverable? Which files, or objects have to be packaged together and released outside of the development environment? (For example, we can elect that only the executable, the dynamic libraries, and the documents are deliverable, and never release the source code files.) Do those objects need to be transformed—for example, renamed, compressed, or encrypted?
- How do I distribute the application image to the thousand target workstations in the worldwide network?

These may look like good-enough questions, but the answers give you only a static view of the problem. Also, even if you have answered all of these questions, how long would it take to formalize the result?

2.5 Changing an Application in Your Development Environment

If you answered the questions in the previous section, your application is no longer a mystery to you. You know its architecture by heart, and you have found out how to build, package, distribute, and use it. Now you face the problem of using what you've learned.

Even if the legacy application works fine, it will nevertheless need to be changed to handle new requirements or to fix a bug. Here is an example of the change process to upgrade your application (or to fix a bug):

1. **Make a change request.** Take a request form with the right predefined skeleton, then fill it in with useful information that could facilitate the change implementation. The form could include, for instance:
   - An identifier
   - An abstract description
   - A long text description
   - Some attributes to categorize the change, which can also be used to calculate quality metrics, or get some statistics.
   - An ownership field, showing the name of the person owning the request
2. **Assign the change.** Assign the responsibility of implementing the change to the group, department, or company that must control access to the data.
3. **Analyze the impact of the change.** Based on the description, try to estimate the cost, workload, and impact (what other applications are affected? what new lines of code, new classes, or new database tables

are needed?). Then accept or reject the task of implementing the change. Changes that are too expensive or are irrelevant are unlikely to be acceptable.

4. **Create a workspace to make the change.** A famous motto says "Never change something that works fine," so we recommend that you not touch the existing running version of an application. Instead, create a dedicated environment to generate the requested change. For example, you could create a new directory on a specific machine, and copy from the reference domain into the workspace all necessary data to implement the change.
5. **Make the change.** Once a workspace has been set up and contains the right data impacted by the change, it is time to edit the files. Next, scan them to check the dependencies. Finally, build the application (or the parts of it necessary for the implementation), using the correct build tools, such as a compiler, or a link (the well-known Edit/Compile/Debug phase).
6. **Build and validate the change in the operational (or target) environment.** Even if the application runs fine on your own machine, it may not work properly on the end-user's machine. Rebuild your application on a pilot end-user platform and run a complete series of tests before claiming you have completed the change.
7. **Update the reference domain or promote the change.** Once the application in the development workspace has passed all the tests successfully, and the version is ready to become the next official version of your application, promote the changes from the workspace up to the reference domain.
8. **Package and distribute the change.** Your customer is waiting for the new functions of this application. Only the data such as executable, dynamic link libraries, or end-user documentation required to install and run the application still has to be gathered from the reference area. These are deliverable. (Development data such as source files, object files, test cases, or development documents are not deliverable.) To save tape or diskette space, or to decrease transfer time over the network, deliverable are often converted. For example, they can be compressed, encrypted, jarred (Java archive), zipped (compressed) or renamed. A package is a collection of deliverable after conversion. Once built, send the package to the target machine, using any distribution tool you have.
9. **Monitor the quality of the change process and measure customer satisfaction through time.** Cross-check to see if you have correctly estimated the impact of the change, and whether it matches customer expectations.

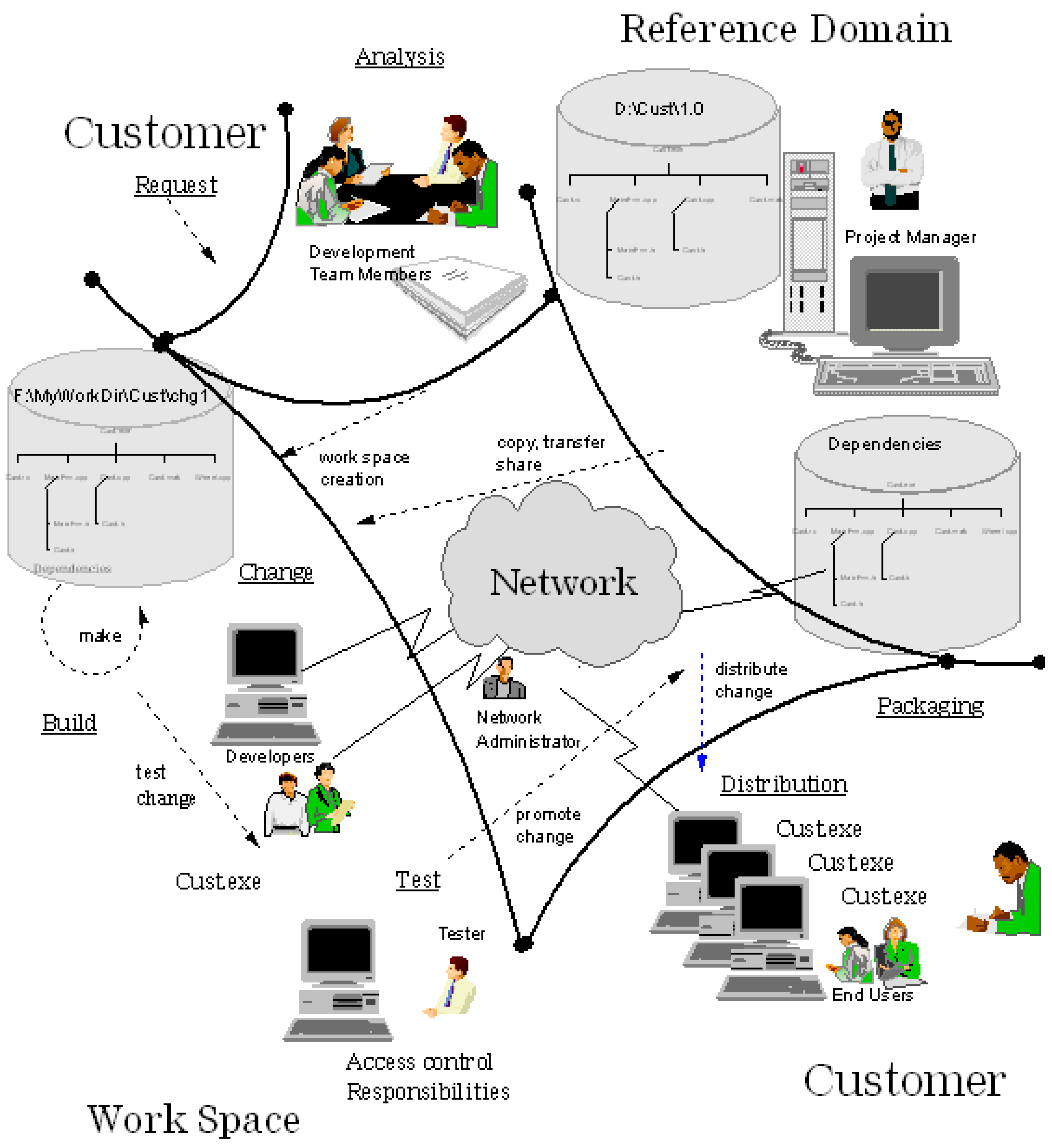


Figure 5. The Transparency of TeamConnection

Now examine Figure 5 on page 17 from top left to bottom right. There you can see the entire life cycle of a change, starting from the customer request and ending with the shipment of the change. Note that the customer initializes and ends the cycle. Moreover, the changes live in a distributed development environment.

2.6 Insert TeamConnection in Your Development Environment

The two previous phases show what you do when you haven't migrated your application into TeamConnection. Choosing the language you are going to use for programming comes at the end of the process, when you specify the functions of an application. The programming language can create constraints, but it can also facilitate development by allowing you to implement additional functions. It is important to follow standardized procedures to enjoy the full advantages of TeamConnection.

The change process explained so far could be either simpler or more complex, but you will run through the described steps for any type of application, even though you may not be aware of it. Can TeamConnection handle such a change process? Let us map your world to the TeamConnection world:

- Groups, departments, companies, or other management entities are TeamConnection *components*.
- TeamConnection changes are known as *defects* or *features*.
- Workspaces are TeamConnection *workareas*. A workarea is created in reference to a feature, or to a defect if problem tracking is turned on.
- Reference domains are TeamConnection *releases*. You can also have reference sub-domains that include a collection of changes made against a reference domain. These are known as TeamConnection *drivers.*
- The copy operation from a reference domain to a workspace is a TeamConnection *workarea create*, or *workarea refresh*. The actions *release link*, or *part link* could also be used to import parts from another reference domain, or from more than one reference domain.
- Data changes are handled as follows:
    - For source parts, such as C code or header files, TeamConnection uses *part check-in/check-out* actions (this is not an exhaustive list of all possible actions),
    - For generated parts such as package file, object file, executable, dynamic or static libraries, TeamConnection uses *part build* action.
    - For all types of parts, TeamConnection uses *part rename*, *part create*, and *part delete* actions.

- Build tools are TeamConnection *builder*s. A TeamConnection builder must be associated with each generated part. A builder is a routine designed to generate output files from a collection of input files. A builder is executed on a TeamConnection build server. If more than one build server exists, the build operation is distributed and run in parallel. You can see a builder as an alias of the command issued to produce ready-to-use parts from input parts. If you want to produce an object file from a C-source file by using a C++ compiler, when the target processor is a Pentium, you want to use the header files from the d:\custExe\include directory, and you are in unit test phase, then you could create a builder named *CPPCompiler* and link it to the command:

  ```
  cl /c /G6 /Id:\custbody\include /DDEBUG=YES
  ```

- Dependency scanners are TeamConnection *parsers*. A TeamConnection parser can be associated with a nongenerated part. It points to a program that scans each line of the source, searches for a specific character string pattern, and returns a list of the contents of whatever lines are found while excluding that pattern. For example, a header file scanner in C scans the line for #include and returns the list of all the header file names.
- Packages are particular build processes that use the TeamConnection Gather program associated with TeamConnection builders. The Gather tool automates the movement of software and data from one directory to another on the same machine, to prepare a package for electronic distribution. It can transform, copy, or erase files; it can also create or delete directories.
- Distributions are also a particular type of build process using specific TeamConnection utilities associated to a TeamConnection builder. Those utilities include bridges with Netview Distribution Manager, and Tivoli/Courier. Both products are parts of TME 10 Software Distribution. The NVBridge tool supports automated distribution between a single NetView DM/2 CC server and its LAN-connected CC clients. It also supports remote distribution to APPC-connected NetView DM/2 servers and mainstream servers. Tivoli/Courier supports the synchronized distribution of client/server software packages across the enterprise.
- For a quality measurement of the change process, the TeamConnection *report* command allows you to know the status of your project at any time, and you can also use it to build statistics to measure customer satisfaction.

Provided requirements are well defined, it is not difficult to implement a solution: just ask the right questions and you will get the right answers. In Figure 6 on page 21, you can visualize your development environment with something more called *VisualAge TeamConnection*. The main differences are:

- Any workspace is enclosed in the TeamConnection family server, which is the guarantor for the consistency of the changes made against your application. Data is moved out of TeamConnection only to be edited, built, packaged, and distributed.
- A new function, but not necessarily a new person, appears in the development team: the TeamConnection administrator
- Parallel and distributed builds are both available for applications.

Once you have done the necessary work to load your legacy application and puts it under TeamConnection control, here are some jobs you can delegate to TeamConnection:

Make sure that users are authorized to perform the actions they are requesting.

- Each time you check a part back in, TeamConnection looks to see if the modified part belongs to other versions of the application. If it does, you are asked to specify which other versions you also want to update. (You check-in a part when you copy it into the TeamConnection family and check-out the part when you extract a copy from the family to work with.) In such a case, a flag is set in place by TeamConnection in order to indicate or forbid further changes till the part is checked in again by the user who checked the part out (in serial development). In parallel development the flag is used to handle the conflicts created by the fact that many users are able to check the part out and will probably never check-in parts in the same order as they checked them out.
- Find out whether or not the requested actions are synchronized with the current status of the change. For example, you cannot create a workarea (a workspace) in reference to a defect object if the defect has not yet been accepted by the defect owner; and you cannot integrate a workarea if the build process has not ended successfully.

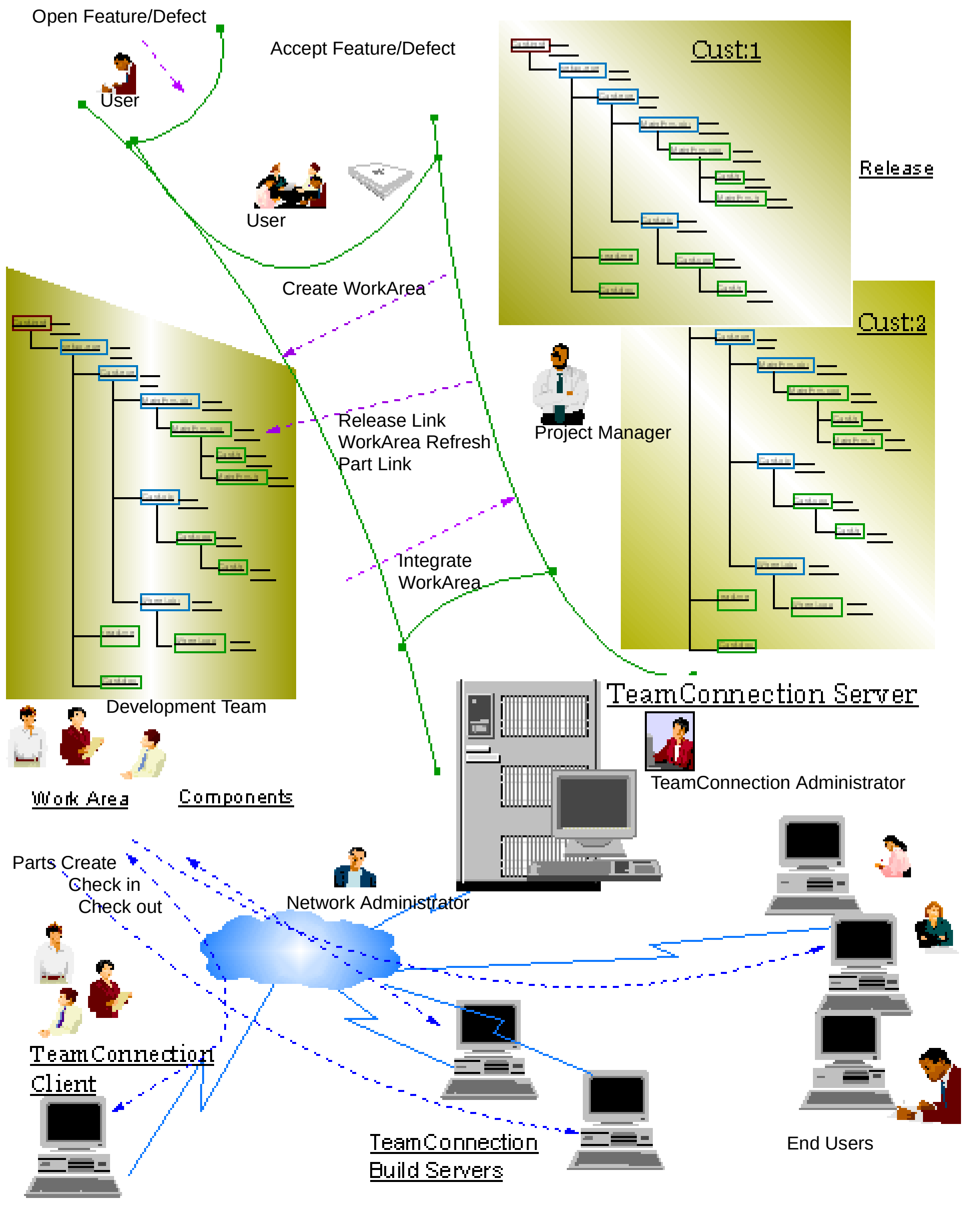


Figure 6. Inserting TeamConnection into Your Development Environment

2.7 Change Your Application With TeamConnection

We have explained the main functionalities of TeamConnection, but we continue to examine the life of your application. So far, your application is under control and you have defined the process to modify it. See Figure 7 on page 22, in which the initial version, labeled with the numeric identifier 1.0, has already been shipped to the customer. In other words, Version 1.0 of the application has been put into production.

You now create a new version, let us say Version 1.1, to fix some defects in the original version. Before updating the production environment, you decide (correctly) to run some tests against the new version. The customer keeps asking every day about the availability date of the fixes, which means that Version 1.1 will be most welcome.

One month later, the customer asks you to add some new functionalities into the initial version, to meet new requirements. Implementing the additional functions requires time, and after discussing the matter with the customer, you decide to ship Version 1.1 first and deliver a new version later (Version 2.0) that includes both the fixes and the new functions. As shown in Figure 7 on page 22, to manage the different versions in the different phases (coding, test, and production) more easily, you must create for each version a work environment with a specific build process, and well-defined responsibilities.

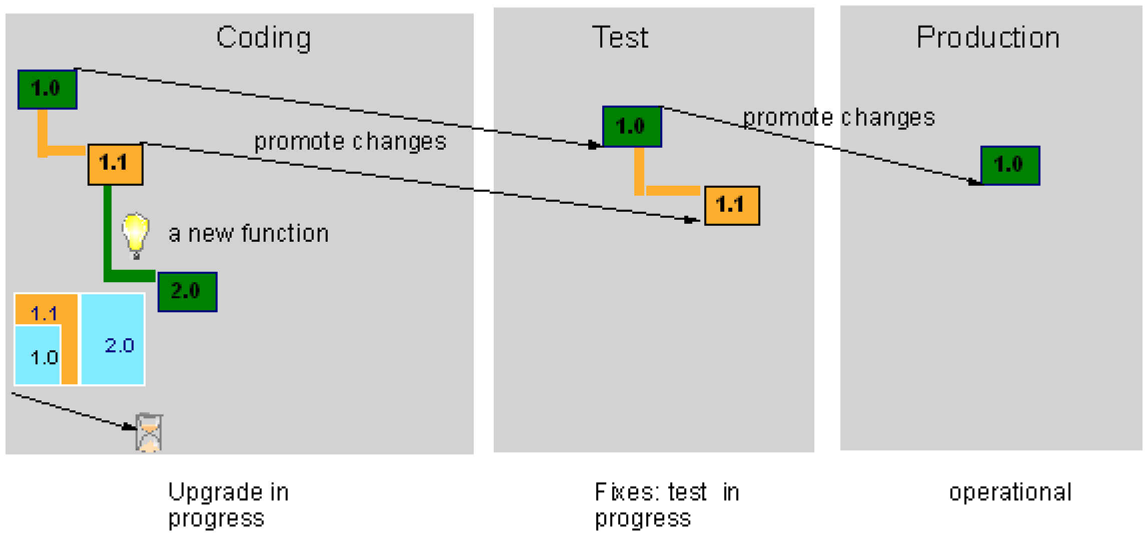


Figure 7. Project Status Before It Gets Crazy

So far, so good. Unfortunately, the customer finds a new bug in the Version 1.0 that is running, and this bug offers the highest risk of damage, which means it must be fixed now. Here are the steps you need to cover first:

1. Analyze the impacts of the bug. You need to know, for example, which data have to be changed, and which existing versions must be updated.
2. Based on the result of the impact analysis, fix the bug by changing the proper data in the bugged version.
3. Based on the result of the impact analysis, propagate the fix to the other affected versions of your application.

2.7.1 Impact Analysis

At this point, you are back to the question: “If this file changes, what other files also need to be changed and what applications, or application versions, have to be rebuilt?" To answer that, you must once again find the answers to these questions:

- What are the data cross-dependencies, or how can I check them? For example, how do I figure out the class hierarchy for an application developed with an object-oriented language or, even worse, the cross-references if 3GL is used?
- Does my application share data with other applications, or with various versions of the same application? And when does it do so? Is it:
    - At run time (execution), such as dynamic libraries, distributed objects?
    - At build time (compilation, link edition), such as include files, classes, static libraries, database connection?
    - At creation time (automatic generation), such as document templates, existing frameworks?

The first question concerns the dependencies inside one application, and calls for a local impact analysis. The second question deals with a collection of application-sharing data, and calls for a cross-impact analysis. (These definitions reappear later in this chapter.)

Suppose your current application is developed in C. To fix the bug, you have to modify the type of one parameter of a C function, which is part of a dynamic link library (.dll or .a), and to tune a constant in a header file. You now face these two questions:

- Into which C source files do versions of the applications need to be edited to correct the function call statement?

First, to get the list of the affected application versions, you could scan all the make files to search for the names of static libraries (.lib or .a) associated with the modified dynamic link library. Then, for each affected version, you could scan all C source files to search for the affected function name. It is easy enough to describe, but it takes a long time to do.

- Which versions of the application require a new build?

  You could scan all the C source and header files to search the name of the modified header file.

If you are currently developing under UNIX, with a small shell script using the find, and grep commands, this is not so difficult.

(For example: `find . -type  f -exec or grep -l ì$1î {} \; ).`

If, unfortunately, your development environment is not UNIX, then go to the Internet to download a UNIX-like package. You could also use a data dictionary for a 4GL application. What is most important is to gather all the necessary data from all affected versions at the right place otherwise you can miss some affected versions.

TeamConnection can help greatly in the impact analysis phase, as we explain later in the chapter.

2.7.2 Bug Fixing

Now that you have all the required information to take care of the changes across all the versions of your application, let us start the so-called *hot fix* process depicted in Figure 8 on page 25:

1. Make the necessary changes to fix the bug in Version 1.0 and put the changes into a new version of your application (Version 1.0.1).
2. Run the tests again. (This is called a *nonregression test*.) Ship the new version to your customer.

At this point, you can wait for customer feedback or, better yet, you can be proactive and ask how it works now.

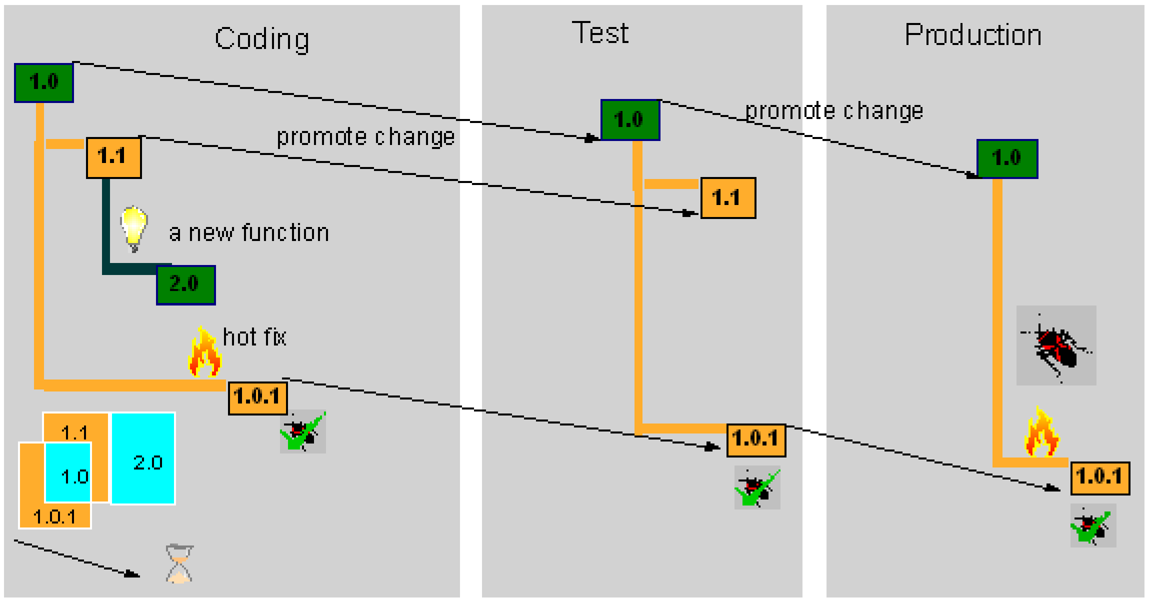


Figure 8. Project Status at the End of the Hot Fix Process

Let us suppose the feedback is: “Great! the bug has disappeared!” Nevertheless, the change process is not yet finished.

2.7.3 Fix Propagation

Based on the impact analysis, you now have to propagate the changes to two other versions: 1.1 and 2.0. The propagation process consists of identifying data being changed in all the versions, reconciling the changes between the existing versions, and preparing new versions.

Identifying the changed data consists of cross-checking the different versions to build the list of data changes. Viewed from the target version, the reconciliation process can entail getting data from one previous version, or merging data from one or more previous versions. Figure 9 on page 26 shows the changes made in four different versions and how to merge them.

This is one of the worst cases you might have. Each area of intersection is a collection of data changed for more than one version. The remaining area is a group of data created for a given version. Whereas Intersection Area A shows data modified from Version 1.0 to Version 1.1, Intersection Area F groups the data that have been changed from 1.0 to 1.1, from 1.0 to 1.0.1, and from 1.1 to 2.0.

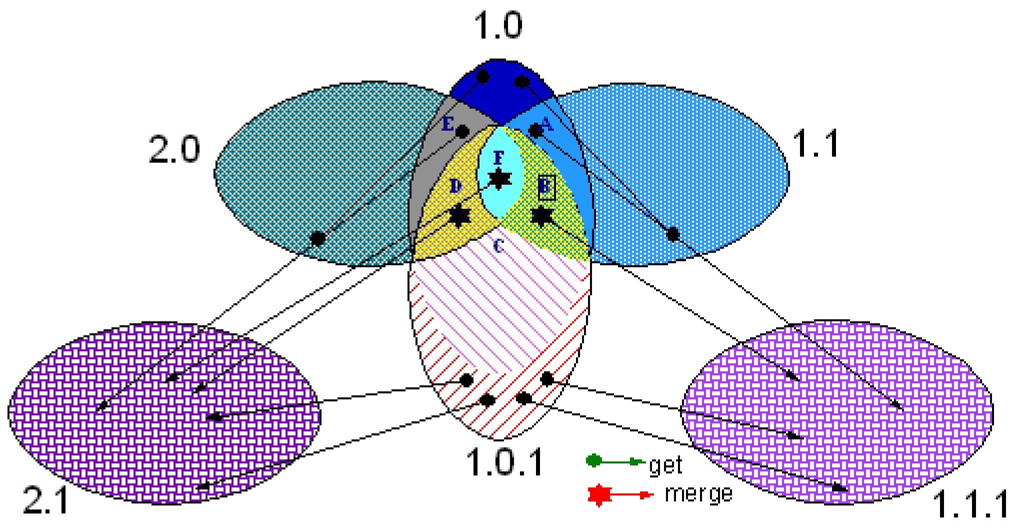


Figure 9. The Merge Nightmare

Table 1 on page 26 summarizes the meaning of the intersections denoted with a capital letter in Figure 9 on page 26 and what you have to do for each of them. The cross (X) means data have been changed in the version. When there is only one cross in a row, you need to get data from only that version. When a row has more than one cross, you have to merge different versions.

Table 1. Merge Cases for Four Versions

	Version 1.1	Version 2.0	Version 1.0.1	Version 1.1.1	Version 2.1
Intersection A	X			get from 1.1	get from 1.1 or 1.1.1
Intersection B	X		X	merge 1.1 and 1.0.1	get from 1.1.1
Intersection C			X	get from 1.1	get from 1.1
Intersection D		X	X		merge 2.0 and 1.0.1
Intersection E		X			get from 2.0
Intersection F	X	X	X		merge 2.0 and 1.1.1

The complexity of the propagation depends on various parameters such as:

- The project management: What should we do first? Fix the problem and ship the fix, or fix the problem in the next version?
- The development environment: Process, programming language, tools (editor, incremental compiler).
- The type and design of the application.
- The kind and extent of changes: New data created, one or more pieces of data changed, one or more pieces of data changed several times in different versions.

Imagine, for instance, that you are using a programming language such as C++, COBOL, or Java, and that you are storing your code in files. You can create a new file for each new function. Identification of any modified files will result from the list of new files, coupled with the list of files having undergone changes to fix the bugs. The resulting reconciliation then merely consists of copying the changed files into the new version, because the new files could not have been affected by the change. There is no merge. From the same evidence, you can see that the smaller the files, the lower the probability of collisions.

Another example shows a worse case. Say you have a file, File 1, in the initial version, which you have already modified for Version 1.1, and then for Version 2.0. A bug is then found in Version 1.0, and File 1 must be modified again for Version 1.0.1. You further have to merge Version 1.0.1 with Version 1.1 to create Version 1.1.1, then later merge Version 1.0.1 (or 1.1.1) with Version 2. If File 1 is a text file, you can find a merge tool easily; if not, if it is a binary file such as a Word document; you face a real problem.

Figure 10 on page 28 ends our story. It shows that the bug is first fixed in the coding environment, then propagated to test, and then to production. That way of working forces us to run nonregression tests. We could have fixed the bug in the production environment first, had the need for bug resolution been sufficiently urgent. Whatever the case, the customer is satisfied because the production version works fine and it is bug free. The test team is running the test cases on the correct next version, and the coding team is implementing new functions on the right bases.

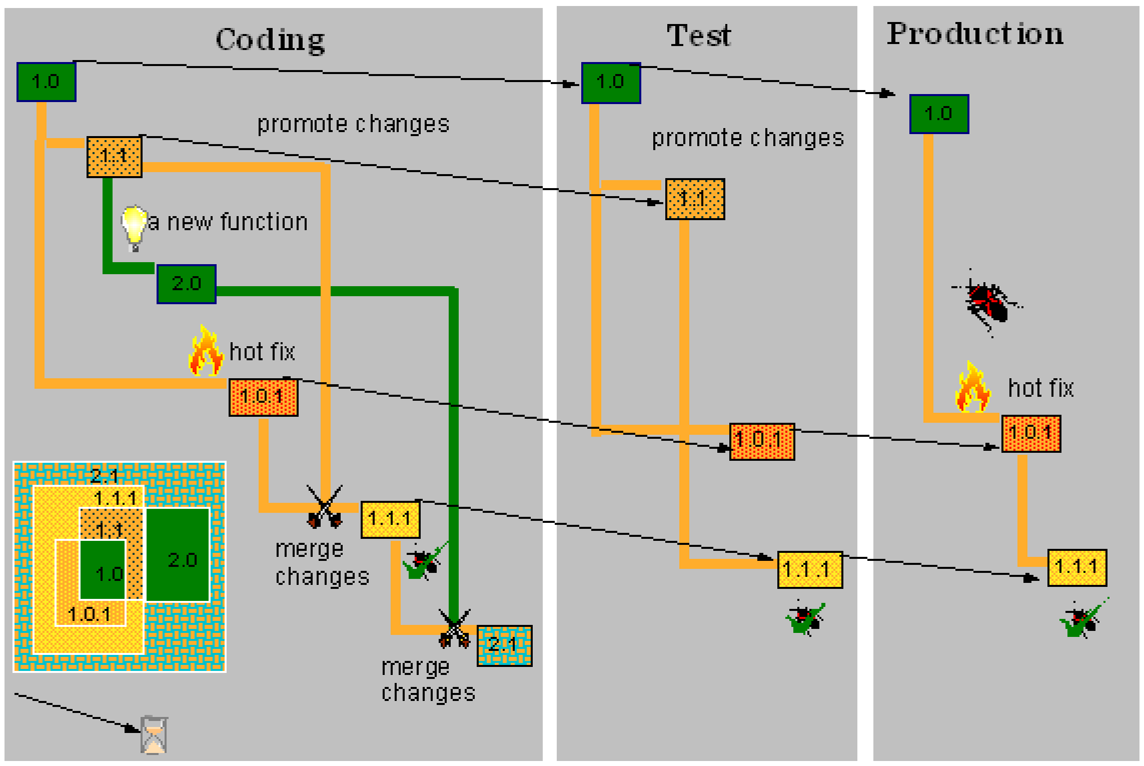


Figure 10. Getting Out of the Nightmare

TeamConnection can manage the changes in your applications. Even when you have read the documentation, you are likely to find some points you cannot figure out how to take most advantage of. Once you have understood the major concepts of the tools, you are likely to want to play with them. Each of the next three sections gives an example of using TeamConnection for one of the stages described so far.

2.8 Control Your Change with TeamConnection: An Example

Here is an example of how you could use TeamConnection to handle change management. As shown in Figure 11 on page 29, you create a TeamConnection release to group all parts of the application and to set up the right build process by defining TeamConnection builders and parsers. You also create a TeamConnection driver for both test and production environment, to collect, build, package, and distribute changes.

Because an input part cannot be changed in a driver, you could say that a driver is a kind of baseline. The coding environment is the bundle of active workareas in which you can make changes on any parts. Controlling access to the parts, as well as controlling responsibilities, are obviously defined through a TeamConnection component structure, but for simplicity, the structure is not drawn in Figure 11 on page 29.

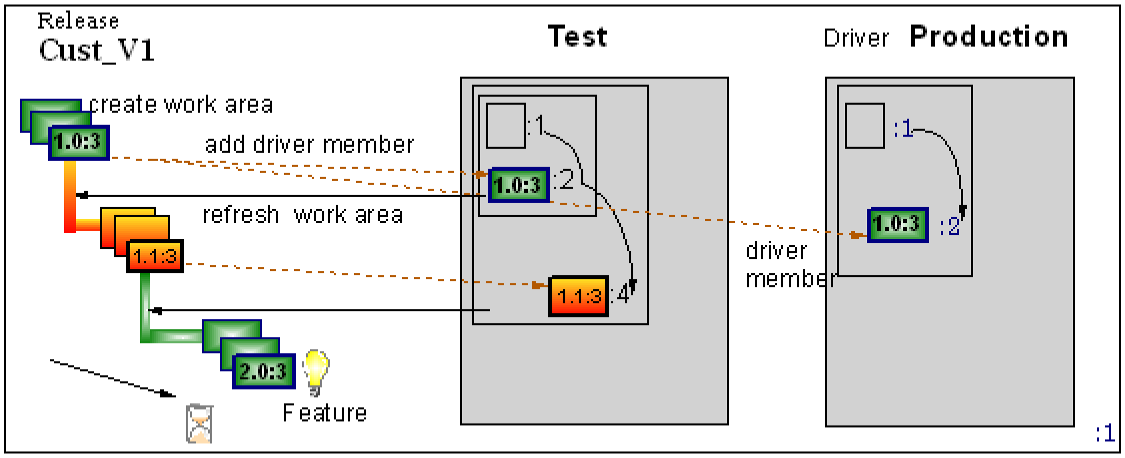


Figure 11. Project Status before Change Management Implementation.

Look at back at Figure 8 on page 25, and compare it with Figure 11 on page 29: there is little fundamental difference. In Figure 8, the creation of the link of succession between two versions, say 1.0 and 1.1, remains a mystery. (Some changes may have been forgotten during the linking operation.) In Figure 11, on the contrary, you see the successive links in the driver. However, we recommend considering a given version of a driver as nested groups of changes instead of as a tree structure. For example, Version 4 of the driver test contains the changes made for Versions 1.0 and 1.1.

2.8.1 Impact Analysis

The two basic types of impact analysis are explained here.

2.8.1.1 Local Impact Analysis

At this stage, you know only that version of your application on which changes have been opened. What you want to know now is which data must be changed (edited, or built) in the current version. In TeamConnection, you always modify a part that is an element of one or more build trees (or a node on a graph). The part, which has been defined in a given workarea, was created for a given release, with reference to a specific defect or feature. The group workarea, release, defect or feature is often called a *work context*.

Whenever you change a part, TeamConnection walks through all the build trees, or through the build graph defined in the current workarea, and updates the build status of all the parts that have the changed part as a successor through a relation of the type *OutputOf* or *InputTo*. To get a list of the impacted parts, make a query that includes *Build status = out_of_date*. That query engine selects only the impacted parts linked by relations of the type *InputTo*, or *OutputOf*, but not by the *DependentOf* relation.

If you do not want to change parts (*check-out*, *check-in*) and intend to do an impact analysis preview, you can use the *Touch_part* action, which changes the time stamp only on the part in the workarea.

The *DependentOf* relations are automatically checked when you start a build. Before building anything, TeamConnection invokes the parser for all parts that depend on the changed part, then updates the build tree automatically. Updating the build tree means creating or deleting *DependentOf* relations, according to the results of the parsers.

A parser can be used to check any kind of dependency, such as C header file inclusion, or C external function calls. To obtain the list of impacted parts without building your application, use the Report mode of the *part_build* command, which gives you a preview of which parser would be run and on which parts, and what would be built if you invoked a build. The report identifies what steps would occur without any translations taking place.

Here is a simple procedure to analyze the local impact:

1. Create a temporary workarea for the release for which you found the bug.
2. Start a build (with the Report mode turned on) of the part that is the root of the build tree. You get the list of all parts impacted by the *DependentOf* relationships.
3. Touch all parts of the list returned by the fake build. To get the list of parts impacted by the *InputOf* relationships, ask TeamConnection to show all parts with status *out_of_date*.
4. Destroy the workarea by performing an *Undo_work_area* action, then a *Cancel_work_area* action. Please note that if you made other changes to the workarea, you may need to undo the workarea many times before it is truly empty of changes. (If you undo the workarea all the way through, the last undo action actually cancels the workarea and it vanishes.)

Apply the procedure to the build tree sample shown in Figure 12 on page 31.

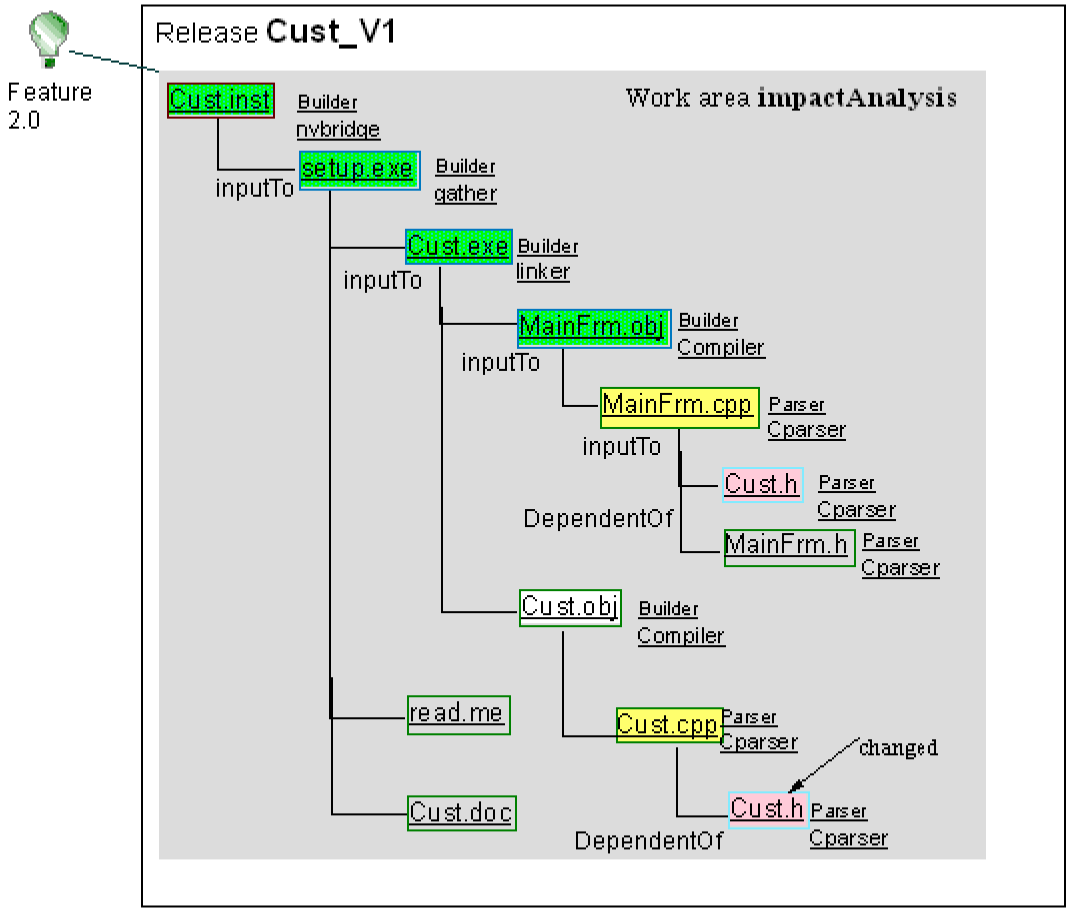


Figure 12. Local Impact Analysis TeamConnection Facilities

Assume that you change the part Cust.h that belongs to Application Cust_V1. Now do this:

1. Create a temporary workarea called *impact Analysis* for the release Cust_V1.
2. Run a build of Cust.inst part. The fake build lists MainFrm.cpp and Cust.cpp.
3. Touch both parts MainFrm.cpp and Cust.cpp. The query based on the status out_of_date lists the predecessors of the two parts.

4. Undo the workarea impactAnalysis, then cancel it.

2.8.2 Cross-Impact Analysis

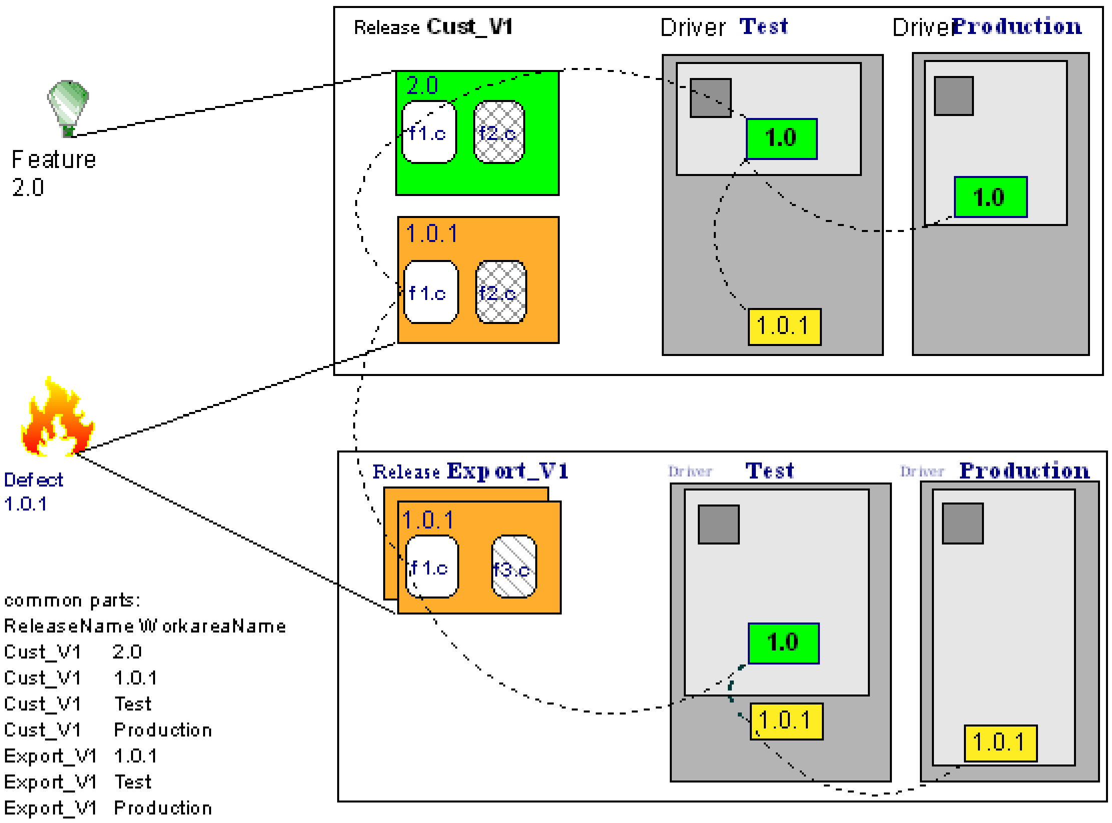
You may also need to know what versions of other applications are impacted by the changes. Within TeamConnection, parts can be shared among as many workareas (or drivers of whatever releases) as you like, much as a file could be located on different directories (workspaces) of different computers (reference areas). TeamConnection keeps track of all the links, cross-parts, cross-workareas, cross-drivers, and cross-releases. It is therefore easy to see how TeamConnection can help you analyze the potential impacts of any changes efficiently.

If you want to estimate the work load for a given change, you can use one of two types of procedures:

- You can ask TeamConnection to view information about the impacted parts, and get a common parts section (if not, scroll down the information window) in which TeamConnection enumerates all the common versions of the parts, so that you can see the list of impacted workareas and releases.
- You can request TeamConnection to show all impacted parts and versions from the PartFull window (ordered first by Release name and then by Current version).

  WARNING: Avoid using the PartFull window unless you really need it. PartFull looks at ***all*** the parts in TeamConnection, one by one, in order to find out the release name and current version. This process may takes a very long time once your family is fully operational.

Figure 13 on page 33 shows various versions of two applications, Cust_V1 and Truck_V1, sharing one file, f1.c. The dashed lines are the sharing relationships. If you look at the information associated with File f1.c in workarea 1.0.1, you should read in the common parts section something like the text in the bottom left corner of the drawing.



common parts:

ReleaseName	WorkareaName
Cust_V1	2.0
Cust_V1	1.0.1
Cust_V1	Test
Cust_V1	Production
Export_V1	1.0.1
Export_V1	Test
Export_V1	Production

Figure 13. Cross-Impact Analysis TeamConnection Facilities

2.9 Bug Hot Fix

Let us run through the hot-fix process by translating it into TeamConnection language. If you want a quick view of the various TeamConnection actions performed during the fix process, see Figure 14 on page 35. The process is:

1. Report a bug: Open a defect, assign it, then accept it.
2. Create a workspace to solve the problem: create a workarea in reference to the defect.
3. Copy the bugged version into the defect workarea to be sure you make changes against the right parts: refresh the defect workarea from the production driver. (We could also have said, derive the bugged version to change it, or the new version is a successor of the bugged version.) Each

time a workarea is refreshed, a new version of the workarea is created in which the new changes can be made. The *refresh* action actually freezes the workarea twice: once before the refresh and once after the refresh. The source of the refresh, either a workarea or a driver, is frozen once before the refresh. Freezing the workarea or driver before a refresh ensures that you can roll back the workarea or driver if an error occurs during the refresh action.

4. Fix the bug: Modify the parts in the defect workarea:
   - The most used part actions are:
     - *Check-out*: Transfer the part from the TeamConnection family server machine to the working directory of the TeamConnection client workstation.
     - *Check-in*: Send back the part from the working directory of the TeamConnection client workstation to the TeamConnection family server machine. The previous content of the part is overwritten in the workarea.

   WARNING: When remarks are entered in an unfrozen check-in, if you check-in twice and enter remarks each time, ***only*** the remarks in the last check-in remain; the remarks in the first check-in disappear.

   - *Delete*: Delete does not destroy the part; it only moves its status to delete.
   - *Recreate*: Put the part back in the game.
   - *Modify properties:* Modify properties such as path name (rename), the associated builder and parser, or the build options,
   - *Undo*: Reverts back to the last frozen part version within the workarea. In other words, if you check-out parts and check them back in several times without freezing the workarea, then perform the Undo action, you discard all the changes made by all the check-out and check-in actions.
   - workarea actions:
     - *Freeze*: Create a new version of the workarea; in other words, make a backup of the changes made in the parts.
     - *Undo*: Return the workarea to its original state before a freeze; it means rolling back the changes.
5. Promote the changes to test: Integrate the defect workarea, then add it as a driver member of the test driver. Each time a driver member is added to a driver, a new version of the driver is created, in which the new changes are stored. No change can be made in an integrated workarea.

6. Test the fix: Build parts in the test driver with the test options if they exist, then extract the parts required to test the application to the test computers. You could also build a part representing your application (usually it is a *none* part, also called *collector*, and it is the root of the build tree) with a builder that allows you to install the application on the test computers.
7. Promote the changes to production: Add the defect workarea as a member of the production driver
8. Package and distribute the fix: Build parts in the production driver with the final options, if they exist, then extract the parts composing the application, package them, and transfer them to the media used to install the application in the production environment. You can also build the part by grouping the parts to be shipped (usually a none part, called *collector*) with a builder that allows you to package and distribute the application to the end user-computers.

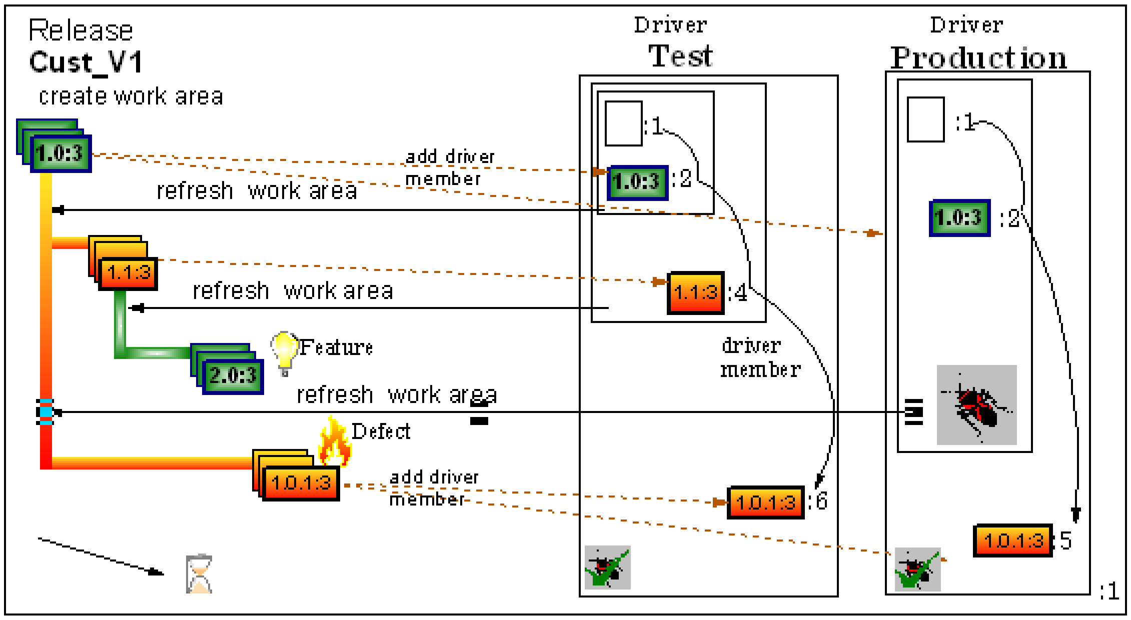


Figure 14. Hot Fix with TeamConnection

As you can see from Figure 14 on page 35, it is not so difficult to fix a bug with TeamConnection, and it is close to what you usually do. What about propagating the fix to the existing versions?

2.10 Fix Propagation

The two stages of the propagation process, identification of data changed in the different versions, and reconciliation of the different changes, can also benefit from TeamConnection.

The first stage is automatically done by TeamConnection when you refresh a workarea from another workarea or driver. (If no source is specified, the refresh takes the changes from the release.) As a result, you have to refresh as many times as there are alternative versions.

TeamConnection does the following whenever you refresh a workarea:

1. Makes a backup of the workarea to be refreshed as well as a backup of the source. In other words, a new version is created for both target and source, so that you can always go back if something goes wrong.
2. Checks the different versions of the parts to detect any inconsistencies.
3. Creates a collision record for each part that differs between the target and the source. (Collision records are generated only when the concurrent development process has been activated for a release, not in the default serial mode.)

What do you still have to do? For each active collision record, you can either:

- Accept the alternative version. This is a sort of copy function,
- Reject the alternative version.
- Reconcile the current version with the foreign version. TeamConnection reconciliation consists of automatically performing the following actions:
    - Checking out the current version of the part.
    - Extracting the alternative version of the part.
    - Invoking the merge tool. From here, TeamConnection builds a composite version that includes the changes from both versions. The build can be more or less easy, depending on the changes made. For example, if lines were appended, it is enough to save the composite file. If, however, lines were deleted, changes inserted. or, worst of all, modifications were introduced in a C function interface, then TeamConnection builds a consistent file.
    - Checking the current version back in after you have saved the composite file.

If the parts to be merged are text files, TeamConnection provides a merge tool called TCMERGE, and you can run through the entire reconciliation process described above. But for binary parts, such as word processor files, this is quite tricky because no binary merge tool exists, except in a few specific cases. You must then run through the reconciliation process manually, step by step. To compare the two versions, it is easiest to start the editor for each of them (or maybe to print them) and to use the well-known Cut/Paste function. The human factor becomes very important in the merge stage.

Figure 15 on page 38 concludes the hot fix process using TeamConnection. Here are the few actions you have to perform to keep all three versions consistent in parallel, thanks to TeamConnection:

1. Create a workspace to merge Version 1.0.1 and 1.1: Create a workarea, in reference to defect calls 1.1.1, for example. You can no longer use workarea 1.1 because it is in integrated state. (You could, in fact, reuse workarea 1.1, but if you do, you lose the link between the changes made against the parts and the defect report.) To reactivate an integrated workarea that is also a driver member, you must first remove the associated driver member and then move its status to Fix.
2. Get the bugged Version 1.1: Refresh workarea 1.1.1 from workarea 1.1.
3. Cross-check Version 1.0.1 and Version 1.1: Refresh workarea 1.1.1 from the driver test that contains the fix. It could be from driver production, or from workarea 1.0.1. TeamConnection creates collision records.
4. Integrate the fix into Version 1.1.1: Reconcile the collision records, either accepting, rejecting, or merging parts
5. Promote the new Version 1.1.1 to test, then production: Integrate the workarea 1.1.1, then add it as driver member to both driver test and production.
6. Cross-check Version 1.0.1 and Version 2.0: Refresh workarea 2.0 from the driver test that contains the fix. It could be from driver production, or from workarea 1.1.1.. TeamConnection creates collision records.
7. Integrate the fix into Version 2.0: Reconcile the collision records, either accepting, rejecting, or merging parts.

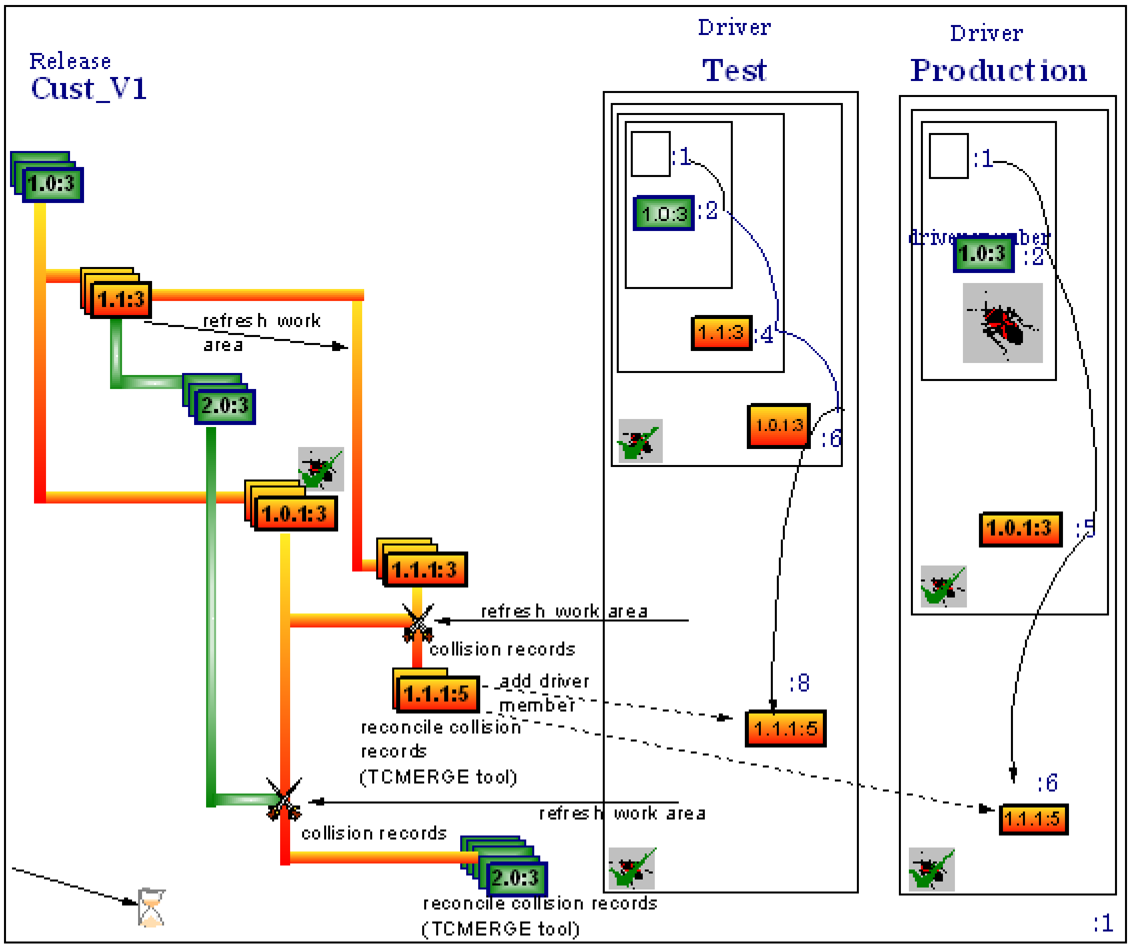


Figure 15. How to Propagate a Fix With TeamConnection

Figure 15 on page 38 shows the process with the addition of TeamConnection. This ends the chapter and our exploration of TeamConnection.

Chapter 3. Preparing to Install TeamConnection

This chapter explains how to prepare for the installation of TeamConnection servers and clients on all supported platforms. It contains some preliminary information you will need before you start the installation process and explains:

- Hardware and software requirements for each platform
- Information on setting TCP/IP addresses and ports

3.1 Migrating Your Database

If you have previously installed CMVC or a previous version of TeamConnection, and you want to migrate to TeamConnection Version 3, you need to install TeamConnection Version 3 on a separate machine and migrate your database to TeamConnection Version 3. See the relevant chapters in section three of this book for instructions.

Installing TeamConnection Version 3 causes any previous version of all of the components of TeamConnection to be overwritten without any warning messages. You will not be able to go back to the version of TeamConnection that was previously installed without completely reinstalling that version. If you already have an existing version of TeamConnection installed, ensure that none of the TeamConnection daemons and tools are running.

3.2 Hardware and Software Requirements

The following sections provide information about DB2 requirements and list the hardware and software requirements for each platform supported by TeamConnection. Each platform section contains the following tables:

- Server hardware requirements
- Client hardware requirements
- Software requirements

The last section contains requirements for the MVS build server. TeamConnection requires auxiliary software, such as the Adobe Acrobat Reader, a Web browser such as Netscape, and the Java Development Kit. The Download page of the TeamConnection Web site contains information about obtaining this auxiliary software. To access this information, go to the VisualAge TeamConnection Enterprise Server home page at URL

http://www.software.ibm.com/ad/teamcon

and select **downloads**.

3.3 DB2 Universal Database

The TeamConnection server requires DB2 Universal Database Version 5. During TeamConnection server installation, you have the option of installing the appropriate edition of DB2 on your server (Personal Edition on Intel platforms or Workgroup Edition on UNIX), if you do not already have it installed. On some platforms, the installation program also checks for the appropriate version of DB2 and gives you the option to install it.

For specific instructions on installing DB2 during TeamConnection server installation, refer to the chapter specific to your platform in Part 2, “UDB Mysteries Unfolded” on page 59. The TeamConnection installation CD contains the *DB2 Universal Database Quick Beginnings* manual in HTML format. You can access Quick Beginnings from the following directory paths on the TeamConnection installation CD:

\db2pubs\en_US\db2ix.htm, for UNIX platforms

\db2pubs\enu\db2i2.htm, for OS/2 platforms

\db2pubs\enu\db2i6.htm, for Windows NT platforms

On Solaris platforms, you need DB2 FixPak U453783. If your Team-Connection server already has DB2 installed, make sure you apply FixPak U453783 before installing TeamConnection. If you do not have DB2 installed and it is installed as part of the TeamConnection server installation, apply FixPak U453783 after TeamConnection and DB2 are installed.

3.4 Requirements for TeamConnection for AIX

Tables 2, 3, and 4 present the hardware and software needed to run TeamConnection on AIX. Table 2 shows the server hardware needed.

Table 2. Hardware Requirements for an AIX TeamConnection Server

Family server	**Processor:** IBM RS/6000 with PowerPC architecture (recommended), such as 43P. PowerServer architecture can be used.
	Pointing device: A mouse or other pointing device
	Monitor: Any X11 graphics display supported by the RS/6000 workstation
	Memory: 128 MB minimum (256 MB recommended); more may be needed according to number of users and database size

	Disk space: - 500 MB for operating system and prerequisites - 200 MB in the target file system for user data (1+ GB recommended) - 160 MB for TeamConnection server code - 65 MB for TeamConnection documentation - 200 MB for DB2 - Amount recommended by operating system for swapper space **Note:** Older buses, I/O controllers, and hard drives may not provide adequate I/O rates for a DB2 application such as TeamConnection to demonstrate minimally acceptable performance.
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product
Build server	**Processor:** IBM RS/6000 with PowerPC architecture (recommended), such as 43P. Power Server architecture can be used.
	Pointing device: A mouse or other pointing device
	Monitor: Any X11 graphics display supported by the RS/6000 workstation
	Memory: 64 MB memory minimum (128 MB recommended); more may be needed according to the compilers and linkers used
	Disk space: – 500 MB for operating system and prerequisites – 60 MB for TeamConnection server code – Amount recommended by operating system for swapper space
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product

Table 3 shows the hardware requirements for an AIX TeamConnection Client.

Table 3. Hardware Requirements for an AIX TeamConnection Client

Client	**Processor**: IBM RS/6000 with PowerPC architecture (recommended), such as 43P. PowerServer architecture can be used.
	Pointing device: A mouse or other pointing device.

	Monitor: Any X11 graphics display supported by the RS/6000 workstation.
	Memory: 64 MB memory minimum.
	Disk space: – 300 MB for operating system and prerequisites – 60 MB for TeamConnection client code – Amount recommended by operating system for swapper space
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation.
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product.

The software requirements for TeamConnection for AIX are shown in Table 4.

Table 4. Software Requirements for TeamConnection for AIX

Server	AIX version 4.2.1 or higher version that includes TCP/IP
	Java Development Kit 1.1. We recommend 1.1.2, but you can also use 1.1.4 with the April 30, 1998 fixes applied (²JDK 1.1.4 IBM build a114–19980430²)
	A Web browser such as Netscape Navigator to display the help panels for the GUI
Client	AIX version 4.2.1 or higher version that includes TCP/IP
	Java Development Kit 1.1 (1.1.2 or higher recommended)
	A Web browser such as Netscape Navigator to display the help panels for the GUI
Build server	AIX version 4.2.1 or higher version that includes TCP/IP
Distribution function	TME 10 Software Distribution Version 3.1 Revision A (LK2T-6047-03)
	TME 10 Framework Upgrade to Version 3.1 Revision C (LK2t-6073-02)
Bridge between VisualAge Smalltalk and VisualAge TeamConnection	TeamConnection client
	VisualAge Smalltalk Version 4.0 (4231060)

3.5 Requirements for TeamConnection for HP-UX

Tables 5, 6, and 7 show the hardware and software needed to run TeamConnection on HP-UX. Table 5 shows the hardware required for an HP-UX TeamConnection server.

Table 5. Hardware Requirements for an HP-UX TeamConnection Serve

Family server	**Processor:** Any HP 9000 Series 700 or 800 workstation
	Pointing device: A mouse or other pointing device
	Monitor: Any X11 graphics display supported by the processor
	Memory: 128 MB minimum (256 MB recommended); more may be needed according to number of users and database size
	Disk space: – 500 MB for operating system and prerequisites – 200 MB in the target file system for user data (1+ GB recommended) – 160 MB for TeamConnection server code – 200 MB for DB2 – Amount recommended by operating system for swapper space **Note:** Older buses, I/O controllers, and hard drives may not provide adequate I/O rates for a DB2 application such as TeamConnection to demonstrate minimally acceptable performance.
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product
Build server	**Processor:** Any HP 9000 Series 700 or 800 workstation
	Pointing device: A mouse or other pointing device
	Monitor: Any X11 graphics display supported by the processor
Build server (continued)	**Memory:** 64 MB memory minimum (128 MB recommended); more may be needed according to the compilers and linkers used
	Disk space: – 500 MB for operating system and prerequisites – 60 MB for TeamConnection build server code – Amount recommended by operating system for swapper space

	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product

Table 6 shows the hardware requirements for an HP-UX TeamConnection client.

Table 6. Hardware Requirements for an HP-UX TeamConnection Client

Client	**Processor:** Any HP 9000 Series 700 or 800 workstation.
	Pointing device: A mouse or other pointing device.
	Monitor: Any X11 graphics display supported by the processor.
	Memory: 64 MB memory minimum.
	Disk space: – 300 MB for operating system and prerequisites – 60 MB for TeamConnection client code – Amount recommended by operating system for swapper space
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation.
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product.

Table 7 shows the software required to meet the needs of the hardware recommended in Tables 5 and 6.

Table 7. Software Requirements for TeamConnection for HP-UX

Server	HP-UX version 10.20, which includes TCP/IP. The following patches are required for DB2: – PHSS_10556 – PHSS_10436
	Java Development Kit1.1 (1.1.2 or higher recommended)
	A Web browser such as Netscape Navigator to display the help panels for the GUI.
Client	HP-UX version 10.20, which includes TCP/IP
	Java Development Kit 1.1 (1.1.2 or higher recommended)

	A Web browser such as Netscape Navigator to display the help panels for the GUI
Build server	HP-UX version 10.20, which includes TCP/IP
For the distribution function	TME 10 Software Distribution Version 3.1. Revision A (LK2T-6047-03)
	TME 10 Framework Upgrade to Version 3.1 Revision C (LK2t-6073-02)
Bridge between VisualAge Smalltalk and VisualAge TeamConnection	TeamConnection client
	VisualAge Smalltalk Version 4.0 (4231060)

3.6 Requirements for TeamConnection for Solaris

Tables 8, 9, and 10 present the hardware and software required to install VisualAge TeamConnection on Solaris. Table 8 begins with server hardware.

Table 8. Hardware Requirements for a Solaris TeamConnection Server

Family server	**Processor:** SPARC or UltraSPARC workstation.
	Pointing device: A mouse or other pointing device
	Monitor: Any X11 graphics display supported by the processor
	Memory: 128 MB memory minimum (256 MB recommended); more may be needed according to number of users and database size
	Disk space: – 500 MB for operating system and prerequisites – 200 MB in the target file system for user data (1+ GB recommended) – 200 MB for TeamConnection server code – 65 MB for TeamConnection documentation – 200 MB for DB2 – 100 MB for temporary space – Amount recommended by operating system for swapper space **Note:** Older buses, I/O controllers, and hard drives may not provide adequate I/O rates for a DB2 application such as TeamConnection to demonstrate minimally acceptable performance.

Family server (continued)	**Communications support:** Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product
Build server	**Processor:** SPARC or UltraSPARC workstation.
	Pointing device: A mouse or other pointing device
	Monitor: Any X11 graphics display supported by the processor
	Memory: 128 MB memory minimum (256 MB recommended); more may be needed according to the compilers and linkers used
	Disk space: – 500 MB for operating system and prerequisites – 200 MB for TeamConnection server code – 100 MB for temporary space – Amount recommended by operating system for swapper space
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	CD-ROM drive:A CD-ROM drive (internal or external) for installation of the product

Table 9 shows the hardware needed for a TeamConnection client on Solaris.

Table 9. Hardware Requirements for a Solaris TeamConnection Client

Client	**Processor**: SPARC and UltraSPARC workstation.
	Pointing device: A mouse or other pointing device.
	Monitor: Any X11 graphics display supported by the processor.
	Memory: 64 MB memory minimum.
	Disk space: – 300 MB for operating system and prerequisites – 60 MB for TeamConnection client code – Amount recommended by operating system for swapper space
	Communications support: Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation.
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product.

What TeamConnection requires in terms of software to run on Solaris is shown in Table 10.

Table 10. Software Requirements for TeamConnection for Solaris

Server	Sun Solaris 2.5.1 which includes TCP/IP. The following patches are required: – 103663-11 – 103600-13 – 103640-08 **Note:** Contact your Solaris service support representative for information on obtaining and applying these patches.
	Java Runtime Environment 1.1. We recommend 1.1.6 or higher.
	A Web browser such as Netscape Navigator to display the help panels for the GUI
Client	Sun Solaris 2.5.1 which includes TCP/IP
	Java Runtime Environment 1.1 We recommend 1.1.6 or higher.
	A Web browser such as Netscape Navigator to display the help panels for the GUI
Build server	Solaris 2.5.1, which includes TCP/IP.
Distribution function	TME 10 Software Distribution Version 3.1, Revision A (LK2T-6047-03)
	TME 10 Framework Upgrade to Version 3.1, Revision C (LK2t-6073-02)
Bridge between Visual Age Smalltalk and VisualAge TeamConnection	TeamConnection client
	VisualAge Smalltalk Version 4.0 (4231060)

3.7 Requirements for TeamConnection for OS/2

Tables 11, 12, and 13 present the hardware and software needed to run VisualAge TeamConnection on OS/2. Table 11 shows the hardware required for a server.

Table 11. Hardware Requirements for an OS/2 TeamConnection Server

Family server	**Processor:** 133 MHz Pentium-based processor or higher

	Pointing device: A mouse or other pointing device
	Monitor: VGA or higher resolution with the appropriate adapter
	Memory: 64 MB memory minimum; more may be needed according to number of users and database size
	Disk space: – 200 MB for operating system and prerequisites – 200 MB for user data (1+ GB recommended) – 60 MB for TeamConnection server code – 200 MB for DB2 – 128 MB for swapper space (150+ MB recommended) **Note:** Older buses, I/O controllers, and hard drives may not provide adequate I/O rates for a DB2 application such as TeamConnection to demonstrate minimally acceptable performance.
	Communications support: Network card supported by TCP/IP for OS/2
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product
Build server	**Processor:** 133 MHz 486-based processor or higher
	Pointing device: A mouse or other pointing device
	Monitor: VGA or higher resolution with the appropriate adapter
	Memory: 32 MB memory minimum; more may be needed according to compilers and linkers used
	Disk space: – 75 MB for operating system and prerequisites – 15 MB for TeamConnection build server code – 45 MB for swapper space
	Communications support: Network card supported by TCP/IP for OS/2
Build server (continued)	**CD-ROM drive:**A CD-ROM drive (internal or external) for installation of the product

Table 12 shows the hardware a client needs to run TeamConnection.

Table 12. Hardware Requirements for an OS/2 TeamConnection Client

Client	**Processor:** 66 MHz 486-based processor or higher
	Pointing device: A mouse or other pointing device
	Monitor: VGA or higher resolution with the appropriate adapter

	Memory: 16 MB minimum
	Disk space: – 75 MB for operating system and prerequisites – 25 MB for TeamConnection client code – 32 MB for swapper space
	Communications support: Network card supported by TCP/IP for OS/2
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product

In addition, you will need the software shown in Table 13.

Table 13. Software Requirements for TeamConnection for OS/2

Server	One of the following: – OS/2 Warp Version 4 (84H1426) and IBM TCP/IP Version 3.0 for OS/2 Warp (33H9749) – OS/2 Warp Server Version 4 (25H8002) – OS/2 Warp Server Advanced Version 4 (25H8030)
	Java Development Kit 1.1 We recommend 1.1.4 with the 1998 fixes applied [²JDK 1.1.4 IBM build o114–19980304²]
	A Web browser such as Netscape Navigator to display the help panels for the GUI
Client	One of the following: – OS/2 Warp Version 4 (84H1426) and IBM TCP/IP Version 3.0 for OS/2 Warp (33H9749) – OS/2 Warp Server Version 4 (25H8002) – OS/2 Warp Server Advanced Version 4 (25H8030)
	Java Development Kit 1.1 We recommend 1.1.4.
	A Web browser such as Netscape Navigator to display the help panels for the Web client and TeamConnection Merge
Client (continued)	One of the following: – OS/2 Warp Version 4 (84H1426) and IBM TCP/IP Version 3.0 for OS/2 Warp (33H9749) – OS/2 Warp Server Version 4 (25H8002) – OS/2 Warp Server Advanced Version 4 (25H8030)
Build server	One of the following: -- OS/2 Warp Version 4 (84H1426) and IBM TCP/IP Version 3.0 for OS/2 Warp (33H9749) -- OS/2 Warp Server Version 4 (25H8002) -- OS/2 Warp Server Advanced Version 4 (25H8030)

Distribution function	TME 10 Software Distribution Version 3.1. Revision A (LK2T-6047-03**)**
	TME 10 Framework Upgrade to Version 3.1 Revision C (LK2t-6073-02)
Bridge between VisualAge Smalltalk and VisualAge TeamConnection	TeamConnection client
	VisualAge Smalltalk Version 4.0 (4231060)

3.8 Requirements for TeamConnection for Windows

Tables 14, 15, and 16 show the hardware and software needed to run TeamConnection on Windows. Table 14 shows the hardware needed.

Table 14. Hardware Requirements for a Windows TeamConnection Server

Family server	**Processor:** 133 MHz Pentium-based processor or higher (however, the PowerPC is not supported)
	Pointing device: A mouse or other pointing device
	Monitor: VGA or higher resolution with the appropriate adapter
	Memory: 64 MB memory minimum; more may be needed according to number of users and database size
Family server (continued)	**Disk space:** –200 MB for operating system and prerequisites –200 MB for user data (1+ GB recommended) – 60 MB for TeamConnection server code –200 MB for DB2 –128 MB for swapper space (150+ MB recommended) **Note:** Older buses, I/O controllers, and hard drives may not provide adequate I/O rates for a DB2 application such as TeamConnection to demonstrate minimally acceptable performance.
	Communications support: Network card supported by TCP/IP for Windows NT
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product

Build server	**Processor:** Any processor supported by the required version of Windows, except the PowerPC
	Pointing device: A mouse or other pointing device
	Monitor: VGA or higher resolution with the appropriate adapter
	Memory: 32 MB memory minimum; more may be needed according to compilers and linkers used
	Disk space: – 75 MB for operating system and prerequisites – 15 MB for TeamConnection build server code – 45 MB for swapper space
	Communications support: Network card supported by TCP/IP for Windows NT or Windows 95
	CD-ROM drive:A CD-ROM drive (internal or external) for installation of the product

The hardware needed for a client to run TeamConnection on Windows is shown in Table 15.

Table 15. Hardware Requirements for a Windows TeamConnection Client

Client	**Processor:** Any personal workstation supported by the operating system, except the PowerPC
	Pointing device: A mouse or other pointing device
	Monitor: VGA or higher resolution with the appropriate adapter
	Memory: 16 MB minimum
Client (continued)	Disk space: – 75 MB for operating system and prerequisites – 25 MB for TeamConnection client code – 32 MB for swapper space
	Communications support: Any network card supported by the above workstation that supports TCP/IP
	CD-ROM drive: A CD-ROM drive (internal or external) for installation of the product

Table 16 shows the software required to run TeamConnection on Windows.

Table 16. Software Requirements for TeamConnection for Windows

Family Server	Microsoft Windows NT 4.0, which includes TCP/IP

	Java Runtime Environment 1.1 (1.1.6 recommended)
	A Web browser such as Netscape Navigator to display the help panels for the GUI
Client	Java Runtime Environment 1.1 (1.1.6 recommended)
	A Web browser such as Netscape Navigator to display the help panels for the Web client and TeamConnection Merge
	One of the following: – Microsoft Windows 95 – Microsoft Windows NT 4.0, which includes TCP/IP
Build server	Microsoft Windows NT 4.0 which includes TCP/IP
Distribution function	TME 10 Software Distribution Version 3.1, Revision A (LK2T-6047-03)
	TME 10 Framework Upgrade to Version 3.1, Revision C (LK2t-6073-02)
Bridge between VisualAge Smalltalk and VisualAge TeamConnection	TeamConnection client
	VisualAge Smalltalk Version 4.0 (4231060)

3.9 Requirements for MVS Build Servers

The MVS build server needs TCP/IP Version 3.2 for MVS and OS/390 R3 LE.

Chapter 4. Inserting VisualAge TeamConnection in Your World

Before installing TeamConnection in your software development environment, it is best to organize a meeting with the two key people: the network administrator (NetAdm), and the TeamConnection administrator (TCAdm).

The TCAdm needs the NetAdm to provide a TCP/IP socket number for each TeamConnection product component to be installed. The NetAdm needs to choose those numbers carefully; they must not be already used by any internal application or reserved by standard applications, such as FTP or TelNnet. Moreover, since the product components use their host names to keep in touch with each other, the files of etc/services and etc/hosts must be updated.

The situation is complicated because more than one instance of each system can be active at the same time on the same machine. For example, several family servers or build servers can be running on one machine. However, only the family server requires TCP/IP sockets. The client must know the socket number of a family server.

If several family servers run on the same machine, the host name is not enough to identify the family; therefore, an alias should be chosen for each family. The alias must be unique; it is required to update the etc/hosts file on the family server machine, or the tables in the TCP/IP Name Server.

Using an alias for the sockets, however, may make it impossible to update etc/services. In order to deal with the problem, the NetAdm needs to know how many instances of each product component the TCAdm will be using.

4.1 Updating TCP/IP Files

Note: For OS/2, Windows NT, and Windows 95 or 98, if you use the Wizards provided to install TeamConnection, then your TCP/IP services and host files are set up for you and you can skip this chapter.

Before installing the TeamConnection code, you must have the correct version of TCP/IP installed on your workstation. See “Hardware and Software Requirements” on page 39 for the communications software needed for your operating system. After TCP/IP is installed, update your TCP/IP services and host files. You can update these files using smit.

4.2 Host and Service Files

4.2.1 General Considerations

AIX: In many installations, a name server is used instead of /etc/hosts. Also, many installations distribute /etc/services. Here, smit can be used to make the necessary updates.

HP: You can update these files using sam. In many installations, a name server is used instead of /etc/hosts. Also, many installations distribute /etc/services. You can use the sam tool to make the necessary updates.

4.2.2 Service File Update

Carry out the following steps:

- **UNIX:** Update the services file, which is located in \etc\services in the directory where TCP/IP is installed.
- **OS/2:** To determine the directory name, type `echo %etc%` at a prompt.
- **Windows NT and Windows 95:** If you have a services file, it is located in the system32/drivers/etc subdirectory of the Windows NT installation directory. If the services file does not exist, you must configure it through TCP/IP.

ALL: Include the family name and port address of the TeamConnection server. The port address can be any four-digit number, as long as it does not already exist in your services file. You may want to ask your TCP/IP administrator to assign you a number. Type the following entry in your services file, replacing ffff with an appropriate port address and following the second line with a carriage return:

```
# TeamConnection servers
testfam ffff/tcp # port address for the TeamConnection test family
```

4.2.3 Host File Update

- **UNIX:** Update the TCP/IP hosts file, which is located in \etc\hosts.
- **OS/2:** To determine the directory name, type `echo %etc%` at a prompt. If the hosts file does not exist, you must configure it through TCP/IP.
- **Windows NT and Windows 95:** If you have a hosts file, it is located in the system32/drivers/etc subdirectory or the Windows NT installation directory. If the hosts file does not exist, you must configure it through TCP/IP.

ALL: Add the following:

- IP address.
- Server name.
- Alias name of the TeamConnection family server, which is your family name. For the initial installation of TeamConnection, the family name is *testfam*.
- Alias name for the build socket. For the initial installation of TeamConnection, use *bldsock*.

The following is an example of the entry you would type in your hosts file. Follow the line with a carriage return. You can use the hostname command to get the name of the server.

```
9.12.345.67 teamserv.company.com testfam bldsock
```

4.3 Verifications and Controls

4.3.1 Controlling the Hostname

Do the following to verify that the hosts file is specified correctly:

- **OS/2 & AIX:** Type `host` *`family_name`*, where family_name is the name of your TeamConnection family.
- **Windows NT** The tchostw.exe utility is available from the samples directory on the TeamConnection CD-ROM. Type `tchostw` *`family_name`*, where family_name is the name of your TeamConnection family.
- **HP:** The tchost.hp utility is available from the *misc* directory on the TeamConnection CD-ROM. Type `tchost.hp` *`family_name`*, where family_name is the name of your TeamConnection family.
- **SUN:** The tchost.sol utility is available from the misc directory in the CD-ROM for TeamConnection for UNIX. Type `tchost.sol` *`family_name`*, where family_name is the name of your TeamConnection family.

The information returned should match the number and name specified in your hosts file entry. For example, using the entry given in the previous step, the system response would be as follows:

```
teamserv.company.com = 9.12.345.67
```

4.3.2 Controlling the IP Address

- **OS/2 & AIX**: Type `host` *`ip_address`*, where ip_address is the IP address of your machine.
- **Windows NT:** Type `tchostw` *`ip_address`*, where ip_address is the IP address of your machine.
- **HP:** Type `tchost.hp` *`ip_address`*, where ip_address is the IP address of your machine.
- **SUN:** Type `tchost.sol` *`ip_address`*, where ip_address is the IP address of your machine.

The information returned should match the number and name specified in your hosts file entry. For example, again using the entry given in the previous step, the system response would be as follows:

```
9.12.345.67 = teamserv.company.com
```

If you do not receive the expected response, contact your TCP/IP administrator to solve the problem.

If the servers are defined in the domain name server, then you can use the UNIX utility "nslookup" instead.

4.3.3 Connection to the Family

Do the following to verify that you can connect to your TeamConnection family:

ALL except SUN:

- At a prompt, type `ping testfam.`
- Press Ctrl+C to end the command.

SUN:

- At a prompt, type `ping -s testfam.`
- Press Ctrl+C to end the command.

If you receive information that is similar to the following, you can successfully connect to your TeamConnection family:

```
PING teamserv.company.com: 56 data bytes
64 bytes from 1.23.457.78: icmp_seg:0. time=0. ms
64 bytes from 1.23.456.78: icmp_seg:1. time=0. ms
64 bytes from 1.23.456.78: icmp_seg:2. time=0. ms
```

For **Windows N**T if you can successfully connect to your TeamConnection family, you should receive information that is similar to the following:

```
PINGING teamserv.company.com (9.12.345.67): with 32 bytes of data:
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
```

In Windows, the PING command will send four requests and then it will stop.

If you receive the message `unknown host testfam`, you cannot connect to the family. Verify that the data you entered in the hosts and services files is correct, and then try the command again. If you still do not get the correct response, contact your TCP/IP administrator to solve the problem.

4.3.4 Connecting to the Build Server

Do the following to verify that you can connect to your TeamConnection build server:

ALL Except SUN:

- At a prompt, type ping bldsock.

Press Ctrl+C to end the command.

SUN:

- At a prompt, type ping -s bldsock.
- Press Ctrl+C to end the command.

If you receive information that is similar to the following, you can successfully connect to your TeamConnection family:

```
PING teamserv.company.com: 56 data bytes
64 bytes from 1.23.457.78: icmp_seg:0. time=0. ms
64 bytes from 1.23.456.78: icmp_seg:1. time=0. ms
64 bytes from 1.23.456.78: icmp_seg:2. time=0. ms
```

For **Windows N**T if you can successfully connect to your TeamConnection family you should receive information that is similar to the following:

```
PINGING teamserv.company.com (9.12.345.67): with 32 bytes of data:
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
Reply from 9.12.345.67: bytes=32 time=10ms TTL=255
```

The PING command will send four requests and then it will stop.

If you receive the message unknown host bldsock, you cannot connect to the build server. Verify that the data you entered in the hosts and services files is correct, and then try the command again. If you still do not get the correct response, contact your TCP/IP administrator to solve the problem.

Note: Do not install the TeamConnection components until the ping commands complete successfully.

Part 2. UDB Mysteries Unfolded

Chapter 5. Installing Universal Database: General Considerations

The ten chapters that make up part two of this book examine the installation of the universal database on a variety of platforms and various aspects of its use.

5.1 Overview

VisualAge TeamConnection Enterprise Server Version 3 uses DB2 Universal Database V5 (DB2 UDB V5) to store objects and parts. There is little information about the operational details for configuring and maintaining DB2 UDB in the VisualAge TeamConnection manuals. The main objective of these pages is to fill this gap.

These pages are intended to provide a collection of hints and tips for the installation, configuration, and maintenance of DB2 UDB V5 for the VisualAge TeamConnection on the following supported platforms:

- AIX
- HP-UX
- Solaris
- Windows NT
- OS/2 Warp

This is neither a tutorial for DB2, nor a substitute for the appropriate manuals and training materials. If you need training in the administration aspects of DB2, the DB2 UDB Administration class will help. See the DB2 home page for more details on this type of education:

http://www.software.ibm.com/data/db2/

The structure of this Part 2 is as follows:

1. Installation of DB2 for UNIX and creation of DB2 instances. See “Installation of DB2 for UNIX” on page 71. At this time there is no silent installation option for the DB2 Installer utility in UNIX; that is why, when installing VisualAge TeamConnection in UNIX, you have to interact with DB2 Installer. Thus, we present in “Installing Universal Database: General Considerations” on page 61, a suggested installation sequence, with snapshots of all the screens displayed to the user, together with suggested values to be entered and default values to be accepted.

2. Installation of DB2 for Windows NT and creation of DB2 instances. See “Installation of DB2 for Windows” on page 109. When installing the VisualAge TeamConnection family server, the DB2 installation utility for Windows NT is invoked in silent mode. However, in case you need to interact directly with this DB2 installation utility, the installation sequence is presented in this document.
3. Installation of DB2 for OS/2 and creation of DB2 instances. See “Installation of DB2 for OS/2” on page 115. When installing the VisualAge TeamConnection family server, the DB2 installation utility for OS/2 is invoked in silent mode. However, in case you need to interact directly with this DB2 installation utility, the installation sequence is presented in this document.
4. Creating a VisualAge TeamConnection Family using the newly created DB2 instance. See “Creation of a Family Using the New DB2 Instance” on page 121.
5. Configuring the DB2 Control Center to control remote UNIX databases. See “Configuring the Db2 Control Center” on page 134.” We highly recommend using the DB2 Control Center, which is available only for Intel workstations, to control the DB2 databases in UNIX.
6. Overview of the administration of databases with DB2 tools. See “Administration of Databases with DB2 Tools” on page 139
7. Working with DB2 instances: See “Working with DB2 Instances” on page 145
8. Using the DB2 Administrator Server in UNIX: how to create, start, stop, remove, etc. See“Working with the DB2 Administration Server in UNIX” on page 153.

The administrative tasks linked to everyday management of a VisualAge TeamConnection family is not covered.

5.2 Important Information about Installation

Unfortunately, the hardcopy documentation for VisualAge TeamConnection needs to be sent to the printer months before the product becomes generally available. Thus, after we send the hardcopy documentation material to the printer, we sometimes discover that we need to add or modify information in the manuals, and we provide the following mechanisms:

- For the most current installation instructions for VisualAge TeamConnection Enterprise Server Version 3, refer to the softcopy Installation Guide, install.pdf, install.htm (master file), or install.txt. All

Installation Guide formats are located in the following directories in the CD-ROMs for VisualAge TeamConnection:

`softpubs/<lang>` (for UNIX)

or

`nls\doc\<lang>` (for Intel)

- To obtain auxiliary software for VisualAge TeamConnection, download the latest fixes for DB2 Universal Database and the auxiliary code for TeamConnection, using:

  1. Acrobat Reader for PDF files.
  2. Java for tcadmin and tcmerge.
  3. Netscape Navigator for displaying the on-line help.

Select the item to be downloaded from:

`http://www.software.ibm.com/ad/teamcon`

5.3 DB2 UDB V5 Packaging with VisualAge TeamConnection V3

The installation CD-ROMs for VisualAge TeamConnection Enterprise Server Version 3 include the installation images for DB2 Universal Database Version 5. This means that a customer who buys VisualAge TeamConnection gets a copy of the necessary DB2 UDB components to run the product.

It is important to clarify that the version of DB2 UDB packaged on the CD-ROM is the same version the customer would get if buying DB2 UDB directly. In other words, the DB2 code that is bundled with VisualAge TeamConnection is the same; only the packaging is different.

Should you have already bought DB2 UDB V5 independently of VisualAge TeamConnection, and have it installed on your system, there is no reason for uninstalling it and installing the version provided by VisualAge TeamConnection.

5.4 Applying Fixes for DB2 Universal Database

After installing VisualAge TeamConnection and DB2 UDB, you must apply the latest DB2 UDB FixPak and interim fix for your operating environment.

For detailed instructions on how to apply these fixes, see the appropriate readme file for your operating environment, as follows:

- UNIX: `<cdrom>/<platform>/db2/fixes/readme.db2`

- Windows NT: `<cdrom>\db2filesw\fixes\readmew.db2`
- OS/2: `<cdrom>\db2fileso\fixes\readmeo.db2`

Here, `<cdrom>` is the CD-ROM drive/mount point, and `<platform>` is `aix4, hpux10,` or `solaris`. These files are ASCII format, and can be browsed using any text browser in your operating system.

5.5 Miscellaneous Installation Details

The VisualAge TeamConnection installation utility will check for DB2 Universal Database Version 5 and attempt to install DB2 if it is not present.

For Intel environments, if you issue one of the following commands from the root directory of the VisualAge TeamConnection CD-ROM, the installation utility will bypass all DB2 checking and installation.

- `install -nodb` (for OS/2)

or

- `setup -nodb` (for Windows)

Chapter 6. DB2 UDB under UNIX: General Considerations

6.1 Assumptions

We make these assumptions:

- No version of DB2 has yet been installed in your target system. This document does not cover migration from previous versions nor the coexistence of DB2 UDB V5 with other versions of DB2.
- We also assume that this is the first installation of DB2 UDB V5 in your system.If you tried before but did not achieve a complete and successful installation, then you need to clean up your system. If you do not, you may encounter problems when running the procedures specified in this document. See the *DB2 Quick Beginnings* manual for details.
- In UNIX, the DB2 Installer utility will be used. We strongly recommend that you install and configure DB2 Universal Database products in UNIX using DB2 Installer because it takes care of creating the necessary group IDs, user IDs, and DB2 instances.

  Furthermore, this utility updates the /etc/services file by adding the new port numbers. If you are familiar with DB2 V1 or V2, you may not know that the current DB2 Installer is a great utility, and you may be tempted to extrapolate from your knowledge of DB2 V1 or V2 and try to create the DB2 instance by hand. If you decide to do it manually, then you need to be aware of the required Fenced User ID and of the highly recommended DB2 Administration Server.
- To facilitate the maintenance of a DB2 database in UNIX, the DB2 Control Center utility is used on an Intel workstation (UNIX has no such utility yet). In fact, the examples in this book use the Control Center from Windows NT. The DB2 Control Center greatly facilitates the maintenance activities of local and remote databases.
- The only communication protocol discussed here is TCP/IP because it is the only one used by VisualAge TeamConnection.
- The default values from the DB2 Installer that are used in a pristine system will be used in this document. For example, the first DB2 instance to be created will be called *db2inst1*.
- Only the English prompts and messages are shown.
- For illustration purposes, the VisualAge TeamConnection family name and DB2 database name of *tcfamily* will be used, both located on the host machine called *oem-ppc3*.

6.2 DB2 Instances and VisualAge TeamConnection Relations

The relationship between a DB2 instance, which is a logical database server environment, and a VisualAge TeamConnection family is explained in this section.

To clarify the subject, we use an AIX system (IBM UNIX) because this environment clearly distinguishes all the players. The objects used by a VisualAge TeamConnection Version 3 family (such as the family `tcfamily` which resides in `/home/tcfamily`) are stored inside a DB2 database, which is controlled by one DB2 instance (such as `db2inst1` which resides in `/home/db2inst1`). The VisualAge TeamConnection family server daemon runs the code from TC_HOME (such as `/usr/teamc`), which in turn uses services from the DB2 instance that runs DB2 code from `/home/db2inst1` and from where the real DB2 code is installed (`/usr/lpp/db2_05_00,` for example).

Table 17. Relationship of TeamConnection Families to DB2 Instances

VA TC Family	DB2 Instance
DB2DBDFT=tcfamily	DB2INSTANCE= db2inst1
TC_DBPATH=/home/tcfamily	DB2INSTANCE_HOME=/home/db2inst1
DB2_DBPATH=database	/home/db2inst1/db2inst1
* Teamcd Server Daemon	

Table 18. Relationship between TeamConnection Codes and DB2 UDB Codes

VA TC Code	DB2 UDB Code
TC_HOME=/usr/teamc	DB2_HOME=/usr/lpp/db2_05_00

Based on these relationships, please keep in mind the following recommendations when you create DB2 instances and VisualAge TeamConnection families:

- Each VisualAge TeamConnection family that is used to maintain production code and documents (that is, not merely for tutorial purposes), should be isolated in one DB2 instance
- This DB2 instance should not be shared by any other production family. By having one production family for every DB2 instance, you ensure that if this DB2 instance stops working, it will not affect other DB2 instances (and consequently, it will not affect other families).
- In UNIX, even though we recommend having one DB2 instance for each family, we strongly recommend using different names for the family itself,

and the DB2 user IDs (instance, fenced user ID, administration server). For example,

  - Use `db2inst1` for the DB2 instance and `tcfamily` for the TeamConnection family.
  - Use `db2as` for the DB2 Administration Server and use `db2fenc1` for the DB2 Fenced User ID.
  - Do not use the same user ID (such as "tcfamily" or "db2inst1") for both the DB2 instance and the TeamConnection family, because there will be actions that you want to do only for the DB2 instance or the family but not both. By having different names you will avoid confusion.

- In a UNIX machine, for medium or large production families, we recommend isolating each DB2 instance and its associated VisualAge TeamConnection database in its own file system. Better yet, try to set up this file system on its own hard disk. In that way, the hard disk will be totally dedicated to the DB2 instance and the TeamConnection family. This should have better performance than a setup in which a single hard disk has to serve other applications, besides the desired pair of DB2 instance and TeamConnection family.
- In a UNIX machine, you can have several DB2 instances, each with a different user ID and group ID. This means that you can have multiple TeamConnection families running on that server, one family per DB2 instance.
- Although in principle it is possible to have multiple DB2 instances in an Intel machine (with degraded performance), in reality you should count on having only ***one*** DB2 instance, and thus, only one TeamConnection family for each Intel machine.

What are the UNIX DB2 environment variables needed in a family?

The following environment variables need to be present in the .profile of a family:

- TeamConnection variable that specifies the location of the family directory

  ```
  export TC_DBPATH=$HOME
  ```

- DB2 variables

  ```
  export DB2_HOME=/usr/lpp/db2_ð5_ðð # directory of DB2
  code
  export DB2INSTANCE=db2inst1 # DB2 instance name
  export DB2INSTANCE_HOME=/home/db2inst1
  ```

- DB2 Instance home directory

  export `DB2DBDFT=tcfamily # Default DB2 family`

- By default, the actual database will be created under the DB2 instance directory; however, you could specify another location such as under the TeamConnection family directory:

  export `DB2_DBPATH=/home/tcfamily/ <===` directory to store the DB2 database files>

- Start the db2profile

  . ${DB2INSTANCE_HOME}/sqllib/db2profile

  export `PATH=$PATH:$DB2INSTANCE_HOME/sqllib/bin`

  export `PATH=$PATH:$DB2INSTANCE_HOME/sqllib/adm`

- For HP_UX: (Location of TeamConnection libraries)

  export `SHLIB_PATH=${SHLIB_PATH}:${TC_HOME}/lib`

  export `SHLIB_PATH=${SHLIB_PATH}:${DB2_HOME}/lib`

- For Solaris: (Location of TeamConnection libraries)

  export `LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${TC_HOME}/lib`

  export `LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:${DB2_HOME}/lib`

6.3 Performance Tuning for DB2

Each VisualAge TeamConnection family is different—has a different usage pattern, a different operating environment, and so on. As a result, the tuning of performance is unique to each family. DB2 is a major component of this performance tuning.

One of the main precepts of performance tuning is to change only one variable at a time and compare the behavior of the system with the appropriate baseline to determine if this change improved performance. However, the DB2 Performance SmartGuide or Wizard, available from the DB2 Control Center, changes many variables at once, so that you will be unable to do an educated comparison with your baseline.

Thus, we advise you to avoid running this DB2 Performance SmartGuide. DB2 has many parameters that affect performance (for better or for worse) and we strongly recommend that anyone seriously considering performance tuning take a DB2 UDB V5 Performance class. See the DB2 home page for more details on this type of education:

http://www.software.ibm.com/data/db2/

Performance tuning is beyond the scope of this book. To get some hints, please visit the IBM VisualAge TeamConnection Enterprise Server Library home page by selecting the item Library at URL:

http://www.software.ibm.com/ad/teamcon

Chapter 7. Installation of DB2 for UNIX

7.1 Overview

After completing the installation of DB2 according to these instructions and using the DB2 Installer, you will have done the following:

1. Installed DB2 UDB Workgroup Edition, including the appropriate NLS related files and on-line documentation.
2. Created a normal DB2 Instance.

   This is where the database for TeamConnection will be created later by the TeamConnection Administration tool. This DB2 instance has its separate user ID and primary group ID. A separate user ID should be used for each DB2 Instance. We recommend creating a new group ID, to be used as a primary group ID for the DB2 instance user ID.
3. Created a DB2 fenced user ID with its separate user name and group name.

   Although this is not used by VisualAge TeamConnection, it is needed by the DB2 tools. It does not consume significant resources. Accept the defaults for this user ID; do not try to delete or change it. Because DB2 to allows the creation of user-defined functions (UDFs), requires that these functions should be placed in a so-called *fenced* area, and not in the DB2 instance. Thus, a dedicated user name and group name are needed to keep the fenced UDFs and stored procedures. For security reasons, we recommend that you do not use the DB2 instance user name and group name for the fenced UDFs and stored procedures.
4. Created a DB2 Administrator Server (DAS) instance which is used to allow remote administration by means of the DB2 Control Center running on an Intel workstation.

   This DB2 Administration Server instance must have its separate user name and group name. For security reasons, we recommend that you do not use the DB2 instance user name and group name for the Administration Server.
5. Created a DB2 Sample database.

   You can use this database as your sand box to practice some of the administration tasks, without affecting the database used by TeamConnection.

7.2 Installation Methods

There are several installation methods, but we focus on only one.

7.2.1 Invoking tcinst.ksh

When you install the TeamConnection server, the VisualAge TeamConnection installation utility, tcinst.ksh, asks if you want to invoke the DB2 Installer utility to install DB2 in your system. We strongly recommend that you use the DB2 Installer to install DB2 in your system.

7.2.2 Other Methods to Install on a UNIX Platform

If you encounter problems with DB2 Installer, you may need to look at other methods to install DB2 in UNIX; see the following chapters in the manual *Quick Beginnings for UNIX*:

- Chapter 21, "Other Methods to Install DB2 for AIX."
- Chapter 22, "Other Methods to Install DB2 for HP-UX."
- Chapter 23, "Other Methods to Install DB2 for Solaris."

These methods are manually intensive, require a lot of work, and have many steps, such as unpacking the installation images, creating the user IDs and group IDs, setting up the configuration files, and so on.

7.3 Planning and Prerequisites

For more details on the planning for installation of DB2 UDB V5 in UNIX platforms (AIX, HP-UX, and Solaris), such as hardware and software requirements, consult Chapter 4 "Planning for Installation" of the manual *Quick Beginnings for UNIX*.

The hardware and software requirements for VisualAge TeamConnection already take into account the corresponding requirements for DB2 UDB, and they are described in the *VisualAge TeamConnection Administrator's Guide*.

7.3.1 The Full Path Where DB2 Is Installed

The full path (DB2_HOME) where DB2 will be installed is shown below. Thus, you need to ensure that you have enough available space in the appropriate file systems.

For AIX, use
/usr/lpp/db2_05_00

For HP-UX, use
/opt/IBMdb2/V5.0

For Solaris, use
/opt/IBMdb2/V5.0

7.3.2 Space Used While Installing DB2

After performing the installation described in this document, the following disk space was used for the DB2 code and the DB2 instance only (this does not take into account the DB2 databases).

- 91 MB in `/usr` (AIX) or in `/opt` (HP-UX and Solaris), where around 30 MB is for documentation.
- 39 MB in `/home` (AIX and HP-UX) or `/export/home` (Solaris) for the actual code and all the three user IDs (the DB2 instance, the DB2 Administrator Server instance and the DB2 Fenced user ID). This includes the SAMPLE database.

Note: In order to create the DB2 database of a VisualAge TeamConnection family, you must have at least 100 MB of free disk space in the file system where the DB2 instance resides.

7.4 Uninstalling ObjectStore

In case you have installed ObjectStore, the database used by VisualAge TeamConnection Version 2, you could uninstall this database management system in order to recover disk space. You can uninstall ObjectStore after you have migrated your TeamConnection family from Version 2 to Version 3. To uninstall ObjectStore in UNIX, do the following:

1. Log in as root.
2. Ensure that the TeamConnection servers are not running. If they are running, then shut them down.
3. Ensure that the ObjectStore servers are not running. If they are running, then shut them down.
4. Remove the following directory:

   For AIX, use

   /usr/lpp/ODI

   For HP-UX, use

   /usr/local/ODI

7.5 Installation Steps

To install the DB2 products on UNIX systems, perform the following steps:

1. Identify and record the parameter values. See “Identify and Record the Parameter Values” on page 74
2. If desired, you can create a file system dedicated to the DB2 instance and the VisualAge TeamConnection family. See “Create a File System for the DB2 Instance and TC Family” on page 76.
3. Update kernel configuration parameters. (This step is not required on AIX). See “Update Kernel Configuration Parameters (Not Required for AIX)” on page 77
4. Mount the CD-ROM. See “Mount the CD-ROM” on page 81.
5. Install VisualAge TeamConnection. See “Using VisualAge TeamConnection Install” on page 84
6. Install DB2 UDB Workgroup Edition. See “Installing the Code for DB2 UDB Workgroup Edition” on page 84.
7. Perform the post-installation tasks. See “post-installation Tasks” on page 104
8. Verify that the DB2 configuration is correct by creating the DB2 Sample database. See “Verification of DB2 Configuration” on page 105

7.5.1 Identify and Record the Parameter Values

After completing these installation instructions, the values shown below will be the values of all the appropriate parameters used during this installation procedure for DB2:

Note: We recommend that you change the passwords of the DB2 user IDs mentioned in the tables below.

The parameters for the normal DB2 instance are shown in Table 19 on page 74

Table 19. Parameters for the DB2 Instance

Parameter	Description
Full path name	/home/db2inst1
User Name	db2inst1
UID	System-generated UID

Parameter	Description
Group Name	db2iadm1
GID	System-generated GID
Password	ibmdb2
TCP/IP Connection Service Name	db2cdb2inst1
TCP/IP Connection Port Number	50000
TCP/IP Interrupt Service Name	db2idb2inst1
TCP/IP Interrupt port number	50001

The parameters for the normal DB2 Fenced User ID are shown in Table 20 on page 75

Table 20. .Parameters for the Normal DB2 Fenced User ID

Parameter	Description
Full path name	/home/db2fenc1
User Name (UDF)	db2fenc1
UID (UDF)	System-Generated UID
Group Name (UDF)	db2fadm1
GID (UDF)	System-generated GID
Password (UDF)	ibmdb2

The parameters for the DB2 Administration Server (DAS) are shown in Table 21 on page 75.

Table 21. Parameters for the DB2 Administration Server

Parameter	Description
Full path name	/home/db2as
User Name (UDF)	db2as
UID	System-Generated UID
Group Name	db2asgrp
GID	System-generated GID
Password	ibmdb2
TCP/IP Port Number	523

7.5.2 Create a File System for the DB2 Instance and TC Family

If desired, you can create a file system dedicated to the DB2 instance and the VisualAge TeamConnection family. In this way, you can increase performance and ease maintenance of the family by isolating it from other applications that might compete for file system resources. Isolation also allows you an easy backup and restoration of the entire file system.

Just for completeness, place the DB2 fenced user ID that corresponds to the DB2 instance in the same file system. However, because the DB2 Administration Server is not really associated with a particular DB2 instance, there is no need to place it in the dedicated file system for DB2 instances.

The following instructions explain how to create the physical directories for the DB2 instance, the DB2 fenced user ID, and the VisualAge TeamConnection family in the same dedicated file system (such as `/home2`).

Because the DB2 Installer does not allow you to specify the physical directories when creating a DB2 instance (it assumes that they will be in `/home`), we need to create the symbolic links from the `/home2` directory to `/home`; in this way, the DB2 Installer will think that `/home/db2inst1` is real, when actually, it is physically located in `/home2/db2inst1`.

Follow these steps:

1. Log in as root.
2. Let’s assume that the file system `/home2` is created and mounted.
3. Under the new file system, create the home directory for the DB2 instance (`db2inst1`) and the VisualAge TeamConnection family user ID (`tcfamily`):
   - cd /home2
   - umask 000
   - mkdir db2inst1
   - mkdir db2fenc1
   - mkdir tcfamily

   Note: For the moment, take note that because the user IDs `db2inst1, db2fenc1,` and `tcfamily` have ***not*** been created yet.The owner of these home directories is still root. Also, note that because the umask was set temporarily to 000 (assuming that the previous umask was 022), the file permissions for these directories are 777 (read-write-execute to all). After these user IDs are actually created, we can change the ownership and the file permissions for these home directories.

4. Create the following symbolic links from `/home2` to `/home`:
   - ln `-s /home2/db2inst1 /home/db2inst1`
   - ln `-s /home2/db2fenc1 /home/db2fenc1`
   - ln `-s /home2/tcfamily /home/tcfamily`
   - umask 022

   See the notes in step 3 about ownership and file permissions of the linked directories.
5. Use the DB2 Installer to create the DB2 instance. See Table 7.5.6 on page 84
6. Create the VisualAge TeamConnection family. See “Create a VisualAge TeamConnection Family” on page 104
7. Modify the ownership and file permissions for the home directories. See “Changing Ownership and File Permissions of Home Directories” on page 104

7.5.3 Update Kernel Configuration Parameters (Not Required for AIX)

Depending on your workstation's operating system and its kernel configuration, you may have to update the kernel configuration parameters.

This step is not required on AIX.

Because changing the kernel in UNIX is a delicate operation, we strongly recommend that you make a backup of your UNIX system before attempting the kernel changes needed by DB2.

7.5.3.1 Recommended Values for HP-UX

The values in Table 22 on page 77 are recommended for the HP-UX kernel configuration parameters, based on the available physical memory. To maintain inter- dependence among kernel parameters, change parameters in the same sequence in which they appear in this table

Table 22. HP-UX Kernel Configuration Parameters (Recommended Values)

Kernel Parameter	64MB - 128MB	128MB - 256MB	256MB+
nproc	512	768	1024
maxfiles	256	256	256
maxuprc	256	384	512
nflocks	2048	4096	8192

Kernel Parameter	64MB - 128MB	128MB - 256MB	256MB+
vninode	512	1024	2048
nfile	(4 * NINODE)	(4 * NINODE)	(4 * NINODE)
msgseg	8192	16384	32768
msgmnb	65535 (1)	65535 (1)	65535 (1)
msgmax	65535 (1)	65535 (1)	65535 (1)
msgtql	128	256	256
msgmap	(2 + MSGTQL)	(2 + MSGTQL)	(2 + MSGTQL)
msgmni	128	256	256
msgssz	16	16	16
semmns	256	512	1024
semmni	128	256	512
semmap	(2 + SEMMNI)	(2 + SEMMNI)	(2 + SEMMNI)
semmnu	256	512	1024
shmmax	67108864 (2)	134217728 (2)	268435456 (2)
shmseg	16	16	16
shmmni	300	300	300

Notes:

1. Parameters msgmnb and msgmax must be set to 65535.
2. Parameter shmmax should be set to 134217728 or 90% of the physical memory (in bytes), whichever is higher. For example, if you have 196 MB of physical memory in your system, set shmmax to 184968806 (176*1024*1024). When using the sam tool, these values are actually represented in hexadecimal:
   - 67108864 = 0X40000000 (0X4 followed by 7 zeros)
   - 134217728 = 0X80000000 (0X8 followed by 7 zeros)
   - 268435456 = 0X100000000 (0X1 followed by 8 zeros)

Warning: Ensure that your system has the HP-UX 10 Transitional Links for HP-UX 9; otherwise, the rebuilding of the kernel will fail. The following commands in HP-UX 10 handle the transitional links, which are installed by default:

`/opt/upgrade/bin/tllist` -> lists the transitional links, if available

`/opt/upgrade/bin/tlremove` -> removes the transitional links

`/opt/upgrade/bin/tlinstall` -> establishes the transitional links

To change the values, do the following:

1. Log in as root.
2. Invoke the SAM tool:

   /usr/sbin/sam &
3. Select **Kernel Configuration**.
4. Select **Configurable Parameters**.
5. Highlight the parameter to be changed.
6. Select **Modify Configurable Parameter** from the **Actions** menu and make the appropriate changes.

   In some cases, the recommended value to be used with DB2 for a parameter will replace an existing formula. We decided to delete the formula and use the actual value from Table 22 on page 77, rather than fudge with the formula by modifying the parameters used in the formula, in order to avoid side-effects (The objective was to avoid making changes to other parameters not listed in the table.)
7. Repeat the previous two steps for every kernel parameter that needs to be updated.
8. Create a new kernel by selecting **Create a New Kernel** from the **Actions** menu.
9. Reboot the system so that the changes can take effect.

We accepted the defaults from the window **Reboot the system**. To continue with the installation on HP-UX systems, proceed to “Mount the CD-ROM” on page 81

7.5.3.2 Recommended Values for Solaris

The values in Table 23 on page 80 are recommended for Solaris kernel configuration parameters, based on the available physical memory.

Table 23. Solaris Kernel Configuration Parameters (Recommended Values)

Kernel Parameter	64MB - 128MB	128MB - 256MB	256MB+
msgsys:msginfo_msgmax	65535 (1)	65535	65535
msgsys:msginfo_msgmnb	65535	65535	65535
msgsys:msginfo_msgmap	130	258	258
msgsys:msginfo_msgmni	128	256	256
msgsys:msginfo_msgssz	16	16	16
msgsys:msginfo_msgtql	128	256	256
msgsys:msginfo_msgseg	8192	16384	32768
shmsys:shminfo_shmmax	67108864	134217728 (2)	268435456 (2)
shmsys:shminfo_shmseg	16	16	16
shmsys:shminfo_shmmni	300	300	300
semsys:seminfo_semmni	128	256	512
semsys:seminfo_semmap	130	258	514
semsys:seminfo_semmns	256	512	1024
semsys:seminfo_semmnu	256	512	1024

Notes:

1. Parameters msgsys:msginfo_msgmnb and msgsys:msginfo_msgmax must be set to 65535.
2. Parameters shmsys:shminfo_shmmax should be set to 134217728 or 90% of the physical memory (in bytes), whichever is higher. For example, if you have 196 MB of physical memory in your system, set the shmsys:shminfo_shmmax 184968806 (176*1024*1024).

To change the values, do the following:

1. Log in as root.
2. Just in case you have an existing `/etc/system file`, make a backup of it.
3. Edit the file `/etc/system` as follows:

   To set a kernel parameter, add a line at the end of the file:

   ```
   set parameter-name = value
   ```

   For example, to set the value of the parameter msgsys:msginfo_msgmax, add the following line:

   ```
   set msgsys:msginfo_msgmax = 65535
   ```

4. Depending upon the amount of physical memory in your system, append the appropriate kernel configuration parameter file to the /etc/system file. If necessary, change the value of shmsys:shminfo_shmmax as described in Note 2 above.
5. After updating the /etc/system file, Reboot the system:

```
shutdown -i6 -y -g0
```

To continue with the installation on Solaris systems, proceed to 11.5.4 “Mount the CD-ROM.”

7.5.4 Mount the CD-ROM

To install DB2 products using the DB2 Installer, you must first mount the platform-specific CD-ROM for either the VisualAge TeamConnection Version 3 or DB2 UDB Version 5. After you have mounted the CD-ROM, you can start installing DB2:

Refer to the following procedures to mount the CD-ROM on different UNIX operating systems.

7.5.4.1 Mounting on AIX Systems

Perform the following steps to mount the CD-ROM on AIX operating systems:

1. Log in as root.
2. Insert the CD-ROM in the drive.
3. Create a root directory, such as /cdrom, by typing the following command:
   mkdir -p /cdrom
4. Allocate a CD-ROM file system by typing the following command:
   smit storage
5. Select **File Systems**.
6. Select **Add/Change/Show/Delete FileSystems**.
7. Select **CD-ROM FileSystems**.
8. Select **Add CDROM FileSystems**.
9. Select the **DEVICE Name**.
10. Respond to the prompt, mount point, by typing the following:
    /cdrom
11. Mount the CD-ROM file system by typing the following command:
    smit mountfs
12. Select the **File System name**. For example, the name can be `/dev/cd0`.
13. Select the **Directory name**: `/cdrom`.
14. Select the **Type of file system**: `cdrfs`.
15. Set the **Mount as READ-ONLY** system to **Yes**.
16. Log out.
17. After mounting the CD-ROM, proceed to “Install DB2 UDB Workgroup Edition Using DB2 Installer” on page 84.

7.5.4.2 Mounting on HP-UX Systems

Perform the following steps to mount the CD-ROM on HP-UX operating systems:

1. Log in as root.
2. Issue the mount command to determine if the CD-ROM file system is mounted and operational. Verify that /cdrom is listed; if not, then do the following:
   1. . To create a directory for the CD-ROM, type mkdir /cdrom.
   1. . Type the following command:

      sam &

Notice that the & (ampersand) at the end of the line will start the command in background mode.

1. . Select **Disks and File Systems**.
1. . Select **Disk Devices**.
1. . Select the entry **CDFS** (for CD-ROM File System), and select **Actions** and then **View more information** from the menu bar.
1. . Click on **Show Device File** and from the list, select the Device file value for the **Block device file**, such as `/dev/dsk/c0t2d0.`
1. . Close the **View More Information** window.
1. . Close the **Disks and File Systems** window.
1. . Exit sam.

3. Insert the CD-ROM in the drive and mount it as in the following example:

```
/usr/sbin/mount /dev/dsk/cðt2dð /cdrom
```

where /cdrom is the CD-ROM mount directory.

Issue the mount command again and verify that /cdrom is listed.

4. Log out.

After mounting the CD-ROM, proceed to “Install DB2 UDB Workgroup Edition Using DB2 Installer” on page 84

7.5.4.3 Mounting on Solaris Systems

Perform the following steps to mount the CD-ROM on Solaris operating systems:

1. Log in as root.
2. Insert the CD-ROM in the drive.
3. If the Volume Manager (vold) is installed on your system, the CD-ROM is automatically mounted as:

   ```
   /cdrom/volume_name
   ```

   where `/cdrom/volume_name` is the CD-ROM mount directory.
4. If the Volume Manager is not installed on your system, mount the CD-ROM by entering commands as shown in the following example:

   ```
   mkdir -p /cdrom/volume_name
   ```

   ```
   mount -F hsfs -o ro /dev/dsk/cðt6dðs2 /cdrom/teamcv3
   ```
5. Log out.

After mounting the CD-ROM, proceed to “Install DB2 UDB Workgroup Edition Using DB2 Installer” on page 84

7.5.5 Using VisualAge TeamConnection Install

When installing the server component of the VisualAge TeamConnection Enterprise Server Version 3, the installation utility will ask you if you want to install DB2 UDB at this time. If you do,

1. Log in as root.
2. Execute the command:

   tcinst.ksh
3. Provide the appropriate information.
4. If you install the Server component of VisualAge TeamConnection, then the installation utility will scan your system looking for an existing installation of DB2 UDB V5. If one is found, then there is no need to reinstall DB2. If it is not found, you see a prompt asking you if you want to install DB2 UDB now. If you do, continue with “Install DB2 UDB Workgroup Edition Using DB2 Installer” on page 84.

7.5.6 Install DB2 UDB Workgroup Edition Using DB2 Installer

The DB2 Installer is a tool that is consistent across all UNIX platforms. It performs the following independent tasks:

- Install the code for DB2 UDB Workgroup Edition. See “Installing the Code for DB2 UDB Workgroup Edition” on page 84 for details.
- Create a DB2 Instance, create the DB2 Administration Server, create all the related IDs, and make the necessary file changes. See “Creating the DB2 Instances” on page 95 for details.

We explain how to perform these two tasks separately. Although it is possible to perform them in the same session with DB2 Installer, for the sake of simplicity, we do not recommend that approach.

Note: If you are using DB2 Installer from a remote server, it is better to open a telnet session instead of using the `rlogin` command to connect to your remote server.

7.5.7 Installing the Code for DB2 UDB Workgroup Edition

If you are installing DB2 UDB as part of your installation of VisualAge TeamConnection, please start with Step 4.

If you are installing DB2 UDB by itself from the CD-ROM (either the one from VisualAge TeamConnection or the one you obtained by direct purchase of DB2), please start with Step 1.

The steps are these:

1. Log in as root.
2. Change to the directory where the CD-ROM is mounted by typing the following command:
   - On AIX or HP-UX:

     ```
     cd /cdrom
     ```

     where `/cdrom` is the mount point of the CD-ROM drive on AIX and HP-UX.
   - On Solaris, if using the CD-ROM from VisualAge TeamConnection, issue the following command:

     ```
     cd /cdrom/volume_name
     ```

     where `/cdrom/volume_name` is the mount point of the CD-ROM on Solaris.
   - On Solaris, if using the CD-ROM from DB2 UDB, issue the following command:

     ```
     cd /cdrom/unnamed_disk
     ```

     where `/cdrom/unnamed_disk` is the mount point of the CD-ROM on Solaris.
3. If this is the CD-ROM with VisualAge TeamConnection, then do the following step; otherwise, skip this step and continue with Step 4.

   ```
   cd <platform>/db2
   ```

   where <platform> is the desired operating system (aix4, hpux10, or Solaris). Continue with Step 4.
4. Type the following command to start the DB2 Installer:

   ```
   ./db2setup
   ```

   It will take some time for the DB2 Installer to start up, as it is scanning your system for information.(Figure 16 on page 86).

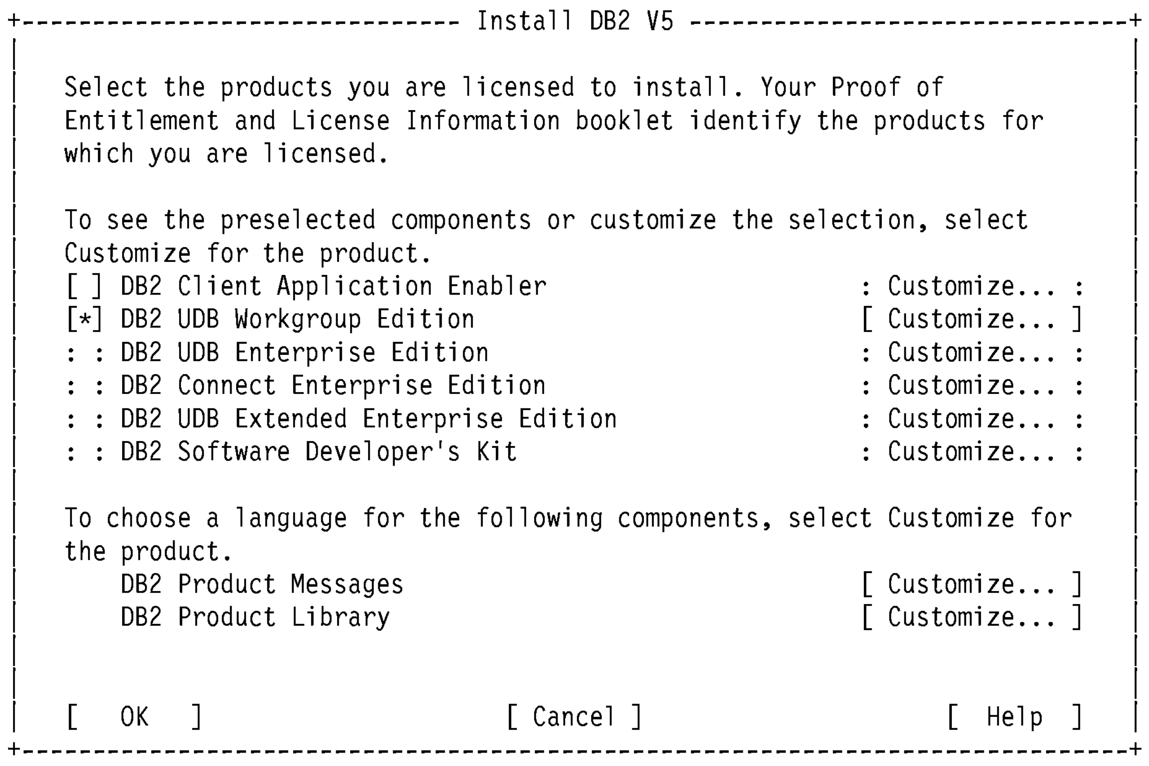
```
+------------------------------ DB2 Installer -------------------------------+
|                                                                            |
|   Select Install to select products and their components to install, or    |
|   select Create to create the DB2 services.                                |
|                                                                            |
|   To select products and their components, select          [ Install... ]  |
|   Install.                                                                 |
|                                                                            |
|   To create a +--- Please Wait -----------------------------+[ Create... ]  |
|   Server, sele|                                             |               |
|               |   Scanning your system for information...   |               |
|               |                                             |               |
|               |                                             |               |
|               +---------------------------------------------+               |
|                                                                            |
|   [  Close  ]                                                  [  Help  ]  |
+----------------------------------------------------------------------------+
```

Figure 16. DB2 Installer Screen

5. From the product list on the **Install DB2 V5** screen (Figure 17 on page 87), select to install: **DB2 Universal Database Workgroup Edition.**

Navigation Hints: Use the left tab key to move around; the Shift-tab key does not move backward. If you keep pressing the tab key, then when you reach the last entry at the bottom right corner, it will start again from the upper left corner. You may also use the arrow keys to move up and down, and left to right. Use the tab key to move to the entry "DB2 UDB Workgroup Edition" then press the space bar to select it.

Select **Customize** and press Enter



```
+------------------------------ Install DB2 V5 ------------------------------+
|                                                                            |
|   Select the products you are licensed to install. Your Proof of          |
|   Entitlement and License Information booklet identify the products for   |
|   which you are licensed.                                                  |
|                                                                            |
|   To see the preselected components or customize the selection, select    |
|   Customize for the product.                                               |
|   [ ] DB2 Client Application Enabler                      : Customize... :  |
|   [*] DB2 UDB Workgroup Edition                           [ Customize... ]  |
|   : : DB2 UDB Enterprise Edition                          : Customize... :  |
|   : : DB2 Connect Enterprise Edition                      : Customize... :  |
|   : : DB2 UDB Extended Enterprise Edition                 : Customize... :  |
|   : : DB2 Software Developer's Kit                        : Customize... :  |
|                                                                            |
|   To choose a language for the following components, select Customize for |
|   the product.                                                             |
|       DB2 Product Messages                                [ Customize... ]  |
|       DB2 Product Library                                 [ Customize... ]  |
|                                                                            |
|                                                                            |
|   [   OK   ]                 [ Cancel ]                    [  Help  ]      |
+----------------------------------------------------------------------------+
```

Figure 17. Install DB2 V5 Screen

6. From the **DB2 Universal Database Workgroup Edition** screen, you can deselect components, such as the **Code Page Conversion Support** if you are using English only. We recommend installing the code to create the sample DB2 database; you can use it later on to verify the DB2 setup and to practice your DB2 administration commands. See "Verification of DB2 Configuration" on page 105 for more details. When you have finished choosing product components, select **OK**. To undo any selections you made, select **Cancel**.

```
+--- DB2 Universal Database Workgroup Edition --------------------------------+
|                                                                              |
|   Required:     DB2 Client                                                   |
|                 DB2 Run-time Environment                                     |
|                 DB2 Engine                                                   |
|                 DB2 Communication Support - TCP/IP                           |
|                 DB2 Communication Support - IPX/SPX                          |
|                 DB2 Communication Support - SNA                              |
|                 DB2 Communication Support - DRDA Application Server          |
|                 Administration Server                                        |
|                 License Support                                              |
|   Optional:     [ ] Open Database Connectivity (ODBC)                        |
|                 [ ] Java Database Connectivity (JDBC)                        |
|                 [ ] Replication                                              |
|                 [*] DB2 Sample Database Source                               |
|                 Code Page Conversion Support:                                |
|                     [*] Japanese     [*] Simplified Chinese                  |
|                     [*] Korean       [*] Traditional Chinese                 |
|                                                                              |
|   [ Select All ]              [ Deselect All ]                [ Default ]    |
|   [   OK   ]                     [ Cancel ]                   [  Help  ]    |
+------------------------------------------------------------------------------+
```

Figure 18. DB2 Universal Database Workgroup Edition Screen

7. You are returned to the **Install DB2 V5** screen, where you could specify, if you want, other locales for installation. Please note that the messages for the default `en_US locale` is installed always (which is why there is no explicit choice for it). Move the tab key to the [Customize...] for DB2 Product Messages. Press Enter.

8. From the **DB2 Product Messages** screen, select the desired locales by moving the cursor (tab key) and then pressing the space bar to select. For example, `ja_JP`. Select the **OK** button that is below Select All (in the inside window). Press Enter.

```
|+--- DB2 Product Messages -----------------------------------------------+|
||                                                                        ||
||   Required:                                                            ||
||   Optional:     DB2 Product Messages:                                  ||
||                     : : Fr_FR       : : fr_FR       : : De_DE          ||
||                     : : de_DE       : : Es_ES       : : es_ES          ||
||                     : : pt_BR       [ ] Ja_JP       [*] ja_JP          ||
||                     [ ] ko_KR       [ ] zh_CN       [ ] Zh_TW          ||
||                     [ ] zh_TW       : : Da_DK       : : da_DK          ||
||                     : : Fi_FI       : : fi_FI       : : No_NO          ||
||                     : : no_NO       : : Sv_SE       : : sv_SE          ||
||                     : : cs_CZ       : : hu_HU       : : pl_PL          ||
||                     : : ru_RU       : : bg_BG       : : sl_SI          ||
||                                                                        ||
||   [ Select All ]            [ Deselect All ]             [ Default ]   ||
||   [   OK   ]                   [ Cancel ]                 [  Help  ]   ||
|+------------------------------------------------------------------------+|
```

Figure 19. DB2 Product Messages screen

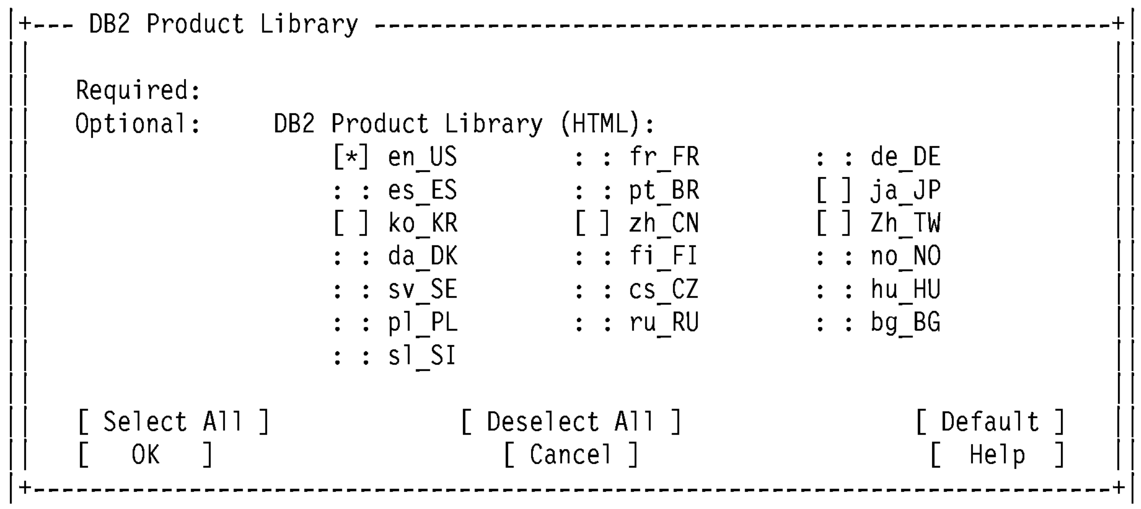
9. You are returned to the **Install DB2 V5** screen, where you could specify, if you want, to install the Product Library (HTML).

   Unlike the Product Messages, in which the en_US is installed by default, the Product Library in the en_US locale is not installed by default, and thus, this time there is an explicit entry for it.

   Move the tab key to the [Customize...] for DB2 Product Library. Press Enter.

10. From the **DB2 Product Library** screen, select the desired locales by moving the cursor (tab key) and then pressing the space bar to select, for example, en_US.

    Select the **OK** button that is below Select All (in the inside window). Press Enter.



```
|+--- DB2 Product Library ---------------------------------------------------+|
||                                                                           ||
||   Required:                                                               ||
||   Optional:     DB2 Product Library (HTML):                               ||
||                     [*] en_US       : : fr_FR       : : de_DE             ||
||                     : : es_ES       : : pt_BR       [ ] ja_JP             ||
||                     [ ] ko_KR       [ ] zh_CN       [ ] Zh_TW             ||
||                     : : da_DK       : : fi_FI       : : no_NO             ||
||                     : : sv_SE       : : cs_CZ       : : hu_HU             ||
||                     : : pl_PL       : : ru_RU       : : bg_BG             ||
||                     : : sl_SI                                             ||
||                                                                           ||
||   [ Select All ]            [ Deselect All ]             [ Default ]      ||
||   [    OK    ]                [ Cancel ]                  [  Help  ]      ||
|+---------------------------------------------------------------------------+|
```

Figure 20. DB2 Product Library Screen

11. You are returned to the **Install DB2 V5** screen, where you can select **OK** and press enter to go to the **Create DB2 Services** screen.

12. From the **Create DB2 Services** screen, select **OK** and press Enter. Do not choose to create a DB2 Instance nor to create the Administration server at this time. They will be created in the next session with DB2 Installer, see "Creating the DB2 Instances" on page 95.

13. Ignore these two warning messages:

    DBI1756W DB2 Instance is not created.

    DBI1755W The Administration Server is not created.

    Select **OK** and press Enter

14. A summary report will be shown (Figure 21 on page 91).

```
+------------------------------- DB2 Installer --------------------------------+
|                                                                              |
|   +-- Summary Report ---------------------------------------------------+   |
|   |                                                                    |   |
|   |                                                                    |   |
|   |  Installation                                                      |   |
|   |  ------------                                                      |   |
|   |                                                                    |   |
|   |  Product components to be installed:                               |   |
|   |                                                                    |   |
|   |    DB2 Client                                                      |   |
|   |    DB2 Run-time Environment                                        |   |
|   |    DB2 Engine                                                      |   |
|   |    DB2 Communication Support - TCP/IP                              |   |
|   |    Administration Server                                           |   |
|   |    DB2 Communication Support - SNA                                 |   |
|   |                                                     [ More... ]    |   |
|   +--------------------------------------------------------------------+   |
|                                 [ Continue ]                                 |
+------------------------------------------------------------------------------+
```

Figure 21. Summary Report screen for the installation of the code

The navigation here is a bit tricky:

- Select **Summary Report**.

  Now a highlighted + sign should appear in the lower right corner, near the *More...* string.
- Now you can use PageDown and PageUp to scroll the list.

  Notice that the last entry in the summary report indicates the log file for the db2setup utility:

  The log file is in /tmp/db2setup.log.
- If the selections are not correct, press the F3 key to return to the previous screen.
- If the selections are correct, select **Continue** and press Enter.

15.The warning shown in Figure 22 on page 92 appears:

```
|   |          +--- Warning -----------------------------------+          |   |
|   |  Adminis|                                                |          |   |
|   |         |    (X) This is your last chance to stop.       |          |   |
|   |  Note:  |                                                |          |   |
|   |         |        Select OK to start, or Cancel to abort. |          |   |
|   |  * The l|                                                |          |   |
|   |         |    [   OK   ]                    [ Cancel ]    |          |   |
|   |          +-----------------------------------------------+          |   |
```

Figure 22. Warning screen before unpacking the installation images

Press Enter to start the installation.

WARNING!: You have been pressing the Enter key a lot, but avoid pressing it in the next sub-windows. If you do press it, you will stop the installation process!

16. The **Installing...** screen in Figure 23 on page 92 pops up, indicating that the installation is in progress.

    WARNING!: ***Do not press Enter*** or you will stop the installation! Wait for the process to complete.

```
|   |          +--- Installing... -------------------------------+          |   |
|   |  Adminis|                                                  |          |   |
|   |         |    DB2 Run-time Environment                      |          |   |
|   |  Note:  |                                                  |          |   |
|   |         |        [ Cancel ]                                |          |   |
|   |          +-------------------------------------------------+          |   |
```

Figure 23. Installation progress screen

17. When the installation process has completed, you see the window shown in Figure 24 on page 92.

```
|   |  Administration S+--- Notice --------------------+          |   |
|   |                  |                                |          |   |
|   |  Note:           |    Completed successfully.     |          |   |
|   |                  |                                |          |   |
|   |  * The log file i|          [   OK   ]            |          |   |
|   |                   +-------------------------------+          |   |
```

Figure 24. Notice of Completion of the Installation

Press **OK** to see the Status Report window.

18.Use the Up or Down arrow keys to review the Status Report shown in Figure 25 on page 93. Ensure that the status of all items is reported as SUCCESS.

```
+------------------------------ DB2 Installer --------------------------------+
|                                                                             |
|   +-- Status Report ---------------------------------------------------+    |
|   |                                                                    |    |
|   |                                                                    |    |
|   |  Installation                                                      |    |
|   |  ------------                                                      |    |
|   |                                                                    |    |
|   |    DB2 Client                                            SUCCESS   |    |
|   |    DB2 Run-time Environment                              SUCCESS   |    |
|   |    DB2 Engine                                            SUCCESS   |    |
|   |    DB2 Communication Support - TCP/IP                    SUCCESS   |    |
|   |    Administration Server                                 SUCCESS   |    |
|   |    DB2 Communication Support - SNA                       SUCCESS   |    |
|   |    DB2 Communication Support - DRDA Application Server   SUCCESS   |    |
|   |    DB2 Communication Support - IPX/SPX                   SUCCESS   |    |
|   |                                                     [ More... ]   |    |
|   +--------------------------------------------------------------------+    |
|   [ View Log ]                                               [   OK   ]     |
+-----------------------------------------------------------------------------+
```

Figure 25. Status Report Screen

19.Select **View Log** to view the installation log file (Figure 26 on page 94).

```
+------------------------------ DB2 Installer --------------------------------+
|                                                                             |
|   +-- View Log File ---------------------------------------------------+   |
|   |                                                                    |   |
|   |                                                                    |   |
|   |  DB2 Installer Log File                                            |   |
|   |  ----------------------                                            |   |
|   |                                                                    |   |
|   |  Log started at Fri Mar 27 16:38:59 1998                           |   |
|   |                                                                    |   |
|   |  Command to be executed:                                           |   |
|   |                                                                    |   |
|   |  /usr/sbin/installp -acgqX -d /cdrom/db2/ db2_05_00.client         |   |
|   |                                                                    |   |
|   |  Output log of the above command:                                  |   |
|   |  +---------------------------------------------------------------- |   |
|   |        Pre-installation Verification...                            |   |
|   |                                                       [ More... ]  |   |
|   +--------------------------------------------------------------------+   |
|                                  [   OK   ]                                 |
+-----------------------------------------------------------------------------+
```

Figure 26. View Log File Screen

Select **OK** to close the **View Log File** screen.

Select **OK** to close the **Status Report** screen.

20. The DB2 Installer will perform some cleanup and then you will then see the DB2 Installer screen

21. Select **Close** from the **DB2 Installer** screen to terminate the DB2 Installer.

    Ignore these two warning messages:

    DBI1756W DB2 Instance is not created.

    DBI1755W The Administration Server is not created.

    Select **OK** and press Enter for all the remaining screens, to exit from the DB2 Installer.

22. Change the directory away from `/cdrom`, so you can unmount the CD-ROM:

    cd

    umount /cdrom

7.5.8 Creating the DB2 Instances

After successful installation of the code for DB2 UDB V5, you can create the DB2 instance and the DB2 Administration Server.

Notice that the DB2 Administration Server is a special DB2 instance and it should be the only one of its type in the host. Thus, if you come to this section to create another normal DB2 instance, and if you already have created a DB2 Administration Server, make sure that you do not create another DB2 Administration Server.

The steps are these:

1. Login as root
2. The DB2 Installer is now available in `$DB2_HOME/install`, so there is no reason to mount the CD-ROM again. Thus, depending on your UNIX system, execute the following command:

   For AIX, `/usr/lpp/db2_05_00/install/db2setup`

   For HP-UX, `/opt/IBMdb2/V5.0/install/db2setup`

   For Solaris, `/opt/IBMdb2/V5.0/install/db2setup`
3. You will see the **DB2 Installer** screen shown in Figure 16 on page 86. This time, select **Create...** in order to create a DB2 Instance or the Administration Server.

```
+------------------------------- DB2 Installer --------------------------------+
|                                                                              |
|   Select Install to select products and their components to install, or     |
|   select Create to create the DB2 services.                                  |
|                                                                              |
|   To select products and their components, select          [ Install... ]    |
|   Install.                                                                   |
|                                                                              |
|   To create a DB2 Instance, or the Administration          [ Create...  ]    |
|   Server, select Create.                                                     |
|                                                                              |
|   [  Close  ]                                                  [  Help  ]    |
+------------------------------------------------------------------------------+
```

Figure 27. DB2 Installer Screen - Create Instances

4. From the **Create DB2 Services** screen shown in Figure 28 on page 96, select **Create a DB2 Instance**.

```
+--------------------------- Create DB2 Services ----------------------------+
|                                                                            |
|   Select the items you want to create, and select OK when finished.        |
|                                                                            |
|   A DB2 Instance is an environment where you store data and run            |
|   applications. An instance can contain multiple databases.                |
|                                                                            |
|   [*] Create a DB2 Instance.                           : Customize... :    |
|                                                                            |
|   An Administration Server provides services to support client tools that  |
|   automate the configuration of connections to DB2 databases.              |
|                                                                            |
|   [ ] Create the Administration Server.                : Customize... :    |
|                                                                            |
|   [   OK   ]                [ Cancel ]                      [  Help  ]     |
+----------------------------------------------------------------------------+
```

Figure 28. Create DB2 Services Screen, Snapshot 1

If this is your first time, pressing the space bar to select the item automatically takes you to the **DB2 Instance screen** shown in Figure Figure 29 on page 97.

5. We recommend that you take the defaults in the **DB2 Instance** screen, although you may want to modify the password.

   Select only **Auto start DB2 Instance**.

   Do not select the option to create a sample database. We will create that after the DB2 Instance has been successfully created. For details, see “Verification of DB2 Configuration” on page 105

   If you are using TCP/IP, you need not customize it, although the corresponding screen is shown in Figure 31 on page 98. Press Enter to continue.

```
|+--- DB2 Instance ------------------------------------------------------------+|
||                                                                             ||
||   Authentication:                                                           ||
||       Enter User ID, Group ID and Password that will be used for            ||
||       the DB2 Instance.                                                     ||
||       User Name           [db2inst1]                                        ||
||       User ID             :        :                [*] Use default UID     ||
||       Group Name          [db2iadm1]                                        ||
||       Group ID            :        :                [*] Use default GID     ||
||       Password            [********        ]                                ||
||       Verify Password     [********        ]                [ Default ]     ||
||                                                                             ||
||   Protocol:                                                                 ||
||       Select Customize to change the default           [ Customize... ]     ||
||       communication protocol.                                               ||
||                                                                             ||
||   [*] Auto start DB2 Instance at system boot.                               ||
||   [ ] Create a sample database for DB2 Instance.                            ||
||                                                                             ||
||   [   OK   ]                 [ Cancel ]                     [  Help  ]      ||
|+-----------------------------------------------------------------------------+|
```

Figure 29. DB2 Instance screen

6. In case you do want to customize the default communication protocol, or if you are just curious, then select **Customize** and you will see the **DB2 Instance Protocol** screen shown in Figure 30 on page 98.

   Because TCP/IP is used, the sub-windows in Figure 31 on page 98 is also shown.

   Note: When showing subwindows (windows inside another window), the outer window is not shown completely.

```
+--- DB2 Instance Protocol ------------------------------------------+
|                                                                    |
|   Select protocols and then select Properties to modify the        |
|   protocol values.                                                 |
|                                                                    |
|   [*] TCP/IP   Detected                        [ Properties... ]   |
|   : : IPX/SPX                                  : Properties... :   |
|                                                                    |
|   [   OK   ]              [ Cancel ]              [  Help  ]       |
+--------------------------------------------------------------------+
```

Figure 30. DB2 Instance Protocol Screen

Then by selecting **Properties** and pressing Enter, the sub-sub-window in Figure 31 on page 98 is shown.

```
+--- +--- TCP/IP -----------------------------------------------+----+
|    |                                                          |    |
|   S|   Enter the Service Name and Port Number that will       |    |
|   p|   be used for TCP/IP connection.                         |    |
|    |                                                          |    |
|   [|   Service Name        [db2cdb2inst1  ]                   |]   |
|   :|   Port Number         [50000]            [ Default ]     |:   |
|    |                                                          |    |
|   [|   [   OK   ]          [ Cancel ]          [  Help  ]     |]   |
+----+----------------------------------------------------------+----+
```

Figure 31. Customization of TCP/IP Protocol Screen

Press Enter until you return to the screen **Create DB2 Services**, **DB2 Instance**. Then press **OK** to continue.

7. From the **User-Defined Functions** screen in Figure 32 on page 99, accept the defaults, although you may want to change the password.

   The Fenced User Name and Group Name are used for user-defined functions (which need to be fenced in a separate account, away from the actual account of the DB2 instance). So far, they are not supported by TeamConnection and there are no plans to support them, but it is a good idea to go ahead and create this user ID because they are needed for the creation of the DB2 instance. It will not hurt to have this user ID (and then to ignore it). Press Enter.

```
|+--- User-Defined Functions -------------------------------------------------+|
||                                                                            ||
||   Fenced User-Defined Functions enable application developers to           ||
||   create their own suite of functions specific to their application        ||
||   or domain.                                                               ||
||                                                                            ||
||   Authentication:                                                          ||
||       Enter User ID, Group ID and Password that will be used for           ||
||       the fenced User-Defined Functions.                                   ||
||       User Name           [db2fenc1]                                       ||
||       User ID             :        :              [*] Use default UID      ||
||       Group Name          [db2fadm1]                                       ||
||       Group ID            :        :              [*] Use default GID      ||
||       Password            [                ]                               ||
||       Verify Password     [                ]              [ Default ]      ||
||                                                                            ||
||   Note: It is not recommended to use the DB2 Instance user ID for          ||
||         security reasons.                                                  ||
||                                                                            ||
||   [   OK   ]                  [ Cancel ]                   [  Help  ]      ||
|+----------------------------------------------------------------------------+|
```

Figure 32. User-Defined Functions Screen

8. You are returned to the **Create DB2 Services** screen.

   We strongly recommend that you also create the DB2 Administration Server (DAS), because it will allow your UNIX database to be administered remotely by the DB2 Control Center from an Intel platform. Even if you do not plan to use the DB2 Control Center now, your plans may change in the future and by installing it now you will be prepared. Also, if a UNIX version for the DB2 Control Center is ever made available, you will be ready.

   The administration of a DB2 instance and its databases is greatly simplified by using the DB2 Control Center.

   Select the item **Create the Administration Server**. As soon as you press the space bar, you will be taken to the next screen (see Figure 33 on page 100).

```
+---------------------------- Create DB2 Services -----------------------------+
|                                                                              |
|   Select the items you want to create, and select OK when finished.          |
|                                                                              |
|   A DB2 Instance is an environment where you store data and run              |
|   applications. An instance can contain multiple databases.                  |
|                                                                              |
|   [*] Create a DB2 Instance.                              [ Customize... ]   |
|                                                                              |
|   An Administration Server provides services to support client tools that    |
|   automate the configuration of connections to DB2 databases.                |
|                                                                              |
|   [*] Create the Administration Server.                   [ Customize... ]   |
|                                                                              |
|                                                                              |
|   [   OK   ]                  [ Cancel ]                        [  Help  ]   |
+------------------------------------------------------------------------------+
```

Figure 33. Create DB2 Services Screen, Snapshot 2.

9. From the **Administration Server** screen shown in Figure 34 on page 101, accept the defaults, although you may want to change the password. Select **OK** and press Enter.

```
|+--- Administration Server ------------------------------------------------+|
||                                                                          ||
||   Authentication:                                                        ||
||       Enter User ID, Group ID and Password that will be used for         ||
||       the Administration Server.                                         ||
||       User Name           [db2as   ]                                     ||
||       User ID             :        :               [*] Use default UID   ||
||       Group Name          [db2asgrp]                                     ||
||       Group ID            :        :               [*] Use default GID   ||
||       Password            [                ]                             ||
||       Verify Password     [                ]               [ Default ]   ||
||                                                                          ||
||   Protocol:                                                              ||
||       Select Customize to change the default            [ Customize... ] ||
||       communication protocol.                                            ||
||                                                                          ||
||   Note: It is not recommended to use the DB2 Instance user ID for        ||
||         security reasons.                                                ||
||                                                                          ||
||   [   OK   ]                  [ Cancel ]                   [  Help  ]    ||
|+--------------------------------------------------------------------------+|
```

Figure 34. Administration Server Screen

10..A small sub-windows (Figure 35 on page 101) appears, indicating that your host name is assigned to the variable DB2SYSTEM. Press Enter.

```
||       Group I+--- Notice -----------------------------------+default GID   ||
||       Passwor|                                               |              ||
||       Verify |   (!) DB2SYSTEM will be set to "oem-ppc3".    |[ Default ]   ||
||              |                                               |              ||
||   Protocol:  |                  [   OK   ]                   |              ||
||       Select +-----------------------------------------------+tomize...    ||
```

Figure 35. Notice Screen about DB2SYSTEM

11..You are returned to the **Create DB2 Services** screen.

Select **OK** and press Enter.

12.A summary report, shown in Figure Figure 36 on page 102 appears.

```
+------------------------------ DB2 Installer ------------------------------+
|                                                                           |
|   +-- Summary Report ----------------------------------------------------+  |
|   |                                                                    |  |
|   |  DB2 Services Creation                                             |  |
|   |  ---------------------                                             |  |
|   |                                                                    |  |
|   |  DB2 Instance                                                      |  |
|   |                                                                    |  |
|   |    Group Name                                        db2iadm1      |  |
|   |    User Name                                         db2inst1      |  |
|   |    Password                                            ibmdb2      |  |
|   |    Service Name                                  db2cdb2inst1      |  |
|   |    Port Number                                          50000      |  |
|   |    Update DBM Configuration file for TCP/IP                        |  |
|   |                                                                    |  |
|   |  User-Defined Functions                                            |  |
|   |                                                    [ More... ]     |  |
|   +--------------------------------------------------------------------+  |
|                                [ Continue ]                               |
+---------------------------------------------------------------------------+
```

Figure 36. Summary Report Screen for the Configuration Tasks

13.If the selections are not correct, press F3 to return to the previous screen.

14.If the selections are correct, select **Continue** and press Enter.

15.You now see the warning shown in Figure 37 on page 102.

```
|   |         +--- Warning -----------------------------------+        |   |
|   |  Adminis|                                               |        |   |
|   |         |   (X) This is your last chance to stop.       |        |   |
|   |  Note:  |                                               |        |   |
|   |         |       Select OK to start, or Cancel to abort. |        |   |
|   |  * The l|                                               |        |   |
|   |         |   [   OK   ]                    [ Cancel ]    |        |   |
|   |         +-----------------------------------------------+        |   |
```

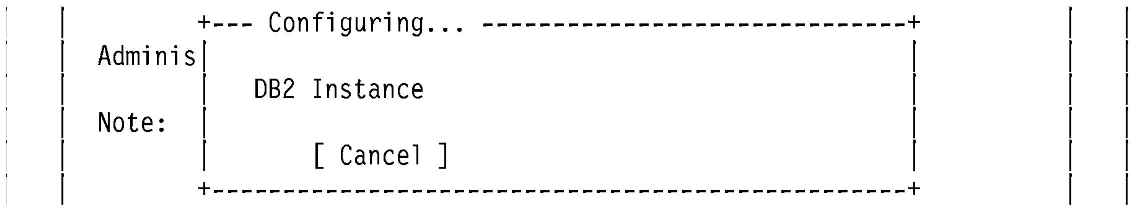
Figure 37. Warning Screen before Configuring the DB2 Instance

Press Enter to start the configuration.

WARNING!: Avoid pressing the Enter key in the next subwindows, because pressing it stops the configuration process!

16.The **Configuring...** screen in Figure 38 on page 103, which indicates the progress of the configuration, appears.

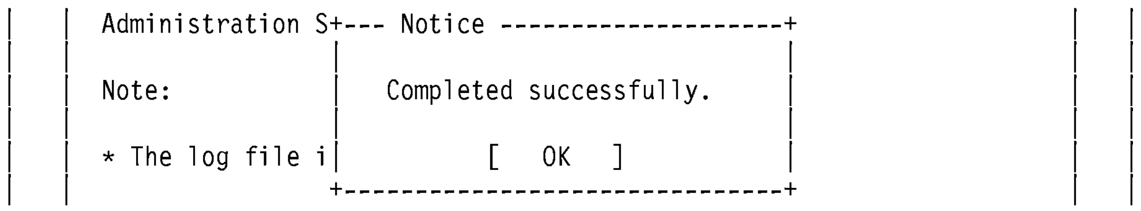
WARNING!: Again, avoid pressing the Enter key here and in the next subwindows. Pressing it stops the configuration process. Wait for the configuration to finish.



```
 |         +--- Configuring... -----------------------------+          |   |
 |  Adminis|                                                |          |   |
 |         |   DB2 Instance                                 |          |   |
 |  Note:  |                                                |          |   |
 |         |       [ Cancel ]                               |          |   |
 |         +------------------------------------------------+          |   |
```

Figure 38. Configuration Progress Screen

17.When the installation process is completed, you see the window shown in Figure 39 on page 103.



```
|   |  Administration S+--- Notice --------------------+          |   |
|   |                  |                                |          |   |
|   |  Note:           |   Completed successfully.      |          |   |
|   |                  |                                |          |   |
|   |  * The log file i|          [   OK   ]            |          |   |
|   |                  +--------------------------------+          |   |
```

Figure 39. Notice Screen –Completion of the Configuration of Instances

Press **OK** to see the next window with the status report.

18.When the installation is complete, use the Up or Down arrow keys to review the status report.

Ensure that all the items show a status of SUCCESS.

Select **OK** to continue.

19.The DB2 Installer will perform some cleanup and then you will see the **DB2 Installer** screen.

20.Select **Close** from the **DB2 Installer** screen to terminate the DB2 Installer.

Select **OK** to exit.

If this is the first time a DB2 Instance is created, then continue with the post-installation tasks mentioned in “post-installation Tasks” on page 104.

If this is the subsequent creation of another DB2 Instance, then there is no need to perform the post-installation tasks.

7.6 post-installation Tasks

7.6.1 Apply the Fixpak and Interim Fixes for DB2 UDB

It is necessary to apply the latest FixPak and interim fixes for DB2 UDB for your platform. For more details, see “Apply the Fixpak and Interim Fixes for DB2 UDB” on page 104

7.6.2 Create a VisualAge TeamConnection Family

Now that you have created a DB2 Instance, you can create a VisualAge TeamConnection family. For details, see “Creation of a Family Using the New DB2 Instance” on page 121. Then, after the family is created, return to this section to continue with the post-installation tasks. If you are following the instructions to use a dedicated file system for the DB2 Instance and the VisualAge TeamConnection family, then proceed with “Changing Ownership and File Permissions of Home Directories” on page 104 after you have created the VisualAge TeamConnection family user ID.

7.6.3 Changing Ownership and File Permissions of Home Directories

If you are following the instructions to use a dedicated file system for both the DB2 Instance and the VisualAge TeamConnection family, then continue with this section after you have created the VisualAge TeamConnection family. Perform the following steps to change the ownership and the file permissions of the home directories in `/home2`, where the physical directories are located.

1. Log in as root.
2. Verify the current settings in `/home2`, where the physical directories are located:

   cd /home2

   ls -dl /home2/db2\

   ls -dl /home2/tcfamily

At this point, the owner should be root. If not—that is, if the owners are the appropriate user IDs—then terminate this section now.

3. Modify the ownership:

   chown db2fenc1:db2fadm1 db2fenc1

   chown db2inst1:db2iadm1 db2inst1

chown tcfamily:db2iadm1 tcfamily

4. If you want, you can modify the file permissions:

   chmod 755 db2fenc1

   chmod 755 db2inst1

   chmod 755 tcfamily

5. Verify the new settings: ensure that the owners are those with the appropriate user IDs.

Notice that there is no need to modify the ownership of the symbolic links in `/home`.

7.6.4 Update the Profile for the DB2 Instances

The default profiles used by the DB2 Instance and the DB2 Administration Server do not set the complete operating environment because the DB2 support group did not want to assume which UNIX shell was being used (such as Bourne/Korn or C, which have different syntax). Instead, you have to invoke an additional profile manually. However, in order to simplify the handling of the commands for the DB2 instances, we suggest the following change in the .profile to set up the complete environment automatically upon login (using the profile for the Korn shell):

1. Edit the .profile.
2. Add the following line after the line "export PATH":

   ../sqllib/db2profile

3. Save and exit the .profile.

Note: By the way, the DB2 Administration Server is a special DB2 instance; therefore, you may want to also make the change mentioned above into its profile.

7.6.5 Verification of DB2 Configuration

Warning for Solaris: It is necessary to apply FixPak 4 (or later) for DB2 UDB for Solaris in order to overcome a problem (core dump) when issuing the command `db2sampl`. For more details, see “Apply the Fixpak and Interim Fixes for DB2 UDB” on page 104.

It is a good idea to create the DB2 Sample Database for the following reasons:

- To verify that you have installed and configured DB2 correctly.

- To practice your DB2 administration commands, such as `backup` and `restore`. In this way, you can experiment without the fear of damaging the real VisualAge TeamConnection database.
- To have a practice database for the DB2 Control Center.

To create the DB2 Sample database, do the following:

1. Log in as the DB2 instance owner (such as db2inst1).
2. Enter the following from a command prompt to create the database with alias SAMPLE that will take around 10 MB of hard disk. This command stands alone; it does not work if you run it within the DB2 command line processor:

   ```
   $ db2sampl
   ```

This command may take a few minutes to process. There is no completion message— when the command prompt returns, the sample database has been created.

To remove the DB2 Sample database, type the following as the instance owner:

```
$ db2 drop database sample
```

7.6.6 Verify the License Information in the Nodelock File

When DB2 is installed using the TeamConnection Installer utility (tcinst.ksh), the utility automatically updates the nodelock file in your system with the proper licensing information for DB2 UDB.

If you install DB2 UDB by itself, however, the proper licensing information is not automatically added to the nodelock file.As a result, when you manually stop (`db2stop`) and restart (`db2start`) a DB2 instance, you will see this warning message.

```
SQL8007W There are "xx" day(s) left in the evaluation
period...
```

Furthermore, after this evaluation period ends, your DB2 instance does not start at all. In case you have this situation, you must add the licensing information into the nodelock file. Once this information is added to the nodelock file, there are no more warning messages and your license will expire in the year 2047. Here are the necessary instructions to do so:

- If you installed DB2 UDB as part of the installation of VisualAge TeamConnection, then the following action is performed to append the licensing information into the appropriate nodelock file:

  For AIX:

  ```
  mkdir -p /var/ifor
  touch /var/ifor/nodelock
  cat $CDROM/aix4/db2/db2/license/db2work.nod >> /var/ifor/nodelock
  ```

  For HP-UX:

  ```
  mkdir -p /usr/netls
  touch /usr/netls/nodelock
  cat $CDROM/hpux1ð/db2/db2/license/db2work.nod >>
  /usr/netls/nodelock
  ```

  For Solaris:

  ```
  mkdir -p /var/netls
  touch /var/netls/nodelock
  cat $CDROM/solaris/db2/db2/license/db2work.nod >>
  /var/netls/nodelock
  ```

  where CDROM is the directory where you mounted the CD-ROM for TeamConnection.

- If you purchase DB2 UDB and you install from its CDROMs, then you will need to follow the instructions mentioned in the relevant section of the manual *DB2 Quick Beginnings for UNIX*.

Chapter 8. Installation of DB2 for Windows

8.1 Overview

Note: When installing the family server of VisualAge TeamConnection, the necessary DB2 UDB code is also installed, by invoking (in silent mode) the DB2 installation utility with a predefined response file. Thus, you do not have to manually install DB2 in your machine. However, in case you bought DB2 UDB and decided to install it in your machine, the following instructions are provided.

After completing the installation of DB2 according to these instructions, you will have done the following:

1. Installed DB2 UDB Personal Edition, including the appropriate NLS-related files and on-line documentation.
2. Created the appropriate groups and shortcuts.
3. Registered the DB2 Security Server, which you can start or stop by: `Start -> Settings -> Control Panel -> Services.`
4. Updated the appropriate configuration files and the Windows registry.
5. Created a normal DB2 instance (this is where the database for TeamConnection will be created later on by the TeamConnection Administration tool).
6. Created a DB2 Administrator Server (DAS) instance, which is used to allow remote administration by means of the DB2 Control Center running on an Intel workstation.
7. Activated the DB2 First Steps following the first reboot after installation.
8. Created a DB2 sample database. You can use this database as your sand box to practice some of the administration tasks, without affecting the database used by TeamConnection.

8.2 Planning and Prerequisites

For more details on the planning for installation of DB2 UDB V5 in Windows NT, such as hardware and software requirements, consult Chapter 4 "Planning for Installation" of the manual *DB2 Personal Edition Quick Beginnings* which came with the software.

The hardware and software requirements for VisualAge TeamConnection already take into account the corresponding requirements for DB2 UDB, and they are described in the *VisualAge TeamConnection Administrator's Guide.*

8.3 The Full Path Where DB2 Is Installed

The full path (DB2_HOME) where DB2 will be installed is shown below. Thus, you need to ensure that you have enough available space in the appropriate disk units.

C:\SQLLIB for example

8.4 Space Used During the Installation of DB2

After performing the installation described in this document, the following disk space was actually used:

- 10 MB in C:\DB2, for the sample database.
- 10 MB in C:\IFOR, for the handling of run-time nodelock licenses.
- 130 MB in C:\SQLLIB, for the actual code and the two user IDs (the DB2 instance, and the DB2 Administrator Server instance).

8.5 Uninstalling ObjectStore

In case you have installed ObjectStore, the database used by VisualAge TeamConnection Version 2, then you could deinstall this database management system in order to recover disk space. You can deinstall ObjectStore after you have migrated your TeamConnection family from Version 2 to Version 3. To deinstall ObjectStore in Windows NT, do the following:

1. Ensure that the TeamConnection servers are not running. If they are running, shut them down.
2. Ensure that the ObjectStore servers are not running. If they are, shut them down.
3. Select **Start -> Settings -> Control Panel -> Add/Remove Programs.**
4. From the list of installed programs, select **ObjectStore** and click on the **Add/Remove** button.
5. Reboot the workstation for the changes to the configuration files (registry) to take effect.

8.6 Installation Steps

To install the DB2 products on Windows NT, perform the following steps:

1. Identify and record parameter values. See “Identify and Record Parameter Values” on page 111.
2. Create a DB2 administration ID. See “Create a DB2 Administration ID in Windows NT” on page 112.
3. Install DB2 UDB Personal Edition. See “Install DB2 UDB Personal Edition in Windows NT” on page 112.

8.6.1 Identify and Record Parameter Values

Note: We recommend that you change the passwords of the DB2 user IDs.

After completing these installation instructions, the values of all the appropriate parameters used during this procedure are set.

The parameters for the normal DB2 instance are as shown in Table 24 on page 111

Table 24. Parameters for the DB2 Instance

Parameter	Description
Full path name	c:\sqllib
User Name	db2admin
Password	db2admin
TCP/IP Connection Service Name	db2cDB2
TCP/IP Connection Port Number	50000
TCP/IP Interrupt Service Name	db2iDB2
TCP/IP Interrupt Port Number	50001

The parameters for the DB2 Administration Server (DAS) are shown in Table 25 on page 111.

Table 25. .Parameters for the DB2 Administration Server

Parameter	Description
Full path name	c:\sqllib
User Name	db2das00
TCP/IP service name	db2cDB2admi

Parameter	Description
TCP/IP Port Number	523

8.6.2 Create a DB2 Administration ID in Windows NT

You need to have a user name that will be used to install DB2. The user name must belong to the Administrators group, and must also be a valid DB2 user name or have the right of an advanced user to act as part of the operating system.

A valid DB2 user name is eight characters or fewer, and complies with DB2's naming rules. Thus, the usual Windows NT login of "Administrator" is ***not*** valid.

To create a new user ID, db2admin, to install and administer DB2, do the following:

1. Log in as Administrator.
2. Select **Start** -> **Programs** -> **Administrative tools (common)** -> **User Manager**
3. Create a user ID, db2admin, that belongs to the Administrator group. For a new user, db2admin, by default, is only a member of the Users group. In order to make it belong to the Administrator group, do the following:

   a. Select the button **Groups** in the lower left corner.

   b. Specify that the new user should be part of the Administrator group.

8.6.3 Install DB2 UDB Personal Edition in Windows NT

The steps are these:

1. Log off as Administrator and log in as db2admin.
2. Shut down any other programs in order that the setup program can update files as required.
3. Insert the CD-ROM into the drive. The auto-run feature automatically starts the setup program.

   If auto-run is disabled, then you can manually invoke the setup program as follows:

   a. Click on the **Start** button and select **Run**.

   b. Type the following in the **Open** field:

   ```
   x:\setup /i=LANGUAGE
   Where
   ```

```
x: represents your CD-ROM drive
LANGUAGE represents the two-character country code for your
language (for example, EN for English).
```

c. Select **OK**

4. The **Welcome** window opens.
5. Click on the **Next** button to open the **Select Products** window.
6. Select **DB2 Universal Database Personal Edition** and click on the **Next** button.
7. Select **Custom Install** (to see what is going to be installed in your system). We suggest that you accept these:

    Graphical tools

    Documentation

    You can select a new disk drive to install the code. Please keep the directory SQLLIB.

    Then click on **Next**.
8. It is very likely that you will get the following warning message:

    ```
    Setup is unable to validate the password.
    Setup will continue using the user name and password provided.
    ```

    Just ignore this message. We could not find any way to avoid it and ignoring it did not cause any problem.
9. Respond to the setup program’s prompts. On-line help is available to walk you through the remaining steps. Invoke on-line help by clicking on the **Help** button at any time. You can click on the **Cancel** button at any time to end the installation.
10. After you install the product, you must reboot your workstation before you can begin to use it. Select a reboot option and click on the **Finish** button. This completes the installation.
11. When your system restarts, log on with the db2admin user name.
12. The DB2 First Steps tool executes automatically (but only after the first reboot when the installation is completed). Use First Steps to create the sample database.
13. (Optional) It is important to ensure that you can work with the SAMPLE database. From the DB2 First Steps, do the following, in sequence:

    a. Log on to an administrative user ID.

    b. Create a SAMPLE database.

    c. View the SAMPLE database (which brings up the Command Center).

   d. Work with the SAMPLE database (from the Control Center).

      To invoke DB2 First Steps at a later time, select

      Start -> DB2 for Windows NT -> First Steps

14. After installing DB2, run the following DB2 command to correctly set the CPU speed that DB2 uses when calculating the optimal access plan for queries:

    db2 "update dbm cfg using cpuspeed -1"

8.6.4 Apply the Fixpak and Interim Fixes for DB2 UDB

It is necessary to apply the latest Fixpak and interim fixes for DB2 UDB for your platform.

Chapter 9. Installation of DB2 for OS/2

9.1 Overview

Note: When installing the family server of VisualAge TeamConnection, the necessary DB2 UDB code is also installed, by invoking (in silent mode) the DB2 installation utility with a predefined response file. Thus, you do not have to manually install DB2 in your machine. However, in case you bought DB2 UDB and decided to install it in your machine, the following instructions are provided.

After completing the installation of DB2 according to these instructions, you will have done the following:

1. Installed DB2 UDB Personal Edition, including the appropriate NLS related files and on-line documentation.
2. Created the appropriate folders.
3. Updated the appropriate configuration files.
4. Created a normal DB2 Instance. This is where the database for TeamConnection will be created later on by the TeamConnection Administration tool.
5. Created a DB2 Administrator Server (DAS) instance which is used to allow remote administration by means of the DB2 Control Center running on an Intel workstation.
6. Activated DB2 First Steps following the first reboot after installation.
7. Created a DB2 Sample database. You can use this database as your sand box to practice some of the administration tasks, without affecting the database used by TeamConnection.

9.2 Planning and Prerequisites

For more details on the planning for installation of DB2 UDB V5 in OS/2, such as hardware and software requirements, consult Chapter 4 "Planning for Installation" of the manual *DB2 Personal Edition Quick Beginnings*.

The hardware and software requirements for VisualAge TeamConnection already take into account the corresponding requirements for DB2 UDB, and they are described in the *VisualAge TeamConnection Administrator's Guide.*

9.2.1 The Full Path Where DB2 Is Installed

The full path (DB2_HOME) where DB2 will be installed is shown below. Thus, you need to ensure that you have enough available space in the appropriate disk units.

For OS/2, use `C:\SQLLIB`

9.2.2 Space Used for Installation of DB2

After we installed DB2 as described in this document, the following disk space was actually used:

- 10 MB in `C:\DB2`, for the Sample database.
- 10 MB in `C:\IFOR`, for the handling of run-time nodelock licenses.
- 130 MB in `C:\SQLLIB,` for the actual code and the two user IDs (the DB2 instance, and the DB2 Administrator Server instance).

9.3 Uninstalling ObjectStore

If you have ObjectStore installed—the database used by VisualAge TeamConnection Version 2— you can uninstall this database management system in order to recover disk space. You can uninstall ObjectStore after you have migrated your TeamConnection family from Version 2 to Version 3. To uninstall ObjectStore in OS/2, do the following:

1. Ensure that the TeamConnection servers are not running. If they are running, then shut them down.
2. Ensure that the ObjectStore servers are not running. If they are, shut them down. To do so,
   a. Open `ObjectStore for OS/2` folder, select **ObjectStore setup**.
   b. Select **Shutdown Services**.
   c. Select **Uninstall**.
3. Once the code is uninstalled, do the following:
   ```
   cd c:\ostore\bin
   del ossetup.exe
   del oscp437.dll
   cd \
   rmdir c:\ostore\bin
   rmdir c:\ostore
   ```
4. Reboot to activate the changes in CONFIG.SYS.

9.4 Installation Steps

To install the DB2 products on OS/2, perform the following steps:

1. Identify and record parameter values. See “Identify and Record Parameter Values” on page 117.
2. Create a DB2 administration ID. See “Create a DB2 Administration ID in OS/2” on page 118.
3. Install DB2 UDB Personal Edition: See “Install DB2 UDB Personal Edition in OS/2” on page 119.

9.4.1 Identify and Record Parameter Values

Note: We recommend that you change the passwords of the DB2 user IDs.

After completing these installation instructions, the values of all the appropriate parameters used during this procedure will be set.

The parameters for the normal DB2 instance are shown in Table 26 on page 117.

Table 26. Parameters for the DB2 Instance

Parameter	Description
Full path name	c:\sqllib
User Name	db2admin
Password	db2admin
TCP/IP Connection Service Name	db2cDB2
TCP/IP Connection Port Number	50000
TCP/IP Interrupt Service Name	db2iDB2
TCP/IP Interrupt Port Number	50001

The parameters for the DB2 Administration Server (DAS) are shown in Table 27 on page 117.

Table 27. Parameters for the DB2 Administration Server

Parameter	Description
Full path name	c:\sqllib
User Name	db2das00
TCP/IP service name	db2cDB2admi
TCP/IP Port Number	523

9.5 Create a DB2 Administration ID in OS/2

Before you begin the installation, be sure that you have the proper user ID and password. This user ID should have a local administrator or administrator authority in the User Profile Management (UPM). The DB2 Administration Server uses this user ID to log on when it is started.

If UPM is installed, then this user ID should have the authorities mentioned above. If necessary, create a user ID with the appropriate characteristics.

If UPM is not installed, DB2 will install it, and the user ID and password entered will be used to create a user ID with the correct authorities.

If you have LAN NetBIOS already installed, do the following to ensure that you will be able to log on after the installation:

1. From an OS/2 command prompt, log off from any possible LAN connection:

   c:> logoff

2. Do a logon to the LOCAL LAN domain and specify your user ID and password:

   c:> logon userid /p:password

   where the user ID and password values are the appropriate ones for your situation.

   Warning: If you cannot log in to your LOCAL LAN domain, ***stop***! You must fix this problem before you install DB2. If it is not fixed, you ***cannot*** use DB2 at all.

9.6 Setup of UPM

The UPM Services are located in:

`OS/2 System -> System Setup -> UPM Services`.

Follow the instructions in Chapter 13 of *DB2 Personal Edition Quick Beginnings*, to use UPM for the first time.

Note: If `c:\muglib\accounts` does not exist, then `c:\ibmlan\accounts\net.acc` will be used.

9.7 Install DB2 UDB Personal Edition in OS/2

The steps are these:

1. Log on as db2admin.
2. Shut down any other programs so that the setup program can update files as required.
3. Insert the CD-ROM into the drive.
4. Open an OS/2 window and set the drive to x:, where x is the letter that represents your CD-ROM drive.
5. Enter the command:

   x:\db2\language\install

   where

   `x:` represents your CD-ROM drive

   `language` represents the two-character country code for your language (for example, EN for English).
6. The **IBM DB2 for OS/2 Version 5 Installation** dialog opens.
7. Select **DB2 Universal Database Personal Edition**

   Once you select the option, `Operation Type` label `Install` is enabled. Click on **Continue**.
8. Answer the question if you want the installation program to update your `config.sys` file.
9. The **Install – Directories** window provides a list of the components to install. We suggest that you accept these:

   Graphical tools
   Documentation

   You can select a new disk drive on which to install the code. (Please keep the directory SQLLIB.) Then click on **Next**.
10. We recommend that you answer **Yes** to the question on auto-start Control Center.
11. The **Specify System Name** window opens. Specify the System Name for your host, which should be unique.
12. The **Customize Communication Protocols** window opens. The default DB2 instance is called *DB2*. Click on **customize**, then click on **Auto start DB2 instance at boot time**. For the Administration Server, click on **customize**. It uses the defaults for TCP/IP.

13. The **Enter User ID and Password** window opens. Use this to enter the user ID and password that will be used to log on and start the Administration Server each time your system is initialized.
14. The DB2 components are now installed on your system.
15. After you install the product, you must reboot your workstation before you can begin to use it. Select a reboot option and click on the **Finish** button. This completes the installation.
16. When your system restarts, log on with the db2admin user name.
17. The DB2 First Steps tool executes automatically (but only after the first reboot when the installation is completed). Use First Steps to create the sample database.
18. (Optional) Ensure that you can work with the SAMPLE database. From the DB2 First Steps, do the following:

    a. Log on with an administrative user ID.

    b. Create a SAMPLE database.

    c. View the SAMPLE database (which brings up the Command Center).

    d. Work with the SAMPLE database (from the Control Center).

    To invoke DB2 First Steps at a later time, search for the folder `DB2 for OS/2`, then open it and click on the First Steps icon.
19. After installing DB2, run the following DB2 command to correctly set the CPU speed that DB2 uses when calculating the optimal access plan for queries:

    db2 "update dbm cfg using cpuspeed -1"

9.8 Apply the Fixpak and Interim Fixes for DB2 UDB

It is necessary to apply the latest fixpak and interim fixes for DB2 UDB for your platform.

Chapter 10. Creation of a Family Using the New DB2 Instance

10.1 Overview of this Chapter

Note: In order to create the DB2 database of a VisualAge TeamConnection family, it is necessary to have at least 100 MB of free disk space in the file system where the DB2 instance resides.

After a DB2 instance has been created (see "Creating Instances" on page 145), you can create a VisualAge TeamConnection family which in turn will create a DB2 database using that DB2 instance.

Note about the primary group ID for the family

In the UNIX environment, in order for the DB2 instance to allow the creation of the DB2 database by the VisualAge TeamConnection family, the primary group ID of the instance (such as the default db2iadm1) MUST be either the primary group ID or part of the group set for the family user ID. If the family user ID does not have the proper group ID then when trying to create the family by using tcadmin (or the sample utility "dbcreate"), you will get a DB2 error message saying that you do not have enough authority to create a database.

If you want to use a dedicated file system for both the DB2 instance and the VisualAge TeamConnection family, see "Create a File System for the DB2 Instance and TC Family" on page 76.

You can perform the following sequence to create the VisualAge TeamConnection family:

1. Ensure that you have installed the auxiliary software for VisualAge TeamConnection:
   - Ÿ Acrobat Reader for PDF files.
   - Ÿ Java for tcadmin and tcmerge.
   - Ÿ Netscape Navigator for displaying the on-line help.

For details, see "Using TCADMIN to Create the Family Database" on page 122.

2. 2. Create the user ID for the family. See"Creation of the User Id for the Family" on page 123. If you are following the instructions for a dedicated file system, then you will need to change the ownership and the file permissions for the home directories, as shown in "Changing Ownership and File Permissions of Home Directories" on page 104.

3. 3. Copy and customize the sample profile, teamcv3x.ini and local Teamcgui file. See “Customization of Local Files (Profile, Teamcv3x.Ini and Teamcgui)” on page 124.
4. 4. Use the tcadmin tool to create the database for the family. See “Using TCADMIN to Create the Family Database” on page 122.

The parameters used in this chapter for the sample VisualAge TeamConnection family are:

Table 28. Parameters for the sample VisualAge TeamConnection family

Parameter	Description
Full path name	/home/tcfamily
User Name	tcfamily
UID	System-generated UID
Group Name	db2iadm1
GID	System-generated GID
Group Set	db2asgrp,staff
Password	<your password>
TCP/IP Port Number	4567

For more details on how to create group IDs and user IDs, and how to update the hosts and services files, see the technical report "VisualAge TeamConnection Version 3: how to do routine operating system tasks".

10.2 Using TCADMIN to Create the Family Database

Ensure that you have installed the auxiliary software for VisualAge TeamConnection:

- Acrobat Reader for PDF files.
- Java for tcadmin and tcmerge.
- Netscape Navigator for displaying the on-line help.

For details, see “Important Information about Installation” on page 62. If you do not install Java, then you cannot use the VisualAge TeamConnection Family Administration GUI (tcadmin).

10.3 Creation of the User Id for the Family

It is necessary to manually create a user ID for the VisualAge TeamConnection family:

1. Login as root.
2. Use the appropriate administration tool from your platform to create a user.

Note about the primary group ID for the family

In the UNIX environment, in order for the DB2 instance to allow the creation of the DB2 database by the VisualAge TeamConnection family, the primary group ID of the instance (such as the default db2iadm1) MUST be either the primary group ID or part of the group set for the family user ID.

If the family user ID does not have the proper group ID then when trying to create the family by using tcadmin (or the sample utility "dbcreate"), you will get a DB2 error message saying that you do not have enough authority to create a database.

3. In this chapter, it is assumed that the user ID that is created is tcfamily and its primary group ID is db2iadm1. You can login into the new user ID and issue the following command to verify its settings:

   ```
   $ id
   ```

   The output should look like this:

   ```
   uid=237(tcfamily) gid=250(db2iadm1)
   ```

   In case the primary id of the DB2 instance is not the primary id of the user for the family but it is part of its group set, then the output of the "id" command, should look like this:

   ```
   uid=237(tcfamily) gid=1(staff) groups=250(db2iadm1)
   ```

4. Modify the /etc/hosts file and add the family name in the entry for the desired host, such as:

   ```
   9.12.345.678 oem-ppc3 tcfamily
   ```

5. Modify the /etc/services file and add the port number for the family, such as:

   ```
   tcfamily 4567/tcp # VisualAge TeamConnection family
   ```

   If you are following the instructions for a dedicated file system, then you will need to change the ownership and the file permissions for the home directories, as shown in “Changing Ownership and File Permissions of Home Directories” on page 104.

10.4 Customization of Local Files (Profile, Teamcv3x.Ini and Teamcgui)

For the steps in this section, please notice the values for the following variables:

- $TC_HOME is by default located in:

Table 29. TC_HOME Location

Environment	Location
AIX	/usr/teamc
HP-UX	/opt/teamc
Solaris	/opt/teamc

- $LANG is the national language, such as en_US for English in the USA.

After the user ID has been created, you have to copy and customize the following set of files.

1. Login as the user ID for the family.
2. Copy the file that has the initial data for the Tasks window:

   ```
   cp $TC_HOME/nls/cfg/$LANG/teamcv3x.ini $HOME/.
   chmod u+w $HOME/teamcv3x.ini
   ```

3. (Optional) Would you like to customize your fonts, copy the sample local Motif resource file:

   ```
   cp $TC_HOME/nls/cfg/$LANG/Teamcgui.user $HOME/Teamcgui
   chmod u+w $HOME/Teamcgui
   ```

4. Copy the sample profile for a family:

   ```
   mv $HOME/.profile $HOME/.profile.original
   cp $TC_HOME/install/$LANG/profile.family $HOME/.profile
   chmod u+w $HOME/.profile
   ```

You will need to customize your new profile. Please see the instructions in the header of the file.

5. After you customize your profile, log out of your user ID and log in again in order to ensure that you will be working from a fresh environment.

It is NOT a good idea to cut corners and just simply execute the profile again. There are some variables, such as PATH, that append the new value to the previous value, and thus, you will not have a fresh environment to work with.

10.5 Using Default settings to Create the Family Database

In this section, a new VisualAge TeamConnection family will be created using the default settings. Only the required values that do not have defaults will be entered in this example.

1. Login as root.
2. Modify the hosts and services files to add the new family. In /etc/hosts, find the entry for your host, and add an alias for the family name, such as:

   9.37.199.98 oem-ppc3 tcfamily

   In /etc/services, add a new entry with a port number that is not currently assigned to another service, such as:

   tcfamily 3420/tcp
3. Login into the family user ID.
4. In UNIX, ensure that your DISPLAY variable is setup to the proper X server.
   - From a local host, you can use shared memory, which will be faster:

     Using Korn shell:

     export DISPLAY=:0.0

     Using Bourne shell:

     DISPLAY=:0.0

     export DISPLAY

     In Bourne shell you cannot combine the definition and the export in the same statement. The rest of this document will show only the syntax in Korn shell.
   - From remote hosts, if you are logged into a host A, then telnet to another host B, you need to use sockets (which are slower than shared memory):

     export DISPLAY=hostName:0.0

     It is important to note that the terminal where the window will be actually displayed (host A) needs to allow the use of the X server, by doing the following in host A:

```
xhost +
```

Note: If this DISPLAY variable is not properly set, the actual tcadmin window will not be displayed in your terminal (it will be displayed in someone else's terminal).

5. In UNIX, ensure that you can display correctly a X Windows application. You can try the following harmless application that will display a clock in a window (as a background process):

   ```
   xclock &
   ```

   If you do not see the xclock window, then STOP! This means that your DISPLAY variable is not setup correctly, and that you will NOT see the window from the tcadmin tool. You will need to fix the DISPLAY variable or issue the xhosts command to properly display an X Windows application. If you are still having problems, consult your UNIX manuals or local help desk.

6. To invoke the VisualAge TeamConnection Administration tool issue the command:

   ```
   tcadmin
   ```

   Notes:

   - In case of problems during the creation of the family, you can specify to create the file tcadmin.log that will contain the commands that tcadmin is using:

     ```
     tcadmin -log
     ```

   - To monitor the commands that are being written in tcadmin.log, issue the following command:

     ```
     tail -f tcadmin.log
     ```

     You need to press Ctrl+C to terminate this monitoring task.

7. If this is the first time this tool is invoked, then it will display a message saying that the settings file ($HOME/tcadmin.ini) could not be found. Just press OK to acknowledge this warning.

8. If you get the following error message:

   Unable to identify the directory containing the files required by tcadmin. Please verify your NLSPATH or use the -f parameter.

   Then press OK to exit.

   This error message is intermittent and so far, we have not found the cause. The work-around is to exit and start tcadmin again.

9. You will see the "TeamConnection Family Administrator" window.

Select File -> Create Family.

10. In the "Untitled - Properties" dialog, enter the new following information

    Name: tcfamily

    Path: /home

    Port: 3420

    Password: <enter your password>

    The tcadmin tool will use a physical directory that is the concatenation of the values of the "Path" field + the appropriate directory separator + the "Name" field, such as the UNIX sample "/home" + "/" + "tcfamily" = "/home/tcfamily". Based on this behavior, do not use the $HOME directory as the value for the Path field (such as /home/tcfamily), because tcadmin will append the Name field (such as tcfamily) and use /home/tcfamily/tcfamily (which is rather cumbersome).

11. Click on the OK button to start the process to create your family. This should take less than one hour. You will then see an information message that will indicate that the VisualAge TeamConnection family and its database were successfully created.

    Note: If after one hour tcadmin has not finished with the creation of family, then it is likely that there is not enough available space in the file system where the DB2 instance is located (the default is /home). If this is the case, kill the tcadmin process, drop the DB2 family database (issue "db2 drop database tcfamily"), expand the file system and try tcadmin again.

12. To start the family servers (teamcd and notifyd) and the activity monitor tool, select the family from the main tcadmin window, then click on Family in the menu bar and the "Family Servers" window will appear.

13. Click on the button "Start Both Servers" located in the lower right corner, to start all the servers.

14. Minimize the "Family Services" window, if you want. If you intend to work with your family, do not close this window, because if you close it, tcadmin will stop the servers.

10.6 Db2 Configuration Parameters when Creating a Family

When using tcadmin to create a VisualAge TeamConnection family, the corresponding DB2 database is created by the TeamConnection utility "fhcirt" which sets the following Database Configuration parameters to non-default values to improve database performance and functionality.

- Ÿ All platforms:

logfilsiz = 4000

applheapsz = 1280

logprimary = 5

logsecond = 30

buffpage = 12000

dlchktime = 1000

- Solaris:

  dbheap = 2400

- Intel:

  dbheap = 600

  catalogcache_sz = 32

  maxappls = 40

  locklist = 50

  app_ctl_heap_sz = 128

Note: These non-default values are based on our testing experience, and may evolve with the product life.

10.7 What To Do Next

Now that you have created a VisualAge TeamConnection family, these are some hints on what to do next:

- At this point you will be ready to start using the newly created VisualAge TeamConnection family.
- Configure the Control Center. See “Configuring the Db2 Control Center” on page 134.
- Find out the DB2 tools to administer databases. See “Administration of Databases with DB2 Tools” on page 139.
- Learn how to work with the DB2 instances. See “Working with DB2 Instances” on page 145.
- For UNIX, you can learn how to work with the DB2 administration server. See “Working with the DB2 Administration Server in UNIX” on page 153.

Chapter 11. Control Center Configuration

11.1 Overview of the Chapter

The scope of this chapter is to learn how you can configure the Control Center in order to gain control on remote UNIX databases. In order to be ready to proceed to the next chapter, “Administration of Databases with DB2 Tools” on page 139, it is necessary to configure the Control Center in an Intel workstation to be able to control remote databases in a UNIX server. The structure of the chapter is as follows:

- The relationship between the Control Center and the Administration Server is shown in “Relationship between Control Center and Administration Server.”
- If you do not have already the DB2 GUI Tools installed in an Intel workstation, then proceed to “Installing the DB2 GUI Tools” on page 131.
- If you already installed the DB2 GUI Tools in an Intel workstation, then proceed to “Configuring the Db2 Control Center” on page 134.

11.2 Relationship between Control Center and Administration Server

The relationship between the DB2 Control Center and the DB2 Administration Server is shown in Figure 40 on page 130.

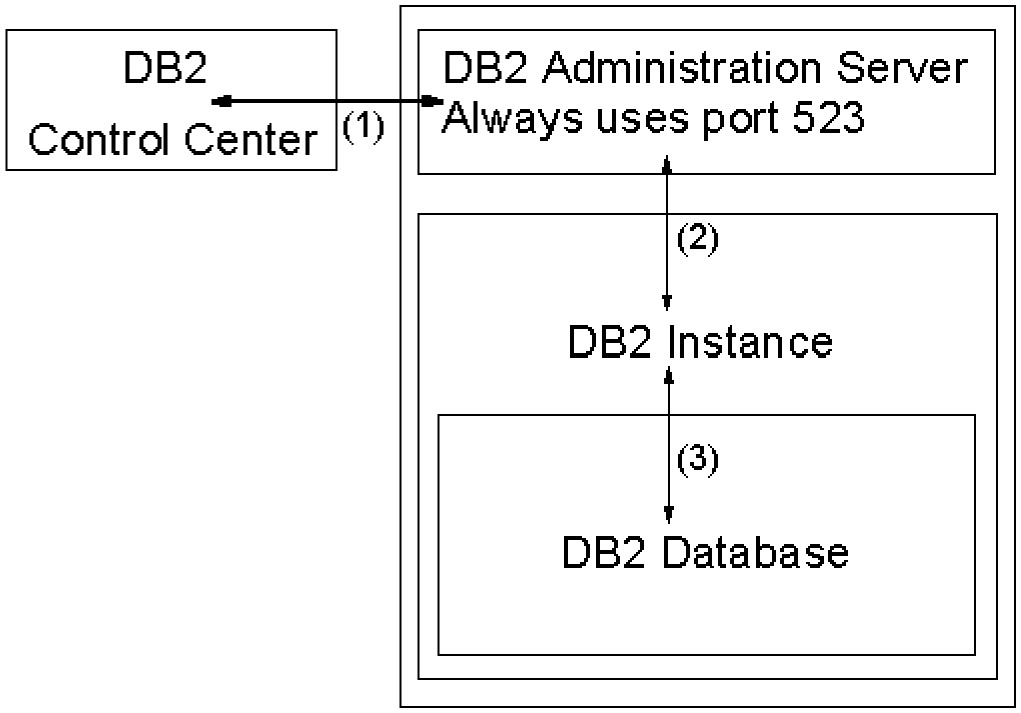


Figure 40. Relationship between Control Center and Administration Server

The DB2 Control Center so far is available only on Intel platforms. On the other hand, the

DB2 Administration Server is available on all platforms. You can use the DB2 Control Center to control local or remote DB2 instances and DB2 databases. In fact, the DB2 Control Center needs to interact with DB2 Administration Server in order to subsequently interact with the DB2 instances known to the Administration Server, and the DB2 databases controlled by those DB2 instances. In other words, the DB2 Control Center cannot communicate directly with remote DB2 instances or DB2 databases: all communications need to be done through the DB2 Administration Server.

You need to perform the following sequence from the DB2 Control Center in order to communicate with a DB2 instance (and its DB2 databases). The numbers in Figure 40 on page 130 correspond to each step in the sequence.

1. From the DB2 Control Center, select a system, and then you have to "attach" to the DB2 Administration Server by providing the user ID (the default is "db2as") and the password (the default is "ibmdb2").

2. After you are attached to the DB2 Administration Server, expand the icon for the system, select the desired instance, and then you have to "attach" to the DB2 instance by providing the user ID (the default is "db2inst1") and the password (the default is "ibmdb2").
3. After you are attached to the DB2 instance, expand the icon for the instance, select the desired database, and then you have to "connect" to the desired DB2 database. At this time, by default, there is no need to provide another user ID or password.

11.3 Installing the DB2 GUI Tools

This section describes how to install the DB2 GUI Tools in an Intel workstation. This section is targeted for those users who have VisualAge TeamConnection servers installed in UNIX workstations but do not have VisualAge TeamConnection servers installed in an Intel workstation. In case you have installed VisualAge TeamConnection Version 3 in UNIX, you can still install the DB2 GUI Tools in an Intel workstation if you have access to one. These DB2 GUI Tools are part of the DB2 Client Application Enabler (CAE) component of DB2 Universal Database for the Intel platforms. In case it is not feasible to install only the DB2 CAE component from the CDROMs provided with VisualAge TeamConnection, you can download the DB2 CAE code, by selecting the item Client Application Enabler from the IBM DB2 Service and Support web page at:

http://www.software.ibm.com/data/db2/db2tech

Follow the installation instructions for DB2 on Windows NT (see “Installation of DB2 for Windows” on page 109) or for DB2 on OS/2 (see “Installation of DB2 for OS/2” on page 115) for details on how to install DB2 from CD-ROM in your platform. However, a simplified version is shown in the next subsections.

If you do not want to install the complete "DB2 UDB Personal Edition", you can specify to install only the "DB2 Client Application Enabler". Be sure to select to install the DB2 GUI Tools and the on-line Documentation. After the installation of the DB2 Client Application Enabler (CAE), you will have the following:

- DB2 CAE installed in the C:\SQLLIB directory.
- The GUI Control Center and the Documentation will also be installed.

11.3.1 Installing the Client Application Enabler in Windows NT

Install the Client Application Enabler as follows:

1. Start -> Settings -> Control Panel -> Add/Remove Programs
2. Click on Install.
3. Specify: x:\SETUP.EXE (where x: is your CD-ROM drive)
4. Click on Finish.
5. At the "Welcome" window, click on Next.
6. At the "Enable Remote Administration" window, select the checkbox "Install components required to administer remote servers".
7. At the "Select Installation Type" window, select Custom.
8. At the "Select DB2 Components" window, ensure that you select ALL of the following components (they are all selected by default):
   - Graphical Tools, which in turn has 2 subcomponents:
     - Control Center
     - Client Configuration Assistant
   - Documentation (you may select the desired subcomponents).
9. At the "Select DB2 Components" window, verify that the default disk drive is the one that you want to select. It is strongly recommended to keep the directory name SQLLIB, because the documentation refers to it frequently.
10. At the "Select Start Options" window, you can decide if the Control Center is started automatically at boot time.A shortcut icon is added into the Startup folder; later on you could remove this icon from the folder in order to not start the Control Center at boot time.
11. At the "Customize NetBIOS" windows, you may decide to select on the check box for NetBios and look at the predefined Properties. This could be useful if you install DB2 UDB V5 on an Intel workstation later on.
12. At the "Start Copying Files" you will have the last opportunity to review/update/cancel the installation. Once you are totally sure, then click on Install.
13. Continue with “Configuring the Db2 Control Center” on page 134.

11.3.2 Installing the Client Application Enabler in OS/2

Install the Client Application Enabler as follows:

1. Logon as db2admin.
2. Shut down any other programs so that the setup program can update files as required.
3. Insert the CD-ROM into the drive.

4. Open an OS/2 windows and set the drive to x:, where x is the letter that represents your CD-ROM drive.
5. Enter the command:

   x:\db2\language\install

   Where x: represents your CD-ROM drive, LANGUAGE represents the two-character country code for your language (for example, EN for English).
6. The "IBM DB2 for OS/2 Version 5 Installation" dialog opens.
7. Select the following item:

   DB2 Client Application Enabler

   Once you select the option, then the "Operation Type" label "Install" will be enabled. Click on the button "Continue".
8. Answer the question if you want the installation program to Update your config.sys file.
9. The Install - Directories window provides a list of the components to install. It is suggested to accept:

   Graphical tools

   Client Configuration Assistant

   Documentation

   You can select a new disk drive where to install the code. Please keep the directory "SQLLIB". Then press Next.
10. It is recommended to answer "Yes" to the question on auto-start Control Center.
11. The Specify System Name window opens. Specify the System Name for your host, which should be unique.
12. The DB2 components are now installed on your system.
13. After you install the product, you must reboot your workstation before you can begin to use it. Select a reboot option and click on the Finish button. This completes the installation.
14. When your system restarts, log on with the "db2admin" user name.
15. Continue with “Configuring the DB2 Control Center.”

11.4 Configuring the Db2 Control Center

This section describes how to configure the UNIX server to allow the remote control of DB2 databases from a DB2 Control Center in an Intel workstation. You need to perform the following tasks:

1. Setup of the UNIX Server. See “Setup of the UNIX Server” on page 134
2. Setup of the Intel workstation. See “Setup of the Intel workstation (adding a remote system)” on page 135.

11.4.1 Setup of the UNIX Server

If you used "db2setup" to setup the DB2 Instance and the DB2 Administration Server, then everything is already setup for you. If you did not create the DB2 Administration Server, then you should create it by using the DB2 Installer. See “Creating The Administration Server” on page 153for similar information, but instead of requesting to create a normal DB2 Instance, select to create a DB2 Administration Server instance. You can perform the following tasks to verify that the setup in UNIX is correct:

1. Login as the DB2 instance owner.
2. Verify the TCP/IP Connection Service name (SVCENAME) for the DB2 instance by doing the following:

   ```
   $ db2 get dbm cfg | grep SVCENAME
   ```

   ```
    TCP/IP Service name (SVCENAME) = db2cdb2inst1
   ```

3. Verify the TCP/IP connection service name and port number for the DB2 instance are defined in the /etc/services file by doing the following. Use the value for SVCENAME obtained in previous step:

   ```
   $ grep db2cdb2inst1 /etc/services
   ```

   ```
   db2cdb2inst1 50000/tcp # Connection port for DB2 instance db2inst1
   ```

4. Verify that the TCP/IP interrupt service name and port number for the DB2 instance are defined in the /etc/services file by doing the following. The port number is based on the connection service port of the DB2 instance (such as the default 50000) plus 1. In this case the default is 50001:

   ```
   $ grep 50001 /etc/services
   ```

   ```
   db2idb2inst1 50001/tcp # Interrupt port for DB2 instance db2inst1
   ```

5. Verify that the Node type is "Database Server with local and remote clients". Obtain the "Node type" for the instance:

   ```
   $ db2 get dbm cfg | more
   ```

The value will be shown in the first 3 lines of the output and it has to be the following:

Database Manager Configuration

Node type = Database Server with local and remote clients

6. Verify that the communications support, DB2COMM, is defined as tcp/ip:

   $ db2set -all

   [i] DB2AUTOSTART=TRUE

   [i] DB2COMM=tcpip

   [g] DB2SYSTEM=oem-ppc3

   [g] DB2ADMINSERVER=db2as

   If DB2COMM is not "tcpip" (for TCP/IP) then specify it as follows and restart the DB2 instance:

   - db2set DB2COMM=tcpip
   - db2stop
   - db2start

For more details, see chapter 25 "Setting up Communications on the Server Using the Command Line Processor", from the Quick Beginnings for UNIX book.

11.4.2 Setup of the Intel workstation (adding a remote system)

You can setup the Intel workstation to administer a remote database in UNIX. Basically the remote system and the desired instances and databases need to be added to the DB2 Control Center. In this example the DB2 Control Center from Windows NT will be shown.

1. Logoff as Administrator (or other user ID that does not have administration authority). If you use Administrator, then some DB2 actions will fail with the error message (notice that the user name is truncated to the first 8 characters).

   SQL1092

   "Administ" does not have the authority to perform the requested command.

2. Login as db2admin (or another user ID with administration authority and which name complies with the DB2 rules of 8 characters or less.

3. Start the Control Center:

   Start -> DB2 for Windows NT -> Administration Tools -> Control Center

4. Select the item Systems; then click the right button and select "Add...". In the "Add System" dialog enter the following info:

   Protocol parameters:

   Host name: oem-ppc3

5. Click on Retrieve.

   Then the other values will be filled in; you should verify them. Notice that the "System Name" is the local (to the client) name by which the remote server will be known. It could be different than the actual remote server name (you can provide a name that is more meaningful for you).

6. You will see that the oem-ppc3 entry was added to the Systems tree in the Control Center.
7. Click on the system entry (such as oem-ppc3) and expand it by clicking on the + sign. So far, there are no instances defined to it.
8. Select "instances" from the tree, click on the right mouse button and select "Add...". In the "Add Instance" dialog, click on "Refresh" to populate the fields. It is suggested to have the same instance name for both local and remote. Thus, copy the value from the "Remote instance" field into "Instance name". Then the instance will be added to the tree.
9. To add a database to the instance, see “Adding a Database in the Control Center” on page 136.

11.5 Adding a Database in the Control Center

To add a database to a DB2 instance in the Control Center, do the following:

1. Logoff as Administrator (or other user ID that does not have administration authority). If you use Administrator, then some DB2 actions will fail with the error message (notice that the user name is truncated to the first 8 characters).

   SQL1092

   "Administ" does not have the authority to perform the requested command.

2. Login as db2admin (or another user ID with administration authority and whose name complies with the DB2 rules of 8 characters or less.
3. Start the Control Center:

   Start -> DB2 for Windows NT -> Administration Tools -> Control Center

4. Click on the + sign on the item Systems; then click on the + sign on the desired system. Before you can actually work with a given system, you need to "attach" to the DB2 Administration Server of that system:

   Click on the right mouse button and select "Attach...".

   Enter the user name and password of the DB2 Administration Server (such as db2as).

5. Click on the + sign on the item Instances.

6. Before you can actually work with a given DB2 instance, you need to "attach" to it. Select the desired DB2 instance (such as db2inst1) from the tree, click on the right mouse button and select "Attach...".

   Enter the user name and password of the DB2 instance owner.

7. Select the item Databases, click on the right mouse button and select "Add...". In the "Add Database" dialog, click on the Refresh button to see the available databases. Select the desired one. Add a comment to identify the use or purpose of this database. You may need to use a different local alias, in case you get a warning message) and click on Apply.

8. Now you should be able to see the desired remote databases in your DB2 Control Center. Select the desired database and click on the + sign to expand it and show the tables, views, etc.

11.6 What to do Next

Now that you have configured the Control Center, these are some hints on what to do next:

- Find out the DB2 tools to administer databases. See “Administration of Databases with DB2 Tools” on page 139.
- Learn how to work with the DB2 instances. See “Working with DB2 Instances” on page 145.
- For UNIX, you can learn how to work with the DB2 administration server. See “Working with the DB2 Administration Server in UNIX” on page 153.

Chapter 12. Administration of Databases with DB2 Tools

12.1 Overview of this Chapter

There are two ways to administer DB2 databases:

- The DB2 GUI Tools (Recommended).

  You can perform database administration tasks from an OS/2, Windows NT, or Windows 95 client by using DB2 GUI tools:

  - Use the Control Center to graphically perform server administrative tasks such as configuring, backing up and recovering data, managing directories, scheduling jobs, and managing media.
  - Use the Command Center to access and manipulate databases from a graphical interface.

- Command Line Processor.

  You can administer local databases using the command line processor, the DB2 GUI tools greatly facilitates the maintenance activities of local and remote databases. These tools are only available in Intel platforms and are not available yet in UNIX platforms. In fact, these tools are so useful that even in the official DB2 Administration for UNIX education classes from IBM, these tools (running in an Intel workstation) are used to administer remote databases in UNIX. Thus, it is strongly recommended that if you have DB2 databases in UNIX you should try to administer them remotely from the DB2 GUI tools running on an Intel workstation. This is the mechanism described in this chapter.

12.2 Control Center

The Control Center displays database objects (such as databases, tables, and packages) and their relationships to each other. You can manage a local database server or multiple remote database servers and the database objects within them, all from a single point of control. From the Control Center you can perform the following tasks on database objects:

- Create and drop a database.
- Create, alter, and drop a table space or table.
- Create, alter, and drop an index.
- Backup and restore a database or table space.

The administration of a VisualAge TeamConnection family is not covered in this chapter.

12.3 Additional Facilities from the Control Center

The Control Center provides additional facilities to assist you in managing your DB2 servers:

- Use the Command Center to enter DB2 commands and SQL statements in an interactive window and see the execution result in a result window. You can scroll through the results and save the output into a file.
- Use the Script Center to create mini-applications called *scripts*, which can be stored and invoked at a later time. These scripts can contain DB2 commands, SQL statements, as well as operating system commands. Scripts can be scheduled to run unattended. These jobs can be run once or set up to run on a repeating schedule; a repeating schedule is particularly useful for tasks like backups.
- Use the Journal to view all available information about jobs that are pending execution, executing, or that have completed execution; the recovery history log; the alerts log; and the messages log. The Journal also allows you to review the results of jobs that are run unattended.
- Use the Alert Center to monitor your system for early warnings of potential problems or to automate actions to correct problems discovered.
- Use the Tools Setting to change the settings for the Control Center, Alert Center, and Replication. You can run these facilities from the Control Center toolbar or from icons in the Administration Tools folder. You can find additional information in the Administration Getting Started Guide or in the Control Center's on-line help.

12.4 Understanding System Administrative (Sysadm) Authority

System Administrative authority is required to perform administration tasks such as cataloging, starting the database manager, or creating a database. Throughout this document, the IDs that have this authority are referred to as having SYSADM authority.

By default, System Administrative (SYSADM) privileges are granted to the following:

- UNIX

  Any user belonging to the primary group of the instance owner's user ID.

- OS/2

  A valid DB2 user ID which belongs to the UPM Administrator or Local Administrator group.

- Windows NT

  A valid DB2 user name which belongs to Administrator group. The length of the user name should be less than or equal to eight characters.

- Windows 95

  Any Windows 95 user that explicitly logs in into the system.

12.5 Where Are The Files Used with The Profile Registry?

The profile registry in DB2 contains the registry values, the environment variables and the configuration parameters that control the DB2 UDB environment. The profile registry is made up of the following distinct parts, which are located according to the DB2 platform:

System Profile Registry

Contains the listing of the local instance names:

AIX /var/db2/v5/profiles.reg

HP-UX /var/opt/db2/v5/profiles.reg

OS/2 %DB2INSTPROF%\profiles.req

Windows NT
\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2\PROFILES

Global Profile Registry

Contains global (machine-wide) default variables and DB2 system variable settings:

AIX /var/db2/v5/default.env

HP-UX /var/opt/db2/v5/default.env

OS/2 %DB2INSTPROF%\default.env

Windows NT
\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2\GLOBAL_PROFILE

Instance Profile Registry

Contains instance variable settings per instance:

UNIX $DB2INSTANCE_HOME/sqllib/profile.env

OS/2 %DB2INSTPROF%\<instance>\profile.env

Windows NT
\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\DB2\PROFILES\<instance>

12.6 What Other System-Wide Files Are Needed by Db2?

Several directories and files are related to DB2. These are some hints concerning the main ones:

- For details on the directories where the DB2 code and the DB2 instances are installed, see Table 17 on page 66.
- For details on the directories and files where the DB2 Registry is located, see “Where Are The Files Used with The Profile Registry?” on page 141
- For details on the UNIX directories and files where the runtime licenses for DB2 are located, see “Verify the License Information in the Nodelock File” on page 106. In Intel, the iFor License Use Runtime group is created under the IFOR subdirectory in the same disk unit (the default is C:). The actual file that has the runtime license is IFOR\LS\CONF\NODELOCK.

Miscellaneous other files are also related to DB2:

- Services file, which has the TCP/IP services and port numbers for DB2 and for the VisualAge TeamConnection family.
- Hosts file, which may have the database alias for the VisualAge TeamConnection family.

12.6.1 System-Wide Files in UNIX Needed by DB2

The following system files are also needed by DB2:

/etc/passwd Defines the DB2 user IDs.

/etc/group Defines the DB2 group IDs.

/etc/initab Is used in UNIX to invoke /etc/rc.db2 for the auto start of DB2 instances upon reboot.

/etc/rc.db2 Is used in UNIX for the auto start of the DB2 instances upon reboot.

12.7 What Next?

Now that you know the DB2 tools to administer databases, these are some hints on what to do next:

- Learn how to work with the DB2 instances. See “Working with DB2 Instances” on page 145.
- For UNIX, you can learn how to work with the DB2 administration server. See “Working with the DB2 Administration Server in UNIX” on page 153.

Chapter 13. Working with DB2 Instances

13.1 Overview of this Chapter

The information in this chapter describes DB2 server instances and how to work with them.

A DB2 instance is a logical database manager environment where you catalog databases and set configuration parameters. On UNIX systems, a separate user ID is needed for every DB2 instance and you can use multiple DB2 instances to accomplish the following:

- Use one instance for each VisualAge TeamConnection family.
- Tune a database instance for a particular environment.
- Optimize the database manager configuration for each database instance.
- Limit the impact of an instance crash. In the unlikely event of an instance crash, only one instance is impacted. The other instances may continue to function normally. However, multiple instances have some minor disadvantages:
- Additional system resources (virtual memory and disk space) are required for each instance.
- More administration is required because you have additional instances to manage.

During the installation procedure, a default DB2 instance is also created:

UNIX db2inst1 in /home/db2inst1

Intel DB2 in C:\DB2

To support the immediate use of this instance, the following are set during installation:

- The environment variable DB2INSTANCE is set to DB2.
- The DB2 registry value DB2INSTDEF is set to DB2.

13.2 Creating Instances

The user ID of the instance owner, and the primary group ID that is the system administration group are associated with every instance. These are assigned during the process of creating the instance. One user ID can be used for only one DB2 Instance.

Also, each instance owner must have a unique home directory. All of the files necessary to run the database instance are created in the home directory. The primary group of the instance owner user ID is also important, because it automatically becomes the system administration group for the database instance and gains administrative authority, which allows users that are members of that group to start, stop, or change the database instance. You can use the following methods to create an instance:

- The DB2 Installer utility, which is the recommended method in UNIX.
- The DB2 line command: `db2icrt`.

When an instance is created, its name is also added to the list of instances on the system.

In UNIX, these methods create the DB2INSTANCE_HOME/sqllib, where DB2INSTANCE_HOME is the home of the instance owner.

In Intel, the directory C:\InstanceName is created, where InstanceName is the actual name of the instance.

You should not create files or directories under DB2INSTANCE_HOME/sqllib directory other than those created by DB2 products. This avoids a potential loss of data if an instance is deleted.

13.2.1 Using the DB2 Installer in UNIX

The DB2 Installer is available for AIX, HP-UX, and Solaris operating systems. This is the recommended method. You can use the DB2 Installer to create additional DB2 instances after you have installed the DB2 product on your system. To start the DB2 Installer from the DB2 product directory, type the following command:

AIX `/usr/lpp/db2_05_00/install/db2setup`

HP-UX `/usr/IBMdb2/V5.0/install/db2setup`

Solaris `/usr/IBMdb2/V5.0/install/db2setup`

The next time you start the DB2 Installer, after having installed DB2 Version 5, you see the **DB2 Installer** screen.

To create another DB2 Instance on your system, select **Create**. See “Creating the DB2 Instances” on page 95

To terminate the DB2 Installer, select **Close**.

13.2.2 Using db2icrt

You can also create additional DB2 instances using the db2icrt command as follows:

DB2_HOME/instance/db2icrt -u FencedID InstName

such as:

db2icrt -u db2fenc1 db2inst1

where InstName is a string up to eight alphanumeric characters long (refer to Appendix “Naming Restrictions for DB2 User IDs” on page 295 for more information about naming rules).

Where DB2_HOME is:

AIX /usr/lpp/db2_05_00

HP-UX /opt/IBMdb2/V5.0

Solaris /opt/IBMdb2/V5.0

Notes:

1. For instances to be used by VisualAge TeamConnection, the authentication type defaults to SERVER because DB2 UDB Workgroup Edition is used and because the instance must be created locally with respect to the server.
2. FencedID identifies the user under which the fenced UDFs and Stored Procedures will execute. FencedID may not be root or bin. Also, for security reasons, we strongly recommend that you avoid using the same name as the instance name.
3. InstName. It is the login name of the instance owner.

13.3 Listing Instances

To get a list of all the database instances that are available on a system, you can do one of the following actions (in order of ease of use).

- From the DB2 Control Center in Intel, select Systems, then the desired system, then click on the item Instances.
- Run the db2ilist command, which is located in:

  $DB2_HOME/bin/db2ilist

Where DB2_HOME is:

AIX /usr/lpp/db2_05_00

HP-UX /opt/IBMdb2/V5.0

Solaris /opt/IBMdb2/V5.0

- To determine which database instance applies in the current session, enter:

```
echo $DB2INSTANCE
```

13.4 Enabling the Proper Profile when Logging in to the DB2 Instance

Unfortunately, the default initial profile for the DB2 instance does not have the proper statement to execute the DB2 profile. We recommend that you add the statement corresponding to your environment into your initial profile.

In order to execute this DB2 profile, set up the instance owner environment by executing:

- **Korn or Bourne shell**.

. $DB2INSTANCE_HOME/sqllib/db2profile

- **C shell**.

source DB2INSTANCE_HOME/sqllib/db2cshrc

where DB2INSTANCE_HOME is the home directory of the DB2 instance.

13.5 Starting and Stopping a DB2 Server Instance

You must start a DB2 server instance before you can perform the following tasks related to VisualAge TeamConnection:

- Connect to a database on the instance.
- Bind a package to a database.

13.5.1 Starting a DB2 Server Instance

To start a database instance:

1. Log in as the instance owner.
2. Start the DB2 database manager by entering the following command from a command line:

   db2start

13.5.2 Stopping a DB2 Server Instance

To stop a database instance:

1. Log in as the instance owner.
2. Stop the DB2 database manager by entering the following command from a command line:

   ```
   db2stop
   ```

In case there are pending transactions from a connected database, then db2stop will not work. In order to cancel those transactions and stop the DB2 instance, do the following:

```
db2 force application all
db2 terminate
db2stop
```

13.6 Auto-Starting Instances (UNIX)

13.6.1 Enabling Auto-Start

To enable an instance to auto-start after each system reboot, perform the following steps:

1. Log in as the instance owner.
2. Turn on the auto-start flag in the instance's registry with the command:

   ```
   db2set -i InstName DB2AUTOSTART=YES
   ```

Where InstName is the login name of the instance.

13.6.2 Disabling Auto-Start

To prevent an instance from auto-starting after each system reboot, perform the following steps:

1. Log in as the instance owner.
2. Turn off the auto-start flag in the instance's registry with the command:

   ```
   db2set -i InstName DB2AUTOSTART=
   ```

where InstName is the login name of the instance.

13.7 Updating Instances

Existing instances are designed to be as independent as possible from the effects of subsequent installation and removal of DB2 products. In most cases, existing instances will automatically inherit or lose access to the function of the product being installed or removed. However, if certain executable or components are installed or removed, existing instances do not automatically inherit the new system configuration parameters or gain access to all the additional function. The instance must be updated. If a DB2 product is updated by installing a PTF or a patch, all the existing DB2 instances should be updated using the command:

```
db2iupdt -u db2fenc1 db2inst1
```

Running the `db2iupdt` script will update the specified instance by replacing the files in DB2INSTANCE_HOME/sqllib directory, where DB2INSTANCE_HOME is the home directory of the instance. The `db2iupdt` command is available in the DB2_HOME/instance directory, where DB2_HOME is:

/usr/lpp/db2_05_00 on AIX

/opt/IBMdb2/V5.0 on HP-UX or Solaris

13.8 Removing Instances

To remove a DB2 instance, perform the following steps:

1. Log in as the instance owner.
2. Make sure the database manager instance is stopped. See “Stopping a DB2 Server Instance” on page 149.
3. Back up files in the DB2INSTANCE_HOME/sqllib directory, if needed. For example, you might want to save the database manager configuration file, db2systm, or user-defined function or fenced stored procedures applications in DB2INSTANCE_HOME/sqllib/function, where DB2INSTANCE_HOME is the home directory of the instance owner.
4. Log out as the instance owner.
5. Log in as root.
6. Remove the DB2 instance by executing the db2idrop command:

```
$DB2_HOME/instance/db2idrop InstName
```

where InstName is the login name of the instance. The `db2idrop` command removes the instance entry from the list of instances and removes the DB2INSTANCE_HOME/sqllib directory.

7. As root, remove the instance owner’s user ID and group (if used only for that instance). Do not remove these if you are planning to recreate the instance. Also, you can remove the home directory for the DB2 Instance. This step is optional since the instance owner and the instance owner group may be used for other purposes.

13.9 What to Do Next

Now that you know how to work with normal DB2 instances, these are some hints on what to do next:

- For UNIX, you can learn how to work with the DB2 administration server. See ““Working with the DB2 Administration Server in UNIX” on page 153.

Chapter 14. Working with the DB2 Administration Server in UNIX

14.1 Overview of this Chapter

This section shows you how to manually create the Administration Server. It also describes how to start, stop, list and remove the Administration Server.

Note: You can have only one Administration Server for each host.

14.2 Understanding the Administration Server

The Administration Server is used as a service by the DB2 Administration Tools to satisfy requests. It is implemented as a DB2 instance, and has interfaces to start, stop, catalog, and configure it. The Administration Server resides on every DB2 server that you want to administer and detect. The Administration Server is required in order to use any of the administration tools described above. The relationship between the Control Center and the Administration Server is shown in "Relationship between Control Center and Administration Server" on page 129.

14.3 Creating The Administration Server

You can use the following methods to create the DB2 Administration Server:

- The DB2 Installer utility, which is the recommended method in UNIX.
- The DB2 line command: `dasicrt`.

14.3.1 Using the DB2 Installer in UNIX

The DB2 Installer is available for AIX, HP-UX and Solaris operating systems. This is the recommended method. You can use the DB2 Installer to create the DB2 Administration Server after you have installed the DB2 product on your system. To start the DB2 Installer from the DB2 product directory, type the following command:

AIX /usr/lpp/db2_05_00/install/db2setup

HP-UX /usr/IBMdb2/V5.0/install/db2setup

Solaris /usr/IBMdb2/V5.0/install/db2setup

The next time you start the DB2 Installer, after having installed DB2 Version 5, you will see the **DB2 Installer** screen.

To create the DB2 Administration Server on your system, select **Create**. See "From the Create DB2 Services screen shown in Figure 28 on page 96, select Create a DB2 Instance." on page 95 for additional details.

To terminate the DB2 Installer, select **Close**.

14.3.2 Using the DB2 Line Command dasicrt in UNIX

To create the Administration Server, you must have root authority to run the dasicrt command at a command prompt. The syntax of the `dasicrt` command is as follows:

$DB2_HOME/instance/dasicrt ASName

where ASName is the name of the Administration Server, which is composed of a string of up to eight alphanumeric characters. Refer to "Naming Restrictions for DB2 User IDs" on page 295 for further information. You use the name of the Administration Server to set up the directory structure and access permissions. To start the newly created Administration Server:

- Use the `db2admin` command for a manual start (refer to "Starting the Administration Server").
- Reboot the system and it will be automatically started during the boot up.

14.4 Listing The Administration Server

To obtain the name of the Administration Server on your system, you must log in as the owner of one DB2 instance, and then execute the command:

db2set -g DB2ADMINSERVER

14.5 Enabling the Proper Profile when Logging in to the Server

Unfortunately, the default initial profile for the DB2 Administration Server does not have the proper statement to execute the DB2 profile. It is recommended that you add the following statement into your initial profile in order to execute this DB2 profile:

- **Korn or Bourne shell**.

  . $DB2INSTANCE_HOME/sqllib/db2profile

- C shell.

  source DB2INSTANCE_HOME/sqllib/db2cshrc

Where DB2INSTANCE_HOME is the home directory of the DB2 Administration Server.

14.6 Starting the Administration Server

To start the Administration Server, you must perform the following steps:

1. Log in as the Administration Server owner.
2. Start the Administration Server using the db2admin command as follows:

   db2admin start

The Administration Server is automatically started after each system reboot.

14.7 Stopping the Administration Server

To stop the Administration Server, you must perform the following steps:

1. Log in as the Administration Server owner.
2. Stop the Administration Server using the db2admin command as follows:

   db2admin stop

14.8 Removing the DB2 Administration Server

You need to stop the DB2 Administration Server before you can remove it. See “Stopping the Administration Server” on page 155 To remove the Administration Server, you must perform the following steps:

1. Back up the files in the ASHOME/sqllib directory, if needed, where ASHOME is the home directory of the Administration Server.
2. Log in as root and remove the Administration Server using the `dasidrop` command as follows:

   $DB2_HOME/instance/dasidrop ASName

   where ASName is the name of the instance being removed. The `dasidrop` command removes the sqllib directory under the home directory of the Administration Server.
3. As root, remove the Administration Server's user ID and group. You can also remove the home directory for the Administration Server.

   This step is optional since the instance owner and the instance owner group may be used for other purposes.

Part 3. The Light through the Migration Process

Chapter 15. Comparing of CMVC and TeamConnection V3.1

This chapter compares the functions of the latest version of CMVC 2.3.1 (Configuration Management and Version Control) and the latest version of its successor product, VisualAge TeamConnection Enterprise Server Version 3. The objective is to provide to the CMVC user the relevant information about what is the same, what is different, and what is new between CMVC and VisualAge TeamConnection.

15.1 Introduction

The purpose of this section is to compare and contrast the latest version of CMVC 2.3.1 (Configuration Management and Version Control) with the latest version of its successor, VisualAge TeamConnection Enterprise Server Version 3. The focus will be on enhanced capabilities in TeamConnection derived from the addition of an object-oriented methodology and its object repository.

This chapter is designed to provide enough information for current CMVC users to become productive VisualAge TeamConnection users quickly. It is assumed that readers are familiar with the basic functions of CMVC.

The organization of this chapter is as follows:

- Comparison of functions common to both products. See “Comparison of Common Functions” on page 164.
- New VisualAge TeamConnection functions and differences affecting general users. See “New User Functions of TeamConnection” on page 182.
- New VisualAge TeamConnection functions and differences affecting family administrators See “Changes That Impact System And Family Administrators” on page 1987.
- Improvements after migrating from CMVC to VisualAge TeamConnection. See Appendix E, “Process Improvements after Migration” on page 287.

This chapter is not a replacement for the documentation of these products; therefore, the functions will be explained only briefly. For more details, refer to the appropriate documentation listed in “Related Publications” on page 307.

The best way to find out how the client functions really work is to review and run the following samples:

- In CMVC, /usr/lpp/cmvc/samples/README.demo3 and demo3.tar.
- In VisualAge TeamConnection, /usr/teamc/samples/univunix. The univunix utility runs the TeamConnection commands in univunix.cmds that you can modify. Because the CMVC servers are only available for UNIX, there is a tendency in this chapter to focus on the UNIX capabilities of VisualAge TeamConnection.

For brevity, sometimes the term "TeamConnection" is used to refer to "VisualAge TeamConnection Enterprise Server."

15.1.1 Compatibility between CMVC and VisualAge TeamConnection

VisualAge TeamConnection is the successor product to CMVC. VisualAge TeamConnection has evolved from CMVC 2.3 and has been extended to support object-oriented development.

The products have a lot in common; however, because of changes in VisualAge TeamConnection, these products are not compatible with each other. That is, you cannot use a CMVC client with a TeamConnection server or a TeamConnection client with a CMVC server.

A migration facility is provided with TeamConnection to move your CMVC family to TeamConnection. There are no utilities to migrate a TeamConnection family to CMVC.

15.1.1.1 Migration from CMVC to TeamConnection

The chapter “Migrating from CMVC to TeamConnection Version 3” on page 220 describes how to prepare a CMVC family to be migrated to TeamConnection.

15.1.2 The Big Difference for Users

Here are a few things that will be significant changes for CMVC users migrating to TeamConnection. There is more detail in “Comparison of Common Functions” on page 164 and “New User Functions of TeamConnection” on page 182.

- The names of several objects have changed. For example, the CMVC TRACKS are now TeamConnection WORKAREAS and CMVC LEVELS are now TeamConnection DRIVERS. For more details see “Command Names” on page 164
- The versioning scheme is different:

    - CMVC Files are versioned by check-out/check-in (file versioning); that is, the first time a file is created it has a version 1.1, then after the first check-in it is 1.2, and so on.
    - In contrast, TeamConnection Parts are versioned by the number of freezes in a workarea (system versioning); that is, the first time a part is created it is considered version 1 based on the workarea, such as workarea1:1, then after the first check-in and explicit freeze, it is workarea1:2, and so on.
    - Version numbers in TeamConnection are named relative to the workarea, the driver, and the release. For example, the part a.c can have a version number workarea1:3, or driver2:2 or release1:1. See “Version Numbers of TeamConnection Objects” on page 185.
    - The TeamConnection drivers and workareas are essentially specialized releases, in which they have a snapshot of all the versioned parts in a release. For more details see “Versioning” on page 183
- Workareas are much more important than tracks and behave differently. For more details see “Workareas Are More Than Renamed Tracks” on page 187.
    - A TeamConnection workarea needs to be specified in all "teamc part -checkin" and "teamc part -checkout" commands. Even if you are just using TeamConnection to store parts in a single release without any defects or features, you need to create a workarea and periodically integrate and commit your changes.
    - You may periodically be required to link parts from other workareas (preferable) or to refresh your workarea before you perform actions like integrating the workarea, or checking parts in or out. For example, if you try to check out a file that another user has updated in a different workarea, but has not committed, you will need to do a part link or a refresh from that workarea.
    - You can freeze a workarea in order to save intermediate changes of the changed parts in that workarea. However, only the last version of a workarea will be integrated to the release.
- When performing queries in TeamConnection, such as filling in a filter window, you are often querying within a context instead of the entire family. You need to specify that context by entering a release, a driver, a workarea or a version. If you do not provide enough context you will get an error or not be able to press the **OK** button. For more details see “Query Facility” on page 176.

- A Web Client has been added in Version 3 to allow interacting with TeamConnection commands via a Web browser. For more details see "New Web Client" on page 194.

15.1.3 The Big Differences for Family Administrators

Here are a few things that will be significant changes for CMVC Family Administrators migrating to VisualAge TeamConnection. There are more details in "Family Administration" on page 180 and "Changes That Impact System And Family Administrators" on page 198. The significant changes are these:

- The only database supported by VisualAge TeamConnection 3 is DB2 Universal Database (UDB) Version 5, and its installation files are bundled with the CD-ROMs for VisualAge TeamConnection. The installation utilities of TeamConnection will check whether DB2 UDB is already installed in your system; if not, then DB2 UDB will be installed in your system.
- In VisualAge TeamConnection all of the data is stored in the database, including files. The TeamConnection family administrator does not need to know SCCS in order to keep the database and file versions synchronized. In CMVC, in some rare cases it was necessary to know some SCCS commands in order to resynchronize the database and the file versions; for example, when the data was committed in the database, but because of a problem in the file system, the actual SCCS archive file was not created, and thus the database and file system became out of synch. This also makes backup much easier, because in CMVC you need to back up both the database and the version control ($CMVC_FAMILY_HOME/vc) file system at the same time.
- Most of the administration commands have changed. See the TeamConnection Administrator Guide for more details. Further, the TeamConnection Family Administration GUI (tcadmin) is provided for family administration.
- TeamConnection build administration requires maintaining pools of build machines so that TeamConnection can properly perform release builds. This means that TeamConnection requires more extensive planning and management for the network.
- The migration facility is different in TeamConnection than in CMVC. For more details, see "Migration from CMVC to TeamConnection" on page 160.

15.1.4 The Big Differences for Project Administrators

The Lotus Notes federation with TeamConnection is a significant benefit to a large project because it allows users to handle Requirement, Design, and Test Case documents via Lotus Notes and still handle defects and features that are tracked in TeamConnection. For more details see “Lotus Notes Integrated Databases” on page 194.

The Web interface enables a customer, with a little work to customize some html, to provide a very flexible 'helpdesk' system tracking all the way from the end user to development. For more details see “New Web Client” on page 194.

15.1.5 The Big Difference for Tools Integrators

CMVC relied on framework tools such as SDE/WORKBENCH to provide tool integration. SDE/Workbench provided build integration through "make," GUI integration and messaging, and so on.

TeamConnection is actually a point of integration, and it provides:

- A generic build interface that allows for traditional builds, as well as packaging and distribution. See “Build Support” on page 193 for more details.
- An API to allow client applications to perform TeamConnection tasks using function calls instead of client line commands. See “Integration with Other Products” on page 176 and “Object Repository and Information Model” on page 197 for more details.
- A Web Client that allows the interaction with TeamConnection commands via a web browser. This function was added in Version 3. For more details see “New Web Client” on page 194.
- A Lotus Notes interface that allows users to use Lotus Notes databases for requirements, test cases, development and other tasks (based on the generic template) to interact with TeamConnection features and defects. This function was added in Version 3. For more details see “Lotus Notes Integrated Databases” on page 194.
- The client in Version 3 supports the Source Code Control API to allow other applications to checkout and checkin parts. For more details see “Source Code Control API” on page 195.

15.2 Comparison of Common Functions

15.2.1 Command Names

When comparing TeamConnection and CMVC, the most obvious change is the naming of the client commands:

All TeamConnection commands now begin with "teamc", for example, "teamc report". Most of the CMVC and TeamConnection commands are the same. However, Table 30 on page 164 lists commands that have changed names:

Table 30. *CMVC terms that have different name/function in TeamConnection*

CMVC	TeamConnection	Comment
File	Part	To better describe the range in granularity of the objects that can be managed.
Level	Driver	To conform with common industry terms
LevelMember	DriverMember	To conform with the change for Level
Track	Workarea	To conform with the functionality change in workarea.
cmvcd	teamcd	To reflect the name of the product
cmvc	teamcgui	To reflect the name of the product
stopCMVC	tcstop	To reflect the name of the product
mkfamily, rmfamily, mkdb, rmdb, chfield, ch*	tcadmin	Consolidation of administration tools.
cmvccomp (OS/2)	tcmerge	Availability for all platforms.

NOTES:

1. The actions in the Authority table reflect the corresponding names, such as FileView in CMVC and PartView in TeamConnection.
1. In Windows and OS/2, where commands are limited to 8 characters some of the CMVC names have been abbreviated.

15.2.2 Configuration Management

Both CMVC and TeamConnection provide support for the process of identifying, organizing, managing and controlling software modules. This is accomplished by components and a component hierarchy.

15.2.2.1 Components and Component Hierarchy

A component hierarchy allows software modules to be grouped for organization purposes and to control the access of users to those software modules. For example, an application can create components to separately manage access to server code, client code and documentation.

A component can have zero, one or more child components, as well as one or more parents. The top component of the hierarchy, "root", has no parent; consequently, all the rest of the active components are descendants of the "root" component. However, deleted components also do not have parents.

The access list and the notification lists are inherited from the parent component to its descendants.

15.2.2.2 Access to Components

An access list associated with a component specifies an authorization level for each user, which determines the action that the user can perform on that component and its descendants. For example, a user with a "general" access has the CompView authority that allows the user to view the description of the specified component (or its descendants) but does not have the PartAdd authority that is needed to create a software module.

The above authorities can be restricted in a specific component. However, child components do not inherit these restrictions. There are certain actions that all users are able to perform, regardless of their access list. These actions are called "base authorities", such as opening a defect and adding remarks to it.

There are other actions that each user has implicit authority for, such as modifying the properties for the user object in CMVC or in TeamConnection.

The users who have "superuser" authority are allowed to perform any action that is legal, given the current process configurations and the state of the TeamConnection object being manipulated. For example, a superuser can cancel any defect that is not already associated with a workarea.

Differences

- Some actions from the CMVC authority.ld have been renamed in the TeamConnection authorit.ld file; which is loaded into the database as the

Authority table. For example, the FileView action has been renamed to PartView

- Furthermore, some new actions have been added to TeamConnection to reflect the new functions.

15.2.2.3 Notification Mechanism

A notification mechanism sends mail to team members as software modules change and as other actions are taken with objects in the product.

There are some explicit notifications that the user may want to receive and this is accomplished by registering to a notification list in a given component. For example, if a user wants to be notified when a new release is created in that component, then the user can add the "developer" notification list.

There is no support for finer granularity of events, such as the notification for all changes for one particular defect.

The actual notification is performed by means of the Notification Daemon (notifyd). In CMVC, the notification daemons only supports the UNIX utility "sendmail". In TeamConnection, the notification daemon runs a script which can be customized, for example, to use a mechanism other than sendmail (such as Lotus Notes or Microsoft Exchange in Windows NT).

Differences

- Some actions from the CMVC interest.ld have been renamed in the TeamConnection interest.ld file; this file is loaded into the database to populate the Interest table. For example, the FileView action has been renamed to PartView.
- Furthermore, some new actions have been added to TeamConnection to reflect the new functions.

15.2.2.4 Customized Processes for Components

The components are the objects from which defects and features (see “Defects” on page 170 and “Features” on page 170) are opened and manipulated. The process to handle the defects and features is customizable by component; that is, one component can have one process, and another component can have another process.

The process for a component could include the following subprocesses:

- Design-Size-Review (DSR)
- Verify subprocess

Depending on the subprocesses, it is possible to have the following records:

- Sizing records to identify the sizing activities during design.
- Verification records to accept or reject the defect/feature.

15.2.3 Release Management

Both CMVC and TeamConnection provide support performing a logical organization of objects that are related to an application. This is accomplished by releases.

15.2.3.1 Releases

A release provides a logical view of objects that must be extracted, built, tested and distributed together. Furthermore, a release can be linked to another release.

Both CMVC and TeamConnection provide support for handling serial development; that is, only one user at a time can have a lock on a file. The release may have an approval list and environment list.

Differences

TeamConnection has added the following functionality to releases:

- Concurrent development. A release can be created to allow concurrent development, that is, more than one developer at a time can update a part. For more details see “Sequential and Concurrent Development” on page 182.
- Collision records. When using concurrent development, if two or more users are updating the same part, then a collision record is created upon the first checkin of that part (that is, the first one in wins, the second gets a collision record). The workarea receiving the collision record needs to resolve the differences prior to integration. We suggest using the VisualAge TeamConnection Merge tool (tcmerge). For more details see “Sequential and Concurrent Development” on page 182.
- tcmerge: Graphical tool to merge different versions of a part. To help the handling of collision records, TeamConnection provides a GUI utility called tcmerge that shows the differences between two or three parts and allows the user to manually merge the parts into one single part. You can select to view the differences and collisions, as well as view the composite output of the merged files.
- Versioning releases. The TeamConnection drivers and workareas are essentially specialized releases, in which they have a snapshot of all the versioned parts in a release. As such, they are versioned objects that can

be queried for change history, can be extracted, can be built, etc. The CMVC "current" release concept has been removed. For more details see “Versioning” on page 183.

- Building and packaging releases. By using the new build and packaging functions in TeamConnection, the releases can be built and packaged from within TeamConnection, without the need to extract the release and invoke the make utilities to build the code. For more details see “Build Support” on page 193.
- Pruning. A release can be explicitly or automatically pruned to conserve space in the family account and to remove change history that is no longer needed. For more details see “Pruning of Releases” on page 183.
- Configurable fields. In TeamConnection Version 3 you can now add configurable fields to releases.
- Part shadowing. In TeamConnection Version 3 you can now exploit the part shadowing capabilities for releases. For details, see “Part Shadowing Capability” on page 195.
- Coupling (commonality of files between releases). It is possible to specify the type of coupling (commonality between the files) that a release has with other releases; this does not affect the commonality of files between workareas in the same release. This is specified by the -coupling flag for release -create and release -modify:
  - Default. You must use force to break links (same as VA TC V2 and CMVC).
  - Loose. If a common file is checked in for one release and if the -common flag is not used, then the commonality will be broken. This is the same behavior as if using -force; in this coupling, -force is not necessary.
  - LooseRestrict. A release cannot be kept common with any other release by the -common mechanism.

15.2.3.2 Approval List

The approval list contains the users who can approve the changes to be made in a release.

15.2.3.3 Environment List

The environment list contains the platforms or environments that need to be tested when the changes are incorporated into the release.

15.2.3.4 Customized Processes for Releases

The process to handle the releases is customizable, that is, one release can have one process, and another release can have another process. The process for a release could include the following subprocesses:

TRACK
Require defects or features to be associated with all tracks (CMVC) or workareas (TeamConnection). In other words, you need a reason to make a change.

APPROVAL
Require approval for each track/workarea before changes can be made to the track/workarea. This is usually used to limit development activity as you approach delivery of the release.

FIX
Used to identify components where files/parts are changed. Completing a fix record is a useful mechanism for developers to indicate that their track/workarea is ready to be integrated.

LEVEL (CMVC) or DRIVER (TEAMCONNECTION)
Levels/Drivers are used to incrementally integrate and commit a set of changes in a release. This makes incremental development and the delivery of beta drivers easy to manage and recreate.

TEST
Used as a verification method for someone other than the originator of a defect or feature. Useful when a test group or some other users need to evaluate a change prior to the originator taking final action. Depending on the subprocesses being used, the following records may be used:

- Approval records which are used by those users that are in the approval list.
- Fix records which are used by those users who are responsible for the components where the changes will be incorporated.
- Test records which are used by those users who need to test the changes in the specified environments.

Differences

- The CMVC Level subprocess is called Driver in TeamConnection.
- In TeamConnection it is possible to have a process in which the release could have the driver subprocess but not the track subprocess (this is not

possible in CMVC), because a workarea is always needed in TeamConnection and these workareas can be placed in a driver.

This is not recommended for production releases because there is no change history (defects/features) associated with the parts. In our experience, the customer sooner or later is going to ask why was this part changed? And without a change history it is not easy to find out the answer.

15.2.4 Change Control

15.2.4.1 Overview

Both CMVC and TeamConnection keep track of any file changes you make and the reasons you make them. Defects and features are used to identify the requirements for the file changes. After changes are made, you integrate the changes and build the application.

Both products ensure that the change process is followed and that the changes are authorized. The change control process is configurable and your team can decide how strict the change control should be, from loose to very tight. You can also adjust the level of control as you move through a development cycle.

There is an audit trail associated with each defect and feature. The history information includes who opened it, who accepted it, a description of the problem or suggestion, etc.

The defects and features can be "aged" in order keep track of how long the defect or feature has been active.

Note: It is not necessary to make a change prior to integrating tracks (CMVC) or workareas (TeamConnection). An empty track/workarea can be promoted for documentation or record keeping purposes.

15.2.4.2 Defects

A defect is opened (but not created) to report a problem.

The process to handle the defect depends on the process configuration for the component where the defect was opened. See “Components and Component Hierarchy” on page 165.

15.2.4.3 Features

A feature is opened (but not created) to request a design change to a future function. The process to handle the feature depends on the process configuration for the component where the feature was opened. See “Components and Component Hierarchy” on page 165.

15.2.5 Version Control

15.2.5.1 Overview

Both CMVC and TeamConnection provide support for the process of tracking the relationships among the versions of the various software modules that form an application. Version control with configuration management enables you to build your product using stable levels of code, even if the code is constantly changing.

Version control allows you to view the different snapshots of a file over time: as the file was created, as it was one month ago, as it is today.

The file changes need to be done in the context of a release. These changes are done by means of CMVC Tracks or TeamConnection Workareas, which in turn are integrated into a CMVC Level or a TeamConnection Driver. These integrated changes will eventually be committed by following the release process.

Differences

The implementation of versioning has been completely replaced in TeamConnection. For more details see “Versioning” on page 183.

15.2.5.2 Files (CMVC) or Parts (TeamConnection)

The software modules for your application are stored in files in CMVC (and parts in TeamConnection). Files are then created and modified to address defects and features. These files are associated with a component for the authorization of who can do what, and with a release for the actions of extraction, subsequent build and packaging.

The files are versioned and both products handle the versioning of text (such as a file that contains source code in C) and binary files (such as an executable file).

Both CMVC and TeamConnection provide expandable keywords that can be embedded in the files. This information identifies the context of the build (for example, which release the file came from), and when the files are extracted, these expanded keywords provide useful information to developers trying to determine where a source file or executable came from.

Differences

- TeamConnection parts are only versioned as part of the workarea. What this means is that when you check out a part, change it, then check it back in, the previous version will be replaced (unless this is the first change in the workarea).

  New versions of parts can also be created by freezing the entire workarea; in this case, only those parts that have changed in the workarea get new version numbers.

- Part shadowing.

  In TeamConnection Version 3 you can now exploit the part shadowing capabilities for releases. For details, see “Part Shadowing Capability” on page 195.

  See next bullet item for additional information about the CR/LF conversion.

- In CMVC, all text files were normalized to the UNIX standard: only use line-feed or LF between each line, and delete the carriage-return or CR. Thus, only when using Intel workstations it is necessary to use the -crlf flag during a file extract or check out to convert the LF to the CR/LF pair.

  In contrast, in TeamConnection, files are checked in "as is", that is, they are not normalized to the UNIX standard. By default, the text files are not normalized during extract or checkout. In order to normalize the text files to match the appropriate standard of the client, the -crlf flag should be used when extracting or checking out a file/part.

  However, in part shadowing, the flag -crlf means to explicitly add the CR/LF pair to all text while not using -crlf means to add only LF.

- TeamConnection parts cannot be explicitly destroyed. However, it is possible to reclaim space by pruning workareas in the release.

- TeamConnection parts are more than CMVC files. The granularity was expanded to include fine-grained objects that are not extracted as files, such as VisualAge Generator objects. Further, TeamConnection parts have attributes that are used during build and packaging. For more details see “Versioning” on page 183 and “Parts Are More Than Just Files” on page 186.

- The TeamConnection Part command adds actions to support the building of parts. When building a part that is defined as an "output", this causes the building of all of the "input" parts. Therefore, building the appropriate output part will build a complete executable.

- In CMVC, there are 2 sets of expandable keywords, one if the family is configured to use SCCS (the most widely used type) and another if the family uses PVCS. TeamConnection provides a different syntax for the expandable keywords.

- The TeamConnection Part object uses two attributes to retain important date information:
  - - Last update.

    This is the timestamp when the part was last changed.
  - - File system timestamp.

    During check-in, the date and time of a part from the file system is stored in this attribute.

  In TeamConnection, during the extraction of a part, the file system timestamp will be used; this means that the part will have the same date and time that the part had during the most recent check-in.

  In contrast, in CMVC, the last update attribute changed for ANY action that affected either the contents or the attributes of the file, and this date and time was used during the extraction.

15.2.5.3 Tracks (CMVC) or Workareas (TeamConnection)

In CMVC, depending on the process of the release, if the track subprocess is activated, then you need to specify a CMVC Track that will identify the file changes with a defect/feature and a release. For more details see “Workareas Are More Than Renamed Tracks” on page 187.

You may specify that a given Track/Workarea needs to be integrated with another one, and thus, they will become corequisites.

Differences

TeamConnection workareas are more powerful and useful than CMVC tracks due to the TeamConnection's object-oriented design.

- Workareas in TeamConnection are the context by which you view a release (a logical view). As such, you must always specify a workarea.
- Further, you may open multiple workareas for a defect or feature in one release. For more details, see “Workareas Are More Than Renamed Tracks” on page 187.
- TeamConnection adds the concept of a prerequisite so that two workareas do not need to change the same part in order to establish a relationship that will insure they are integrated to the same driver.

15.2.5.4 Levels (CMVC) or Drivers (TeamConnection)

In CMVC, depending on the process of the release, if the CMVC Level subprocess is activated, then you need to integrate the CMVC Track into a CMVC Level by means of a CMVC LevelMember.

Similarly, in TeamConnection, depending on the process of the release, if the TeamConnection Driver subprocess is activated, then you need to integrate the TeamConnection Workarea into a TeamConnection Driver by means of a TeamConnection DriverMember.

Differences

TeamConnection Drivers are also more powerful and useful than CMVC Levels. See "Driver Has More Options Than Level" on page 189 for more details.

15.2.6 Users and Authentication of Users

15.2.6.1 Overview

Each user has a unique ID in CMVC or TeamConnection. Be default, it is used in combination with a hostname and a system login to control access to the family.

In OS/2 and in Windows 95 (when the users bypass the login screen), because there is no system login, a variable is used instead; in CMVC the variable is USER and in TeamConnection the variable is TC_USER.

Differences

The functionality of the user is the same, but the authentication of users has been expanded in TeamConnection to allow the use of passwords and to optionally require users to login into TeamConnection. The use of passwords makes optional the use of the hostname and system login as part of the authentication. For more details, see "Password Loging to TeamConnection" on page 191.

15.2.7 Architectural Issues

15.2.7.1 Client/Server Design

Both CMVC and VisualAge TeamConnection have a client/server architecture, in that the client communicates across a TCP/IP socket to a family server. The socket is identified by

`"FamilyName@FamilyHost@SocketNumber".`

The client consists of a command line interface and a graphical user interface (GUI). The GUI constructs and issues line commands. The line commands issued are stored in a log file, which is specified in the GUI's settings pages.

Differences

- One single executable file with all client commands.

CMVC has one executable file for each command, such as Report. In contrast, VisualAge TeamConnection has a single executable "teamc" which contains all functions for specific command such as "teamc report".

This approach allows to have the same name for UNIX and Intel commands. For example, in CMVC, the UNIX command "LevelMember" had to be renamed in Intel as "levelmem".

- Object Repository API.

TeamConnection adds another client/server interface, the Object Repository. This allows C++ programs to access data in the family server through an object-oriented application programming interface. See "Object Repository and Information Model" on page 197 for more details.

15.2.7.2 Customization Aspects

The family can be customized in the following aspects:

- Configuration types (config.ld)
  - The family administrator can add more new types for items that have multiple-choice values.
  - A help text can be defined, which can be displayed from the GUI.
- Configurable fields

It is possible to add extra fields to the following objects:

  - Defects
  - Features
  - Users
  - Files (CMVC) or Parts (TeamConnection)
  - Releases (only in TeamConnection)
  - Workareas (only in TeamConnection).

For the family administrator there are several small changes. See "Family Administration" on page 180 for more details.

- User exits (only on the server side). Both products allow you to extend their functionality to a certain degree by allowing the family server to invoke utilities developed by the family administrator when certain actions/conditions occur. For more details, see “User Exits” on page 199.

15.2.7.3 Integration with Other Products

In CMVC and TeamConnection it is possible for other applications to use the command line interface to interact with the product.

Differences

- In CMVC, a UNIX client GUI is provided that is integrated with the product Softbench; however, because Softbench has been withdrawn from the market, then VisualAge TeamConnection is not integrated to Softbench.
- In TeamConnection Version 3, an interface was added for Lotus Notes, in which you can use Lotus Notes to manage requirements, test cases or development, and then manage the corresponding TeamConnection features or defects.
- In TeamConnection Version 3, an interface was added for Source Code Control (SCC) API, in which other applications such as PowerBuilder can checkout and checkin parts from TeamConnection.
- In TeamConnection, API's are provided to access TeamConnection data through an Object Repository (see “Object Repository and Information Model” on page 197). Applications such as VisualAge Generator use the API's to integrate to TeamConnection. The VisualAge brand family includes several other products that integrate directly to TeamConnection; there is more integration work under development.

15.2.7.4 Query Facility

CMVC and VisualAge TeamConnection Version 3 simply pass through the SQL queries to the database management system (DBMS) of the relational database and the DBMS responds to the queries.

Differences

- CMVC and TeamConnection have basically the same common tables and views, with the exception of the new functions for TeamConnection (such as ParserView) and the renaming of some functions (such as Levels to Drivers). As a result, TeamConnection reports in raw format often have more fields, and some of the fields (for example, versionSID) may be interpreted differently.
- The TeamConnection teamc report command offers two new options:

-TestClient
verifies the configuration of the client without contacting the server.

-userExitInfo
provides information on user exit to super-users. To find our more details on the parameters and configurable fields that a family administrator can use in the user exits, a superuser can issue the command:

```
teamc report -userExitInfo
```

An example of the output is shown below:

Action Name:

```
DefectModify User Exits 0 and 3
```

Parameter List: defectname, newdefectname, severity, environmentname, prefix, reference, drivername, abstract, originator, answer, remarks, release, configFields, MessageBuffer, notesDB, notesID, effectiveUserID, TeamcUserID, VerboseFlag

User Exit 1:

Parameter List: defectname, newdefectname, severity, environmentname, prefix, reference, drivername, abstract, originator, answer, remarks, release, configFields, dateoflastupdate, notesDB, notesID, effectiveUserID, VerboseFlag

User Exit 2:

Parameter List: defectname, newdefectname, severity, environmentname, prefix, reference, drivername, abstract, originator, answer, remarks, release, configFields, notesDB, notesID, effectiveUserID, VerboseFlag

Configurable Fields: symptom, phaseFound, phaseInject, priority, target, pubsImpact, guiImpact, duration, solution, customerName, pmrNumber, testImpact, installImpact, testerlogin

This command does not give the list of existing user exits for the family.

15.2.7.5 Migration Utilities

CMVC and TeamConnection provide utilities to migrate to the most current versions of each product. Further, TeamConnection provides a utility to migrate from CMVC to TeamConnection 3.

CMVC provides utilities to migrate from SCCS to CMVC.

CMVC and TeamConnection provide utilities to migrate to the most current versions of each product. Further, TeamConnection provides a utility to migrate from CMVC to TeamConnection 3.

15.2.7.6 Year 2000 Readiness

CMVC 2.3.0 uses 2 digits to store the information about the year. Although CMVC does not parse or use this information directly, when the user requests sorting by date (the actual sorting is done by the database) it is possible that the year 2000 might not be sorted properly.

VisualAge TeamConnection and CMVC 2.3.1 use 4 digits to store the information about the year. Thus, VisualAge TeamConnection and CMVC 2.3.1 are fully ready for the Year 2000.

15.2.8 Supported Platforms And Databases

15.2.8.1 Family Server

Both products support family servers in the following platforms:

- AIX Version 4.2.1
- HP-UX Version 10.20
- Solaris Version 2.5.1
- OS/2 Warp
- Windows NT

CMVC supports the following legacy server platforms. This support will not be added to TeamConnection:

- AIX Version 3.2.5
- HP-UX Version 9
- SunOS 4.1.3

15.2.8.2 Clients

Both products support clients (GUI and line commands) in the following platforms:

- AIX Version 4.2.1
- HP-UX Version 10.20
- Solaris Version 2.5.1
- OS/2 Warp
- Windows 95 and Windows NT

NOTES:

In VisualAge TeamConnection Version 3, there is no longer support for Windows 3.1.

CMVC supports the following legacy client platforms. This support might be added to TeamConnection at a later date (depending upon demand):

- AIX Version 3.2.5
- HP-UX Version 9
- SunOS 4.1.3

15.2.8.3 Build Servers

The build function is only supported by TeamConnection (see “Build Support” on page 193) and the supported platforms are:

- The same platforms for the TeamConnection family server, “Family Server” on page 178.
- MVS OS/390 and MVS/OE (Open Edition).

15.2.8.4 Databases

CMVC supports the following relational databases:

- DB2 Versions 1 and 2
- DB2 Universal Database (UDB) Version 5
- Oracle Version 7.x (including 7.3)
- Informix Version 5 and 7
- Sybase Version 4.9, 10 and 11

The only database supported by VisualAge TeamConnection 3 is DB2 Universal Database (UDB) Version 5, and its installation files are bundled with the CD-ROMs for VisualAge TeamConnection. The installation utilities of TeamConnection will check if DB2 UDB is already installed in your system; if negative, then DB2 UDB will be installed in your system.

15.2.8.5 License Handling

CMVC uses NetLS to enforce that the maximum number of concurrent users complies with the number of licenses that the customer acquired. For more details, see “License Handling in CMVC” on page 200.

TeamConnection, on the other hand, provides a utility named tclicmon (TeamConnection License Monitor) to monitor the audit.log of the family to see if the maximum number of concurrent users is within the bounds.

15.2.9 Family Administration

CMVC and TeamConnection use a family to contain the objects and files associated with a project or a set of projects. For each family there is a database that is dedicated only to that family.

In UNIX operating systems, where file security is based on a separate user login, it is recommended that each TeamConnection family be in a separate family login. The sample profile for families assumes a separate user login for each family.

Operating systems that run on the Intel platform, although some offer user login capabilities, do not benefit from this added file security. As such, OS/2 and Windows NT set up documentation only recommend separate directories.

15.2.9.1 Structure of Family Account

CMVC stores the file changes in a directory structure from the family file system and stores the information about the objects of the family inside the relational database.

In contrast, TeamConnection stores both the part changes and the information about the objects of the family inside the same database. As a result all updates happen within a single database transaction to improve data integrity.

The directory structure for CMVC and TeamConnection are similar; They are shown in Figure 31 on page 180 and in Figure 32 on page 181; the comments describe the differences:

Table 31. CMVC Family Directory Structure

Directory	Comments
authority.ld *	
config.ld *	
cfgcomproc.ld *	
cfgrelproc.ld *	
config.ld *	
interest.ld *	
audit/log *	Separate directory for audit log files
configField/ *	Definition of the configurable fields

Directory	Comments
maps/ *	The driver information is stored in the database in TeamConnection
queue/messages *	Separated addressing information and contents.
vc/ *	The file data is stored in the database in TeamConnection

Table 32. TeamConnection Family Directory Structure

Directory	Comments
authorit.ld *	
comproc.ld *	
config.ld *	
interest.ld *	
relproc.ld *	
audit.log *	Audit directory removed: only one file
cfgField/ *	Definition of the configurable fields
config/ *	Place for the Security and userExit files
queue/ *	Messages stored in one file in one directory, - The addresses are in the i* files - Tthe contents are in the m* files.
tctmp/ *	Temporary directory

NOTES:

1. The items marked with * indicate items that are needed to create a family.
1. The file names in configField (CMVC) differ from the ones in cfgField (TeamConnection).

The differences in the family directory structure between CMVC and TeamConnection are shown in Figure 33 on page 182.

Table 33. Changes from CMVC to VisualAge TeamConnection

CMVC	TeamConnection
authority.ld	authorit.ld
cfgcomproc.ld	comproc.ld
cfgrelproc.ld	relproc.ld
configField/	cfgField/
queue/messages	queue/
maps	REMOVED
vc	REMOVED
<none>	tctmp/

15.2.9.2 Family Administration Tools

In TeamConnection, a GUI tool called tcadmin is the utility that is used to create and to maintain a family. Most other tools have changed names. For more details see "TCADMIN - Family Administrator's GUI" on page 198.

The format for the views of the configurable fields was changed from CMVC to TeamConnection.

15.3 New User Functions of TeamConnection

This section elaborates on new TeamConnection functions of interest to general users.

15.3.1 Sequential and Concurrent Development

CMVC and TeamConnection provide a sequential development mode, in which a file can be locked only to one user at a time.

TeamConnection also provides a concurrent development mode in which a part can be locked by more than one user at a time. In concurrent mode, parts can be checked out and back in. However, when the workarea is refreshed, such as prior to adding the workarea to a driver, collision records are created for workareas that change a part already modified by another workarea. Collisions records must be resolved, usually through the use of the VisualAge TeamConnection Merge tool (tcmerge) to combine the changes to a part in multiple workareas, prior to adding the workarea to a driver.

This concurrent mode is available on a per release basis and can be specified at the time a release is created. The mode can be changed from serial to concurrent, but it cannot be changed from concurrent to serial.

NOTE: In concurrent mode, the *workarea -check* ignores implicit prerequisites and corequisites.

15.3.2 Pruning of Releases

In CMVC, every check-in of a file created a new version number. This leads to saving versions of files that were not useful in the long term. TeamConnection lets you control the versions you keep during your development process without limiting the number of times you check your files/parts into a workarea.

By using release pruning and workarea freeze you can specify how many of the intermediate check-ins you perform will be kept and how many can be discarded after you integrate and commit a workarea.

There are two ways to accomplish the pruning:

- Manual.

You can prune a committed branch (workarea or driver) of a release by using the command "*teamc release -prune*".

- Automatic.

It is possible to specify that a release can be pruned automatically for committed branches (work areas or drivers); this is accomplished with the *+autopruning* flag in the command "*teamc release -create*" or "*teamc release -modify*".

For more information about this topic, consult the Versioning Model in Page 82, Chapter 2 "TeamConnection", in the manual "Introduction to the IBM Application Development Team Suite". In the same manual see also C.3, "TeamConnection 1, useful to TeamConnection 3 users" on page 75.

15.3.3 Versioning

In CMVC, files are versioned either by SCCS or PVCS. These utilities determine the version numbers used and reported by CMVC. Creating new versions of files increment these version numbers, then creating new releases, linking releases and breaking those releases cause branching that results in more complex version numbers for parts (for example, 1.2.1.4).

While these numbers show the relationship of all file changes across all releases that share the file, the numbers do not have a relationship to the tracks that manage the changes to the file. In other words, there are two different tracking mechanisms that need to be manually reconciled.

TeamConnection version numbers reflect the workarea used to change a particular part. For example, if part a.c is created in workarea Work1, then the first version of a.c will be named Work1:1. If more than one version of a part is checked in and frozen, the version numbers will increment sequentially (for example Work1:2, Work1:3, etc.). See “Version Numbers of TeamConnection Objects” on page 185.

The releases are the objects that are versioned, which means that the drivers, workareas and parts are also versioned. The benefit of this is that you have access to all the frozen parts of all workareas, drivers and releases that are not pruned. This provides a consistent view of all current and frozen workareas, including their change history. For more details, see “Workareas Are More Than Renamed Tracks” on page 187.

15.3.3.1 Versioning in CMVC

CMVC uses the UNIX utility SCCS as the underlying mechanism for the versioning of the files. The product PVCS is only used with CMVC on AIX 3.x (which is now discontinued).

In SCCS, numbers begin with 1.1 and the second digit is incremented with each version checked in (for example, 1.2, 1.3, etc). When a release or file is linked and then a link is broken during a check-in, a branch occurs in the numbering. For example, if release_1 is linked to release_2 when file_a is version 1.3, both releases will point to version 1.3. If only release_2 checks in a new version of file_a, the link will be broken to release_1 and the new version number will be 1.3.1.1.

The approach in PVCS is similar, but not identical.

The important point is that the numbering scheme is global and managed by SCCS/PVCS. It does not show any practical relationship between changes and the workarea/release where the change occurred.

Space is saved when storing files in CMVC in the following manner:

- SCCS/text files are stored as forward deltas by SCCS. This saves space by comparing each file to the previous version and only saving the changes. The drawback is that it takes longer to extract the most current version (it must be constructed from the previous changes).
- SCCS/binary files may be stored as reverse deltas using a reverse delta versioning scheme internal to CMVC. If the changes between the current and previous version are beyond a specified level then the whole current file is stored. Reverse delta means that the most current version is a whole file and previous versions are stored as deltas. As a result it is faster to extract the current version but slower to extract old versions.

SCCS has significant limitations with respect to the types of data that can be stored. As a result, any file that cannot be stored in SCCS is stored as a binary using the CMVC internal binary mechanism.

- In PVCS all files are stored as forward deltas by PVCS.

One drawback of the separate processing for text and binary files in SCCS is that files cannot be changed on the fly from one type to another. If you want to change the type, then you must delete and destroy the file and then recreate it with the new type.

15.3.3.2 Versioning in TeamConnection

In TeamConnection, all parts are managed and versioned in the database. Here is what we do to them:

- Text files are compressed and versioned using reverse deltas.
- Binary files are compressed and each version is stored whole.
- The release versions use reverse deltas.
- The file versions within a workarea and driver use forward deltas.

You can change the type of the part, from text to binary and vice versa, by checking it out and then checking it in and specifying the new type.

For more information about this topic, consult the Versioning Model in Page 82, Chapter 2 "TeamConnection", in the manual "Introduction to the IBM Application Development Team Suite". In the same manual see also C.3, "TeamConnection 1, useful to TeamConnection 3 users" on page 75.

15.3.3.3 Version Numbers of TeamConnection Objects

The version control of TeamConnection maintains different versions of the following objects:

- Releases

- Work areas (and driver members)
- Drivers
- Parts

When you want to find and retrieve previous versions of these objects, it is helpful to know how TeamConnection creates and deletes previous versions of each object.

- When you first create an object, the initial version name is the object name suffixed with ":1". When you create a new work area called "*myWorkArea*", for example, its version is "*myWorkArea:1*". Subsequent versions are identified in numerical order: *myWorkArea:2, myWorkArea:3, myWorkArea:4*, and so on.

Versions of releases and drivers are identified similarly:

myRelease:1, myRelease:2, myRelease:3; myDriver:1, myDriver:2, myDriver:3; and so on.

- o Unique versions of parts are identified by association with a specific version of a release, work area, or driver. Your family may have three different versions of a part called "*myPart*", one associated with release myRelease:2, one associated with work area myWorkArea:1, and one associated with work area myWorkArea:2.

15.3.4 Parts Are More Than Just Files

TeamConnection parts are more than just files. The most important changes are:

- Parts can be more than just files (of type text and binary). Parts may also be type "none" indicating that they will not be extracted as files; this is used for object level integration with other tools.
- Parts include attributes that define their behavior during build and packaging. For example, a COBOL program will be associated with a different parser than a C program, as well as a different builder that calls a different compiler with different parameters.
- Parts that are type "output" will be checked in after a build. The number of output files that can be stored in a release is set for that release.
- Parts that are "temporary" are discarded after the build, without being checked in.
- Tools may define subclasses of parts which may contain other attributes, relationships and dependent objects.

15.3.4.1 File/Part Links

From a user's point of view, parts are linked between releases and these links can be broken in the same manner in CMVC and TeamConnection. However, in TeamConnection, the version numbers are not dictated by the when links are broken, rather, the numbering is based on the workarea and release.

In TeamConnection the database information for a part is copied into each release, regardless of whether there is a link. So if a part is linked in two releases, then the database information about the file will be stored twice (in each release), but the actual contents of the file will be stored only once. That is, a part linked in two releases does not take the double amount of space as if the part were not linked.

Furthermore, in TeamConnection there is commonality between workareas when a part is linked or refreshed from another workarea in the same release.

15.3.4.2 Changing the Type of a File (Text <=> Binary)

Because text files are subject to some character manipulation (such as replacing line feeds with carriage return/line feed in OS/2 and Windows, and the insertion of keyword values), there are times when a file that has been stored as text needs to be changed to binary, or vice versa.

In TeamConnection you can change the type of a file without the need to destroy the file and then to recreate it again. You can change the type of the part, from text to binary and vice versa, by checking it out and then checking it in and specifying the new type.

NOTES:

1. The "type" field in CMVC is called "fileType" in TeamConnection.
2. There is a syntax change from CMVC to TeamConnection:
   - File -type text --> Part -text
   - File -type binary --> Part -binary
3. The Part -type flag has a new meaning in TeamConnection: it is used to define fine-grained objects.

15.3.5 Workareas Are More Than Renamed Tracks

A workarea provides a logical view of a release that includes all of your changes and excludes the changes of all other workareas that have not been promoted.

While a CMVC track is just a list of the changes for a given defect or feature within a release, a TeamConnection workarea lets you see the impact of a defect or feature without needing to compare the version numbers of every file that has uncommitted versions.

In other words, a CMVC Track just contains file changes, but a TeamConnection Workarea is a view of the entire release. You cannot build a Track, but you can build a Workarea because it includes all the parts.

In CMVC you could use a release process such as 'prototype' which would allow you to create and checkin files without the need to specify a track.

By contrast, in TeamConnection, you always have to specify the information about the workarea in order to create and checkin parts. However, in release processes such as 'prototype' it is not necessary to name a workarea after a defect or feature, that is, you can use any name that may help you identify the workarea (it has to be unique).

15.3.5.1 Multiple Workareas per Defect or Feature

A significant improvement in TeamConnection over CMVC is the ability to create more than one workarea for a defect or feature within a release. This allows you to break up the work you are doing into pieces, even to commit the work a piece at a time. You can include parts from other workareas, including currently committed drivers or releases.

15.3.5.2 Workarea Performance

For users of prototype releases it is important to periodically integrate a workarea into the release in order to maintain good performance.

15.3.5.3 The "teamc workarea -undo" Command

Workareas are versioned objects. As a result, you can undo new versions of workareas in TeamConnection. You can also undo individual part changes.

If a workarea has been frozen or refreshed from another workarea (refreshing causes an implicit freeze), then "teamc part -undo" is not enough to remove all changes from the workarea (as File -undo could do for a CMVC track). After using Part -undo on any files that have been changed since the last freeze, use teamc workarea -undo to remove each freeze that occurred.

The "teamc report -view versionView" command reports each freeze that occurred; that is, it reports each version of the workarea. From the GUI Workarea window, select Selected->Show->Versions.

15.3.5.4 Miscellaneous Features

- Configurable fields.

In TeamConnection Version 3 you can now add configurable fields to workareas.

- Part shadowing.

  In TeamConnection Version 3 you can now exploit the part shadowing capabilities for workareas. For details, see “Part Shadowing Capability” on page 195.

- o Extraction.

  In TeamConnection Version 3 you can now extract the parts associated with a workarea (full or delta).

15.3.6 Driver Has More Options Than Level

Because the CMVC map files have been replaced with state data in the database, drivers can now list all of the parts and their versions that are in a committed driver.

One side effect of the change from Levels to Drivers is that it may take longer to integrate workareas into drivers.

The "teamc driver" command has two new options that give you more flexibility.

15.3.6.1 The "teamc driver -freeze" Command

Because a driver is really a specialized workarea, you can save a snapshot of the driver's contents prior to committing. Depending on how you have set the pruning of the release, this allows you to either temporarily or permanently version the changes in a driver. This is particularly useful if you do not commit drivers frequently.

15.3.6.2 The "teamc driver -restrict" Command

Restrict is a state for drivers between integrate and commit. The purpose is to allow a release's build administrator to limit the ability of other users to change the contents of a driver while building and verifying the build.

The typical use of restrict is in conjunction with some automated promotion and build process. For example, at 7 PM every night a process is started that adds all workareas in a release that are in integrate state, that is, all the fix records are completed, to the driver by creating driver members. After adding the workareas to the driver, a build is started. Without the restrict state the only way to prevent changes to a driver while build is running is to either change all users access (affecting the entire release) or to commit the driver (preventing further changes).

By using restrict, the tool can commit the driver after a successful build or regression test. If the build or regression test fails, then notifications can be sent to the development team to make the necessary changes to complete the build and pass the regression test. At that point the build administrator can promote the changes, build, test and commit the driver without anyone else changing the driver contents.

15.3.7 Updated VisualAge TeamConnection GUI

The VisualAge TeamConnection GUI has evolved significantly from the original CMVC GUI for UNIX. Actually, some of the new features in the TeamConnection GUI were added to the CMVC GUIs for OS/2 and Windows.

Examples of new features in the TeamConnection GUI are:

- The TeamConnection Part command adds an -edit action that performs checkout, opens an editor session, then checks the part back in after the editor session is closed.
- Better context sensitive menu options when selecting an entry in a dialog, then pressing the right-mouse button.
- More options on the Settings dialog, such as selecting whether the read-only attribute should be set.

NOTE: This compares the TeamConnection V3 Intel GUI with the CMVC version 2.3.0.18 GUI for Intel.

15.3.8 New UNIX GUI

CMVC users of the OS/2 or Windows GUI are already familiar with the TeamConnection Intel GUI. The TeamConnection UNIX GUI is now based on the Intel design.

As a result of using the Intel design, the following has changed for UNIX GUI users:

- You now have a settings page for changing the defaults for options like the Component name.
- The task list queries support the use of environment variables. As a result, the default tasks that already include environment variables do not need to be customized to be useful.
- There is a significant increase in the use of icons that are associated with each TeamConnection object. For example, there is a generic icon for defect, another for component, etc; then when you are showing a list of

defects you will see the same generic icon on the left hand side for each defect.

Each generic icon is augmented with minor graphics to provide additional information on the state of the objects, such as a defect in working state has a different icon than a defect that was canceled.

- The GUI uses "containers" to show the components and drivers, and you can select a Tree-View which will show the icons with a plus or a minus sign. Then you can click on these signs to expand or contract a specific component. This provides an equivalent function to the CMVC Component Tree hierarchy.
- A command window has been added. This allows you to use line commands without out opening a shell window.
- After selecting an object (for example, a defect in the defect window), a right-mouse click provides a context sensitive menu of actions.
- You can edit and invoke queries on each object window without opening the filter window.

15.3.9 Password Loging to TeamConnection

TeamConnection adds another method of authentication that is complementary to the entries in the Host List table. Through the TCLOGIN command a client can enter a password in order to log the client session into TeamConnection. If the user is authorized, then the family server and the login manager in the client keep track of the login status and allow the user to issue other TeamConnection commands.

The family administrator can configure the family to use one of the following authentication methods:

- HOST_ONLY: only use host entries, which is the default.

  This is the only option in CMVC.
- PASSWORD_ONLY: only use the method of login with password.
- PASSWORD_OR_HOST: either a login with password or a host entry.
- NONE: no authentication method at all.

This method should be used with extreme care because every user has the potential to become a superuser.

This method should be used as the last resource in emergency situations, such as when migrating a family from one host to another and no provisions were made (such as adding a new host list entry for the new host) before the migration. In this case, shutdown the family, change the authentication method for the new family to NONE, restart the family, add the proper new host list entry, shutdown the family, change the authentication method to a more restrictive type, and restart the family.

The password option solves several problems associated with using host lists:

- Users that access a family from a large number of workstations require less effort to manage a password than a long list of host table entries.
- The potential security issue of one workstation taking over another workstation's IP address in order to gain another user's access to a family, is eliminated by passwords.
- Multiple users can access the family from the same workstation and user account without inadvertently using another's TeamConnection User ID.
- A user's login can be explicitly terminated and not taken over by another user. A user's access is terminated whenever the family server is recycled, the user logs off by using tclogin or the user logs out of the workstation.
- It is easier for a family administrator to temporarily disable a user account with passwords than deleting a user's host list entries.

NOTE: The authentication type can be changed at any time by using the tcadmin utility or by modifying the file $FAMILY_HOME/config/security. It is necessary to restart the family in order for the new teamcd daemons to use the new authentication type.

15.3.10 Planning for "teamc release/driver/workarea -extract" Actions

When doing a teamc release/driver/workarea extract, TeamConnection extracts all files directly to the client. It is no longer necessary to set the userid (uid) and the group-id (gid), or to worry about the ability to mount a client directory to the server (like with CMVC). However, planning is still required whenever you want to manage an extract to a system or to another user other than the client invoking the extract.

You may try to do an NFS mount or use any other tool at your disposal (for example, OS/2 NET USE) to place the target directory within reach of the local host where the family daemon is running and then extract to the client as you normally would.

Another option would be to issue a rexec to the target machine so that a client installed on that machine can run the extract.

15.3.11 Build Support

The function that enables you to define the structure of your application and then to create it within TeamConnection from your input parts.

You can build applications for platforms in addition to the one TeamConnection runs on; currently, you can use TeamConnection to build applications on AIX, HP-UX, Solaris, OS/2, Windows NT, Windows 95, MVS OS/390 and MVS/OE. A single build pool can include systems running any or all of the operating systems and TeamConnection will route the build to the appropriate machine.

With TeamConnection's distributed build, managed components or releases can also be built for the enterprise client/server environment (that is, for the target platforms that may be different from the server platform used for development).

With TeamConnection, a development team can set up the build structure once and be able to identify the build rules depending on the task at hand. The build rules are associated with the objects being built so there is no need to search for related build steps in a file used by a separate build tool. The software configuration management services rebuild only those objects that need to be rebuilt, based on criteria such as date changes. This minimizes the system resources required to accomplish each build reliably.

The TeamConnection software configuration management services also determine which parts of a build need to be built in serial and which can be in parallel; it then can parcel out the work to the available machines.

Building objects from different technologies, such as procedural and object-oriented languages, through a single rule-based build process reduces integration risks and complexity.

15.3.12 Packaging Support

After you complete a build, you can run a specialized build to package and even distribute your product. Currently, TeamConnection supports a generic packaging tool called "gather" and the Tivoli Software Distribution facility.

While distribution is part of the build subsystem, the TEAMCPAK utility can be run separately.

Although the build function is heterogeneous, from a practical perspective the packaging function is essentially homogeneous. Currently, the executable files are operating specific, as are most of the installation utilities used for packaging (for example, InstallShield, installp, etc.). The increasing popularity of languages like Java may allow for heterogeneous packaging in the future.

15.3.13 New Web Client

The Web Client is a TeamConnection Client that provides a subset of the functions available in the Intel or UNIX client. The name Web Client is a slightly misleading name, because it really is a specialized Web Server built into the TeamConnection family server.

The Web Client provides access to a TeamConnection family from a frame-enabled web browser (such as Netscape Navigator or recent versions of Microsoft Internet Explorer), without having to install any TeamConnection components on the client system.

The function missing from the GA version of the Web Client is the ability to manipulate part contents, that is, it is not possible to check-in, check-out, create or extract parts, although it is possible to view part contents.

NOTE: This ability will be added to the code in future fixpaks.

To access a TeamConnection family using the Web Client, all that is required is to know the hostname of the family server (or its IP dotted decimal address) and the port number of the family. The URL to be used is:

```
http://hostname:port
```

To disable the Web client, set the following environment variable in the TeamConnection server:

```
TC_WWWDISABLED
```

15.3.14 Lotus Notes Integrated Databases

The VisualAge TeamConnection Integrated Notes Database feature provides a documentation facility to support software development. A software development group can use this database to communicate with TeamConnection objects from within Lotus Notes documents. The TeamConnection Integrated Notes Database links the technical documents to the TeamConnection objects involved in software enhancement and maintenance.

A single database template is provided that you use to define the database that will assist you with all phases of your development process. From the template, databases can be produced that can assist with requirements tracking, design and development documentation, and test case management, as well as other purposes.

This function was added in Version 3.

15.3.15 Source Code Control API

Source Code Control (SCC) API, in which other applications such as PowerBuilder, Visual Basic, Visual C++ and Content Manager can checkout and checkin parts from TeamConnection, among other functions.

Any tool that this SCC API should work; however, the ones listed above are the only ones officially supported

This support was added in Version 3.

15.3.16 Part Shadowing Capability

TeamConnection now includes support for shadows. A shadow is a collection of parts in a file system that reflects the contents of a workarea, driver, or release. You can use shadows to build your product, or to provide a place outside the library where developers can go to search through code. Shadowing is similar to extracting in that the purpose of each is to provide a set of objects from a TeamConnection version or version context.

In part shadowing it is important to understand the following characteristic with respect to the CR/LF conversion: the flag -crlf means to explicitly add the CR/LF pair to all text (for Intel) while not using -crlf means to add only LF (for UNIX). For more details, see the differences subsection in “Files (CMVC) or Parts (TeamConnection)” on page 171.

This support was added in Version 3.

15.3.17 VisualAge TeamConnection For Smalltalk Bridge

VisualAge TeamConnection for Smalltalk provides a repository with operational support tailored specifically for highly-interactive, prototyping environments that emphasize iterative development, such as VisualAge Smalltalk or VisualAge Generator.

A bridge from VisualAge TeamConnection Enterprise Server Version 3 to VisualAge TeamConnection for Smalltalk provides access to the software configuration management (SCM) support provided by VisualAge TeamConnection for Smalltalk along with the scalable, enterprise-level support provided by VisualAge TeamConnection Enterprise Server's ability to manage all development artifacts (not just source code), to share information in a common model, and to integrate multiple tools and multiple languages across the enterprise on a single baseline that extends the capabilities of software development groups.

This bridge provides essential integration for VisualAge tools which use VisualAge TeamConnection for Smalltalk as their day-to-day operational library.

The bridge supports the import and export of VisualAge Generator objects (parts) to an from VisualAge TeamConnection Enterprise Server.

VisualAge TeamConnection Enterprise Server can be used to manage artifacts (parts) that need to be shared with languages that are not based on a universal virtual machine or tools for purposes of build management, problem tracking and other configuration management functions. These artifacts can be exported from VisualAge TeamConnection for Smalltalk to VisualAge TeamConnection Enterprise Server through the bridge and stored as TeamConnection parts.

Objects stored in a TeamConnection database can be queried and retrieved back into the VisualAge TeamConnection for Smalltalk development environment as needed. The units of storage in TeamConnection include exported VisualAge for Smalltalk and VisualAge Generator components (such as applications and configuration maps) and large grained objects (files). Small-grained objects, such as VisualAge Generator data items, are imported and exported as constituents of applications. The data items in an application are exported to VisualAge TeamConnection Enterprise Server in a form that makes their definitions available to other tools through the data model.

15.3.18 Part/Release Merge Function

TeamConnection provides a merge function to the part and release commands. Parts and releases from multiple sources can now be merged into a new single source.

In addition, an automatic merge tool (Automerge) has been introduced to provide automatic merging of files. In most cases, the user needs only to resolve conflicts. The actual "script writing" and execution for merging of files is now automatic.

This support was added in Version 3.

15.3.19 Translation Support Attributes for the Part Command

The following fields related to the translation of parts related to National Language Support are provided to designate the translatability of parts for the actions part -create, part -modify, and part -mark:

- translation

  This field indicates if the part is part of the translation process.

- translation state

  This field indicates if the part is ready or not ready to be translated.

New versions of a part have an initial translation state of notReady. The most recently committed part version is marked by default as notReady.

The transition of a part to the ready state is not automatic, and it must be done manually with the part -mark command. No translation states are currently defined for parts in other languages that are the result of translation.

This support was added in Version 3.

15.3.20 Object Repository and Information Model

TeamConnection integrates an object-based repository with software configuration management functions to provide a managed environment for data-related assets at all levels of business and application information.

With TeamConnection, team members of different disciplines interact with the repository to work with information in a form that makes sense for their task. The repository provides a central point of controlling and translating information by allowing access to the data through the integration of tools.

A repository provides multiple task-driven views of stored information, as well as, "inherited" services to all the objects that it manages. Tool integration with the TeamConnection repository provides the ability to decompose the processes of each discipline in a team, analyze them, and transform their managed data objects into their conceptual, logical, and physical implementations, each with their respective views.

Underlying all of the views (conceptual, logical, storage) are the object and class definitions required to support these views. This set of definitions represents the Information Model. The TeamConnection repository architecture includes a set of APIs and an open extendible information model so that tools can share data, store their unique data, and use the common software configuration management and repository services available in TeamConnection.

This capability allows for an open repository that can be extended through TeamConnection services by both tool vendors and developers of in-house tools. Specifically, TeamConnection's information meta-meta-model provides a common language for representing various tool-specific meta-models. A tool can reuse or extend TeamConnection classes to define a tool modelreflecting attributes, relationships, and behaviors of the objects to be created and used.

The information model and the functions for the tool builder are provided separately in the Tool Builder's Development Kit.

15.4 Changes That Impact System And Family Administrators

There are significant differences in the way CMVC and TeamConnection work and use system resources. The following sections address some of these differences. Much of this information is not fully documented in either the CMVC or TeamConnection administration manuals.

15.4.1 TCADMIN - Family Administrator's GUI

While TeamConnection still allows you to create, configure and manage your families using commands, there is a new GUI, tcadmin, that makes all of the configurable elements of a TeamConnection family obvious and easy to manage.

Also, you can keep track of all of your families at once by and select which family to modify from the main dialog.

For each family you can specify to start/stop the daemons and to change the following characteristics, which are presented as a notebook:

- Family information: name, location, port number, mail exit,
- Initial superuser and its password
- Security: authentication level, minimum password size, maximum number of invalid attempts.
- Configurable types (config.ld)
- Configurable fields and their report and table formats.
- Release processes
- Component processes
- User exits
- Authority groups

- Interest groups

NOTES:

1. Security (Authentication Level). For more information, see “Password Loging to TeamConnection” on page 191
2. Configurable fields. In CMVC, chfield actually interacts with the database and validates the configurable field definition files in configField. In contrast, in TeamConnection, tcadmin does not interact with the database and relies only in the configurable field definition files in the cfgField directory.

15.4.2 User Exits

The CMVC user exit paradigm is supported in TeamConnection. However, the parameters on almost all commands have changed. Further, the possible values for many of the commands have changed. For example, the file "type" in CMVC is either "text" or "binary" but in TeamConnection the values are "text", "binary" and "none". Furthermore, in several commands, the values are passed as integers (that is, 0, 1, 2) instead of text strings.

TeamConnection adds significant ease of use on top of the CMVC paradigm. You can cause a parameter file to be created that contains the values of specific parameters in a binary files. The sample program teamcenv.c allows you to extract the values from the files.

TeamConnection adds a family administration GUI that supports the setting up of user exits; see “TCADMIN - Family Administrator's GUI” on page 198.

TeamConnection now supports running TeamConnection actions from within a user exit. As with CMVC, there is always the limitation that the action cannot access the same data as the command, in such a way as to cause a database deadlock. You will need to test to determine what is an acceptable action in any user exit.

For user exits that modify the contents of files as they are extracted, checked in, etc. there is one minor change with respect to the CR/LF pair (Intel standard carriage-return/line-feed): in CMVC the text files were stored normalized to the UNIX standard (only LF), while in TeamConnection, the text files are not normalized.

As a result, the temporary file on the server that is accessible by a user exit may contain CR/LF pairs or just LF between lines. For more details see“Files (CMVC) or Parts (TeamConnection)” on page 171.

15.4.3 License Handling

15.4.3.1 License Handling in CMVC

In CMVC, the licenses for the UNIX clients are managed by the use of the product NetLS (also known as iForLS or iFor). This product enforces the number of concurrent users that can work with CMVC. However, NetLS could be difficult to install, to administer and to maintain, specially in complex networks. Further, an inefficiently running NetLS can add up a lot of overhead to each CMVC command. Because of this, the use of NetLS is one of the major causes for dissatisfaction for some CMVC customers. By the way, the licenses for the OS/2 and Windows clients do not use NetLS.

15.4.3.2 License Handling in TeamConnection

TeamConnection chose a simpler approach for the licenses: the honor system. That is, the customers will use only the licenses for which they are entitled to. In order to help them with this task, TeamConnection provides a License Monitor utility that will scan the audit log of a family server and will report the highest number of concurrent users. The number of concurrent users is defined as the number of unique combinations of TeamConnection user id, system login, and host name that use the family server in a period of 15 minutes (this default value can be modified).

15.4.4 Backup and Recovery

The family administrator has to use the DB2 facilities for backup and restore of the database.

There is more documentation on backup and recovery in the VisualAge TeamConnection Administrator's Guide and the DB2 Administration Guide manuals.

15.4.5 Which Processes Are Started

15.4.5.1 Which Processes Are Started in CMVC

CMVC has a fairly complex hierarchy of daemons, where a single daemon (called "the grandparent") is started, followed by the number of daemons requested; for example, "cmvc testfam 3" would start 3 daemon processes. This second set of daemon processes are called "parents". After all the parent processes have started, the grandparent process is terminated.

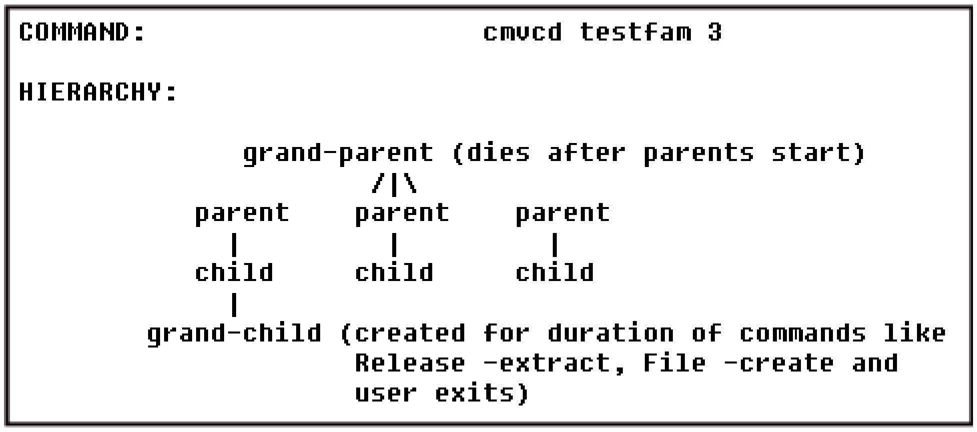
Next, each parent starts a "child" daemon that listens for CMVC clients requesting that a CMVC command be processed. So, from the example above, 1 grandparent starts 3 parents which in turn start 1 child each producing 3 children, but when the grandparent terminates, then there will be 6 processes left running.

When a command is issue that needs to temporarily change to your user ID, like "Release -extract", a grandchild is spawned for the duration of the command (with your user ID as the owner). Grand children are also generated for all file operations, where changing to the family account is necessary before running SCCS commands.

Also, when calling user exits it is necessary to spawn a grandchild running as the family account in order to prevent a user exit from getting access to root authority.

All this was done so that CMVC could handle almost any sort of failure without reducing the number of daemons available to service your requests.

The process hierarchy for CMVC daemons is shown in Figure 41 on page 201.



```
COMMAND:                      cmvcd testfam 3

HIERARCHY:

              grand-parent (dies after parents start)
                      /|\
           parent    parent    parent
              |         |         |
           child     child     child
              |
        grand-child (created for duration of commands like
                     Release -extract, File -create and
                     user exits)
```

Figure 41. CMVC Daemon Process Hierarchy

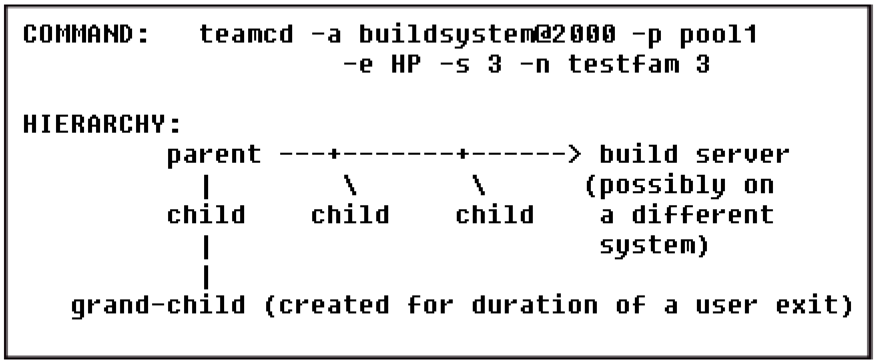
15.4.5.2 Which Processes Are Started in TeamConnection

TeamConnection has a much simpler architecture. A single parent spawns one child for each family daemon you request. Also, the parent will spawn a build server daemon as requested in the startup command.

The parent process stays around to keep an eye on all of the children processes. Should a child process die for any reason, the parent process will start another child process. Because the parent process does not open the database, it does not use database resources.

TeamConnection is not a setuid process and does not need to change user IDs, thus, it only needs to spawn grandchildren for such things as user exits.

The process hierarchy for TeamConnection daemons is shown inFigure 42 on page 202.



```
COMMAND:   teamcd -a buildsystem@2000 -p pool1
                    -e HP -s 3 -n testfam 3

HIERARCHY:
         parent ---+-------+------> build server
           |        \       \      (possibly on
         child    child    child    a different
           |                        system)
           |
   grand-child (created for duration of a user exit)
```

Figure 42. TeamConnection Daemon Process Hierarchy

NOTE: In order for processes in TeamConnection to start and stop properly, you should start all processes using the options on the teamcd command. This will ensure that tcstop will find all processes and terminate them.

15.4.6 Use of Disk Space and Memory

15.4.6.1 Directory /tmp on UNIX

CMVC requires that /tmp (or the directory pointed to by the TMP or TMPDIR variable) must have free space equal to three times (3X) the largest file to be checked into the family. This is pretty vague, because you rarely know how big your largest file will be.

TeamConnection uses /tmp and $FAMILY_HOME/tctmp on a per process basis.

15.4.6.2 Disk Space for the Family Account

In CMVC, almost all of the disk space is allocated to the vc directory structure. This is where the files are stored. When a file system becomes too big, we recommend that you select a subdirectory (such as, $HOME/vc/0/2) and symbolically link that directory into another file system. This allows you to overcome most file system size limits.

The next largest directory is where you store the database backup or export.

Space Used by Database

In TeamConnection, all files and other data are stored in the database. The exception is the set of *.ld files (which contents is loaded into the database by tcadmin) and the contents in the bin, config and cfgField directories in the family HOME directory.

The "bulk data" table resides in its own table space. Thus, it has the potential for expansion by adding more devices.

Space Used by Family Daemons

The TeamConnection temporary directory, /tmp and $FAMILY_HOME/tctmp, stores a temporary copy of each part of type "tcpart" while it is being checked in or out. The other files stored in the temporary directory tend to be small, so no significant space needs to be allocated.

15.4.6.3 Use of AFS, DFS and NFS

Network distributed file systems such as AFS (formerly known as Andrew File System) and NFS (Network File System) are commonly used for the home directories of users. Also, DFS (Distributed File System) is an industry standard component of DCE (Distributed Computing Environment) and a replacement for AFS.

The TeamConnection and CMVC development and support teams have always discouraged their use for the contents of family accounts or databases, because NFS can be unreliable and because AFS/DFS tokens can expire; therefore, we strongly recommend that the database and contents of family accounts must reside physically on the systems running the CMVC daemons. We do understand that customers have successfully moved less frequently used files in the vc directory structure to NFS mounted drives, but we hope that is limited to a last resort.

Please note that the TeamConnection databases will not work in AFS, and currently do not work in DFS file systems.

15.4.7 Family Daemons and Security/Integration Issues

15.4.7.1 Process Privileges

The CMVC daemon processes run with the setuid bit on and are owned by root. This allows CMVC to become a specific user (other than the family account) as needed, as well as running the mount command which is owned by root.

TeamConnection does not need to use the mount command or change the user id, therefore the TeamConnection daemons do not use the setuid bit and need NOT be root processes.

The following CMVC security/integration issues are eliminated by this TeamConnection architecture:

o There is no need for TeamConnection daemons to be owned by root and this eliminates the possibility of users with no root authority to "steal" or "misuse" the root authority.

o The functionality of the CMVC Release/Level extract was changed in TeamConnection Release/Driver extract so that, NFS mounts are NOT needed in TeamConnection. Thus, this change eliminates the need for use of the mount command or to change to a user ID. Another benefit is the elimination of mount failures due to differences in TCP/IP of various operating systems.

15.4.7.2 Access to Data in the Family Account

In CMVC, it is necessary for all of the files and directories in the vc directory to have read access for group and other. Even though it is not possible to determine which file is which without access to the database, this is still a security issue.

In TeamConnection, files are stored in the database in a way that is not directly accessible to end users. Further, you can now be more restrictive in the file and directory permissions you use for your family account.

15.4.7.3 System Integration Issues

The TeamConnection daemon does not have to interact with SCCS. The greatest benefit is that all the work happens in a single database transaction, which improves data integrity.

This also eliminates 2 environment variables, as well as a long list of miscellaneous restrictions on the format of "text" files, as well as a lengthy list of obscure SCCS error messages.

15.4.7.4 Other Issues

The teamc report command now has access controls. It is no longer possible for a user without permissions via component access lists to see the data through the report command.

Chapter 16. Migrating to TeamConnection Version 3

TeamConnection provides tools for migrating from CMVC to TeamConnection (called migcmvc) and from TeamConnection Version 2 to Version 3 (called migtc). This chapter explains how to use these tools.

This chapter contains the following sections:

- Migrating from previous versions of TeamConnection to Version 3
- Migrating from CMVC to TeamConnection
- Using the migration tools

If you are currently using a previous version of TeamConnection or CMVC, and you want to begin using TeamConnection Version 3, then you must migrate your database to TeamConnection Version 3.

For previous TeamConnection versions' customers, because TeamConnection Version 3 uses DB2 Universal Database instead of ObjectStore, you cannot simply install Version 3 over your current installation. Instead, you need to install TeamConnection Version 3 on a different server from your actual version's server and migrate your database to the Version 3 server using the migtc tool.

For CMVC customers, although the TeamConnection product was developed as the next generation of CMVC, much of the underlying functionality has changed. For this reason, you need to migrate your CMVC database to TeamConnection Version 3. To migrate your database from CMVC to TeamConnection, you need to install TeamConnection Version 3 on a different server from your CMVC server and migrate your CMVC database to the Version 3 server using the migcmvc tool.

Refer to “Using the Migration Tools” on page 235 to learn more about the migration commands before beginning the migration process.

The following list of terms used in this chapter will help you understand the migration process:

original database–The original TeamConnection or CMVC database before migration.

migrated database–The new TeamConnection Version 3 database after migration.

source server–The original family server (TeamConnection 2 or CMVC) from which you are migrating your database.

target server–The new TeamConnection Version 3 family server to which you are migrating your database.

new source server–A relocated source server. This term is used to refer to a source server that has been moved from its original production machine.

We recommend that you make a backup copy of your family database and file structure before you begin migration. After you have backed up your family database, restart it in maintenance mode so that no updates can be made to it from client machines while the migration is being performed.

By convention, along this chapter, we will suppose that you pre-migration server is a TeamConnection Version 2.0.9 (or later) family server. The migration process is identical for previous family servers versions, even old TeamConnection Version 1.x family servers.

16.1 Migrating from TeamConnection Version 2 to Version 3

TeamConnection Version 3.0 cannot be installed over any previous version of TeamConnection. Instead you need to migrate your previous database to a new TeamConnection Version 3 server. Existing Team- Connection customers must migrate their data if they want to be able to use it on TeamConnection Version 3.

Note: The following configuration is not necessary on UNIX platforms, but we suggest it.

To migrate to TeamConnection Version 3, you need two separate server

machines:

- A TeamConnection Version 2.0.9 (or later) family server with your original database. This server must also have a superUser ID and a host list for the superUser so that the target server can access the source server. You must have TeamConnection Version 2.0.9 (or later) installed before you begin migration.
- A separate TeamConnection family server with TeamConnection Version 3 client and server components installed, to which you will migrate your database. This server must be installed and configured as described in the Installation Guide.

You can migrate your TeamConnection Version 2 database to any platform supported by TeamConnection Version 3.

16.1.1 General Guidelines

The user ID of the person responsible for the migration must have superUser authority on the source server to guarantee access to all data for migration. This authority will be migrated from the source database to the target database. The superUser must have a host list entry to the target server.

We recommend that the migration be done in stages with frequent backups of the database. You can use where clauses to capture and migrate a limited or expanded set of data. With frequent backups, you can reset the database to the steps prior to this change and then repeat the migration with the modification.

If your TeamConnection Version 2 database contains fine-grained data, you will need to export it to .cdf files and then import the .cdf file into your Version 3 database. The import and export actions are not performed by migtc, but by issuing TeamConnection line commands instead. Migrating fine-grained data requires careful planning. The following is one way you can incorporate migration of fine-grained data into the migration process:

1. On your source server, export your fine-grained data to .cdf files using the teamc release -export and teamc workarea -export commands. See “Preparing the Source Server for Migration” on page 220 for instructions.
2. On your target server, migrate your database to TeamConnection Version 3. See “Performing the Migration” on page 211 for instructions.
3. Copy the .cdf files from your source server to your target server.
4. Import the .cdf files into your migrated database on the Version 3 target server.

See “Importing Fine-Grained Data into Your Migrated Database” on page 219 for instructions.

16.1.2 Preparing the Source Server for Migration

There are several steps you can take before you begin migrating to help the migration process execute more smoothly:

- All drivers and work areas should be integrated and all drivers committed before beginning migration. All test records and verify records should be completed.

- CollisionViews are not migrated. In the 2.0 database, if a part is checked out in two different work areas, you need to resolve all collision records before migrating the part. Refer to the *User's Guide* for instructions on part reconciliation.
- Create a user ID with superUser access to your original and migrated databases and a host list for the user ID. This ID will be used to access the database from the target server.

  You may want to run queries like the following and save the output to a file:

  teamc report -view workareaView >workare1.vew

  teamc report -view releaseView >release1.vew

  teamc report -view compView >compon1.vew

  teamc report -view defectView >defect1.vew

  teamc report -view featureView >feature1.vew

  Note: These queries may result in very large output files.

  These commands capture the data that will be migrated. You can experiment with *where* clauses to limit the objects to be migrated.
- If you have fine-grained data in your original database, you need to export it to .cdf files. If you do not have fine-grained data, you can skip this step.

  If you have releases with at least one integrated work area, issue the following command on your source server for each release to export the release to a .cdf file.(Specify a user ID with superUser authority on the -become attribute: teamc release -export *releaseName* -file *releaseName*.cdf -become *superUserID* .)

  If you have a nonintegrated work areas, issue the following command on your source server for each nonintegrated work area to export the work area to a .cdf file:

  ```
  teamc workarea -export workareaName -release releaseName
  -file workareaName.cdf -become superUserID
  ```

  Be sure to record the following information for each export action:

  - The name of the .cdf file
  - For work areas, the name of the release it belongs to.
- Stop the family server and use the xcopy command with the /s and /e parameters (in Intel), cp -r in UNIX environments, or an equivalent command, to create a backup copy of your TC_DBPATH directory structure and all files in it (including all .ld files, all .fmt and .tbl files). Make sure these files do not get copied to your target server.

- Restart the family server in maintenance mode (using the -m option) so that no updates can be made to the server while migration is in progress. Client users can view information, but not change it.

TeamConnection version 2 databases are not portable. We recommend that you keep the TeamConnection Version 2 server installed and operational on its original machine. If you do move the Version 2 database to a different machine before migration, please do the following:

- Use xcopy, cp, or other environment-appropriate command to copy the files in the source server’s TC_DBPATH directory to the new source server. Use the exact same TC_DBPATH. The drive and directory for the database must exactly match the original.
- On the new source server, on the first line of the TCP/IP hosts file, place the IP address of the server followed by the hostname and address of the source server, a space, and then the alias, as follows:

  1.11.111.111 testfam.company.com testfam

If you plan to use your source server as your TeamConnection Version 3 server, then you need a new IP address and hostname for this machine, since its original hostname has been transferred to the new source server.

16.1.3 Setting up the Target Server

You run the migration tools from the TeamConnection client component of the target server, which communicates with your source server to migrate the database. Install TeamConnection Version 3 on your target server according to the instructions in the Installation Guide:

- If TeamConnection Version 2 is installed on the machine you intend to use as your target server, you must uninstall it before installing TeamConnection Version 3 and then reboot. If you remove a TeamConnection Version 2 installation, you may have to manually delete some files from the TeamConnection Version 2 installation directory.
- Install the TeamConnection Version 3 client and server components and perform the installation verification procedure to create the testfam database. Make sure you complete the installation process, including updating your TCP/IP hosts and services files with IP addresses for your target family.

  Set or modify the following environment variables in the Environment page of System Properties (Windows NT) config.sys (OS/2) or .profile (UNIX) on the target server. Set this variable to the name of your migrated family database on the target server:

SET TC_FAMILY= migratedDatabaseName

Set this variable to the family name, host name, and port ID of your original database on the source server:

SET TC_FAMILY_CLIENT= originalDatabaseName@hostname@port

Set this variable to a user ID with superUser authority:

SET TC_USER= superUserID

Set this variable to a user ID with superUser authority:

SET TC_BECOME= superUserID

The following are examples of these environment variables, using v3dbase as the original family name and v3dbase as the migrated family name:

SET TC_FAMILY=v3dbase

SET TC_FAMILY_CLIENT=v3dbase@v3server@2222

SET TC_USER=migrator

SET TC_BECOME=migrator

- Reboot the machine.
- Determine the directory or path where the target database will be located. This directory path will be your TC_DBPATH. This path is specified when your database administrator uses the TeamConnection Administration GUI to create the new database instance.
- Set up a directory structure from your new subdirectory for your migrated database similar to the testfam sample structure described in the installation verification procedure. You can copy these files from the \testfam subdirectory created when you ran the installation verification procedure for your version 3 installation:

```
teamc
cfgfield
Defect.fmt
Defect.tbl
Feature.fmt
Feature.tbl
Part.fmt
PartView.fmt
User.fmt
config
userExit
security
```

Note: The TeamConnection administrator is responsible for migrating the .ld files after migration. See “Miscellaneous Tables” on page 234.

- When migrating a single, committed version of each file from TeamConnection version 2, you must first do a release -extract, file -extract, or part -extract, to get the files you want to migrate out of the source database. You must then move the files to the TeamConnection version 3 server machine to migrate them into the target database. You can move the files by using a copy command, FTP, or any other available method.
- Verify that your Version 2 server and ObjectStore are running on the source machine.
- Verify that the database is running on your target server.

16.1.4 Performing the Migration

Once you have prepared your source and target servers, you are ready to perform the migration. The TeamConnection migration utility, migtc, is a command-line utility that you run from the TeamConnection client and server installed on your target server. It uses the information you provided in “Setting up the Target Server” on page 209 to locate and access the source database.

TeamConnection provides a file called migtc.lst that contains sample migration commands. This file is located in the /bin subdirectory of your TeamConnection Version 3 installation path. It may be helpful for you to start with this command file, modify it to suit your needs, and then execute the migration using this command file.

16.1.4.1 Migrating the Current Version of Objects

The following are sample contents of migtc.lst. The commands in this file are written for the following migration functions:

- All objects are migrated, for example, all users, all parts, all defects, and so on.
- Only the current versions of parts (files) are migrated.

If you want to select specific objects to be migrated, you will need to modify this command file by adding *where* clauses to the commands. For example, if you have deleted parts from a release, you may choose not to migrate those parts by using a *where dropDate is null* clause. See “Using the Migration Tools” on page 235 for information on these specific migration commands:

```
set batchsize 100
set decache 1
set maxerrors 100
```

```
migrate Users
migrate HostView
migrate Authority
migrate Interest
migrate Cfgcomproc
migrate Cfgrelproc
migrate Config
migrate CompView where 1=1 order by addDate
migrate BcompView where name=©root©
migrate ReleaseView
migrate EnvView
migrate AccessView
migrate NotifyView
migrate DefectView
migrate FeatureView
migrate BuilderView
migrate ParserView
migrate DriverView where state in (©complete©, ©commit©)
migrate WorkAreaView where state in (©commit©,©complete©)
set top x:\sourceCodeTreeStructureforReleaseName1
migrate PartView -release yourReleaseName1
set top x:\sourceCodeTreeStructureforReleaseName2
migrate PartView -release yourReleaseName2
migrate FixView
migrate ApproverView
migrate ApprovalView
migrate SizeView
migrate TestView
migrate DriverMemberView
migrate VerifyView
migrate NoteView
migrate CoreqView
quit
```

To use this command file for your migration, you need to make several (minimal) modifications. For a complete list of migration commands, see “Using the Migration Tools” on page 235. First, it is wise to divide this command file into several smaller files, such as migtc1.lst, migtc2.lst, and so on, and to back up your database after executing each chunk. Dealing with this command file in chunks will enable you to back up your database at regular intervals during the migration.

Next, the set top command is used to point to the source code tree structure of parts when you migrate only the current version of parts. If you want to migrate only the current version of the parts, do the following:

1. Create the source code tree structure on the target server or on a LAN drive to which the target server has read/write access.
2. Change the set top variable in migtc.lst to point to the directory structure.

The following commands migrate TeamConnection parts. Modify these commands to include the release names defined in your original database. If you have only one release, delete one of these commands:

migrate PartView -release yourReleaseName1

migrate PartView -release yourReleaseName2

If your original database contains fine-grained data, you need to modify these commands as follows to exclude the fine-grained data from the migration.

These are the commands and their modifications:

migrate PartView -release yourReleaseName1 where partType=©file©

migrate PartView -release yourReleaseName2 where partType=©file©

After the database is migrated, you will import the fine-grained data that you exported from your original database in “Preparing the Source Server for Migration” on page 207.

16.2 Migrating Previous Versions

We do not recommend that you migrate all of your past versions, as this can be very time consuming, and you will most likely not use all of this information. Instead, keep your Version 2 database as an archive and retrieve previous versions of parts when you need them. If you want to migrate multiple versions, however, you must have TeamConnection Version 2.0.9 or later installed.

To migrate previous versions, you need to create two migration command files containing the following commands. Make the minimum required changes to these command files as described previously. The set top variable must remain null, as shown in this example:

```
set top
set maxErrors 100
set decache 1
set batchsize 100
migrate users
migrate hostView
migrate authority
migrate interest
migrate cfgcomproc
migrate cfgrelproc
migrate config
migrate compView where 1=1 order by addDate
migrate bcompview where name=©root©
migrate releaseView
migrate envView
migrate accessView
migrate notifyView
migrate defectView
migrate featureView
migrate builderview
migrate parserview
migrate driverview
migrate workareaview
migrate changeview
```

Execute the commands in this command file and then quit and save your database. See “Executing migcmvc.lst” on page 231 for instructions. Then execute the second command file containing the following commands:

```
migrate versionview where 1=1 order by adddate
migrate partfullview -release oneOfYourReleaseNames
migrate partfullview -release anotherRelease
migrate partfullview -release anotherRelease
migrate partview -release oneOfYourReleaseNames
migrate partview -release anotherRelease
migrate partview -release anotherRelease
migrate workareaview
migrate fixView
migrate approvalView
migrate approverView
migrate PartsOutView
migrate SizeView
migrate TestView
migrate DriverMemberView
migrate VerifyView
migrate CommonParts
migrate NoteView
migrate CoreqView
```

WorkareaView is migrated twice because the branch point needs to be migrated following migration of the versionView data.

Note: See the product readme.txt file for any additional migration commands and utilities not included in the command files provided in this document.

16.2.0.1 Executing migtc.lst

To migrate your database follow these steps:

1. Make sure your source and target servers are set up properly and the source server is running in maintenance mode.
2. Start the database on the target server.
3. Set TC_NOAUTH=YES.
4. From the target server, run TeamConnection report commands against the source server to make sure you are connected:

   ```
   teamc report -view users -family sourceFamilyName
   ```

5. Ensure that migtc.lst and migtc.exe (migtc in UNIX environments) are in the \bin subdirectory of your TeamConnection Version 3 installation path. Modify migtc.lst to suit your needs.
6. From a command prompt on your target server, type the following command to start the migration tool:

   ```
   migtc
   ```

   The `migtc` command displays a header and the `migtc` command prompt. The header contains information about your source and target servers. The `migtc` command prompt consists of the family name and > symbol.

   ```
   testfam>
   ```

   You can capture output to a file by starting the migration tool using the following command:

   ```
   migtc 1>mig.out 2>&1
   ```

   In a UNIX environment, you can send the output to a file and to your screen by issuing the following command:

   ```
   migtc 2>&1 | tee mig.out
   ```

7. At the migtc command prompt, enter the following command to execute the migration commands in migtc.lst:

   ```
   exec migtc.lst
   ```

As you migrate the database, you may notice the following messages and behavior:

- When users are migrated you will see one fhccomp record migrated. This is the root component for the family.
- When VersionView is migrated, you may see a message that releaseName:1 and releaseName are already loaded in the target database. This is not an error.

- When BuilderView is migrated, you will see a message for null builders indicating that there is no source to extract.
- Migrating NoteView combines the actions from the history section into a single note. The number of notes migrated is likely to be to be less than the number of records for NoteView in the source database.
- If you are migrating all version of parts, there may be more part extract actions than there are records in the report from the source database. This occurs for the following reasons:
  - There are additional part extracts for each part that has been changed in a work area.
  - There are additional part extracts for each part that has been changed in a driver before the driver was refreshed from the release.
- If you are migrating all versions and your database contains parts with no contents, you will see five attempts to extract the part. The part will be migrated to the target after five attempts and the part type will be empty.

8. If you have partitioned this command file into several smaller files, back up your database between the execution of each command file using the following command, or an equivalent:

   ```
   copy d:\migratedDatabasePath\dbname.tcd
   d:\migratedDatabasePath\dbname2.tcd
   ```

   Substitute your TC_DBPATH for migratedDatabasePath and your database name for dbname. Give the target file for the copy command a different name every time you back up the database during the migration process.
9. After migration is complete, exit the migtc command interface and start the ObjectStore server on the target server.
10. Start the migrated family server on the target server.
11. In Version 3, users can view data only if they have been granted access or are:

- A superUser
- A component owner
- A part owner

You can grant users access to each component with the appropriate authority using a command like the following:

```
teamc access -create -login user -authority authGrp -component compName
```

12. To verify that you have migrated all necessary data from the source database, run queries like the following. Compare the output from these queries to the output from the queries you ran from the source server in "Preparing the Source Server for Migration" on page 207. The queries are:

```
teamc report -view workareaView >workare2.vew
teamc report -view releaseView >release2.vew
teamc report -view compView >compon2.vew
teamc report -view defectView >defect2.vew
teamc report -view featureView >feature2.vew
```

For more information on the migration commands, see "Using the Migration Tools" on page 235.

16.2.1 Preparing to Administer Your New Database

You cannot use your existing TeamConnection Version 2 configurable field or user exit files with your version 3 database. The following sections provide guidelines for adding or updating entries from your existing files to your Version 3 database.

16.2.1.1 Miscellaneous Tables

Your config.ld, authorit.ld, interest.ld, relproc.ld, and comproc.ld files have been modified since the Version 2 release, and the Version 2 files are not valid for a Version 3 database. Generate new .ld files from the database by redirecting the raw output of a report. The following commands show how to create these files:

```
teamc report -raw -view config > config.ld
teamc report -raw -view authority > authorit.ld
teamc report -raw -view interest > interest.ld
teamc report -raw -view cfgrelproc > relproc.ld
teamc report -raw -view cfgcomproc > comproc.ld
```

To migrate Version 2 configurable field data, you must merge the content of the Version 3 files created above your previous set .ld files. Your can use the TeamConnection TCMerge tool to update authorit.ld, interest.ld, relproc.ld, and comproc.ld. You must addVersion 2 config.ld entries to the Version 3 config.ld file manually.

Copy the .ld files to your TC_DBPATH where your migrated database exists. You should use these .ld files only if you want to add, delete, or change any of the values. See Appendix D, "TeamConnection Authority Groups" on page 283for creating or modifying these groups. For instructions on using the Family Administrator GUI see Appendix D.2, "Using the Graphical User

Interface” on page 283. For instruction on using family administrator line command see Appendix D.3, “Manual Modifications:” on page 284.

If you have modified the .ld files, you will need to reload the tables according to the instructions in Appendix D.4, “Reloading the authority table” on page 285.

16.2.1.2 User Exit File

User exits from Version 2 cannot be used with Version 3. Use the userExit file shipped with TeamConnection Version 3. Move the userExit file to the \config subdirectory of your TC_DBPATH directory. Any user exits written for Version 2 will have to be rewritten since the infrastructure has changed from Version 2 to Version 3.

Use the sample viewexit.cmd located in the samples subdirectory of the TeamConnection installation path to see the new order of user exit parameters. You need to concern yourself mainly with any parameters being searched and returned. The viewexit.cmd sample will help you determine the new order of parameters.

16.2.1.3 Configurable Field Definitions

You cannot use your Version 2 .tbl files with a Version 3 database. To preserve the Version 2 configurable field information, you must add Version 2 .tbl entries to the Version 3 .tbl files manually. See “Appendix A. Family administration commands” of the TeamConnection Version 3 Administrator Guide, page 205 for additional instructions.

16.2.1.4 Mail Exits

If you are migrating from TeamConnection Version 2, you can use your Version 2 mailexit file unless you are also migrating to another platform. If you are migrating to another platform, use the TeamConnection Version 3 mailexit file designed for your target platform.

16.2.1.5 Initialization File

If you want to preserve queries you’ve written in your task list, you can use the teamcv3.ini file (for Intel) or teamcv3x file (for UNIX), as follows:

```
copy teamc20.ini teamcv3.ini
```

or

```
cp teamc20.ini teamcv3x.ini
```

16.2.2 Importing Fine-Grained Data into Your Migrated Database

Once your database and all related files have been migrated to your target server, you can import your fine-grained data into it. The fine-grained data is contained in a set of .cdf files that you created in “Preparing the Source Server for Migration” on page 207.

To import fine-grained data into your migrated database, follow these steps:

1. Copy the .cdf files from the source server to the target server.
2. Create one work area for each releaseName.cdf file created when preparing for the migration. To create a work area, issue the following command:

   ```
   teamc workarea -create -name workAreaName -release releaseName -family
   familyName
   ```

Make sure you specify the release as defined in your original database. Refer to the notes you made in “Preparing the Source Server for Migration” on page 207.

3. To import your releases, issue the following command for each releaseName.cdf file:

   ```
   teamc workarea -import workareaName -release releaseName
   -file releaseName.cdf -become superUserID
   ```

4. Integrate each work area you created for this process.
5. Create one work area for each workAreaName.cdf file created by issuing a teamc workarea -export command. To create a work area issue the following command:

   teamc workarea -create -name *workAreaName* -release *releaseName* -family *familyName*

   Make sure you specify the release as defined in your original database. Refer to your notes made during “Preparing the Source Server for Migration” on page 207.

6. To import your work areas, issue the following command for each workAreaName.cdf file:

   ```
   teamc workarea -import workAreaName -release releaseName
   -file workareaName.cdf -become superUserID
   ```

7. For -release *releaseName* make sure you specify the release to which the work area belonged in your original database. Refer to the notes you made during “Preparing the Source Server for Migration” on page 207.
8. Integrate each work area you created for this process.

16.3 Migrating from CMVC to TeamConnection Version 3

Existing CMVC customers must migrate their data if they want to be able to use it on TeamConnection Version 3. To migrate to TeamConnection Version 3, you need two separate server machines:

- A CMVC 2.3.12 family server with your original database. This server must also have a superUser ID and a host list for the superUser so that the target server can access the source server.
- A separate TeamConnection family server with TeamConnection Version 3 installed, to which you will migrate your database. This server must be installed and configured as described in the Installation Guide. No database should exist in the TC_DBPATH location of the target server. This server must also contain a CMVC 2.3.12 client with a superUser ID for accessing the source database.

16.3.1 General Guidelines

The user ID of the person responsible for the migration must have superUser authority in the source system to guarantee access to all data for migration. This authority will be migrated from the source database to the target database.

We recommend that the migration be done in stages with frequent backups of the database. You can use *WHERE* clauses to capture and migrate a limited or expanded set of data. With frequent backups, you can reset the database to the steps prior to this change and then repeat the migration with the modification.

16.3.2 Preparing the Source Server for Migration

There are several steps you can take before you begin migrating to help the migration process execute more smoothly:

- Complete all levels and tracks before beginning migration to ensure accurate results.
- Tracks (in CMVC) in the integrate state are migrated to the Fix state (in TeamConnection) because there is no way of knowing which files have been checked out in relation to the track when moving to TeamConnection Version 3. Fix records in the active state are ignored during migration. Ensure that tracks (in CMVC) are in the Complete state before beginning the migration. For any tracks not in the Complete state, the owner will have to manually check out the files, replace them with the appropriate

files containing the modifications, check the files into the work area, and integrate the work area.

- Files from CMVC in the check-out or Locked state are not migrated to the target database because the migration tool does not know the work area or defect that the locked file is associated with. Before migrating, check in or unlock all CMVC files.
- Level members in CMVC are migrated to a driver in the target database. Because only committed and complete levels are migrated to TeamConnection Version 3, level members for levels in the Integrate state are not migrated. Ensure that all levels in the Integrate state are committed or completed before migrating.
- Create a user ID with superUser access to your original database and a host list for the user ID. This ID will be used to access the database from the target server.
- Convert SCCS keywords in text files to be compatible with Team-Connection keyword support. TeamConnection provides several scripts that aid this process. These scripts can be found in the samples subdirectory:
  - `FileExtract2.migcmvc` changes the keywords and generates a list of text files for all releases to be migrated. This script is added to the file -extract step of the migration process as a user exit to the file -extract command to be executed during migcmvc with change history (full) migration.
  - `keywordConversion.migcmvc` is a script to be executed after performing a release extract to prepare for migrating files with keywords.

  See “Converting SCCS Keywords” on page 222 for complete instructions for converting keywords.
- You may want to run queries like the following and save the output to a file. These commands capture the data that will be migrated: (You can experiment with *WHERE* clauses to limit the objects to be migrated.)

  ```
  cmvc report -view trackView >track1.vew
  cmvc report -view releaseView >release1.vew
  cmvc report -view compView >compon1.vew
  cmvc report -view defectView >defect1.vew
  cmvc report -view featureView >feature1.vew
  ```
- Restart the family server in maintenance mode (using the -m option) so that no updates can be made to the server while migration is in progress. Client users can view information, but not change it. Table 34 on page 222 shows how certain objects in a CMVC database are migrated into

TeamConnection. These tables indicate which states can be migrated and how the states of objects might change from CMVC to TeamConnection.

Table 34. Migrating CMVC Objects to TeamConnection

Object	State in CMVC	State in TeamConnection
Tracks	approve	approve
	cancel	cancel
	fix	fix
	integrate	fix
	commit	commit
	complete	complete
Fix	active	(not migrated)
Levels	working	not migrated
	integrate	not migrated
	commit	commit
	complete	complete

Defects in the working state prior to migration will be in the working state following migration. Any file changes that were checked into the database prior to migration but did not have completed fix records are not migrated. Following migration, the owner of the defect must create a work area, check out the necessary files, make changes, check the changes in, and integrate the work area.

16.3.2.1 Converting SCCS Keywords

Keywords that have been inserted into files can be expanded during file extracts so that you can determine their version after the files have been delivered. This operation is accomplished by requesting the **Expand keywords** option during a file extract.

Expanding keywords is essential in large systems where updates do not always replace all files in your product. CMVC and TeamConnection use different formats for keywords, so during a database migration, the CMVC keywords need to be converted to TeamConnection format.

TeamConnection provides the following script that helps you convert CMVC keywords to TeamConnection format:

- `keywordConversion.migcmvc` prepares files with keywords for migration. You use this script only if you extract files before migrating.
- `FileExtract2.migcmvc` changes the keywords. This script is added to the file -extract step of the migration process as a user exit to the file -extract command to be executed during migcmvc with change history (full) migration.

These scripts use the following tools:

- sed to make changes to the files
- grep to make sure files need to be changed
- CMVC to find all files that have a file type other than binary (such as text, long, longSp, special) and have not been deleted

Do the following before you run these scripts:

- Set the CMVC_FAMILY environment variable to your CMVC family name
- Set the CMVC_TOP environment variable to the directory path to which you are extracting.

The conversion will change only those CMVC keywords that have TeamConnection equivalents. Keywords that do not have TeamConnection equivalents will be left as is in the files.

There are two ways to use these scripts:

First, if you extract levels or releases before you begin the migration process, use these scripts as follows:

1. Extract the files to directory structures using the level or release extract command. Do not use the file extract command.
2. Run `keywordConversion.migcmvc` to prepare files before the migration. Start this script as follows:

   ```
   keywordConversion.migcmvc CMVC_FAMILY CMVC_TOP
   ```

   Only .c and .ksh text files will be converted. You will have to modify the script to convert other file types. Binary files will not be converted.
3. Run `migcmvc`.

Use the second way if you extract files as part of a change history migration. When using this scenario to convert keywords, you must start the CMVC family with at least two daemons. Otherwise, the user exit will issue report commands that will deadlock the family. Then, do the following:

1. Add user exit FileExtract2.migcmvc to file -extract user exit ID 2. (See "Chapter 9. Providing User Exits" on page 105 of the TeamConnection Administrator Guide for instructions on setting up a user exit).
2. Run `migcmvc`.

The file /tmp/checkreps.releaseName is a structured file containing the release and path name for each text file in all releases to be migrated. The script appends to this file for each release. Be sure to delete /tmp/checkreps.releaseName each time you want to generate a new list of files in it.

Keywords are converted to TeamConnection format as follows:

- The concept of current date/extract date is not relevant to TeamConnection. The last updated date is used during extract. As a result, the extract dates are commented out and the update dates are used:

```
%D% -> /* Use $ChkD, get date NA */
%H% -> /* Use $ChkD, get date NA */
%T% -> /* Use $ChkD, get date NA */
%E% -> $ChkD;
%G% -> $ChkD;
%U% -> $ChkD;
```

- Some of the keywords do not have a direct translation in SCCS (they were derived from PVCS): $Own is an example.
- All references to internal information exposed by SCCS are not converted to TeamConnection external values. As a result several values convert to a single FileName:

```
%F% -> $FN;
%M% -> $FN;
%P% -> $FN;
```

- Versioning information has been consolidated. As a result, the following keyword changes occur:

```
%I% -> $Ver;
%W% -> $Ver;
%R% -> /* Use $Ver */
%B% -> /* Use $Ver */
%S% -> /* Use $Ver */
```

- Ideally, the sed script should locate the last keyword and insert the end-replace keyword $EKW after it. However, that is a personal preference, left for users to modify as they see fit.

If the conversion scripts detect keywords that are not supported, they will generate comments stating this fact. The following is a brief description of some of the SCCS keywords. If a keyword does not have a TeamConnection equivalent, it will remain as is in the file.

$Own; The owner of the component that manages the part.

$KW=@(#); Begin keyword expansion after this keyword.

$EKW; End keyword expansion until the next $KW=@(#); keyword.

%Z% SCCS keyword converts to **$KW=@(#)**; in TeamConnection.

%A% Not applicable in TeamConnection

%Y% Not applicable.

%C% Not available in TeamConnection

16.3.2.2 Migrating Renamed Parts

In order for renamed parts to be migrated correctly, the change view in the CMVC family must be dropped and recreated. To do this, issue the following CMVC commands in the order shown:

```
stopCMVC familyname
drop view changeview
create view ChangeView
(pathId, trackId, fileId, versionId, levelId, type,
defectPrefix, defectName, defectReference, defectAbstract, defectType,
releaseName, pathName, versionSID, userId, levelName, newPathName,
userName, fileType) as
select
c.pathId, c.trackId, c.fileId, c.versionId, c.levelId, c.type,
d.prefix, d.name, d.reference, d.abstract, d.type,
n.name, p.name, v.SID, c.userId, b.name, f.basename, u.login, f.type
from
Changes c, Tracks t, Defects d, Releases n, Path p, Versions v, Levels
b,
Files f, Users u
where
c.trackId = t.id and
t.defectId = d.id and
t.releaseId = n.id and
c.versionId = v.id and
c.pathId = p.id and
c.levelId = b.id and
f.id = c.fileid and
```

```
u.id = c.userid and
p.id = f.pathid
commit
cmvcd familyname number of daemons
```

16.3.3 Setting up the Target Server

You run the migration tools from the target server, which communicates with your source server to migrate the database. Although the Version 3 installation program cannot detect whether a CMVC server is installed, you need to make sure that you install the Version 3 server on a different machine from your CMVC server.

Follow these guidelines to ensure that your target server is set up properly for migration. Install TeamConnection Version 3 on your target server according to the instructions in the Installation Guide:

- Install TeamConnection Version 3 and perform the installation verification procedure to create the testfam database. Make sure you complete the installation process, including updating your TCP/IP hosts and services files with IP addresses for your target family.
- Install the CMVC 2.3.12 client on the target server. Make sure your config.sys or profile contains the following settings:
  - Ensure that CMVC is in the PATH statement before TeamConnection, for example,

  ```
  SET PATH=C:\CMVC\EXE;E:\teamcpath;
  ```

  - – Make sure CMVC.CAT is in the NLSPATH statement before TEAMCV30.CAT, for example,

  ```
  SET NLSPATH=c:\cmvc23\exe\nls\%N;c:\teamcpath\nls\msg\enu\%N;
  ```

  Several executable programs that are common to both the CMVC and target client are used in the migration. The PATH environment must find the appropriate version of these programs:

  ```
  File.exe
  To migrate files
  Report.exe
  To migrate all views
  Defect.exe
  To migrate notes
  Feature.exe
  To migrate notes
  ```

  The CMVC file path should precede the TeamConnection Version 3 file path.

- Determine the directory or path where the target database will be located. This directory path will be your TC_DBPATH. Do not create a family database in this location. Instead just create a subdirectory from your TeamConnection Version 3 installation directory to contain the migrated database. You could, for example, use the directory path d:\teamc\v3dbase.
- Set the following environment variables in config.sys or .profile on the target server:

  Set this variable to the name of your migrated family database on the target server:

  SET TC_FAMILY= `migratedDatabaseName@hostname@port`

  Set this variable to the family name, host name, and port ID of your original database on the source server:

  SET CMVC_FAMILY= `originalDatabaseName@hostname@port`

  Set these variables to a user ID with superUser authority:

  SET TC_USER= `superUserID`
  SET CMVC_USER= `superUserID`
  SET CMVC_BECOME= `superUserID`
  SET TC_BECOME= `superUserID`

  Set this variable to the directory path you created for your migrated database on the target server:

  SET TC_DBPATH= `migratedDatabasePath`

  Set this variable only if you are migrating from a CMVC95 server. If you are migrating from a CMVC 2.3.12 server, this environment variable is not necessary:

  ```
  SET CMVC_TYPE=
  ```

  Set this variable to yes to convert SCCS keywords; omit it to prevent keywords from being converted:

  ```
  SET CMVC_KEYS=yes
  ```

  `CMVC_ALLCOMMON` is used in conjunction with `migrate commonparts`:

  ```
  SET CMVC_ALLCOMMON=yes
  ```

  Set `CMVC_ALLCOMMON=yes` to migrate all common parts in a CMVC family. If CMVC_ALLCOMMON is not set, only common files in binding releases will be migrated. Migrating only the common files in binding releases will greatly reduce migration times.

The following are examples of these environment variables, using cmvcdbase as the original family name and v3dbase as the migrated family name:

```
SET TC_FAMILY=v3dbase@v3server@1111
SET CMVC_FAMILY=cmvcdbase@v1server@9999
SET CMVC_USER=migrator
SET CMVC_BECOME=migrator
SET TC_USER=migrator
SET TC_BECOME=migrator
SET TC_DBPATH=d:\teamc\v3dbase
```

- Reboot the machine.
- Set up a directory structure from your new subdirectory for your migrated database. Make it similar to the testfam sample structure described in the installation verification procedure. The following directory structure and files must exist off the TC_DBPATH.

  For example, if TC_DBPATH=d:\teamc\v3dbase, then the directory structure should appear as follows. You can copy these files from the \testfam subdirectory created when you ran the installation verification procedure for your Version 3 installation. The structure is:

```
teamc
v3dbase
cfgfield
Defect.fmt
Defect.tbl
Feature.fmt
Feature.tbl
Part.fmt
Part.tbl
PartView.fmt
Release.fmt
RelView.tbl
User.fmt
User.tbl
WAView.fmt
Workarea.tbl
config
userExit
security (new file; did not exist in CMVC)
queue
tctmp
```

Note: The TeamConnection administrator is responsible for migrating the .ld files after migration. See “Miscellaneous Tables” on page 234

- When migrating a single, committed version of each file from CMVC, you must first do a release -extract, file -extract, or part -extract, to get the files you want to migrate out of the source database. You must then move the files to the TeamConnection Version 3 server machine to migrate them into the target database.

  You can move the files by using a copy command, ftp, or any other available method.
- Start the database server.

16.3.4 Performing the Migration

Once you have prepared your source database and set up your source and target servers, you are ready to perform the migration. The TeamConnection migration utility, migcmvc, is a command-line utility that you run from your target server. It uses the information you provided in “Setting up the Target Server” on page 209 to locate the source database.

TeamConnection provides a file called migcmvc.lst containing sample migration commands. This file is located in the /bin subdirectory of your TeamConnection Version 3 installation path. It may be helpful for you to start with this command file, modify it to suit your needs, and then execute the migration using this command file.

16.3.4.1 Migrating the Current Version of Files from CMVC

The following are sample contents of migcmvc.lst as shipped. The commands in this file are written for the following migration functions:

- vOnly the current versions of parts (files) are migrated.
- All objects are migrated, for example, all users, all parts, all defects, and so on. If you want to select specific objects to be migrated, you will need to modify this command file by adding *WHERE* clauses to the commands. See “Using the Migration Tools” on page 235 for information on specific migration commands. These are the commands:

```
set batchsize 100
set decache 1
migrate Users
migrate HostView
migrate Authority
migrate Interest
migrate Cfgcomproc
migrate Cfgrelproc
migrate Config
migrate CompView where 1=1 order by addDate
migrate bCompView where name=©root©
migrate ReleaseView
migrate EnvView
migrate AccessView
migrate NotifyView
migrate DefectView
migrate FeatureView
migrate LevelView where 1=1 order by commitDate
migrate TrackView
migrate FixView
migrate ApproverView
migrate ApprovalView
migrate LevelMemberView
set top x:\location of files in release if migration only the current
version
migrate fileview where releasename=©9604© and dropdate is null
migrate SizeView
migrate NoteView
migrate CommonParts
```

The migration tool supports migration of committed versions of CMVC files directly into the release. The bulk contents are obtained from a local drive. To migrate CMVC files, you need to do the following:

- If you have a very large database, it may be wise to divide this command file into several smaller files, such as migcmvc1.lst, migcmvc2.lst, and so on. Chunk this command file, and this will enable you to backup your database in intervals during the migration.
- Set the CMVC_TOP environment variable or use the migrate command to set the top directory to the directory containing the CMVC files.
  - Create the source code tree structure on the target server or on a LAN drive that the target server has read/write access to.
  - Change the set top variable in migcmvc.lst to point to the directory structure.
- Use the set batchSize command to specify the number of files to process in one transaction.
- Use the migrate FileView command to specify the release to be migrated from the CMVC database to TeamConnection. Files are read from the local drive. They are not extracted from CMVC when migrating a single version. Always include **dropdate is null** with your migrate command to ensure that it does not attempt to select files that have been deleted. You do not need to migrate or create a work area before migrating CMVC files. Modify this command to include the release names defined in your original

database. If you have more than one release, create additional migrate fileview commands.

To use this command file for your migration, you need to make the following minimal modifications. For a complete list of migration commands, see “Using the Migration Tools” on page 235.

16.3.4.2 Migrating Previous Versions

It is not recommended that you migrate all of your past versions, as this can be very time consuming, and you will most likely not use all of this information. Instead, keep your CMVC database as an archive and retrieve previous versions of parts when you need them.

To migrate previous versions, the set top variable must remain null, as shown in this example:

```
migrate LevelView
migrate TrackView
migrate ChangeView
set top
migrate FileView -release releaseName
```

16.3.4.3 Executing migcmvc.lst

To migrate your database, follow these steps:

1. Make sure your source and target servers are set up properly, the source server is running in maintenance mode, and the client on the target server is running.
2. From the CMVC client installed in the source server, run CMVC report commands to make sure you are connected:

   ```
   report -view users -family sourceFamilyName
   report -view hostView -family sourceFamilyName
   ```
3. Ensure migcmvc.lst is in the \bin subdirectory of your TeamConnection Version 3 installation path. Modify migcmvc.lst to suit your needs.
4. From a command prompt in the directory where you copied migcmvc.lst, type the following command to start the migration tool:

   ```
   migcmvc
   ```

   The migcmvc command displays a header and the migcmvc command prompt. The header contains information about your source and target servers. The migcmvc prompt consists of the family name and > symbol:

   ```
   testfam>
   ```

You can capture output to a file by starting the migration tool using the following command:

```
migcmvc 1>mig.out 2>&1
```

5. At the migcmvc command prompt, enter the following command to execute the migration commands in migcmvc.lst:

   ```
   exec migcmvc.lst
   ```

   As you migrate the database, you may notice the following messages and behavior:

   - When users are migrated you will see one fhccomp record migrated. This is the root component for the family.
   - Migrating NoteView combines the actions from the history section into a single note. The number of notes migrated is likely to be to be less than the number o records for NoteView in the source database.

6. If you have partitioned this command file into several smaller files, back up your database between execution of each command file using the DB2 utilities provided for backup:
7. After migration is complete, exit the migcmvc command interface and start the database server on the target server.
8. Start the migrated family server on the target server.

   Note: You are using the shipped version of configurable field files. If you have configurable fields, they may not display until the administrator has modified these files.

9. Grant users access to the data in the migrated database. In Version 3, users cannot access data as they did in CMVC if they are not:

   - A superUser
   - A component owner
   - A part owner

   You will need to grant users access to each component with the appropriate authority using a command like the following:

   ```
   teamc access -create -login userName -authority authorityGroup
   -component componentName
   ```

10. Run queries like the following to verify that the correct number of objects were migrated. Compare the output from these queries to the output from the queries you ran from the source server in “Preparing the Source Server for Migration” on page 207. These are the queries:

    ```
    teamc report -view workareaView >workare2.vew
    ```

```
teamc report -view releaseView >release2.vew
teamc report -view compView >compon2.vew
teamc report -view defectView >defect2.vew
teamc report -view featureView >feature2.vew
```

For more information on the migration commands, see “Using the Migration Tools” on page 235.

16.3.5 Preparing to Administer Your New Database

After migrating your database, you need to set up a directory structure for your family similar to the testfam sample structure described in the installation verification procedure. The following directory structure and files must exist off the TC_DBPATH.

Type **`SET TC_DBPATH`** to get the name of the family directory. For example, if TC_DBPATH=d:\teamc\testfam, then the directory structure should appear as follows:

```
teamc
testfam
cfgfield
Defect.fmt
Defect.tbl
Feature.fmt
Feature.tbl
Part.fmt
Part.tbl
PartView.fmt
Release.fmt
RelView.tbl
User.fmt
User.tbl
WAView.fmt
Workarea.tbl
config
userExit
security (new file; did not exist in CMVC)
queue
tctmp
```

Note: The directory *testfam* contains .ld files, which the administrator is responsible for migrating.

In most cases, you cannot use your existing CMVC files with your Version 3 database. The following sections provide guidelines for migrating your existing files to your Version 3 database or using the Version 3 files.

16.3.5.1 Miscellaneous Tables

You cannot use the configurable files as they existed in CMVC. You need to create and modify these files for TeamConnection. You can create the .ld files from the database by redirecting the raw output of a report. The following commands show how to create these files:

```
teamc report -raw -view config > config.ld
teamc report -raw -view authority > authorit.ld
teamc report -raw -view interest > interest.ld
teamc report -raw -view cfgrelproc > relproc.ld
teamc report -raw -view cfgcomproc > comproc.ld
```

Copy the .ld files to your TC_DBPATH where your family exists. You should only use these .ld files if you want to add, delete, or change any of the values. See Appendix D, “TeamConnection Authority Groups” on page 283for creating or modifying these groups. For instructions on using the Family Administrator GUI see Appendix D.2, “Using the Graphical User Interface” on page 283. For instruction on using Family Administrator line command see Appendix D.3, “Manual Modifications:” on page 284.

16.3.5.2 User Exit File

If you are converting from CMVC, use the TeamConnection Version 3 config\userExit file and add to it any modifications that you made for your CMVC installation. See “Chapter 9. Providing User Exits” on page 105 of the TeamConnection Administrator Guide for instructions on working with user exits.

16.3.5.3 Configurable Field Definitions

The CMVC configurable field files (defectConfigFormat, featureConfigFormat, fileConfigFormat) are replaced by a collection of .fmt files (Defect.fmt, Feature.fmt, Part.fmt, Release.fmt, User.fmt, and Wrokarea.fmt). An additional configurable field file, called PartView.fmt, is available in TeamConnection Version 3.

If you are migrating from CMVC, use the TeamConnection Version 3 .fmt files and add to them any modifications that you made for your CMVC installation.

The configurable field tables from CMVC (ConfigTable files) can be used with TeamConnection Version 3. Refer to the *CMVC Administrator’s Guide* for the name and location of the files. Copy and rename the corresponding files, then place them in the cfgfield directory:defect.tbl, feature.tbl, user.tbl, part.tbl, release.tbl, and workarea.tbl

16.3.5.4 Mail Exits

Use the TeamConnection Version 3 mailexit file designed for your target platform.

16.4 Using the Migration Tools

Note: We do not recommend that you make changes to your database by issuing INSERT, UPDATE, or DELETE statements or by changing or deleting database tables or the columns defined in TeamConnection database tables. Changing your database in these ways, through the DB2 administrator tools, the DB2 command line processor, the TeamConnection migration tools, or the tcupdb tool can corrupt your TeamConnection database. Any such changes are made at your own risk. Please contact your IBM representative for information on the terms of IBM customer support.

The CMVC and TeamConnection migration tools provide a command line environment from which you can issue the migration commands. You start the migration tool using the Version 3 client, which communicates with the source server to migrate data to the version 3 server. To start the migration tools, type one of the following from your operating system command line:

migcmvc To start the migration tool for CMVC-to-TeamConnection

migtc To start the migration tool for TeamConnection-to-TeamConnection

The command presents a prompt consisting of the family name and > symbol. Type the migration commands at this prompt. To see a list of the migration commands, type **?** at this prompt, as in the following example:

```
testfam> ?
```

Valid commands are:

```
migrate view[,view...] [[where] where-clause]
migrate bcompview where name=©top-component or root©
report viewName [query] (for migtc only)
set argument [value]
select field[,field] from view [where query] (for migtc only)
update view set attribute=value where expression
delete [from] view [where] expression
describe view
exec file
!system-command
dump counters
quit
```

For more information, enter the command followed by a "?"

To see additional help information for a single command, type the command followed by ?.

16.4.1 Migration Tool Variables

Certain environment variables for the migration tool can be set to change how the tool operates. These variables use default values, but you can change them to suit your needs. The following is a list of the default values. For information on these variables and instructions for setting them, see “Set Command” on page 242

```
batchSize set to 1000.
maxErrors set to 0.
decache set to 0.
top set to "D:\".
reportStyle set to "stanza".
loadMessage set to "none".
```

16.4.2 Sample Migration Files

Sample files, called migtc.lst and migcmvc.lst, are provided with the install package. You can edit the commands in these files and use them to perform a migration. Use the exec command to execute the commands in these files:

```
exec migtc.lst
exec migcmvc.lst
```

To add comments to migration command files, use a # symbol. Any line preceded by a # symbol is ignored by the migration tool.

16.4.3 Migrate Command

The migration command executes a report command against the source client and parses this data while creating the objects in the TeamConnection target database. You can choose to migrate all or part of the source data. You select the data to be migrated by specifying *WHERE* clauses on the migrate command to generate reports on the data to be migrated. Use caution when writing these where clauses, because there is potential for error if all objects, such as components or releases, are not migrated but files relating to these objects are migrated. It is your responsibility to generate the correct set of data for migration.

To migrate data from CMVC or TeamConnection Version 2 to Team-Connection Version 3, type the following command at the migration command prompt:

```
migrate view[,view...] [[where] whereClause]
```

where:

- view is one of the views shown in the following list.
- whereClause is a whereClause constructed using column (or field) names as shown in the *Commands Reference*. Refer to this manual for more information on writing queries on TeamConnection views.

The following are some examples of the migrate command:

- The command `migrate users` migrates all users.
- The command `migrate compview where dropDate is not null` migrates all components whose dropDate is not null.
- The following command migrates all defects in open and working state:

  ```
  migrate defectview where state in (©open©, ©working©)
  ```

You can migrate the following views using the migrate command:

AccessView—Access list data. Used for migration from TeamConnection Version 2 and CMVC.

ApprovalView—Approval records. Used for migration from TeamConnection Version 2 and CMVC.

ApproverView—Approver records. Used for migration from TeamConnection Version 2 and CMVC.

Authority—Authority list. Used for migration from TeamConnection version 2 and CMVC.

During the process of migrating Authority, the actions defined in the authority table are changed to the TeamConnection action name as follows:

- Track actions are changed to work area actions.
- Level actions are changed to driver actions.
- File actions are changed to part actions.

New actions that exist in TeamConnection but not in the source database are added and paired with an appropriate group. After migration, you can issue a TeamConnection report command to show the actions and authority groups that exist. These can be compared to a similar report from your source database.

Following migration, if the authority table needs modifications, you can produce an authorit.ld file

bCompView—Component members which make up the component hierarchy. Used for migration from TeamConnection version 2 and CMVC.

When migrating components (CompView) followed by component members (bCompView), it is required that the top level component be specified on the migrate command.

For example, if the default top level component exists in the source database and is named *root,* then you need to migrate the component members using the following command:

```
migrate bcompview where name=©root©
```

To see which component members will be migrated for the top-level component, issue the following report command:

```
report -view bcompview -where "name=©yourRootComponent©" -raw
```

where yourRootComponentis the name of the root component on the source database. If you are not sure of the name of the top-level component, the following query on the source client will provide it:

```
report -view compview -where "id not in (select
childid from compmemberview where parentid<>childid) and id in (select
parentid from compmemberview where parentid<>childid)" -raw
```

If the name of the top component is not root and root does not exist as a component, then you should modify the component name to make it root.

BuilderView—Builders are migrated and the corresponding file is extracted and created in the target database. Used for migration from TeamConnection Version 2.

Cfgcomproc—Configurable component processes. Used for migration from TeamConnection Version 2 and CMVC.

Cfgrelproc—Configurable release processes. Used for migration from TeamConnection Version 2 and CMVC. When migrating configurable processes using Cfgrelproc, the name of the process will remain exactly as it was in the source database. For example, track_level process will be named track_level, but the process now includes "driver" instead of "level." If you wish to change a process name, we recommend that you make a duplicate of the current process and name the new one *track_driver*. To do this, first generate a relproc.ld file as described in “Appendix A. Family administration commands” of the TeamConnection Version 3 Administrator Guide, page 205. In this file, change track_level to track_driver and then execute the following command to reload the configurable release processes table:

```
fhclproc relproc.ld databaseName r
```

where databaseNameis the path name of the target database.

ChangeView—All versions of files. Used for migration from CMVC.

CommonParts— Used for migration from TeamConnection Version 2 and CMVC.

CompView—Components, used for migration from TeamConnection Version 2 and CMVC. See further instructions on bCompView.

Config —Configurable field values, used for migration from TeamConnection Version 2 and CMVC.

CoreqView—Corequisites, used for migration from TeamConnection Version 2.

DefectView—Defects, used for migration from TeamConnection Version 2 and CMVC.

DriverMemberView—Driver members, used for migration from Team-Connection Version 2.

DriverView—Drivers, used for migration from TeamConnection Version 2.

EnvView—Environments, used for migration from TeamConnection Version 2 and CMVC.

FeatureView—Features, used for migration from TeamConnection Version 2 and CMVC.

FileView—Files, used for migration from CMVC.

FixView—Fix records, used for migration from TeamConnection version 2 and CMVC.

HostView—Host lists, used for migration from TeamConnection Version 2 and CMVC.

Interest—Interest table, used for migration from TeamConnection Version 2 and CMVC.

During the process of migrating the Interest, the actions defined in the interest table are changed to TeamConnection action names as follows:

- Track actions are changed to work area actions
- Level actions are changed to driver actions

File actions are changed to part actions

New actions that exist in TeamConnection but not in the source database are added and paired with an appropriate group. After migration, you can issue a TeamConnection report to show the actions that exist. These can be compared to a similar report from your Source database.

Following migration, if the interest table needs modifications, you can produce an interest.ld file as described in “Appendix A. Family administration commands” of the TeamConnection Version 3 Administrator Guide, page 205.

LevelMemberView—Level members, used for migration from CMVC.

LevelView—Levels, used for migration from CMVC.

NoteView—Used for migration from TeamConnection Version 2 and CMVC. The migrate NoteView command iterates over the collection of defects and features that have been migrated and issues a **defect -long -view defectName** or **feature -long -view FeatureName** command against the source database. It then searches for the word "history:" and takes all the following lines of data and creates a single note in the TeamConnection database. The user ID for the note is the originator of the defect or feature and the addDate of the defect or feature.

If you do not want to migrate all the notes, then migrate defects and features in stages. For example, if you want only notes for defects and features in the Open and Working state, then migrate defects and features "where state in (’open’,’working’)" and then migrate NoteView. Then proceed to migrated defects and features "where state not in (’open’, ’working’)."

NotifyView—Notification records. Used for migration from TeamConnection Version 2 and CMVC.

ParserView—Parsers, used for migration from TeamConnection Version 2. When migration of ParserView is complete, the parser files must be moved manually from the source server to the new TeamConnection server and located in the appropriate directory.

PartFullView—All versions of parts, used for migration from Team-Connection Version 2.

PartView—Parts, used for migration from TeamConnection Version 2. Issue one migrate PartView command for each release in your original database. Specify the release whose parts you want to migrate by including a -release attribute with the command as follows:

```
migrate PartView -release yourReleaseName
```

If you have fine-grained parts defined in your database, then you need to

exclude them from the migration by specifying this command as follows:

```
migrate PartView -release yourReleaseName where partType=©file©
```

See “Preparing the Source Server for Migration” on page 220 for more information on preparing a database containing fine-grained data for migration.

ReleaseView—Releases, used for migration from TeamConnection Version 2 and CMVC.

SizeView—Sizing records, used for migration from TeamConnection Version 2 and CMVC.

TestView—Test records. Used for migration from TeamConnection Version 2 and CMVC.

TrackView—Tracks from CMVC migrate to work areas. Used for migration from CMVC.

Users—Used for migration from TeamConnection Version 2 and CMVC. Do not use a *WHERE* clause when migrating users. The **migrate users** command is a required step because it migrates the superUser authority and creates the special user named I*nheritedAccess*. After you issue this command, a component called **root** is created after all users are migrated. The root component is the top-level component and is used as the parent of the first component. The top level component on your target database must be named *root*, the default name that is created when the database is first initialized. This should not be modified.

Since users are the owners, originators, and creators of most objects in the database, they must also exist in the target database in order for the related objects to be migrated. For this reason it is recommended that all users be migrated and that you avoid using the where clause.

VerifyView--Verification records, used for migration from TeamConnection Version 2 and CMVC.

WorkAreaView--Work areas, used for migration from TeamConnection version 2.

16.4.4 Report Command

Data in the target database can be displayed with the report command. The format of the data depends on the setting of the reportStyle option (see “Set Command” on page 242). If the view contains configurable fields, the fields will be displayed with titles like configField1, configField2, and so on, because the title of the fields is not known to the migration tool. The set reportStyle command determines the format of the report output.

To generate a report on the target database, type the following command at the migration command prompt:

```
report view [query]
```

where:

- `view` is one of the views shown in “Migrate Command” on page 236.
- `query` is a query constructed using column (or field) names as shown in the Commands Reference. Refer to the *Commands Reference* for more information on writing queries on TeamConnection views.

The following are some examples of the report command:

- The following command reports information on all users:

  ```
  report users
  ```

- The following command reports information on all components whose dropDate is not null:

  ```
  report compview where dropDate is not null
  ```

16.4.5 Set Command

To set options for the migrate and report commands, use the set command as follows:

```
set option value
```

where:

- `option` is one of the options listed below.
- `value` is a valid value for the option.

You need to issue one set command for each option that you want to change. The following list shows the options you can set and the possible values for each. To display current values for each option, type

set ?

The options are these:

batchSize—This option specifies the number of objects to migrate in one transaction. The default is 1000.

maxErrors—The number of errors found before a migrate view is halted. For example, if migrating defects and the originator’s user login is not found, then this counts as one error. The default is 0.

reportStyle—Specifies how reports generated by the report command are to be formatted. The default is stanza. The choices are these:

- `terse`
- stanza

loadMessage—Specifies how messages generated during a migration are to be formatted. The default is none. The choices are:

- terse
- stanza
- none

deadlockRetry—Specifies how many times to retry a transaction if a deadlock occurs. This variable can be set to any value.

top—Specifies the location of the source code, if needed.

16.4.6 Select Command

Use the select command to display objects in a view. You use this command with the view names and field names described in the appendix of the *Commands Reference:*

```
select field[,field] from view [where query]
```

where:

- `field` is a column name from the view.
- `view` is the name of the view from which you want to display objects.
- `query` is a query constructed using column (or field) names as shown in the Commands Reference. Refer to the *Commands Reference* for more information on writing queries on TeamConnection views.

The following example displays the login IDs for all users in the database select * from users

16.4.7 Update Command

Use the update command to change the value of a field in the database. You use this command with the view names and field names described in the appendix of the *Commands Reference:*

```
update view set attribute=value where expression
```

where:

- `view` is the name of the view whose values you want to update.
- `attribute` is the field name of the value you want to update.
- `value` is the new value.
- `expression` selects the rows from the table (or view) for which you want to update values.

The following example updates the superUser field for all users whose login ID is cpersche:

```
update users set superUser=©yes© where login=©cpersche©
```

16.4.8 Delete Command

Use the delete command to delete an object from the database. You use this command with the view names and field names described in the appendix of the *Commands Reference*. The command is:

```
delete [from] view [where] expression
```

where:

- v `view` is the name of the view whose objects you want to delete.
- v `expression` selects the rows from the table (or view) that you want to delete.

The following example deletes all users from the database whose login ID is minnie:

```
delete from users login in (©minnie©)
```

16.4.9 Describe Command

Use the describe command to display the column names for a view:

```
describe view
```

where:

`view` is the name of the view whose relationships you want to display.

The following command shows the column names for the user's view. These column names can then be used in select and in *WHERE* clauses.

```
describe users
```

16.4.10 Exec Command

Use the exec command to issue migration commands from a file:

```
exec filepath
```

where:

`filepath` is the full path name of the command file to execute. If you do not specify a full path name, the migration tool looks for the file in the current directory.

The following example executes migration commands defined in a file called migtc.lst:

```
exec migtc.lst
```

The following example executes migration commands defined in a file called

```
d:\teamc\bin\migtc.lst.
exec d:\teamc\bin\migtc.lst
```

The file migtc.lst is a sample of the migration views that can be used to migrate your database.

16.4.11 ! Command

Use the ! command to issue operating system commands from the migration command prompt.

```
!systemCommand
```

where:

systemCommand is an operating system command.

The following example issues a dir command from the migration command prompt:

```
!dir
```

16.4.12 Quit Command

Use the quit command to exit the migration command prompt and return to the operating system command prompt:

quit

16.4.13 # Command

Use the # command to add a comment to a migration command file.

Appendix A. VisualAge TeamConnection Technical Overview

This section provides a technical overview of IBM VisualAge Team-Connection Enterprise Server, Version 3, and describes its integration with application development tools. It is written for customers who want to know more about the functions that VisualAge TeamConnection Enterprise Server provides. For brevity, the phrase VisualAge TeamConnection Enterprise Server is shortened to TeamConnection.

TeamConnection is a repository-based software configuration management (SCM) tool designed for a team programming environment. It uses IBM's award-winning DB2 Universal Database and repository technology to maintain the integrity of your development assets, while scaling to fit your needs. With VisualAge TeamConnection, you can manage and control your development projects, increase your team's productivity, and improve overall software quality. TeamConnection's multi-platform support enables you to choose your development platform to maximize productivity, scalability, and performance, while targeting application execution to a variety of different platforms.

Among the exciting new developments in TeamConnection's repository functions is use of the emerging industry standard XML. VisualAge Team-Connection provides a Web client interface that allows any data stored in the TeamConnection repository to be retrieved using XML. XML is used to deliver application meta-data (data about data) on demand to XML-enabled tools. XML enables a new class of applications to present and manipulate repository meta-data with ease.

TeamConnection supports a variety of development tool environments, including IBM's VisualAge family of products and products supporting the Microsoft Source Code Control Application Program Interface (SCC API).

A.1 TeamConnection Customer Value

TeamConnection marries repository technology with software configuration management services in a team environment. This marriage allows you to store all of your development data in one place, regardless of form, and to provide a consistent set of services and processes for your entire development team, not just programmers.

TeamConnection helps you manage your development process by organizing your data, controlling changes to it, and providing a reporting capability.

TeamConnection increases your team’s productivity by automating common development tasks, notifying team members when action is required, and providing a vehicle for tools to share data. It also helps improve product quality by providing repeatable reliable processes that assure proper authorization for changes.

Finally, TeamConnection provides an open, customizable, and scalable environment that can evolve and grow with your development team.

A.2 Application Development Challenges and TeamConnection

Today’s application development teams face tremendous challenges. With both applications and application development environments are becoming more and more complex, development teams are challenged to continuously increase code quality through adherence to process standards and audit trail management. The application development tools available today to help address these issues often come up short, addressing only specific problems and not providing sufficient tool integration to deliver a fully integrated software development environment.

Applications development is more complex when it supports advanced graphical user interfaces, object-oriented and client/server technologies, and heterogeneous distributed computing. Client/server applications today include mission-critical, high-volume transaction processing, which brings with it additional complexity in the application design.

The team's development environment is also becoming more complex in an effort to meet these challenges. The development team is distributed on a LAN, yet it requires the same coordinated team programming support that is available on the centralized host environment. Object technology, with an increasing focus on support for reuse and the need for various tools to share data adds to the complexity of the development environment.

Successful software development organizations are measured by Software Engineering Institute (SEI) maturity levels and by compliance with ISO 9000 standards. These problems present new challenges to development teams that are being asked to deliver software products to market at faster and faster rates.

A.3 TeamConnection Can Help

TeamConnection helps address these development issues by:

- Enabling you to manage the changes to your software more efficiently.

- Providing a secure, scalable repository for your software development assets and associated meta-data, with role-based access.
- Enabling you to organize your software parts for reuse.
- Helping you to rebuild your applications more efficiently after they have been modified.
- Improving team communication and deployment through e-mail notification of activity required, application changes, and project status.
- Packaging your applications for delivery so you can get them to your customers more quickly and with greater reliability.
- Providing a development model that can help increase your SEI maturity level and improve your application quality through reliable, efficient, and repeatable processes.

Providing extensive problem tracking and change control for both fine-grained and coarse-grained application development objects.

- Providing extensive reporting used for tasks such as impact analysis, project status, project management, quality analysis, and workload balancing.
- Providing an application development information model, along with repository services, that provides the basis for tool integration.
- Providing backup and recovery facilities.

A.4 How TeamConnection Helps

TeamConnection integrates software configuration management (SCM) services and object-oriented repository services on a semantic model for tool integration to support application development in a LAN-based client/server team programming environment. Applications can be targeted for any platform and can be stand-alone, client/server, or distributed applications.

TeamConnection delivers the function your application development teams need to manage development data, application versions, and application configurations. The integrated problem tracking and change control system ensures that, while your application developers are more productive, your project leaders can still effectively manage the development process and track progress.

TeamConnection automates and streamlines your application build process and integrates it with the version and change control processes. The build process is tied to release management and extended to provide a framework for delivery of software.

A.5 Configuration Management

TeamConnection's powerful SCM capability enables you to identify, organize, manage, and control access to your development data. Data within TeamConnection is divided into one or more families. A *family* represents a complete and self-contained collection of related *parts* (development data, whether files or modeled objects) and data about the parts (sometimes called *meta-data*).

Data within a family is completely isolated from data in all other families, even those on the same server. One family cannot share data with another, except through external synchronization. TeamConnection has mechanisms that can be used to implement customer-defined synchronization processes.

A.6 Views

The data in the family can be viewed in a number of different ways, depending on the need:

- The parts view is used to show the parts that can be worked on.
- The component or organizational view is used for access control, e-mail notification control, and other configuration management properties.
- The release or application view is used to group the appropriate versions of all relevant parts of an application.
- The build or dependency view shows how the different parts of a release are related to one another.
- The historical view shows different versions of objects, such as parts and releases, over time.

A.7 Parts

A *part* or managed object is development data that is stored by TeamConnection and retrieved by name. The parts controlled by TeamConnection can be text parts, binary parts, modeled parts, meta-data parts (special TeamConnection parts used to manage the other parts) or empty. Parts are always owned by a component.

Text parts are ASCII files, which are generally created by an editor. Examples are source code, test cases, documentation, README files, and install files.

Binary parts are files that are generally created by a tool. Examples are OBJ files that a compiler produces, EXE files that a linker produces, and other files a tool produces whose format is understood only by that tool.

Modeled parts are defined in the information model. Empty parts are place-holders, for example, for the output of a future build.

When you create a TeamConnection part (from the TeamConnection client), you are placing an existing development part under TeamConnection control. The part is made persistent when it is placed into the TeamConnection server, even if it is erased by the TeamConnection client.

To change a part, you check out the part from the TeamConnection server to the TeamConnection client, make changes, and then check in the changed part to the TeamConnection server. TeamConnection stores additional information (such as who changed it and why) about the part each time an action is performed against it. This information can be queried at any time.

Common actions against parts include:

- **Create–**Store a part from your workstation in the TeamConnection server; store a part from a tool in the TeamConnection server; or create a part as a place-holder.
- **Check out**–Get a copy of the part onto your workstation for update. You can lock the part so that other users cannot change it.
- **Check in**–Put an updated part back into TeamConnection.
- **Extract**–Get a copy of a part onto your workstation without locking it.
- **Build**–Execute one or more build steps, such as compile, link, or generate.

Tools such as VisualAge Generator might hide these details from the end user by invoking the actions on behalf of the end user.

A.8 Component Hierarchy

The structure used to organize data within a family is called the *component hierarchy* or *family tree.* Each node within the hierarchy is called a *component.* A component is a TeamConnection object that simplifies project management, organizes project data into structured groups, and controls configuration management properties.

The hierarchy provides a mechanism for organizing components of data into structured groups. Some common ways to group data in a component hierarchy are by function, by execution platform, for access control, or for communication needs. The component hierarchy should reflect the organizational requirements of your development team and can be modified over time as these requirements change.

Each part in TeamConnection is managed by a component. Components that manage one or more parts usually represent the leaves or outer components of your hierarchy, while the branches or inner components are used for organization, access control, notification, and problem reporting.

A.9 Role-Based Access

Control of TeamConnection parts and the authority to access TeamConnection data is managed by components and granted to users through component ownership and the access lists associated with the components. Users are given access to parts in a specific family through their TeamConnection user IDs. Each family has at least one superuser, who has privileged access to the family and acts as the TeamConnection administrator. All TeamConnection users have some basic capabilities which can be further extended by the superuser or his/her delegates.

Ownership of each component is assigned to one user, who is responsible for managing all development data related to that component and the parts it manages. The owner has automatic authority to perform many actions.

In addition, each component has an access list, which specifies the roles or authority group a user has for that component. The roles (such as Developer, Project Leader, and Tester) can perform specific TeamConnection actions against parts owned by that component. This means that user access (beyond the owner) to parts can be controlled by the component owner by assigning users specific roles. Default roles, shipped with TeamConnection, and can be tailored to meet your needs. Access authority is inherited by lower-level components.

Each component can also have a notification list. Notification messages are sent using Simple Mail Transfer Protocol (SMTP) to an electronic mailing address that is specified when the user’s TeamConnection ID is created. Some notification is automatic. For example, the owner of a Team-Connection part is always notified when actions are performed on that part. Users receive notification when an action affects their user ID or requires them to perform an action in return. Additional notifications can be configured through the notification list, which maps user IDs to interest levels. Like access authority, notification is inherited by lower-level components.

A.10 Release Management

A *release* is a logical organization, or mapping, of all parts related to an application. A release does not affect the physical location of a part; instead it provides a logical view of the parts that must be built, tested, and distributed together.

Each release is created against a managing component, which provides the access and notification mechanisms. However, the parts in a release can be managed by many components. Also, a specific part can be in multiple releases and different versions of the same part can be in different releases.

Every time a development cycle begins for the next version of an application, a separate release can be defined. Each subsequent release of an application typically references many of the same parts.

The release manages whether or not part changes are tied to defects and features. If the track process is turned on, then all part changes require a defect or feature number. If the track process is turned off, then part changes can be made without a defect and feature number. Because you can configure your change process by release and modify the change control process over time, you can have a low level of control early in the development cycle, add controls as the release comes closer to shipment, and finally, use the highest level of control after the release has been shipped to customers and is being maintained.

A.11 Version Control

Versioning is making copies of data at some meaningful point in order to return to that point at a later date, if necessary. Most programmers version their code, even if they are not using a software configuration management system to manage their applications.

A classic example is the programmer who has just added a new routine to an application and validated that the new routine works. Before writing the next routine for the application, the programmer makes a copy of the code that works. If subsequent changes break the application, she can always go back to the copy of the application that worked.

TeamConnection versions application development activity at a release level. As compared with tools that record and track versions of file changes, the TeamConnection release-level versioning model maintains snapshots of the release in addition to snapshots of parts. This is the crux of configuration management. A version of a release is the set of the correct versions of all the parts that make up a release. As changes are committed to the release, the release is *frozen*, or saved, so that each evolution can be retrieved.

In practice, most customers create a new release whenever they want a major version of that release. For example, they have a release (application) called *Accounts Payable V1.5*. This application is put into production. They want to ship a new version of this application release in six months, while maintaining the old version. They create a new release called *Accounts Payable V1.6*, and link the new release to the old release, so that the people working on the new release start with the parts from the old release. As these people make changes for *Accounts Payable V1.6*, the links back to *Accounts Payable V1.5* are broken. However, if they make maintenance changes to *Accounts Payable V1.5*, these changes are also reflected in *Accounts Payable V1.6.* If they make changes to *Accounts Payable V1.5* after breaking the link to *Accounts Payable V1.6*, then changes must be made in both releases.

The versioning model of TeamConnection not only enables you to work on the correct versions of individual parts, but it also maintains and presents those part versions within the context of the identified version of the release. You can change between work activities, such as problem resolution on an old release and enhancements to the current release, yet the complete version of each release is recreated to reflect the correct versions of its included parts.

The versioning model is implemented through the use of *work areas*. A work area is a view of all the parts in a release at a particular point in time. A work area serves as a sandbox or space in which you can update parts and do builds without affecting the release. Some people call this sandbox a *change set.*

This work area view is not a physical place, such as a directory. It exists inside the TeamConnection server. In order to update a release, you create a work area. When you create a work area, it maps to the most current version of the release. You check out a part from the work area, edit it, and then check in the part to the work area. Your changes now exist in the TeamConnection server, but are visible only in the work area. After building in the work area and testing, you update the release by committing the work area.

On a *commit*, the release is frozen and then updated with the changes from the work area. This provides the history of changes and the ability to recreate each evolution of the release.

A version of a release is the set of the correct versions of all the parts for a complete release. Maintaining copies of all parts in a release for each version would require exorbitant disk space. This is why versions of a release are actually mappings to the correct versions of the included parts. In addition, versions of the text files are stored as reverse delta versions and compressed to conserve space.

TeamConnection also provides optional support for concurrent development, so you can update parts in parallel with other developers. Its 3-files graphical merge tool helps you resolve conflicting changes in text parts. Optionally, the graphical merge tool can also assist you by automatically merging files, although we recommend you verify the results of the merge.

A.12 Integrated Build

TeamConnection's integrated build function automates and optimizes the process of building individual parts, entire releases, or multiple releases. Building can encompass many different processes such as compiling, linking, generating, document processing, binding an application to a database, or invoking a command file. With distributed build, parts or releases can also be built for the enterprise client/server environment, that is, for target platforms different from the server platform. Integrated build functions are available for Windows NT, Windows 95, OS/2, AIX, HP-UX, Solaris and MVS.

Integrating the build capability within TeamConnection provides value that standard make functions cannot match. Of course, you can still use your normal make routines, either while migrating or permanently, as part of the TeamConnection build, thus preserving your investment in these tools.

TeamConnection builds:

- **Are repeatable and reliable**–Once the build structure is set up, TeamConnection can provide a repeatable, reliable build. The same outputs are produced from the same inputs when you use the same set of translation or build rules. You can also build different outputs from the same inputs by using different translation rules, for example, to compile for a different platform or to remove compiler debug options. The build structure can be created using an object-based GUI or the command line interface.

- **Minimize resources required**–TeamConnection detects out-of-date conditions in a build structure and rebuilds only those parts of the release that need to be rebuilt, thus minimizing the resources required for the build. Because of the way time stamps are kept in a LAN environment, out-of-date conditions cannot always be detected in a file system. But because all inputs are stored in TeamConnection, its build function does not rely on file system time stamps, thus improving its performance over make functions.
- **Builds multi-platform applications**–Client/server applications require t code for multiple platforms to be kept in sync. The build function in TeamConnection can incorporate the build procedures for multiple platforms and can synchronize the build of the related applications.
- **Distributes the build across the available machines**–TeamConnection can manage a pool of available machines to build an application. It can determine which parts of a build need to be serial and which can be parallel. Then it can parcel out the work to available machines. For example, a pool of OS/2 and Windows NT machines might be defined for compiling C code. At night, programmer machines can be added to this pool for large build processes. With this capability, a multi-step build is no longer constrained by the speed of a single machine.
- **Builds objects and files**–TeamConnection provides software configuration management services on a broad range of parts (files and modeled objects), all stored in a single repository. With these capabilities, TeamConnection can support 3GL, 4GL, model-driven, and OO development paradigms. Parts for these paradigms can be stored together in TeamConnection. Where applications are written in a mixture of 3GL, 4GL, and OO, TeamConnection can build them together and keep them in sync. For example, a VisualAge Generator application might work with an application written in COBOL. Both of the applications can be stored and managed by TeamConnection within the same build structure.

A.13 Build Mechanisms

A *build tree* is a structure graphically defining how the parts are built together. The build tree is a complete description of the dependencies the parts have on one another and of all the steps required to build the release. Each executable step in the build tree is called a *build event*. For example, the compilation of a C source code file into object code is a build event.

A *builder* is a TeamConnection object describing how to perform a build event: how to translate input parts into output parts. For example, one builder might know how to transform C++ source code into object code. A different builder might know how to transform object code into an executable. The builder identifies the environment where the transform will execute, passes parameters to and from its build script, defines the basic rules of successful completion, and invokes a build script.

The *build script* the builder uses is essentially a binding between TeamConnection and a transform tool. The build script invokes the tool. Because of the way TeamConnection uses build scripts to invoke tools, TeamConnection is highly extensible and can be used with a large variety of build tools. Examples of these include:

- Application generators, such as VisualAge Generator
- Compilers, such as VisualAge C++ and VisualAge for COBOL
- Linkers
- Preprocessors for subsystems such as DB2 and CICS
- Document processors
- Automated test tools

A build script can be as simple or complex as you like. It can be as simple as a single string invoking the tool. In Windows, it could be a batch file or a C program. In OS/2, a build script is usually a command file. In MVS, it is a JCL fragment. In UNIX, it may be a shell script. The only requirement is that the build script be executable on the build server machine. In MVS, the JCL job stream can do anything a normal JCL job stream does. For example, a build script designed to move an application into test might copy the application executable into a test loadlib along with a set of test cases.

A *parser* is a tool that knows how to read a source file and report back a list of dependencies of that source file. A parser simplifies the job of defining a build tree. It frees you from having to know what dependencies a part has on other parts. For example, a C parser knows how to read a C source code file and report back a list of files included by the source file. A COBOL parser reads a COBOL source file and reports back the list included in Copybooks. If dependencies are found during the parse that are not reflected in the build tree, then the build tree is automatically updated. A set of sample language parsers and build scripts is provided with TeamConnection. These can be extended by the user.

Once the build tree, builders, build scripts and parsers are set up, TeamConnection has a reliable, repeatable build process minimizing the time and resource required to build an application. When a build is requested, the build process determines out-of-date parts (changed parts or parts whose inputs have been changed) within the build tree, parses any out-of-date part or parts to find any new dependencies, and uses the builders and build scripts to rebuild only the out-of-date parts in the requested environment.

A.14 Distributed Build

In TeamConnection, the build server is where build tools such as compilers, linkers, and generators execute. In the case of compilers and linkers, this is typically the target environment where the application will run.

In the build process described in the previous section, the build actually consists of putting the build scope on the job queue. The *build scope* is the collection of build events that must occur to complete the build. The *job queue* is the list of build scopes to be completed.

The *build server* is the TeamConnection process that actually invokes the tools, such as compilers and linkers, to construct an application. It sits on a supported platform anywhere in the network, polling the TeamConnection server for work. The TeamConnection server understands which build events within the job queue need to be performed in a specific order.

TeamConnection also understands the environment required for a build event, such as UNIX, MVS, Windows NT, or OS/2. These mechanisms allow parts of a build to occur in parallel, if multiple build servers are available. They also allow builds to be distributed to the appropriate target environment for execution.

This design allows TeamConnection to handle a wide variety of topologies such as

- Building on a remote machine
- Building a distributed application where application parts execute on different machines
- Building an application that can run on multiple machines (same source code built for OS/2, Windows NT, UNIX, and MVS)

As each build event is completed, the TeamConnection server is updated with the return code and any outputs, such as the compiled object code. After the build is completed, the TeamConnection client that requested the build is notified.

This architecture provides considerable flexibility. To illustrate, an MVS programmer can edit, compile, and debug an application on a Windows NT workstation using IBM VisualAge for COBOL. When the application is ready for system test or production, the programmer requests a build for MVS. The build is executed on MVS, and the executable is automatically brought back to TeamConnection so it can be versioned along with the source.

Some programmers still prefer to compile changes locally, prior to checking them into the source code repository. This remains possible, although the ability to retrieve past versions is limited to the last check-in and freeze.

A.15 Packaging and Distribution Support

Packaging and distribution support provide a bridge from development to the production environment. In TeamConnection, packaging is any of the steps necessary to distribute TeamConnection-managed applications to software users or onto the machines where software is to be used. To do this, TeamConnection extends the build process to include transformation of executable and non-executable files into a distribution-ready form.

Your process may require more than one packaging step to prepare a software package for distribution. Examples of these are:

- Compressing files
- Packing files together
- Encrypting files
- Organizing and grouping files
- Adding prerequisite files and programs such as installation utilities
- Updating asset or inventory managers (license tracking)
- Transferring the files to physical media
- Distributing the packaged software electronically

Any or all of these steps can be automated by incorporating them into the build tree. Unlike conventional build steps, packaging build steps often do not change the content of the release, but simply organize or package the release in preparation for some form of distribution. Another distinction of packaging build steps is that their outputs are generally not physical parts that are stored back into TeamConnection; instead, they are abstract or conceptual. So if the build part is up to date, it may not have produced something in TeamConnection; instead it may represent the fact that files were transferred to a remote server, for example.

A.16 TeamConnection’s Packaging Utilities

In addition to this highly customizable packaging support, TeamConnection focuses on solutions to electronically distribute software. TeamConnection includes two utilities:

- The *Gather* tool is an automated data mover for server- or file-transfer-based distribution.
- The *Bridge* tool automates the installation and distribution of software or data using Tivoli Software Distribution as the distribution vehicle.

Together, these two utilities provide the ability to move an application in TeamConnection, along with associated files such as installation files and documentation, to an OS/2, Windows NT, or UNIX subdirectory and then to invoke Tivoli Software Distribution to catalog the application, distribute it, and install it.

The following scenario demonstrates the value of integrating these functions into TeamConnection. An application managed by TeamConnection is in production when an error is detected. The maintenance team opens a defect object for the error. A work area is created to incorporate the modifications into the current release. After testing, the work area is committed to the release. The release build tree includes the gather and distribution steps. When these steps are built, the updated application is moved into the file system and, through Tivoli Software Distribution, it is distributed and installed. Because the build tree was used, the audit team can be sure that the application included the fix when it was distributed.

A.17 Electronic Distribution Support Using Your Distribution Tool

There are many tools for electronic distribution. You can build a bridge similar to the bridge to Tivoli Software Distribution to invoke the distribution tool of your choice. To simplify this task, TeamConnection includes the mini-utilities used to create the bridge to Tivoli Software Distribution. Although these utilities are unique for Tivoli Software Distribution, you can use them as examples to accelerate the development of your own bridge.

A.18 Integrated Problem Tracking and Change Control

The processes involved in problem tracking and change control give you the ability to manage and control your development process. TeamConnection tracks reported problems and design changes and retains information about the life cycle of each. A *defect* is used to record each reported problem. A *feature* is used to record each proposed non-defect change. The information recorded about defects and features allows you to report on who, what, when, why, and where modifications occur and also to report on where a particular defect or feature is in the development cycle and where the release is in the development cycle. For instance:

- How many features have been implemented? Or still need to be implemented?
- How many defect objects are open?
- How many feature objects are still in test? (Do I need to move resources to test?)
- How many defect objects have been opened against each part? (Where are my code quality problems?)

A defect or feature can be used to record any changes you need to monitor. For example, defects and features can be used to record problems, proposals for process improvement, and hardware design changes, as well as proposals for design changes in products being developed under TeamConnection control.

Any TeamConnection user can open a defect or feature object. Each defect and feature must be opened against a specific component within the component hierarchy. The defect or feature can be reassigned to a more appropriate component within the hierarchy if necessary. The owner of the component to which it is assigned automatically becomes the owner of the defect or feature. This ownership can also be reassigned.

Defects and features go through a configurable process. As they move through the process, team members who need to take action are notified through e-mail. This automatic notification helps keep your process running smoothly and aids in team communication. It might even reduce the number of meetings that use development cycle time for unproductive work.

If you choose to require that changes to parts can be made only in association with a defect or a feature, then changes can be made only when the defect or feature is in the Working state. This enables you to require the correct level of authorization before a change can be made.

Although build parts and versions are release-specific, defects and features are family-wide. This allows a specific defect or feature to be implemented for multiple releases. It also helps to ensure that problems fixed in a maintenance release are likewise fixed in the development release, so your customers don't find an old problem in a new release.

TeamConnection ties the process of managing defects and features to the change control process through the use of the tracking process. If you use the tracking process, changes to parts must be tied to defects and features. Because changes are implemented in work areas, when you use that process you have the ability to build, test, and ultimately commit individual changes to the release.

If the tracking process is being used, the driver process can also be used. This process integrates multiple defects and features before committing them to a release. This is done by adding their work areas to a driver. As this is done, changes can be built together and tested together prior to committing them to a release.

With the ability to configure processes, you can tailor TeamConnection to best suit your environment. You can adjust the process over time to fit the stage your release has reached in its life cycle.

A.19 Lotus Notes

VisualAge TeamConnection provides an integration with Lotus Notes (Release 4.5.3 or later) for delivering a comprehensive technical documentation facility, including the capability to manage and track many of the administrative and management functions necessary for the development and maintenance of software.

This integration provides the ability to easily implement a Lotus Notes database tailored to support the creation and tracking of software development activities. The linkage between the Lotus Notes database and TeamConnection repository enables you to use Lotus Notes to create, view, and manage problems and enhancements related to the development project.

A.20 Report Facility

The report facility in TeamConnection is an SQL query facility. End users can form dynamic queries to quickly find parts of specific interest, such as defects assigned to them, parts they have checked out, or all changes associated with a particular feature. The GUI can also be used to form these queries.

Queries can be stored and used for extensive reporting, auditing, and project management, as the following examples demonstrate:

- **Impact analysis**–Tell me all the releases that a part is used in.
- **Quality analysis**–Tell me how many defects have been reported against each component in a release.
- **Workload balancing**–Tell me how many defects are being worked on by each team member. Tell me how many defects are being tested. Use this information to move resource between development and test.

- **Project Status**–For features scheduled in this release, tell me how many are open, implemented, tested, and complete.

Output from queries can be input to other tools such as Lotus 1-2-3, which will graph the results of a report.

A.21 Backup and Recovery

TeamConnection is built on IBM’s award-winning DB2 Universal Database and users can take full advantage of its sophisticated backup and recovery capabilities. The backup and recovery facility allows TeamConnection data to be stored on alternative media for archival purposes.

A.22 Repository and Model Support

TeamConnection leverages DB2 Universal Database and IBM’s repository technology to maintain the integrity of your development assets. Tools or utilities can provide transparent access to the repository services by using the TeamConnection API. The repository information model can also be extended through a set of service offerings and tools to meet specific customer needs.

A.23 Application Development Tool Integration

TeamConnection integrates with the following tools:

- VisualAge for Smalltalk and VisualAge Generator through a bridge provided with TeamConnection
- Tools that conform to the Source Code Compatibility (SCC) API, such as Microsoft Visual Basic and Visual C++, Rational Rose 98, and PowerBuilder from Sybase.
- Tools (such as compilers) that transform a set of inputs into a set of outputs. These can be plugged into TeamConnection through the use of a build script.

TeamConnection is designed for extensibility and easy integration. The software configuration management architecture allows for many IBM and non-IBM tools to be easily used with TeamConnection for storage and control of development data and for building outputs.

Tools can provide transparent access to the software configuration management services by invoking the TeamConnection command line interface. They can provide transparent access to the repository services by using the TeamConnection API from within their tools. The API is available in C++.

TeamConnection functions can be extended by invoking tools from TeamConnection user exits. User exits are available before and after each command.

TeamConnection provides the integrating technology for multiple application development tools to share data in a team environment. The development, compiler, and generator tools provided by IBM that integrate with TeamConnection include these:

- IBM VisualAge for C++
- IBM VisualAge for COBOL
- IBM VisualAge for Java
- IBM VisualAge Generator
- IBM VisualAge for Smalltalk
- IBM VisualAge PL/I

A.23.1 IBM VisualAge for C++

IBM VisualAge for C++ is IBM's offers the following features:

- Integrated edit/compile/debug/browse environment
- Incremental compiler and linker
- Visual Builder for GUI and logic
- Data Access Builder
- Batch C and C++ 64-bit support
- Compiler and class library support for ANSI/ISO C++ Standard from February 1998

For more information on IBM VisualAge for C++, see:

http://www.software.ibm.com/ad/visualage_c++/

TeamConnection provides the infrastructure for team programming and software configuration management services for IBM VisualAge for C++.

TeamConnection greatly enhances team productivity when used in conjunction with IBM VisualAge for C++. In addition to the standard integration points, storage and control of C++ files, and building C++ executable, TeamConnection is integrated with the Workframe component of IBM VisualAge for C++ Version 3 and with the Workbook component of

VisualAge for C++ Version 4. Files stored in TeamConnection can be reflected in Workframe or Workbook projects.

With TeamConnection’s ability to build applications for OS/MVS, the C/370 for OS/390 compiler can also be used as a build tool with TeamConnection.

A.23.2 IBM VisualAge for COBOL

IBM VisualAge for COBOL provides the COBOL programmer with 32-bit, direct-to-SOM-based, object-oriented support. In addition, a COBOL application development environment is provided that is designed specifically to handle client/server, mission-critical applications through visual programming. IBM VisualAge for COBOL also gives the COBOL programmer a set of high-productivity, LAN-based power tools for the development of applications targeting LAN-based execution systems.

For more information on IBM VisualAge for COBOL, see:

http://www.software.ibm.com/ad/cobol/

TeamConnection provides the infrastructure for team programming and software configuration management services for IBM VisualAge for COBOL.

TeamConnection greatly enhances team productivity when used in conjunction with IBM VisualAge for COBOL. In addition to the standard integration points, storage and control of COBOL files, and building COBOL executable, TeamConnection is integrated with the Workframe component of IBM VisualAge for COBOL. Files stored in TeamConnection can be reflected in Workframe projects, and TeamConnection actions can be invoked from a Workframe project. With TeamConnection’s ability to build applications for MVS, the IBM COBOL for OS/390 compiler can also be used as a build tool with TeamConnection.

VisualAge DataAtlas information about customer data elements and structures can be used by VisualAge DataAtlas to generate COBOL copybooks, providing data integration and reuse at a file level.

A.23.3 IBM VisualAge for Java

IBM VisualAge for Java is an award-winning Java application development environment for building Java applications, applets, servlets and JavaBean** components. VisualAge for Java gives you a dynamic Java development environment for power programming and maximum productivity.

For more information on IBM VisualAge for Java, see:

http://www.software.ibm.com/ad/vajava/

For day-to-day development, VisualAge for Java uses an integrated SCM repository, which has operational support tailored specifically for highly interactive, prototyped, iterative development environments. This has been the actual standard for SCM support for Smalltalk (including VisualAge for Smalltalk) for many years. IBM ships the team portion of this repository with VisualAge for Java Version 2 Enterprise edition.

The team repository of VisualAge for Java addresses programmer needs by providing source configuration support. Although the software configuration and source version control is excellent, this may not be sufficient to fulfill other enterprise software development needs. IBM supports the SCC API in VisualAge for Java Version 2 as a way to integrate with VisualAge TeamConnection Enterprise Server. This gives enterprise-wide teams access to the excellent SCM support provided by the team repository of VisualAge for Java, along with the scalable enterprise support provided by VisualAge TeamConnection Enterprise Server.

Examples of enterprise support include VisualAge TeamConnection’s ability to manage all development artifacts—not just source code, defect and feature tracking—to provide full development life-cycle support, to store modeled objects, and to integrate multiple tools and multiple languages across the enterprise.

A.23.4 IBM VisualAge Generator

VisualAge Generator is an extremely powerful high-end application development environment for building and deploying e-business and multi-tier client/server applications. It offers versatile solutions with scalable, multi-platform exploitation across networked systems.

VisualAge Generator's workstation design energizes, streamlines, and empowers greater productivity for Windows NT and OS/2 developers who are building applications for any number of client and server platforms. This powerful application development solution gives you the productivity of object-oriented visual development and scales up to meet your most demanding transactional enterprise requirements.

For more information on IBM VisualAge Generator, see:

http://www.software.ibm.com/ad/visgen/

For day-to-day development, VisualAge Generator uses its integrated SCM repository, which has day-to-day operational support tailored specifically for highly interactive, prototyping, iterative development environments.

The team repository of VisualAge Generator addresses programmer needs by providing source configuration support. Although the software configuration and source version control are excellent, they may not be sufficient to fulfill other enterprise software development needs. VisualAge Generator customers can take advantage of many TeamConnection facilities through a bridge function that allows you to transfer VisualAge Generator source definitions to and from its integrated SCM repository, and the TeamConnection Enterprise Server repository.

In addition, VisualAge Generator users can take advantage of the TeamConnection build facilities when using this bridge. VisualAge Generator creates the build structure, including the dependency structure and the processing steps, freeing you from the need to do this. You can issue a TeamConnection build request and TeamConnection will issue the appropriate commands to perform a VisualAge Generator build. However, GUIs defined in VisualAge Generator cannot currently participate in the TeamConnection build process. This gives enterprise-wide teams access to the excellent SCM support provided by the team repository of VisualAge Generator, along with the scalable, enterprise support provided by VisualAge TeamConnection Enterprise Server.

A.23.5 IBM VisualAge for Smalltalk

VisualAge is an industrial-strength, object-oriented visual application development environment. You can use VisualAge to rapidly develop and execute graphical client-server applications for OS/2, Windows, AIX, HP-UX, and Solaris.

VisualAge Smalltalk customers can take advantage of many VisualAge TeamConnection Enterprise Server facilities thanks to a bridge function that allows you to transfer VisualAge Smalltalk source definitions to and from its integrated SCM repository and the VisualAge TeamConnection Enterprise Server repository.

With the pure object-oriented application development environment of VisualAge Smalltalk, you can easily reuse or extend your applications. With VisualAge, you can immediately begin developing advanced, mission-critical, object-oriented applications and learn its technology at your own pace.

For more information on IBM VisualAge for Smalltalk, see:

http://www.software.ibm.com/ad/smalltalk/

For day-to-day development, VisualAge Smalltalk uses its integrated SCM repository, which has operational support tailored specifically for highly interactive, prototyping, iterative development environments.

A.23.6 IBM VisualAge PL/I

IBM VisualAge PL/I provides a PL/I development environment on Windows NT that is designed to allow you to create applications that can run on mainframe, workstation, or client/server systems with access to DB2 and other data systems.

VisualAge PL/I provides an optimizing compiler that contains a rich implementation of the PL/I language as well as support to provide compatibility with mainframe PL/I. The compiler also includes the power of the macro facility, include preprocessor, and SQL preprocessor.

For more information on IBM VisualAge PL/I, see:

http://www.software.ibm.com/ad/pli/

TeamConnection provides the infrastructure for team programming and software configuration management services for IBM VisualAge for PL/I and can greatly enhance team productivity. In addition to the standard integration points, storage and control of PL/I files, and building PL/I executable, TeamConnection is integrated with the Workframe component of IBM VisualAge for PL/I. Files stored in TeamConnection can be reflected in Workframe projects, and TeamConnection actions can be invoked from a Workframe project. With TeamConnection's ability to build applications for MVS, the IBM VisualAge PL/I for OS/390 compiler can also be used as a build tool with TeamConnection.

A.23.7 Non-IBM Tools

VisualAge TeamConnection also provides a means for non-IBM tools to work across the enterprise. VisualAge TeamConnection supports the Microsoft Source Code Control Application Program Interface (SCC API). Tools implementing this interface gives these tools basic TeamConnection integration from the IDE. This means tools such as Microsoft VisualBasic, Microsoft Visual C++, Rational Rose 98 (Enterprise Edition), and Sybase PowerBuilder can use VisualAge TeamConnection as their source code repository.

A.24 For More Information

For more information about TeamConnection, please contact your IBM Representative. If you don’t have an IBM Representative, please visit the IBM VisualAge TeamConnection Enterprise Server library at:

http://www.software.ibm.com/ad/teamcon/library

for a list of technical reports.

Appendix B. Scripts to Aid in Migration

All the tools used here are written in Korn shell and are available via the Internet or through IBM's Intranet. Because the tools are updated periodically, we do not provide the source in this document.

You can request help for each tool by entering the "-?" parameter, for example:

release.delete.ksh -?

B.1 Details of the Tools

B.1.1 Complete a Release Prior to TeamConnection Migration

The name of the shell script is:

release.complete.ksh

Help contents:

Usage: `release.complete.ksh -f <FAMILY> -r <RELEASENAME>`

Where

`FAMILY` is the CMVC family to process and `RELEASENAME` is the active CMVC release

Note: Script must be run by a superuser.

This utility will complete work in a given release so that all file changes are committed.

B.1.2 Reassign User's Work and Delete User ID

The name of the shell script is:

reassign.work.ksh

Help contents:

Usage: `reassign.work.ksh -f <FAMILY> -s <FROMUSER> -t <TOUSER>`

Where

`FAMILY` is the CMVC family to process

`FROMUSER` is the CMVC login to be deleted

`TOUSER` is the CMVC login to assign work to

Note: Script must be run by a superuser.

This utility will reassign incomplete work from one active CMVC login to another active CMVC login. Incomplete work can be defined as:

- defects owned/originated
- features owned/originated
- verification
- components
- releases
- levels
- environments
- test records
- size records
- approval
- approver
- tracks
- fix records
- notifications deleted
- access deleted
- login deleted

B.1.3 Delete a Release from a Family

The name of the shell script is:

release.delete.ksh

Help contents:

Usage: `release.delete.ksh -f <FAMILY> -r <RELEASENAME> -d <DEFECT> -w`

Where

`FAMILY` is the CMVC family to process

`RELEASENAME` is the active CMVC release

`DEFECT` is the CMVC defect for the track

Notes: Script must be run by the family account superuser.

*O*ptions define what SCCS and mapfiles can be deleted.

This utility will delete work in a given release. It will also identify a list of vc (SCCS/PVCS) files that are not related to any other release, and that can be removed to save disk space.

B.1.4 Creating a `migcmvc` Migration Script File

The name of the shell script is:

mkmiglst.ksh

Two uses: (split into two lines):

For mkmiglst.ksh to **c**reate a CMVC to TeamConnection migration listfile

Usage: `mkmiglst.ksh [-h] [-f|-c] [-o <outputFile>] [-r <release>] ...`

For `mkmiglst.ksh` to create a CMVC to TeamConnection migration listfile:

Usage: `mkmiglst.ksh [-h] [-f|-c] [`-o <outputFile>] [-r <release>] ...

Description: Generates a migration list file to copy releases from CMVC to TeamConnection. This migration listfile is input to the `migcmvc` migration tool. The file contains the steps necessary to migrate all data related to specified releases. It contains comments to assist the user to manually customize the list file for the migration process. Instructions are displayed for using list file with `migcmvc`; the instructions are also saved in a file, myfile.instructions (by default)

If -o `OutFile` is used, the instructions are saved to OutFile.instructions.

Options: -o: Specify file name for migration listfile. Default=myfile.lst.

`-f:` Default. Migrate only FileView (snapshot of committed release).

`-c:` Migrate ChangeView (complete history of release).

`-r` ReleaseName: The release to be migrated. Use `-r` for each release. If no releases are specified then the user is prompted for them.

Sample: # Listfile is myfile.lst, migrate FileView for two releases

mkmiglst.ksh -o myfile.lst -r v203 -r v204

This utility generates a migration script file to be used as input for the `migcmvc` utility. When the migration script is generated, instruction messages are sent to the screen; these messages are stored in the file with a prefix of ".instructions."

B.1.5 Importing a Release into CMVC before Migration

The name of the shell script is:

file.import.ksh

Help contents:

Usage: `file.import.ksh -f <FAMILY> -a <RELPATH> -m <MAPFILE> -r <RELEASE>`

Where:

`FAMILY` is the CMVC family to process

`RELPATH` is the absolute path to the file tree

`MAPFILE` is the path name component map file

`RELEASE` is the active CMVC release

Note: Script must be run by a superuser.

`MAPFILE` format pathname|compname, assuming empty release.

This utility loads files into CMVC from a given root directory. It requires that the user provide a map file that maps the path names of the files to the components into which the files are to be loaded.

B.2 Obtaining the Tools

The tools described can be downloaded as follows:

- From the IBM intranet (only for IBM employees).
- From the Internet (open to everyone).

Notes: The tools referenced in this document are available from FTP sites instead of being included here, so that we can make updates as necessary.

B.2.1 IBM Intranet

B.2.1.1 Web Home Page

You can access the CMVC Service/Development Home Page at:

http://keithp.raleigh.ibm.com/&tilde.cmvcsupt

From the index at the top of the page, select the section **Tools to migrate from CMVC to TeamConnection**.

B.2.1.2 FTP

You can download the code from our internal FTP site, by doing:

1. ftp keithp.raleigh.ibm.com

2. Log in as "anonymous" and for password give your e-mail address.
3. cd pub/cmvc/fixes/tr-tools
4. Binary
5. Get fileName
6. Quit

B.2.2 Internet

B.2.2.1 Web Home Page

Not available.

B.2.2.2 FTP

You can download the code from our external FTP site, by doing:

1. ftp ftp.software.ibm.com
2. Log in as "anonymous" and for password give your e-mail address.
3. cd ps/products/cmvc/fixes/tr-tools
4. Binary
5. Get fileName
6. Quit

Appendix C. TeamConnection V2.x and V3.1, Differences

This appendix provides a summary of the differences between VisualAge TeamConnection Version 2 2 and VisualAge TeamConnection Enterprise Server Version 3. For more details, see the files "readme.txt" and "relnotes.htm" in the CD-ROM for VisualAge TeamConnection.

C.1 Main Differences

The main differences between VisualAge TeamConnection Version 2 and Version 3 are listed below:

- Due to limitations in scalability, the database management system was replaced from ObjectStore (in Version 2) to DB2 Universal Database Version 5 (in Version 3).
- Because of the change in database management system, in Version 3 you cannot longer create a separate database for each release.
- The Solaris platform is now supported in Version 3 for the server and the client.
- A Web Client has been added in Version 3 to allow interaction with TeamConnection commands via a Web browser. For more details, see “New Web Client” on page 194.
- In TeamConnection Version 3, an interface was added for Lotus Notes, in which you can use Lotus Notes to manage requirements, test cases or development, and then manage the corresponding TeamConnection features or defects. For more details see “Lotus Notes Integrated Databases” on page 194.
- In TeamConnection Version 3, an interface was added for Source Code Control (SCC) API, in which other applications such as PowerBuilder can check out and check in parts from TeamConnection. For details, see “Source Code Control API” on page 195.
- The tcmerge utility in Version 2 was available only on Intel. However, in Version 3, the tcmerge utility was rewritten in Java and thus, it is available in all our supported platforms. Furthermore, the paradigm of the tool was changed from two-file merge into three-file merge in order to support concurrent development.
- The tcadmin utility in Version 2 was available only on Intel. Now in Version 3, the tcadmin utility was rewritten in Java and thus, it is available in all our supported platforms. It now includes a utility that monitors family usage and has expanded user exit support.

- Bridge for VisualAge TeamConnection for Smalltalk. For details, see “VisualAge TeamConnection For Smalltalk Bridge” on page 195.
- Tivoli Software Distribution Packaging support. The Tivoli Software Distribution (TSD) packaging tool supports automated distribution between a single Tivoli Software Distribution server and Tivoli-managed clients.
- Part shadowing into a file system. For details, see “Part Shadowing Capability” on page 195.
- Configurable Fields updates in Version 3.
  - Family Administrators can now define configurable field types that allow users to specify multiple values for a configurable field or require users to enter a value that meets specific criteria, such as a six-digit number or a two-character string. They can also define general help text for configurable field types and specific help text for each value defined for a configurable field type.
  - It is possible to add extra configurable fields to the following objects:
    - Releases.
    - Workareas.
  - Parts: propagate values when doing a -link action.
- tcstop utility to stop the TeamConnection family daemons. A new utility, tcstop.exe, has been provided to stop any daemons associated with a TeamConnection family. The syntax is as follows:

  tcstop Family
- Part/Release Merge function. For details, see “Part/Release Merge Function” on page 196.
- Translation support attributes for the Part command. For details, see “Translation Support Attributes for the Part Command” on page 197.
- The new build server in Version 3 replaces the build agent and the build processor in Version 2.
- New build server for MVS and MVS/OE. It uses a single build server component to build OS/390 applications.

C.2 Miscellaneous Differences between TeamConnection V2 And V3

The TeamConnection report command in Version 2 checks the access lists for a user and returns only the rows of data that a user is authorized to see. This function is planned to be added back to Version 3.

A new shipped authority group called "admin" has been added. It includes User and Host actions that were formerly allowed to a superuser only, including the following actions:

- UserCreate
- UserDelete
- UserModify
- UserRecreate
- UserView
- HostCreate
- HostDelete

This new authority group can help to reduce the number of superUsers because previously, some customers had to give superuser authority to team leaders in order to create users and host lists.

New environment variables have been added:

- The environment variable TC_MODPERM controls whether the file attribute of a file on a workstation is changed to read-only or not after a teamc part -create, -checkin, or -unlock command. If it equals off or OFF, the file attribute will not be changed.
- The environment variable TC_BACKUP controls whether a backup file of a file on a workstation is created during a teamc part -checkout or part -extract command. If it equals off or OFF, the file attribute will not be changed.
- The environment variable TC_CATALOG allows the specification of a message catalog file. With this variable, the exact location of a message catalog can be specified to a TeamConnection daemon.

Users can use the keyword "null" to change to null (0 characters) certain fields, such as the release name in Defects, and configurable fields.

"hold" attributes have been added for workarea state transitions. Use of these attributes will cause the current state of the workarea to be retained when processing the indicated commands, rather than having the state move to the normal next state. This change affects work area fix, test, and commit.

For example, you can create a new release process with all the possible subprocesses in $HOME/relproc.ld or via tcadmin:

```
track_all|approval

track_all|fix

track_all|driver
```

track_all|test

track_all|track

track_all|trackfixhold

track_all|trackcommithold

track_all|tracktesthold

Users can receive customized messages when they attempt to run commands that are not "read-only" against a Team Connection family that is running in maintenance mode.

E-mail addresses are treated by TeamConnection as case sensitive.

An abstract has been added to notifications for all workarea related actions.

New user exit parameters have been added and documented.

A message buffer is now available to user exits on errors. The message buffer (msgBuff) will always be null for exit point 0.

The "+" and "-" characters can be used as the first character of a parameter value. The + or - must be doubled in a command to differentiate between the parameter value and a new flag. Only one + or - will be placed in the database for the value.

Users can have separate INI resource files to be used with teamcgui for each of the families they work with.

The action "teamc report -testServer" can now return to users the names and values of certain environment variables that are set on the Team Connection server.

The action "teamc report -general" was added to allow users to customize their reports.

The action "teamc workarea -extract" was added to allow users to extract only those files changed for specified workareas. In addition to a delta extract, you can do a full extract of parts in the workarea.

The action "teamc driver -extract" allows a full extract of an uncommitted driver.

The actions "teamc test -create" and "teamc test -delete" were added to explicitly create and delete test records.

An -ignore flag has been added to the action "teamc driver -commit" to ignore any missing prerequisites that are identified during the prerequisite and corequisite checking phase.

> **Warning**: We consider this feature to be potentially dangerous to the consistency of your baselines. Thus, we suggest that until you fully understand all the ramifications of ignoring the prerequisites when committing a driver, you should ***not*** give authority for the following action, even to the projectlead authority group (see authorit.ld file), which has it by default: DriverCommitIgn.

A -driver flag has been added to the action "teamc driver -check". This change essentially performs the action "teamc workarea -check -driver" for each workarea in the driver being checked; this action determines the requisites of the workarea as if the specified committed driver was the last committed driver.

An exclude flag has been added to workarea -check to list work areas that should be removed from the input list to satisfy any prereq conditions.

User configurable fields have been added to Part -create.

In Version 3, the attributes "answer" and "release" have been added for Feature.

The answer field is required on the feature -accept and feature -return commands as it is for Defects. The release field is optional on the feature -open command and holds the release name, as it does for Defects.

Ownership of an "original" verification record of a defect or feature will be changed when the originator (originlogin) field of the defect or feature itself is modified. The "duplicate" verification records are not affected.

Support for list and regular expression configuration "kinds" has been added in the configurable field types (config.ld).

Appendix D. TeamConnection Authority Groups

There are many actions that users can perform against TeamConnection objects. It would be tedious to grant one action at a time to each of your users. Instead, you can grant a user the authority to perform a group of actions, called an authority group. For example, the managers of a project might want to view only the status of certain TeamConnection objects, while the developers need to view objects and also check in, check out, and extract parts. You can grant access to an authority group for either of these jobs.

The family administrator is responsible for the authority groups that your organization uses. IBM ships a set of default authority groups with TeamConnection. Determine whether these meet your needs or whether you need to change them. As your organization grows and as your needs change, you will probably want to revise your authority groups.

D.1 Creating or modifying authority groups

When the database is initially created, the authority table contains the default values for the authority groups. If the default authority groups are not adequate for your development organization, you can create new authority groups or modify existing groups. Authority groups can be created or modified at any time during your development cycle.

First, decide what group of actions the intended users are required to perform. If there is a shipped authority group that closely matches your needs, you might want to modify that group. Otherwise, you will need to create a new group.

We recommend you use the Family Administrator GUI to create or modify authority groups; however, if you prefer to do this manually, see “Manual Modifications:” on page 284.

D.2 Using the Graphical User Interface

Before you do this task, we recommend that you stop the family server.

1. Do one of the following from a server machine to display the TeamConnection Family Administrator window:

- From the TeamConnection Group folder on the desktop, double-click on the *Family Administrator* icon.
- Type *tcadmin* from a command prompt.

2. 2. Display the family icon's pop-up menu; then select *Properties.* The properties notebook appears.
3. Select the *Groups* page and then select the *Settings* push button under Authority to display the authority group settings.

4. Do one of the following:

- To change an existing group, highlight the group from the Authority Group list, and then select or deselect the appropriate actions from the Actions list.
- To create a new group, follow these steps:
  1. Select the *New* push button, type the name of the new group in New Authority Group window, and then select the *OK* push button.
  2. From the *Actions* list, select the actions you want to include in the new group.
  3. To save the new group, select the *Apply* push button.
- To delete a group, select it from the *Authority Group* list and then select the *Delete* push button. When the conform delete window appears, select *Yes*.
- To rename a group, follow these steps:
  1. Select a group from the *Authority Group* list.
  2. Select the *Rename* push button.
  3. Type a new name in the *New name* field of the Rename Authority Group indow, and then select the *Apply* push button.
  4. When you finish making changes to your notebook pages, select *OK* to save your changes and exit from the notebook.

The changes will not take effect until you start the family server.

D.3 Manual Modifications:

When a TeamConnection family is created, the authority table is primed with default information contained in the authorit.ld file. This section provides instructions for manually editing the authorit.ld file to create new authority groups or change information about existing authority groups. When you change the authorit.ld file, you must also reload the contents of the authority table in the TeamConnection database.

D.3.1 Editing the authorit.ld file

To add new authority groups or to add actions to an existing authority group, edit the authorit.ld file. It is recommended that you keep the authorit.ld file in the family directory. If you want to maintain common authority group definitions for more than one family, however, you can store this file in a common directory; but you will need to specify the fully-qualified path name for the authorit.ld file when you load it using the *fhclauth* command.

Add entries to the file using the following format:

```
AuthorityGroup                              ActionName
```

D.3.1.1 AuthorityGroup

This is the name of an existing authority group or the name of a group that you are creating. The name can be 15 characters long; it cannot contain blanks, tabs, or vertical separators. For an existing authority group, type the name exactly as it appears in the database table. The default names provided by IBM use all lowercase characters.

D.3.1.2 ActionName

This is the name of an existing TeamConnection action. Specify only one action per entry. You must type the name exactly as it appears in the database table. Refer to the list of actions in the TeamConnection User's Guide for the correct spelling and capitalization. Certain actions cannot be included in an authority group. These actions are noted in the table found in "Appendix F. Worksheets" on page 299 of this manual.

D.4 Reloading the authority table

Whenever you change the authorit.ld file, you must reload the contents of the authority table before your users can use the new and changed authority groups. You can reload the authority table as often as necessary. We recommend that you stop the family server before you reload the authority table.

To reload the authority table, issue the following command from the server machine. Before issuing this command, ensure that the TC_FAMILY environment variable is set to the correct family name and TC_DBPATH is set to the correct database path name.

```
fhclauth path\authorit.ld
```

Where:

path\authorit.ld is the path name of the authorit.ld file. If you specify a fully-qualified path name, TeamConnection looks for the authorit.ld file in the path you specify. If you specify only *authorit.ld* with no directory path, TeamConnection looks for the file in the directory specified by the TC_DBPATH environment variable. To verify that the authority table loaded correctly, use the report command to generate a report on the authority table. For example, to verify that a new authority group named general was added to the table, type the following from a client machine on an OS/2 or TeamConnection command line:

```
teamc report -view authority -where "name='general'"
```

If the table loaded correctly, information about the general authority group appears. If the authority table did not load correctly, make the necessary changes to the authorit.ld file and run the fhclauth command again.

Appendix E. Process Improvements after Migration

Now that there has been time for customers to move to VisualAge TeamConnection (both Version 2 and Version 3), they have provided feedback on how TeamConnection allows for process improvements over CMVC. Here are some of the most significant process improvements:

Note: Where a specific customer made the process improvement, the heading begins with "Customer scenario:".

E.1 Automated Software Build and Distribution

The VisualAge TeamConnection build facility is a general purpose facility. When used with the shadow facility and a Web page, the software build and distribution processes can be automated and can be managed easily.

For example, we wanted to take full advantage of VisualAge TeamConnection by adding a few things to the family setup to ensure that a given application would build reliably and automatically. Also, we wanted to track the results of the build.

In addition, once we built the application, we needed to distribute it to team member; some of whom did not have access to our VisualAge TeamConnection family. In order to always distribute the proper version of the application, we shadow the committed release to a directory on the server and use a Web page to point to the shadow. This allows anyone with access to our Web page to quickly select the files that compose our application and then to download them into the client workstation.

Additionally, the Web page used provides customized access to the VisualAge TeamConnection family through the VisualAge TeamConnection Web interface. Included in this customized Web page is a reference to a second Web page the reports the activity of the family server. Although this is not directly related to building our application, it is useful information for family administrators who monitor the build.

E.2 Assigning Multiple Areas to Users

Multiple workareas makes it easier for people to work on the same problem without getting in each other’s way and maintain better records of their changes. Because many defects require the participation of more than one person, several customers have been pleased by the ability to create and assign a workarea to each contributing person. For example, a workarea can be created for the developer changing the GUI, another for the developer changing the database code, and a third for the technical writer updating the documentation. Each person makes changes in their own workarea without interfering with the others. As a result, when all of the workareas are added to drivers (not necessarily the same driver), there is an accurate and easy way to follow the change history for each person's work.

In CMVC, the solution was usually to create extra fix records or to change the ownership of fix records as different people work on their portion of the defect. Of course, the CMVC solutions were really only workarounds because a fix record is automatically created for each component impacted (which does not always correlate with who is doing the work).

E.3 Allowing Incremental Development in One Feature

Multiple workareas for a feature or a defect also allows for easier incremental development.

A single team member, or a group, can open multiple workareas for a feature or defect, then add their changes incrementally, until the solution is complete. Team members can check changes into workareas, one workarea at a time, and have them committed to drivers or directly into releases without the hassle and confusion of using multiple features and defects.

In TeamConnection, a feature or defect remains in working state as long as at least one workarea is still in the Fix state. Any number of workareas can be integrated, committed and completed prior to moving the feature or defect to test, verify or complete.

Further, TeamConnection allows a team member to freeze the state of a workarea at any time, without needing to commit that workarea.

In CMVC, the only available solution was to:

1. Let a feature or defect go to test, verify or complete.
2. Explain to the tester or person opening the feature/defect that they have to wait for additional changes.
3. Open another feature/defect, typically using some naming convention that reflected a relationship between the old and new feature/defect,
4. Work all subsequent features/defects until done.
5. Update the original feature/defect and tell everyone that the change was really done.

6. This is a cumbersome process that TeamConnection has made obsolete.

E.4 Improving Communication between Team Members

TeamConnection encourages better communication between team members by allowing them to link parts from other workareas or by refreshing a workareas from another when changing the same parts in a release. The linking of parts is preferable to the refreshing of workareas because the refreshing action will get all the changed versions from the source workarea and this may have unexpected side effects.

Whenever team members need to make changes to the same parts there is contention for these parts. Even though the first person to check out a part and check it back into their workarea "wins" (that is, there is no need to take any special action in order to integrate their workarea into a driver), other team members can also update these parts before the workarea containing the part is committed.

The second team member that tries to check out the part to their workarea will get a message indicating that they must link the part or refresh from the first workarea in order to change the part (the first workarea is identified in the message) prior to check-out.

After the second team member checks in the part that as linked from another workarea, then the original workarea becomes a prerequisite. Thus, the second developer will need to talk to the owner of the first workarea in order to resolve issues like "when will your workarea be added to a driver?".

Note that this works particularly well when the release is in "serial" mode. In many cases, the customer sees TeamConnection’s "concurrent" (parallel) mode and initially wants to use that mode for all the releases, believing it will allow team members to work more independently and quickly; the main drawback is that when a part is checked out by several developers at the same time, then there is a lot of time involved in resolving the collisions (conflicts) when merging the changes during the check-in process.

However, once these customers see the way these notifications and prerequisites force communication, and how serial mode reduces the need for manually merging of changes, these customers generally choose serial mode as their preferred release process.

CMVC does little to encourage communication other than sending notifications to component owners (unchanged in TeamConnection) and identifying conflicts with Level -check.

E.5 Sharing Part Changes before Promoting

TeamConnection allows the sharing of parts that have changed but that have not been committed yet by using "teamc part -link" (preferable) or "teamc workarea -refresh".

> This eliminates the confusion caused by the CMVC "current" version concept. Using TeamConnection’s ability to do a part -link or to refresh any workarea from any other workarea in a release, including the release itself or a driver, the CMVC concept of the "current" version of a part in a release is no longer needed.

Further, the issue of how best to manage a "current" version is eliminated. Each version of a part is associated with a specific workarea; when the workarea is integrated and a driver member is created, then that workarea ***is*** the driver. When that driver is committed, the workarea ***is*** the release. Therefore, each version of a part is clearly associated with a set of changes, not just the history of that part (which rarely provides much useful information).

Finally, each team member can build the contents of their workarea without picking up any unwanted changes, as happens so often in CMVC when extracting the "current" release. As a result, multiple team members can simultaneously be working on changes that involve the same part without confusion, and without inadvertently using parts changed by others that CMVC would have included in the "current" pool.

E.6 Maintaining a Reference Directory

Because a driver is also a workarea, it is possible to extract the contents of the driver and to list the part changes included in the driver.

For example, a customer built a process around CMVC that maintained a "reference directory" containing all of the changes in the tracks that had been added to an active level; then the developers built from this reference directory for their integration testing. However, it was not easy to determine if the reference directory was up to date, or even if it contained the right versions of files.

In contrast, when using TeamConnection, they have been able to alter their process to allow developers to add workareas to a driver, then let the tools update the contents of the reference directory. As a result, they know exactly what work is ready to be tested and that the testing done in this reference directory is a valid integration test. Finally, they can build the driver using the TeamConnection build facility.

E.7 Saving the Outputs of a Build with the Build Function

Since the TeamConnection build function checks the output files (that is, executables) back into the release, it is possible to recreate an old release simply by extracting those executables. As a result, upgrading systems with new operating systems, compilers, and the like does not prevent a release from being recreated exactly as it originally was.

The issue of saving too much data and running out of disk space in the family account is addressed though pruning options on the release.

E.8 Encouraging More Frequent Check-In of Parts

Programmers, and other team members, often delay checking out, changing and checking in a part until they feel they are done with their changes (using extracted copies of the part). The most common reason for this is simple: no one wants someone else to see their incomplete work.

Unfortunately, this generally misleads other team members into believing that no one is interested in any particular part, since it is has not been updated in any active release. As a result, when a part is finally checked out it might be different than the version that was extracted and lots of manual merging of changes and retesting is necessary.

Once CMVC users hear about TeamConnection's concurrent release process, managers and team members believe that the collision resolution process and merge tool used by TeamConnection will address their problems. Actually, there is often an even easier way.

In TeamConnection, checking in a part merely replaces the previously stored copy of the part, eliminating it from the part history. In order to save a version, the workarea must be frozen. This means that if you check in a part three times with remarks and if you do not freeze the workarea, then only the remarks for the last check in will be kept and the remarks of the two previous stored copies are eliminated.

The result is that only the versions of parts that are worth saving will be preserved when the workarea is added to the driver. What this means to each team member is that they can check out, change and check in as much as they want without having their incomplete work saved and visible to others.

On top of that, workareas (including their change history) can be manually or automatically pruned from the release when their information is no longer of use.

E.9 Surviving a Security Audit

As with all large companies, someone does EDP (Electronic Data Processing) audits. In IBM’s case, the standards they audit against are defined in a document called ITSC 201. In order for CMVC to meet the ITSC 201 requirements, many specific exemptions were required (for example, use of NFS mounts to OS/2 and Windows) and several specific CMVC options were required (for example, use of CMVC_NFS_DISABLE). As we have found out, IBM is not unique.

TeamConnection resolves these security issues:

- No processes should run as root. CMVC runs the cmvcd daemons with root authority in order to do NFS mounts, and then to perform the setuid() function so that the files extracted to the server are owned by a user and not by the CMVC family id.

  Since TeamConnection performs extracts across sockets, there is no need to setuid to another user and therefore no need for "teamcd" to be a root process. Thus, TeamConnection has no root/setuid processes.

- CMVC allows users to see the contents of the SCCS archives in the family accounts "vc" directory structure. This is required in order to perform extracts from a setuid process. The workaround is to not let anyone other than the family administrator log onto the system running CMVC, making the machine a dedicated CMVC server.

  TeamConnection stores all part data in the family database. Since TeamConnection also implements authentication for each row of data in every table, including through the use of the "teamc report" command, parts are not accessible through TeamConnection unless you are authorized.

  Further, since the DB2 UDB databases are controlled by the family account and are only accessed by the TeamConnection daemons, there is no need to set permissions in such a way as to give another user on the system the ability to read the contents of that database.

- TeamConnection client will own all files it writes. As with Part -extract, all other -extract actions use the same socket mechanism to transfer data. This ensures that the client will write and own all files, eliminating the need to set any permissions for sharing files.

- TeamConnection adds password protection on top of host/userid. The "trusted host/userid" mechanism can be defeated through malicious means. As a result, it is not always good enough to satisfy the requirements of an auditor. In those cases, or merely because it could be more appealing to you, TeamConnection provides a password scheme.

  This password scheme is secure because:

  - The password is encrypted on the client (that is, no unencrypted password is ever passed across a socket).
  - If the server goes down, or the user logs out, the "token" for that user expires.
  - The password is entered on a separate prompt so that it does not show up in a "ps" command (to show the processes that are running).

Appendix F. Naming Restrictions for DB2 User IDs

F.1 Group IDs, Instances, and Databases

DB2 has certain restrictions for the names of the DB2 databases. As the TeamConnection family name is also used to name the DB2 database, these DB2 restrictions become TeamConnection restrictions. Furthermore, the DB2 names for user IDs, group IDs, and instances have these restrictions:

The name you specify:

- Can contain 1 to 8 characters.
- Cannot be any of the following:
  - USERS
  - ADMINS
  - GUESTS
  - PUBLIC
  - LOCAL
- Cannot begin with:
  - IBM
  - SQL
  - SYS
- Cannot include accented characters.
- To avoid potential problems, do not use the special characters @, #, and $ in a database name if you intend to have a client remotely connect to a host database.At the same time, these characters are not common to all keyboards, so do not use them if you plan to use the database in countries having different keyboard layouts.
- In general:
  - On OS/2, use uppercase names.
  - On Windows 95 and Windows NT, use any case.
  - On UNIX, use lowercase names.

Appendix G. Uninstalling DB2 UDB

This appendix shows you how to uninstall (remove) DB2 products.

G.1 Uninstalling DB2 from UNIX

You need to perform the following steps:

1. Stop and back up the VisualAge TeamConnection families.
2. Stop the Administration Server.
3. Stop all DB2 Instances.
4. Remove the Administration Server.
5. Remove DB2 Instances (this step is optional).
6. Remove the DB2 products.

G.1.1 Stop and Back up the VisualAge TeamConnection Families

Use tcadmin or tcstop to stop all the VisualAge TeamConnection families. If you want to keep the database for the families, then back it up for each one.

G.1.2 Stop the Administration Server

You must stop the Administration Server before you remove DB2 products. To stop the Administration Server, you need to perform the following steps:

1. Log in as one of the DB2 instances.
2. Obtain the name of the Administration Server using the following command:

   ```
   $DB2_HOME/bin/db2set -g DB2ADMINSERVER
   ```
3. Stop the Administration Server. See “Stopping the Administration Server” on page 155 for details.
4. Exit the session.

G.1.3 Stop All DB2 Instances

You must stop all DB2 Instances before you remove DB2 products. To stop DB2 Instances, you need to perform the following steps:

1. Log in as the owner of a DB2 instance.
2. Obtain a list of the names of all DB2 instances on your system using the following command:

```
db2ilist
```

3. Stop the instance. See “Stopping a DB2 Server Instance” on page 149 for details.
4. Exit the session.
5. Repeat these steps for each instance.

G.1.4 Remove the Administration Server

You must remove the Administration Server before you remove DB2 products. To remove the Administration Server, you need to perform the following steps:

1. Log in as root.
2. Remove the Administration Server. See “Removing the DB2 Administration Server” on page 155 for details.

G.1.5 Remove the DB2 Instances

You can optionally remove some or all of the DB2 Version 5 Instances on your system. Once an instance is removed, all the DB2 databases owned by the instance, if any, will not be usable. Remove DB2 Instances only if you are not planning to use DB2 Version 5 products, or if you do not want to migrate existing instances to a later version of DB2. To remove DB2 Instances, you need to perform the following steps:

1. Log in as root.
2. Obtain a list of the names of all DB2 instances on your system using the following command:

```
$DB2_HOME/bin/db2ilist
```

3. Remove the instance. See “Removing Instances” on page 150 for details.

G.1.6 Clean up the DB2 registry

In order to have a clean system before reinstalling DB2, we recommend cleaning up the DB2 registry:

1. Log in as root.
2. Execute the following command:

```
rm -fr /var/db2/v5
```

For more information about the DB2 registry, see “Where Are The Files Used with The Profile Registry?” on page 141.

G.1.7 Remove the DB2 Products

The following steps describe how you can remove DB2 products from UNIX operating systems.

G.1.7.1 Remove the DB2 Products on AIX Systems

You can remove the DB2 products on Version 4.1 or later of the AIX operating system using SMIT interface as follows:

1. Log in as root.
2. Type `smit install_remove` to proceed directly to the Remove Software Products screen.
3. Press F4 to display a list of the software to remove. Press F7 for each of the entries that have a prefix of db2_05_00.
4. Press Enter to start removing the DB2 products.
5. After the product is removed, exit smit.
6. Change the directory to /usr/lpp/db2_05_00 and see if any directories and files remain. There may be some directories or files that were not removed in /usr/lpp/db2_05_00. If you want to remove them, type the following:

   cd /usr/lpp

   rm -fr ./db2_05_00

   You can also remove all DB2 Version 5 products on Version 4.1 or later of the AIX operating system, using the `installp` command with the uninstall option:

   installp -u db2_05_00

G.1.7.2 Remove the DB2 Products from HP-UX Systems

You can remove the DB2 products on the HP-UX operating system using the swremove program as follows:

1. Log in as root.
2. Use swremove to remove some or all of the DB2 Version 5 products. Select all the file sets that begin with DB2V5. If you have applied patches, you can select the ones that begin with PDB2 or that have a description identifying them as such.

G.1.7.3 Remove the DB2 Products from Solaris Systems

You can remove the DB2 products on the Solaris operating system using the pkgrm program as follows:

1. Log in as root.

2. Determine the packages for all DB2 for Solaris related products you have installed on your system by typing:

   pkginfo | grep -i db2 | grep 50

3. Remove all packages listed in Step 2 with the pkgrm command.
4. Select **Yes** at the prompt for each package to be removed.
5. Before removing a package, all its dependent packages must be removed first. You must remove packages in a particular order, which is mentioned in the Chapter 28 "Removing DB2 Products" from the *Quick Beginnings for UNIX* manual.

G.2 Uninstalling DB2 from Windows NT

1. Log in as Administrator or as another member of the Administrator group.
2. Stop and back up the VisualAge TeamConnection families.
3. Stop all the DB2 processes:
   - Start -> Settings -> Control Panel -> Services
   - Select one by one, all those DB2 services that are active and click on the **Stop** button:

     DB2 - DB2 # This is the DB2 instance

     DB2 - DB2DAS00 # This is the DB2 Administrator Service

     DB2 Security Server

     TME 10 NetFinity Support Program

   Also, stop the above TME service to avoid the following problem when trying to remove the code:

     DB2 is currently running and therefore cannot be updated.

     Stop the DB2 processes and try again.

4. **Start -> Settings -> Control Panel -> Add/Remove Programs**.
5. Select **DB2** or **IBM DataBase 2 Products** and click on the **Add/Remove** button. Handle appropriately all the dialogs during the uninstallation process.
6. After the uninstallation, notice that the Start -> Programs does not show the entry for DB2 anymore. Also notice that the Services (in the Control Panel) does not show the services related to DB2 anymore.
7. Reboot your workstation.

8. (Optional) You may want to delete the directory for the DB2 instances, such as C:\DB2 and C:\SQLLIB.

G.3 Uninstalling DB2 from OS/2

1. Stop and back up the VisualAge TeamConnection families.
2. Stop all the DB2 processes:
   - Select the **DB2 Control Cente**r.
   - Select **Control Center -> Shutdown DB2 Tools**.
3. Double click on the folder **DB2 V5** and then double click on the folder **DB2 for OS/2**.
4. Double click on the **Installation Utility.**
5. Select **IBM DB2 Universal Database** and from the **Actions** menu select **Delete**. Handle appropriately all the dialogs during the uninstallation process.
6. After the uninstallation, notice that the DB2 for OS/2 folder should not exist.
7. Reboot your workstation.
8. (Optional) You may want to delete the directory for the DB2 instances, such as C:\DB2 and C:\SQLLIB.

Appendix H. Special Notices

This publication is intended to help customers and IBM people to migrate from any level of TeamConnection or CMVC to the current level of VisualAge TeamConnection Enterprise Server. The information in this publication is not intended as the specification of any programming interfaces that are provided by VisualAge TeamConnection Enterprise Server. See the PUBLICATIONS section of the IBM Programming Announcement for VisualAge TeamConnection Enterprise Server for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM ("vendor") products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate

Appendix I. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

I.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see "How to Get ITSO Redbooks" on page 309.

- SG24-4648 Introduction to the IBM Application Development Team Suite
- SG26-2008 Family Planning and Application Development

I.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Number	Kit
System/390 Redbooks Collection	SBOF-7201	SK2T-2177	
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022	
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038	
Lotus Redbooks Collection	SBOF-6899	SK2T-8039	
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044	
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849	
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040	
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041	
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043	
Application Development Redbooks Collection	SBOF-7290	SK2T-8037	

I.3 Other Publications

These documents are also relevant as further information sources:

- SC09-1596-01 *IBM CMVC Client Installation and Configuration*
- SC09-1597-01 *IBM CMVC User's Reference*
- SC09-1631-02 *IBM CMVC Server Administration and Installation*
- SC09-1633-00 *IBM CMVC Concepts*

- SC09-1635-01 *IBM CMVC Commands Reference*
- SC34-4551 *TeamConnection, Administrator’s Guide*
- SC34-4552 *Getting Started with the TeamConnection Clients*
- SC34-4499 *TeamConnection, User’s Guide*
- SC34-4501 *TeamConnection, Commands Reference*

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at `http://www.redbooks.ibm.com/`.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Redbooks Web Site on the World Wide Web**

  `http://w3.itso.ibm.com/`

- **PUBORDER** – to order hardcopies in the United States
- **Tools Disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLCAT REDPRINT
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get BookManager BOOKs of redbooks, type the following command:

  ```
  TOOLCAT REDBOOKS
  ```

  To get lists of redbooks, type the following command:

  ```
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  ```

  To register for information on workshops, residencies, and redbooks, type the following command:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1998
  ```

- **REDBOOKS Category on INEWS**
- **Online** – send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** – send orders to:

	IBMMAIL	**Internet**
In United States	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** – send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** – send orders to:

United States (toll free)	1-800-445-9269
Canada	1-800-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 408 256 5422 (Outside USA)** – ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **On the World Wide Web**

Redbooks Web Site	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

Redpieces

For information so current it is still in the process of being written, look at "Redpieces" on the Redbooks Web Site (`http://www.redbooks.ibm.com/redpieces.html`). Redpieces are redbooks in progress; not all redbooks become redpieces, and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

IBM Redbook Order Form

Please send me the following:

Title	**Order**	**Quantit**

First name	Last name	
Company		
Address		
City	Postal code	Country
Telephone number	Telefax number	VAT number

☐ Invoice to customer

☐ Credit card number

Credit card expiration	Card issued to	Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

Glossary

This glossary defines the terms and abbreviations used in this book. If you do not find the term you are looking for, refer to the IBM Dictionary of Computing, New York: McGraw-Hill, 1994.

This glossary includes terms and definitions from the American National Standard Dictionary for Information Systems, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 1430 Broadway, New-York, New-York 10018.

A

absolute path name.   A directory or a part expressed as a sequence of directories followed by a part name beginning from the root directory.

access list.   A set of objects that controls access to data. Each object consists of a component, a user, and the authority that the user is granted or is restricted from in that component. See also authority and restricted authority.

action.   A task performed by the Team-Connection server and requested by a Team-Connection client. A TeamConnection action is the same as issuing one TeamConnection command.

alternate version ID.   In collision records, the name of a version of a driver, release, or work area where the conflicting version of a part is visible.

approval record.   A status record on which an approver must give an opinion of the proposed part changes required to resolve a defect or implement a feature in a release.

approver.   A user who has the authority to mark an approval record with accept, reject, or abstain within a specific release.

approver list.   A list of user IDs attached to a release, representing the users who must review part changes that are required to resolve a defect or implement a feature in that release.

attribute.   Information contained in a field that is accessible to the user. TeamConnection enables family administrators to customize defect, feature, user, and part tables by adding new attributes.

authority.   The right to access development objects and perform TeamConnection commands. See also access list, base authority, explicit authority, implicit authority, and restricted authority.

B

base authority.   The set of actions granted to a user when a user ID is created within a TeamConnection family. See also authority. Contrast with implicit authority and explicit authority

build.   The process used to create applications within TeamConnection.

build agent.   A program that handles access to persistent data on behalf of the build processor. Each build agent is connected to one and only one build processor, through a TCP/IP connection.

build cache.   A directory that the build processor uses to enhance performance.

build dependent.   A TeamConnection part that is needed for the compile operation to complete but will not be passed directly to the compiler. An example of this is an include file. See also dependencies.

builder.   An object that can transform one set of TeamConnection parts into another by invoking tools such as compilers and linkers.

build event.   An individual step in the build of an application, such as the compiling of hello.c into hello.obj.

build input. A TeamConnection part that will be used as input to the object being built.

build output. A TeamConnection part that will be generated as output from a build, such as an .obj or .exe file.

build pool. A group of build servers that resides in an environment. The environment in which several build servers operate. Typically, several servers are set up for each environment for which the enterprise develops applications.

build processor. A program that invokes tools, such as compilers and linkers, that construct an application. Each build processor is connected to one and only one build agent, through a TCP/IP connection. See also build agent and build cache.

build scope. A collection of build events that implement a specific build request. See also build event.

build script. An executable or command file that specifies the steps that should occur during a build operation. This file could be a compiler, a linker, or the name of a batch file you have written.

build server. The combination of a build processor and a build agent. See also build agent and build processor.

build target. The name of the part at the top of the build tree that is the final output of a build. TeamConnection uses the build target to determine the scope of the build. See also build tree.

build tree. A graphical representation of the dependencies that the parts in an application have on one another. If you change the relationship of one part to another, the build tree changes accordingly.

C

change control process. The process of limiting and auditing changes to parts through the mechanism of checking parts in and out of a central, controlled, storage location. Change control for individual releases can be integrated with problem tracking by specifying a process for the release that includes the tracking subprocess.

check-in. The return of a TeamConnection part to version control.

check-out. The retrieval of a version of a part under TeamConnection control. In nonconcurrent releases, the check-out operation does not allow a second user to check out a part until the first user has checked it back in.

child component. Any component in a TeamConnection family, except the root component, that is created in reference to an existing component. The existing component is the parent component, and the new component is the child component. A parent component can have more than one child component, and a child component can have more than one parent component. See also component and parent component.

child part. Any part in a build tree that has a parent defined. A child part can be input, output, or dependent. See also part and parent part.

client. A functional unit that receives shared services from a server. Contrast with server.

collision record. A status record associated with a work area or driver, a part, and one of the following: the work area or driver's release, or another work area. TeamConnection generates a collision record when a changed version of a part conflicts with a previously committed and integrated version of the same part. This is only related to concurrent development mode, when more than one developer can check out the same version of the same part concurrently. (In serial development, the part will be locked after the first check out.)

command. A request to perform an operation or run a program from the command line interface. In TeamConnection, a command consists of the command name, one action flag, and zero or more attribute flags.

command line. (1) An area on the Tasks window or in the TeamConnection Commands window where a user can type TeamConnection commands. (2) An area on an OS/2 window where you can type TeamConnection commands' committed version. The revision of a part that is visible from the release.

common part. A part that is shared by two or more releases, and the same version of the part is the current version for those releases.

comparison operator. An operator used in comparison expressions. Comparison operators used in TeamConnection are > (greater than), < (less than), >= (greater than or equal to), <= (less than or equal to), and = (equal to).

component. A TeamConnection object that organizes project data into structured groups and controls configuration management properties. Component owners can control access to data and notification of TeamConnection actions. Components exist in a parent-child hierarchy with descendant components inheriting access and notification information from ancestor components. See also *access list* and *notification list*.

concurrent development. Several users can work on the same part at the same time. TeamConnection requires these users to reconcile their changes when they commit or integrate their work areas and drivers with the release. Contrast with serial development. See also *work area*.

configuration management. The process of identifying, managing, and controlling software modules as they change over time.

connect. The process of linking parts so that they are included in a build.

context. The current work area or driver used for part operations.

corequisite work areas. Two or more work areas designated as corequisites by a user so that all work areas in the corequisite group must be included as members in the same driver, before that driver can be committed. If the driver process is not used in the release, corequisite work areas must be integrated by the same command. See also *prerequisite work areas*.

current version. The last visible modification of a part in a driver, release, or work area.

current working directory. (1) The directory that is the starting point for relative path names. (2) The directory in which you are working.

D

daemon. A program that runs unattended to perform a standard service. Some daemons are triggered automatically to perform their task; others operate periodically.

database. A collection of data that can be accessed and operated by a data processing system for a specific purpose.

default. A value that is used when an alternative is not specified by the user.

default query. A database search, defined for a specific TeamConnection window, which is issued each time that TeamConnection window is opened. See also *search*.

defect. A TeamConnection object used to formally report a problem. The user who opens a defect is the defect originator.

delete. If you delete a development object, such as a part or a user ID, any reference to that object is removed from TeamConnection. Certain objects can be deleted only if certain criteria are met. Most objects that are deleted can be recreated.

delta part tree. A directory structure representing only the parts that were changed in a specified place.

dependencies. In TeamConnection builds, there are two types of dependencies: **automatic**, build dependencies that a parser identifies;

manual, build dependencies that a user explicitly identifies in a build tree. See also *build dependent*.

destroy. To remove the part record from the database on the TeamConnection server. The only TeamConnection development object that can be destroyed is a part.

disconnect. The process of unlinking parts so that they are not included in a build.

driver. A collection of work areas that represent a set of changed parts within a release. Drivers are associated with only those releases whose processes include the track and driver subprocesses.

driver member. A work area that is added to a driver.

E

environment. (1) A user-defined testing domain for a particular release. (2) A defect field representing the environment where the problem occurred. (3) The string that matches a build agent with a build event.

environment list. A TeamConnection object used to specify environments in which a release should be tested. A list of environment-user ID pairs attached to a release, representing the user responsible for testing each environment. Only one tester can be identified for an environment.

explicit authority. The ability to perform an action against a TeamConnection object because you have been granted the authority to perform that action. Contrast with *base authority* and *implicit authority*.

extract. A TeamConnection action you can perform on a part, driver, or release. A part extraction results in copying the specified part to a client workstation. A driver extraction and release extraction result in all parts for the driver or release being copied to a designated location.

F

family. A logical organization of related data. A single TeamConnection server can support multiple families. The data in one family cannot be accessed from another family.

family administrator. A user responsible for all non-system-related tasks for one or more TeamConnection families, such as planning, configuring, and maintaining the Team-Connection environment and managing user access to those families.

family server. A workstation running the TeamConnection server software.

feature. A TeamConnection object used to formally request and record information about a functional addition or enhancement. The user who opens a feature is the feature originator.

file allocation table (FAT). The DOS/WIN-DOWS and OS/2-compatible file system that manages input, output, and storage of files on your system. File names can be up to eight characters long, followed by a file extension that can be up to three characters.

fix record. A status record associated with a work area and that is used to monitor the phases of change within each component affected by a defect or feature for a specific release.

freeze. The freeze command saves changed parts to the work area. Thus, TeamConnection takes a snapshot of the work area, including all of the current versions of parts visible from that work area, and saves this image of the system. The user can always come back to this stage of development in the work area. However, a freeze action does not make the changes visible to other users working in the release.

full part tree. A directory structure representing a complete set of active parts associated with the release.

G

graphical user interface (GUI). A type of computer interface consisting of a visual metaphor of a real-world scene, often referred as a desktop. Within that scene are icons, representing actual objects, that the user can access and manipulate with a pointing device.

H

high-performance file system (HPFS). In the OS/2 operating system, an installable file system that uses high-speed buffer storage, known as a cache, to provide fast access to large disk volumes. The file system also supports the existence of multiple, active file systems on a single personal computer, with the capacity of multiple and different storage devices. File names used with HPFS can have as many as 254 characters.

host. A host node, host computer, or host system.

host list. A list associated with each Team-Connection user ID indicating the client machine that can access the family server and act on behalf of the user. The family server uses the list to authenticate the identity of a client machine when the family server receives a command. Each entry consists of a login, a host name, and a TeamConnection user ID.

host name. The identifier associated with the host computer.

I

implicit authority. The ability to perform an action on a TeamConnection object without being granted explicit authority. This authority is automatically granted through inheritance or object ownership. Contrast with base authority and explicit authority

import. To bring in data. In TeamConnection, to bring selected items into a field from a matching TeamConnection object window.

inheritance. The passing of configuration management properties from parent to child component. The configuration management properties that are inherited are access and notification. Inheritance within each Team-Connection family or component hierarchy is cumulative.

integrated problem tracking. The process of integrating problem tracking with change control to track all reported defects, proposed features, and subsequent changes to parts. See also *change control process*.

interest group. The list of actions that trigger notification to the user IDs associated with those actions listed in the notification list.

J

job queue. A queue of build scopes. One job queue exists for each TeamConnection family.

L

lock. An action that prevents editing access to a part stored in the TeamConnection development environment so that only one user can change a part at a time.

login. The name that identifies a user on a multiuser system. In OS/2 and Windows NT, the login value is obtained from the TC_USER environment variable.

M

metadata. In databases, data that describe data objects.

N

name server. In TCP/IP, a server program that supplies name-to-address translation by mapping domain names to Internet addresses.

notification list. An object that enables component owners to configure notification. A list attached to a component that pairs a list of user IDs and a list of interest groups. It designates the users and the corresponding notification interest that they are being granted for all objects managed by this component or any of its descendants.

notification server. A server that sends notification messages to the client.

Windows NT file system (NTFS). In the Windows NT operating system, an installable file system that uses high-speed buffer storage, known as a cache, to provide fast access to large disk volumes. The file system also supports the existence of multiple, active file systems on a single personal computer, with the capacity of multiple and different storage devices. File names used with NTFS can have as many as 254 characters.

O

operator. A symbol that represents an operation to be done. See also comparison operator.

originator. The user who opens a defect or feature and is responsible for verifying the outcome of the defect or feature on a verification record. This responsibility can be reassigned.

owner. The user responsible for a Team-Connection object within a TeamConnection family, either because the user created the object or was assigned ownership of the object.

P

parent component. All components in each TeamConnection family, except the root component, are created in reference to an existing component. The existing component is the parent component. See also *child component* and *component.*

parent part. Any part in a build tree that has a child defined. See also *part* and *child part.*

parser. A tool that can read a source file and report back a list of dependencies of that source file. It frees a developer from knowing the dependencies one part has on other parts to ensure that a complete build is performed.

part. A collection of data that is stored by the family server and retrieved by a path name. Parts can be text objects, binary objects, and modeled objects. These parts can be stored by the user or the tool, or they can be generated from other parts, such as when a linker generates an executable file.

path name. The name of the part under TeamConnection control. A path name can be a directory structure and a base name or just a base name. It must be unique within each release.

prerequisite work areas. If a part is changed to resolve more than one defect or feature, the work area referenced by the first change is a prerequisite of the work area referenced by later changes. A work area is a prerequisite to another work area if:

Part changes are checked in, but not committed, for the first work area.

One or more of the same parts are checked out, changed, and checked in again for the second work area.

problem tracking. The process of tracking all reported defects through to resolution and all proposed features through to implementation.

process. A combination of TeamConnection subprocesses, configured by the family adminis-trator, that controls the general movement of TeamConnection objects (defects, features, work areas, and drivers) from state to state within a component or release. See also subprocess and state.

Q

query. A request for information from a database, for example, a search for all defects that are in the open state. See also *default query* and *search.*

R

raw format. Information retrieved on the Report command that has the vertical bar delimiter separating field information, and each line of output corresponds to one database record.

refresh. This TeamConnection command updates a work area with any changes from the release. It also freezes the work area, if it is not already frozen.

relative path name. The name of a directory or a part expressed as a sequence of directories followed by a part name, beginning from the current directory.

release. A TeamConnection object defined by a user that contains all parts to be built, tested, and distributed as a single entity.

restricted authority. The limitation on a user's ability to perform certain actions on a specific component. Authority can be restricted by the superuser, the component owner, or a user with AccessRestrict authority. See also authority.

root component. The initial component created when a TeamConnection family is configured. All components in a TeamConnection family are descendants of the root component. Only the root component has no parent component. See also *component, child component,* and *parent component*

S

search. To scan one or more data elements of a set in a database to find elements that have certain properties.

serial development. While a user has parts checked out from a work area, no one else on the team can check out the part. The user develops new material without interacting with the other developers on the project. TeamConnection provides the opportunity to hold the part until the user is sure that it integrates with the rest of the application. Thus, the lock is not released until the work area as a whole is committed. Contrast with *concurrent development*. See also *work area.*

server. A workstation that performs a service for another workstation.

shell script. A series of commands combined in a file that carry out a function when the file is run.

sizing record. A status record created for each component-release pair affected by a proposed defect or feature. The sizing record owner must indicate whether the defect or feature affects the specified component-release pair and the approxi mate amount of work needed to resolve the defect or implement the feature within the specified component-release pair.

stanza format. Data output generated by the Report command in which each database record is a stanza. Each stanza line consists of a field and its corresponding values.

state. Work areas, drivers, features, and defects move through various states during their life cycles. The state of an object determines the actions that can be performed on it. See also *process* and *subprocess*.

subprocess. TeamConnection subprocesses govern the state changes for TeamConnection objects. The design, size, review (DSR) and verify subprocesses are configured for com-ponent processes. The track, approve, fix, driver, and test subprocesses are configured for release processes. See also *process* and *state*.

superuser. This privilege lets a user perform any action available in the TeamConnection family.

system administrator. A user who is responsible for all system-related tasks involving the TeamConnection server, such as installing, maintaining, and backing up the TeamConnection server and the database it uses.

T

task list. The list of tasks displayed in the Tasks window. The user can customize this list to issue requests for information from the server. Tasks can be added, modified, or deleted from the lists.

TeamConnection client. A workstation that connects to the TeamConnection server by a TCP/IP connection and that is running the TeamConnection client software.

TeamConnection part. A part that is stored by the TeamConnection server and retrieved by a path name, release, type, and work area. See also *part, common part,* and *type*.

TeamConnection superuser. See *superuser.*

tester. A user responsible for testing the resolution of a defect or the implementation of a feature for a specific driver of a release and recording the results on a test record.

test record. A status record used to record the outcome of an environment test performed for a resolved defect or an implemented feature in a specific driver of a release.

track subprocess. An attribute of a Team-Connection release process that specifies that the change control process for that release will be integrated with the problem tracking process.

Transmission Control Protocol/Internet Protocol (TCP/IP). A set of communication protocols that support peer-to-peer connectivity functions for both local and wide area networks.

type. All parts that are created through the TeamConnection GUI or on the command line will show up in reports with the type of file as the part type. The TeamConnection GUI and command line can only check in, check out, and extract parts of the type file. Note: Parts created through an API can have other specified types.

U

user exit. A user exit allows TeamConnection to call a user-defined program during the processing of TeamConnection transactions. User exits provide a means by which users can specify additional actions that should be performed before completing or proceeding with a TeamConnection action.

user ID. The identifier assigned by the system administrator to each TeamConnection user.

V

verification record. A status record that the originator of a defect or a feature must mark before the defect or feature can move to the closed state. Originators use verification records to verify the resolution or implementation of the defect or feature they opened.

version. (1) A specific view of a driver, release, or work area. (2) A revision of a part.

version control. The storage of multiple versions of a single part and information about each version.

view. An alternative and temporary representation of data from one or more tables.

W

work area. An object in TeamConnection that you create and associate with a release. When the work area is created, you see the most current view of the release and all the parts that it contains. You can check out the parts in the work area, make modifications, and check them back into the work area. You can also test the modifications without integrating them. Other users are not aware of the changes that you make in the work area until you integrate the work area to the release. While you work on files in a work area, you do not see subsequent part changes in the release until you integrate or refresh your work area.

working part. The checked-out version of a TeamConnection part.

List of Abbreviations

3GL	Third Level Graphical Language
4GL	Fourth Level Graphical Language
APA	All Points Addressable
API	Application Programming interface
CMSC	Configuration Management and Software Control
CMVC	International Technical Support Organization
DLL	dynamic link library
ECU	European Currency Unit
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IBM	International Business Machines Corporation
IT	Information Technology
ITSO	International Technical Support Organization
JDK	Java Developer's Kit
NLS	National Language Support
OS	Operating System
SCMS	Software Configuration Management Services
SEI	Software Engineering Institute
TUI	Text User Interface
UDB	Universal Database
UDF	DB2 user-defined functions
Y2K	Year 2000

Index

ITSO Redbook Evaluation

VisualAge TeamConnection Enterprise Server From CMVC to TeamConnection Version 3
SG24-2226-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and return it using one of the following methods:**

- Use the online evaluation form found at http://www.redbooks.ibm.com
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
_ Customer   _ Business Partner   _ Solution Developer   _ IBM employee
_ None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction __________

Please answer the following questions:

Was this redbook published in time for your needs? Yes___ No___

If no, please explain:

What other redbooks would you like to see published?

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

VisualAge TeamConnection Enterprise Server From CMVC to TeamConnection Version 3 **SG24-2226-00**

SG24-2226-00
Printed in the U.S.A.