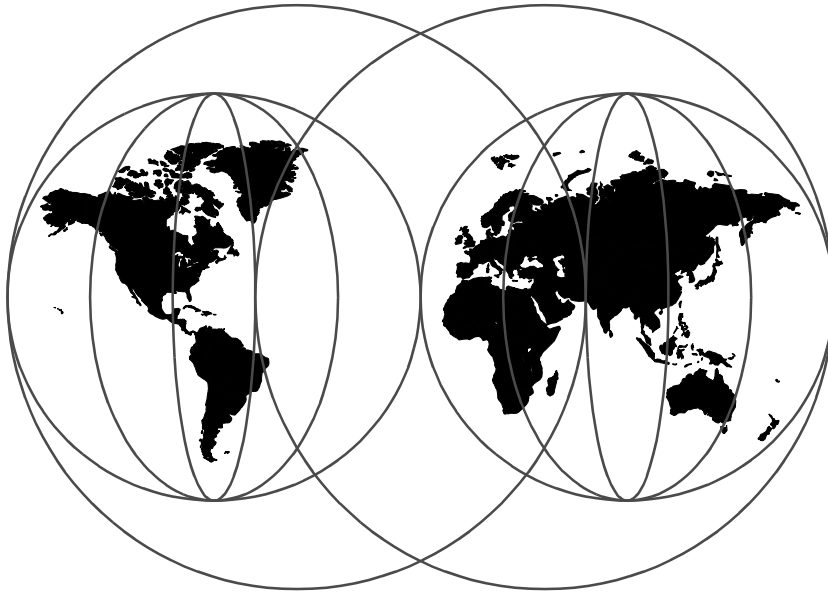


# **VisualAge TeamConnection Enterprise Server Keeping Your Build Process on a Short Leash**

*Jean Gedeon, Glen Johnson, Angel Rivera*



**International Technical Support Organization**

<http://www.redbooks.ibm.com>

SG24-5260-00





International Technical Support Organization

**VisualAge TeamConnection Enterprise Server  
Keeping Your Build Process on a Short Leash**

December 1998

**Take Note!**

Before using this information and the product it supports, be sure to read the general information in Appendix E, "Special Notices" on page 233.

**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.

**© Copyright International Business Machines Corporation 1998. All rights reserved**

Note to U.S Government Users – Documentation related to restricted rights – Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>Figures</b> .....	.xi
<b>Tables</b> .....	.xiii
<b>Preface</b> .....	xv
The Team That Wrote This Redbook .....	xv
Comments Welcome .....	xvi
<hr/>	
<b>Part 1. Products Presentation and Installation</b> .....	1
<b>Chapter 1. Today's Application Development Challenges</b> .....	3
1.1 What the Development Challenges Are .....	3
1.2 How TeamConnection Can Help .....	3
1.3 What TeamConnection Does .....	4
<b>Chapter 2. Meeting VisualAge TeamConnection</b> .....	7
2.1 Managing Software Change: The Power of TeamConnection .....	7
2.1.1 Objectives .....	8
2.2 Types of Problems Encountered .....	10
2.3 Place Your Application Under Control .....	12
2.4 Studying the Anatomy of Your Application .....	12
2.5 Changing an Application in Your Development Environment .....	15
2.6 Insert TeamConnection in Your Development Environment .....	18
2.7 Change Your Application With TeamConnection .....	22
2.7.1 Impact Analysis .....	23
2.7.2 Bug Fixing .....	24
2.7.3 Fix Propagation .....	25
2.8 Control Your Change with TeamConnection: An Example .....	28
2.8.1 Impact Analysis .....	29
2.8.2 Cross-Impact Analysis .....	32
2.9 Bug Hot Fix .....	33
2.10 Fix Propagation .....	36
<b>Chapter 3. Preparing to Install TeamConnection</b> .....	39
3.1 Migrating Your Database .....	39
3.2 Hardware and Software Requirements .....	39
3.3 DB2 Universal Database .....	40
3.4 Requirements for TeamConnection for AIX .....	40
3.5 Requirements for TeamConnection for HP-UX .....	43
3.6 Requirements for TeamConnection for Solaris .....	45
3.7 Requirements for TeamConnection for OS/2 .....	47

3.8	Requirements for TeamConnection for Windows	50
3.9	Requirements for MVS Build Servers	52
<b>Chapter 4. Inserting VisualAge TeamConnection in Your World</b> . . . . . 53		
4.1	Updating TCP/IP Files	53
4.2	Host and Service Files	54
4.2.1	General Considerations	54
4.2.2	Service File Update	54
4.2.3	Host File Update	54
4.3	Verifications and Controls	55
4.3.1	Controlling the Hostname	55
4.3.2	Controlling the IP Address	56
4.3.3	Connection to the Family	56
4.3.4	Connecting to the Build Server	57

---

**Part 2. UDB Mysteries Revealed** . . . . . 59

<b>Chapter 5. Installing Universal Database: General Considerations</b> . . 61		
5.1	Overview	61
5.2	Important Information about Installation	62
5.3	DB2 UDB V5 Packaging with VisualAge TeamConnection V3	63
5.4	Applying Fixes for DB2 Universal Database	63
5.5	Miscellaneous Installation Details	64
<b>Chapter 6. DB2 UDB under UNIX: General Considerations</b> . . . . . 65		
6.1	Assumptions	65
6.2	DB2 Instances and VisualAge TeamConnection Relations	66
6.3	Performance Tuning for DB2	68
<b>Chapter 7. Installation of DB2 for UNIX</b> . . . . . 71		
7.1	Overview	71
7.2	Installation Methods	72
7.2.1	Invoking tcinst.ksh	72
7.2.2	Other Methods to Install on a UNIX Platform	72
7.3	Planning and Prerequisites	72
7.3.1	The Full Path Where DB2 Is Installed	72
7.3.2	Space Used While Installing DB2	73
7.4	Uninstalling ObjectStore	73
7.5	Installation Steps	74
7.5.1	Identify and Record the Parameter Values	74
7.5.2	Create a File System for the DB2 Instance and TC Family	76
7.5.3	Update Kernel Configuration Parameters (Not Required for AIX)	77
7.5.4	Mount the CD-ROM	81

7.5.5	Using VisualAge TeamConnection Install . . . . .	84
7.5.6	Install DB2 UDB Workgroup Edition Using DB2 Installer. . . . .	84
7.5.7	Installing the Code for DB2 UDB Workgroup Edition. . . . .	84
7.5.8	Creating the DB2 Instances . . . . .	95
7.6	post-installation Tasks . . . . .	104
7.6.1	Apply the Fixpak and Interim Fixes for DB2 UDB . . . . .	104
7.6.2	Create a VisualAge TeamConnection Family . . . . .	104
7.6.3	Changing Ownership and File Permissions of Home Directories	104
7.6.4	Update the Profile for the DB2 Instances . . . . .	105
7.6.5	Verification of DB2 Configuration . . . . .	105
7.6.6	Verify the License Information in the Nodelock File . . . . .	106
<b>Chapter 8. Installation of DB2 for Windows . . . . .</b>		<b>109</b>
8.1	Overview . . . . .	109
8.2	Planning and Prerequisites. . . . .	109
8.3	The Full Path Where DB2 Is Installed. . . . .	110
8.4	Space Used During the Installation of DB2. . . . .	110
8.5	Uninstalling ObjectStore. . . . .	110
8.6	Installation Steps . . . . .	111
8.6.1	Identify and Record Parameter Values . . . . .	111
8.6.2	Create a DB2 Administration ID in Windows NT . . . . .	112
8.6.3	Install DB2 UDB Personal Edition in Windows NT . . . . .	112
8.6.4	Apply the Fixpak and Interim Fixes for DB2 UDB . . . . .	114
<b>Chapter 9. Installation of DB2 for OS/2 . . . . .</b>		<b>115</b>
9.1	Overview . . . . .	115
9.2	Planning and Prerequisites. . . . .	115
9.2.1	The Full Path Where DB2 Is Installed . . . . .	116
9.2.2	Space Used for Installation of DB2 . . . . .	116
9.3	Uninstalling ObjectStore . . . . .	116
9.4	Installation Steps . . . . .	116
9.4.1	Identify and Record Parameter Values . . . . .	117
9.5	Create a DB2 Administration ID in OS/2. . . . .	118
9.6	Setup of UPM. . . . .	118
9.7	Install DB2 UDB Personal Edition in OS/2 . . . . .	119
9.8	Apply the Fixpak and Interim Fixes for DB2 UDB . . . . .	120
<b>Chapter 10. Creation of a Family Using the New DB2 Instance . . . . .</b>		<b>121</b>
10.1	Overview of this Chapter . . . . .	121
10.2	Using TCADMIN to Create the Family Database . . . . .	122
10.3	Creation of the User Id for the Family . . . . .	123
10.4	Customization of Local Files (Profile, Teamcv3x.Ini and Teamcgui)	124
10.5	Using Default settings to Create the Family Database . . . . .	125
10.6	Db2 Configuration Parameters when Creating a Family . . . . .	127

10.7 What To Do Next . . . . .	128
<b>Chapter 11. Control Center Configuration . . . . .</b>	<b>129</b>
11.1 Overview of the Chapter. . . . .	129
11.2 Relationship between Control Center and Administration Server . .	129
11.3 Installing the DB2 GUI Tools . . . . .	131
11.3.1 Installing the Client Application Enabler in Windows NT . . . . .	131
11.3.2 Installing the Client Application Enabler in OS/2 . . . . .	132
11.4 Configuring the Db2 Control Center . . . . .	134
11.4.1 Setup of the UNIX Server . . . . .	134
11.4.2 Setup of the Intel workstation (adding a remote system) . . . . .	135
11.5 Adding a Database in the Control Center . . . . .	136
11.6 What to do Next . . . . .	137
<b>Chapter 12. Administration of Databases with DB2 Tools . . . . .</b>	<b>139</b>
12.1 Overview of this Chapter . . . . .	139
12.2 Control Center. . . . .	139
12.3 Additional Facilities from the Control Center. . . . .	140
12.4 Understanding System Administrative (Sysadm) Authority . . . . .	140
12.5 Where Are The Files Used with The Profile Registry? . . . . .	141
12.6 What Other System-Wide Files Are Needed by Db2? . . . . .	142
12.6.1 System-Wide Files in UNIX Needed by DB2 . . . . .	142
12.7 What Next? . . . . .	142
<b>Chapter 13. Working with DB2 Instances . . . . .</b>	<b>145</b>
13.1 Overview of this Chapter . . . . .	145
13.2 Creating Instances . . . . .	145
13.2.1 Using the DB2 Installer in UNIX . . . . .	146
13.2.2 Using db2icrt . . . . .	147
13.3 Listing Instances . . . . .	147
13.4 Enabling the Proper Profile when Logging in to the DB2 Instance .	148
13.5 Starting and Stopping a DB2 Server Instance. . . . .	148
13.5.1 Starting a DB2 Server Instance . . . . .	148
13.5.2 Stopping a DB2 Server Instance. . . . .	149
13.6 Auto-Starting Instances (UNIX) . . . . .	149
13.6.1 Enabling Auto-Start . . . . .	149
13.6.2 Disabling Auto-Start . . . . .	149
13.7 Updating Instances . . . . .	150
13.8 Removing Instances . . . . .	150
13.9 What to Do Next. . . . .	151
<b>Chapter 14. Working with the DB2 Administration Server in UNIX .</b>	<b>153</b>
14.1 Overview of this Chapter . . . . .	153
14.2 Understanding the Administration Server . . . . .	153



14.3	Creating The Administration Server	153
14.3.1	Using the DB2 Installer in UNIX	153
14.3.2	Using the DB2 Line Command dasicrt in UNIX	154
14.4	Listing The Administration Server	154
14.5	Enabling the Proper Profile when Logging in to the Server	154
14.6	Starting the Administration Server	155
14.7	Stopping the Administration Server	155
14.8	Removing the DB2 Administration Server	155

---

**Part 3. How to Build Applications** . . . . . 157

<b>Chapter 15. Basic Build Concepts</b>	159
15.1 The Physical Structure of the Build Function	159
15.2 The Build Object Model	160
15.2.1 TeamConnection Part	161
15.2.2 Build Event	161
15.2.3 Builder	161
15.2.4 Build Script	161
15.2.5 Parser	162
15.3 Parent-Child Relationships in a Build Tree	162
15.3.1 Input Parts	162
15.3.2 Output Parts	162
15.3.3 Dependent Parts	163
15.4 Generating Build Output	163
15.5 Working with a Build Tree	163
15.6 Putting the Pieces Together	167
<b>Chapter 16. A Simple, Yet General Process</b>	169
16.1 The Flow	169
16.1.1 Open State	169
16.1.2 Returned State	170
16.1.3 Canceled	170
16.1.4 Design	170
16.1.5 Size	170
16.1.6 Review	170
16.1.7 Working	170
16.1.8 Verify	171
16.1.9 Testing	171
16.1.10 Integration	171
16.1.11 Production	171
16.1.12 Closed	172
16.2 Typical Evolution of a Build	172
16.2.1 Working with Features	172

16.2.2	Opening a Feature . . . . .	172
16.2.3	Design – Size – Review Process . . . . .	174
16.2.4	Create Work Area . . . . .	175
16.2.5	Modify, Build, and Test Modifications . . . . .	175
16.2.6	Building for Integration . . . . .	183
16.3	An Emergency Hot Fix . . . . .	183
16.3.1	Working with Defects . . . . .	184
16.3.2	Opening a Defect Fix . . . . .	187
16.3.3	Creating a Work Area . . . . .	189
16.3.4	Parts Check-Out Process . . . . .	190
16.3.5	Modify, Check-in, Build, Test Part Fixes (Updates) . . . . .	190
16.3.6	Integrating the Work Area . . . . .	194
16.3.7	Resolving the Conflicts . . . . .	194
16.4	Recapitulation . . . . .	194
<b>Appendix A. VisualAge TeamConnection Technical Overview . . . . .</b>		<b>197</b>
A.1	TeamConnection Customer Value . . . . .	197
A.2	Application Development Challenges and TeamConnection . . . . .	198
A.3	TeamConnection Can Help. . . . .	198
A.4	How TeamConnection Helps . . . . .	199
A.5	Configuration Management. . . . .	200
A.6	Views . . . . .	200
A.7	Parts . . . . .	200
A.8	Component Hierarchy . . . . .	201
A.9	Role-Based Access. . . . .	202
A.10	Release Management. . . . .	203
A.11	Version Control. . . . .	203
A.12	Integrated Build. . . . .	205
A.13	Build Mechanisms. . . . .	206
A.14	Distributed Build . . . . .	208
A.15	Packaging and Distribution Support . . . . .	209
A.16	TeamConnection's Packaging Utilities . . . . .	209
A.17	Electronic Distribution Support Using Your Distribution Tool . . . . .	210
A.18	Integrated Problem Tracking and Change Control . . . . .	210
A.19	Lotus Notes. . . . .	212
A.20	Report Facility . . . . .	212
A.21	Backup and Recovery. . . . .	213
A.22	Repository and Model Support . . . . .	213
A.23	Application Development Tool Integration . . . . .	213
A.23.1	IBM VisualAge for C++ . . . . .	214
A.23.2	IBM VisualAge for COBOL . . . . .	215
A.23.3	IBM VisualAge for Java . . . . .	215
A.23.4	IBM VisualAge Generator . . . . .	216

A.23.5 IBM VisualAge for Smalltalk . . . . .	217
A.23.6 IBM VisualAge PL/I . . . . .	218
A.23.7 Non-IBM Tools . . . . .	218
A.24 For More Information . . . . .	219
<b>Appendix B. Naming Restrictions for DB2 User IDs . . . . .</b>	<b>221</b>
B.1 Group IDs, Instances, and Databases . . . . .	221
<b>Appendix C. Uninstalling DB2 UDB . . . . .</b>	<b>223</b>
C.1 Uninstalling DB2 from UNIX . . . . .	223
C.1.1 Stop and Back up the VisualAge TeamConnection Families . . . . .	223
C.1.2 Stop the Administration Server . . . . .	223
C.1.3 Stop All DB2 Instances . . . . .	223
C.1.4 Remove the Administration Server . . . . .	224
C.1.5 Remove the DB2 Instances . . . . .	224
C.1.6 Clean up the DB2 registry . . . . .	224
C.1.7 Remove the DB2 Products . . . . .	225
C.2 Uninstalling DB2 from Windows NT . . . . .	226
C.3 Uninstalling DB2 from OS/2 . . . . .	227
<b>Appendix D. TeamConnection Authority Groups . . . . .</b>	<b>229</b>
D.1 Creating or modifying authority groups . . . . .	229
D.2 Using the Graphical User Interface . . . . .	229
D.3 Manual Modifications: . . . . .	230
D.3.1 Editing the <code>authorit.id</code> file . . . . .	231
D.4 Reloading the authority table . . . . .	231
<b>Appendix E. Special Notices . . . . .</b>	<b>233</b>
<b>Appendix F. Related Publications . . . . .</b>	<b>237</b>
F.1 International Technical Support Organization Publications . . . . .	237
F.2 Redbooks on CD-ROMs . . . . .	237
F.3 Other Publications . . . . .	237
<b>How to Get ITSO Redbooks . . . . .</b>	<b>239</b>
How IBM Employees Can Get ITSO Redbooks . . . . .	239
How Customers Can Get ITSO Redbooks . . . . .	240
IBM Redbook Order Form . . . . .	241

<b>Glossary</b> .....	243
<b>List of Abbreviations</b> .....	251
<b>Index</b> .....	253
<b>ITSO Redbook Evaluation</b> .....	261

---

## Figures

1. What Goes Wrong when Coherence Is Lost . . . . .	8
2. Change Management versus Configuration Management . . . . .	9
3. Version Control Is A Fact. . . . .	11
4. Anatomy of a Legacy Application. . . . .	13
5. The Transparency of TeamConnection . . . . .	17
6. Inserting TeamConnection into Your Development Environment . . . . .	21
7. Project Status Before It Gets Crazy . . . . .	22
8. Project Status at the End of the Hot Fix Process. . . . .	25
9. The Merge Nightmare . . . . .	26
10. Getting Out of the Nightmare . . . . .	28
11. Project Status before Change Management Implementation. . . . .	29
12. Local Impact Analysis TeamConnection Facilities. . . . .	31
13. Cross-Impact Analysis TeamConnection Facilities . . . . .	33
14. Hot Fix with TeamConnection . . . . .	35
15. How to Propagate a Fix With TeamConnection. . . . .	38
16. DB2 Installer Screen . . . . .	86
17. Install DB2 V5 Screen . . . . .	87
18. DB2 Universal Database Workgroup Edition Screen. . . . .	88
19. DB2 Product Messages screen . . . . .	89
20. DB2 Product Library Screen . . . . .	90
21. Summary Report screen for the installation of the code . . . . .	91
22. Warning screen before unpacking the installation images. . . . .	92
23. Installation progress screen. . . . .	92
24. Notice of Completion of the Installation . . . . .	92
25. Status Report Screen. . . . .	93
26. View Log File Screen. . . . .	94
27. DB2 Installer Screen - Create Instances . . . . .	95
28. Create DB2 Services Screen, Snapshot 1. . . . .	96
29. DB2 Instance screen . . . . .	97
30. DB2 Instance Protocol Screen. . . . .	98
31. Customization of TCP/IP Protocol Screen. . . . .	98
32. User-Defined Functions Screen. . . . .	99
33. Create DB2 Services Screen, Snapshot 2. . . . .	100
34. Administration Server Screen . . . . .	101
35. Notice Screen about DB2SYSTEM . . . . .	101
36. Summary Report Screen for the Configuration Tasks . . . . .	102
37. Warning Screen before Configuring the DB2 Instance . . . . .	102
38. Configuration Progress Screen . . . . .	103
39. Notice Screen –Completion of the Configuration of Instances . . . . .	103
40. Relationship between Control Center and Administration Server . . . . .	130

41. The Physical Structure of TeamConnection . . . . .	160
42. Sample Build Object Model for msgcat.exe . . . . .	164
43. The Build Tree for the hello Application . . . . .	165
44. Two Versions of a Build Tree . . . . .	166
45. Dialog Screen for Open Feature . . . . .	173
46. View Feature Information Field . . . . .	173
47. Screen Showing Design, Size, and Review Tasks . . . . .	174
48. Create Work Areas Screen . . . . .	175
49. Refresh Work Area Dialog Fields . . . . .	176
50. Check Out Parts Dialog Fields . . . . .	176
51. Notepad Showing the Changes . . . . .	177
52. Check-in Parts Window with Dialog Fields . . . . .	178
53. Build Parts Dialog Window . . . . .	179
54. TeamConnection BuildView Screen . . . . .	180
55. Parts View Screen . . . . .	181
56. Result of Using the Web Browser . . . . .	183
57. Create Parts Dialog Window . . . . .	185
58. The Dialog Window that Creates the New Clock2.class Parts . . . . .	186
59. The Connect Parts Window, How Clock2.java Relates to Clock2.class . . . . .	187
60. Open Defect Window Showing Fields with Entries . . . . .	188
61. Accept Defects Window . . . . .	189
62. Create Work Areas Window . . . . .	190
63. Check Out Parts Window . . . . .	190
64. Check-in Parts Window . . . . .	191
65. Create Builder Window with Relevant Fields Filled in . . . . .	192
66. Build Parts Window with Relevant Fields Filled . . . . .	193

---

## Tables

1. Merge Cases for Four Versions . . . . .	26
2. Hardware Requirements for an AIX TeamConnection Server . . . . .	40
3. Hardware Requirements for an AIX TeamConnection Client . . . . .	41
4. Software Requirements for TeamConnection for AIX . . . . .	42
5. Hardware Requirements for an HP-UX TeamConnection Serve. . . . .	43
6. Hardware Requirements for an HP-UX TeamConnection Client . . . . .	44
7. Software Requirements for TeamConnection for HP-UX . . . . .	44
8. Hardware Requirements for a Solaris TeamConnection Server . . . . .	45
9. Hardware Requirements for a Solaris TeamConnection Client . . . . .	46
10. Software Requirements for TeamConnection for Solaris. . . . .	47
11. Hardware Requirements for an OS/2 TeamConnection Server. . . . .	47
12. Hardware Requirements for an OS/2 TeamConnection Client . . . . .	48
13. Software Requirements for TeamConnection for OS/2 . . . . .	49
14. Hardware Requirements for a Windows TeamConnection Server . . . . .	50
15. Hardware Requirements for a Windows TeamConnection Client . . . . .	51
16. Software Requirements for TeamConnection for Windows. . . . .	51
17. Relationship of TeamConnection Families to DB2 Instances . . . . .	66
18. Relationship between TeamConnection Codes and DB2 UDB Codes . . . . .	66
19. Parameters for the DB2 Instance. . . . .	74
20. .Parameters for the Normal DB2 Fenced User ID . . . . .	75
21. Parameters for the DB2 Administration Server . . . . .	75
22. HP-UX Kernel Configuration Parameters (Recommended Values). . . . .	77
23. Solaris Kernel Configuration Parameters (Recommended Values). . . . .	80
24. Parameters for the DB2 Instance. . . . .	111
25. .Parameters for the DB2 Administration Server. . . . .	111
26. Parameters for the DB2 Instance. . . . .	117
27. Parameters for the DB2 Administration Server . . . . .	117
28. Parameters for the sample VisualAge TeamConnection family. . . . .	122
29. TC_HOME Location. . . . .	124





---

## Preface

This redbook will help you understand the build process with VisualAge TeamConnection. Within the book we will introduce you to real examples, the objective being to provide first hand experience and guidance to help you in the design of your own solutions.

Part 1 contains a general introduction to VisualAge TeamConnection and its concepts. It will be especially useful for the newcomers to TeamConnection. Chapter 3 puts emphasis on network needs and gives 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 the build process of 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.

**Glen Johnson** is a Staff Software Engineer with IBM Global Services in Austin TX. He is responsible for development of WEB enabled database tools for work flow automation and business process improvement.

Prior to joining IBM in 1992, he was CFO of Nova Concepts, Inc. (an IBM Business Partner) and Head of Immunochemistry Research at Kallestad

Laboratories. He received a Ph.D. in Biochemistry from the State University of New York at Buffalo.

**Angel 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 Tecnológico 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 261 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](mailto:redbook@us.ibm.com)



---

## Part 1. Products Presentation and Installation

## **2** TeamConnection V3, Keeping Your Build Process on a Short Leash

---

## 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.

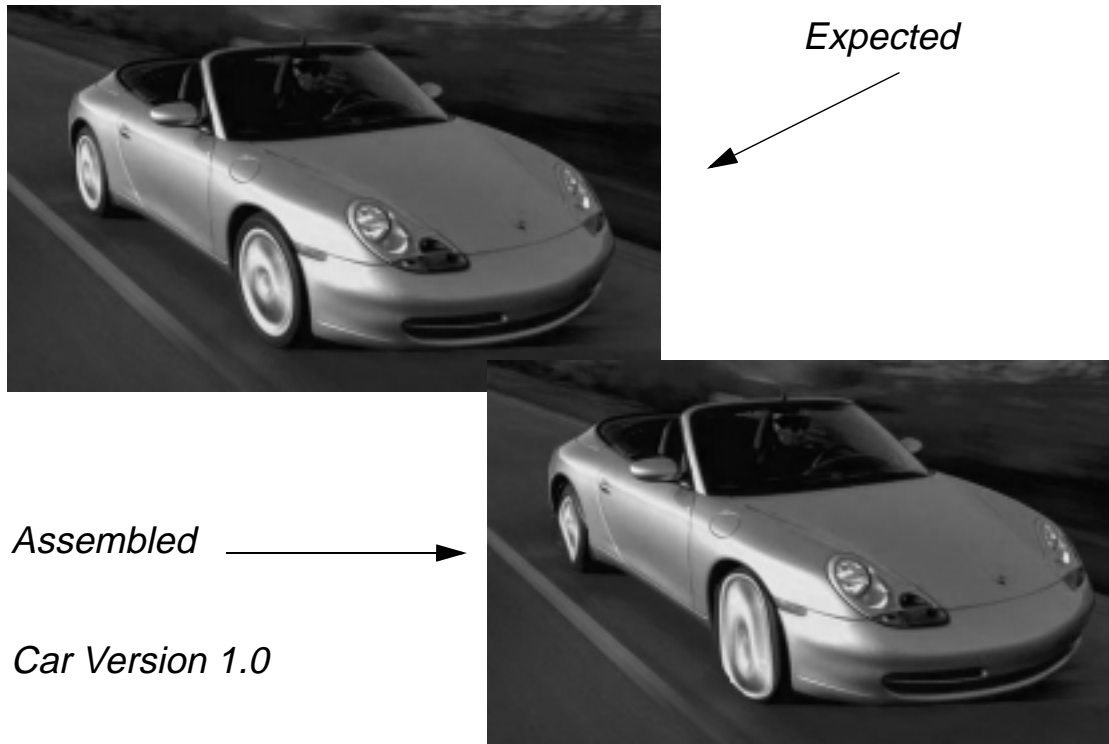


Figure 1. What Goes Wrong when Coherence Is Lost

### 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.

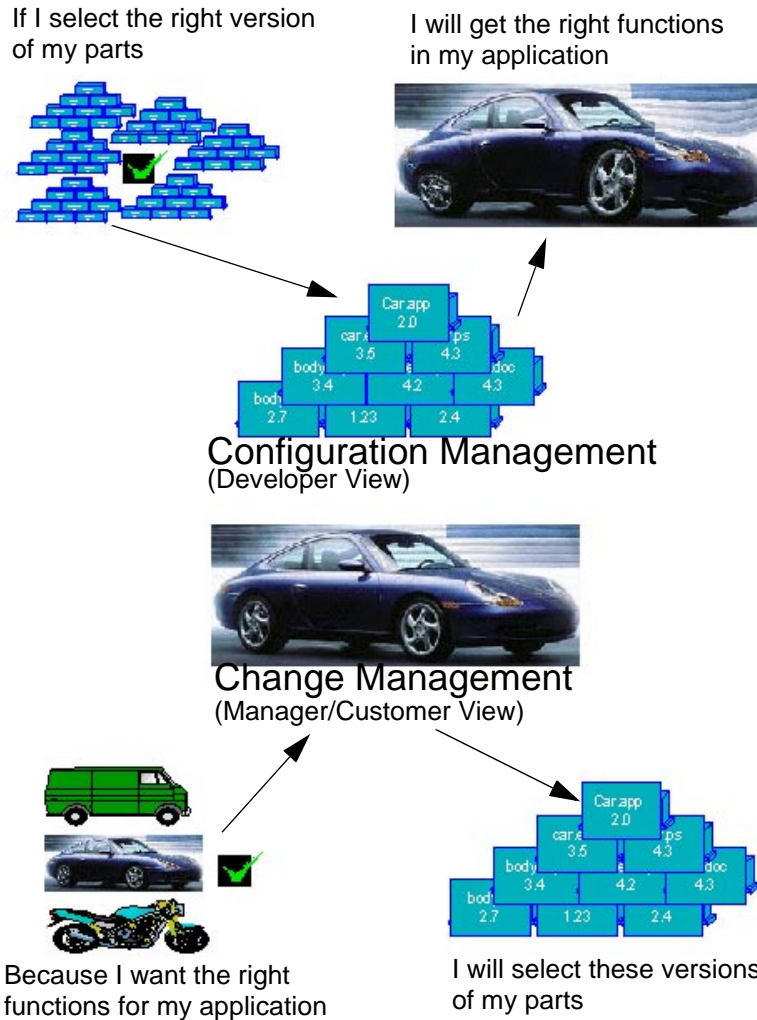


Figure 2. Change Management versus Configuration Management

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.

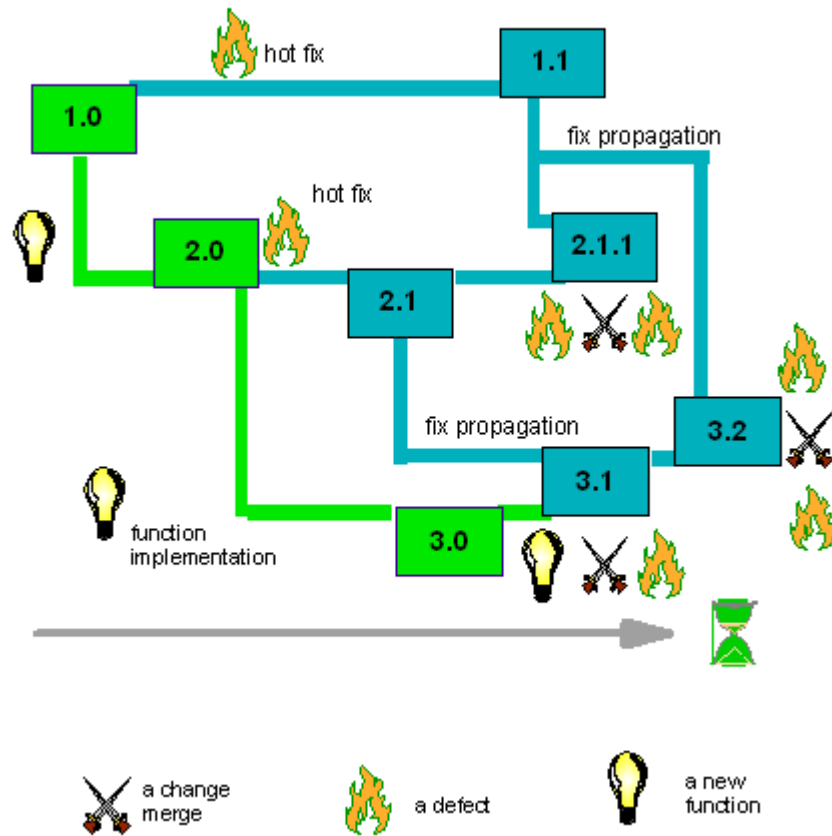


Figure 3. Version Control Is A Fact

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.



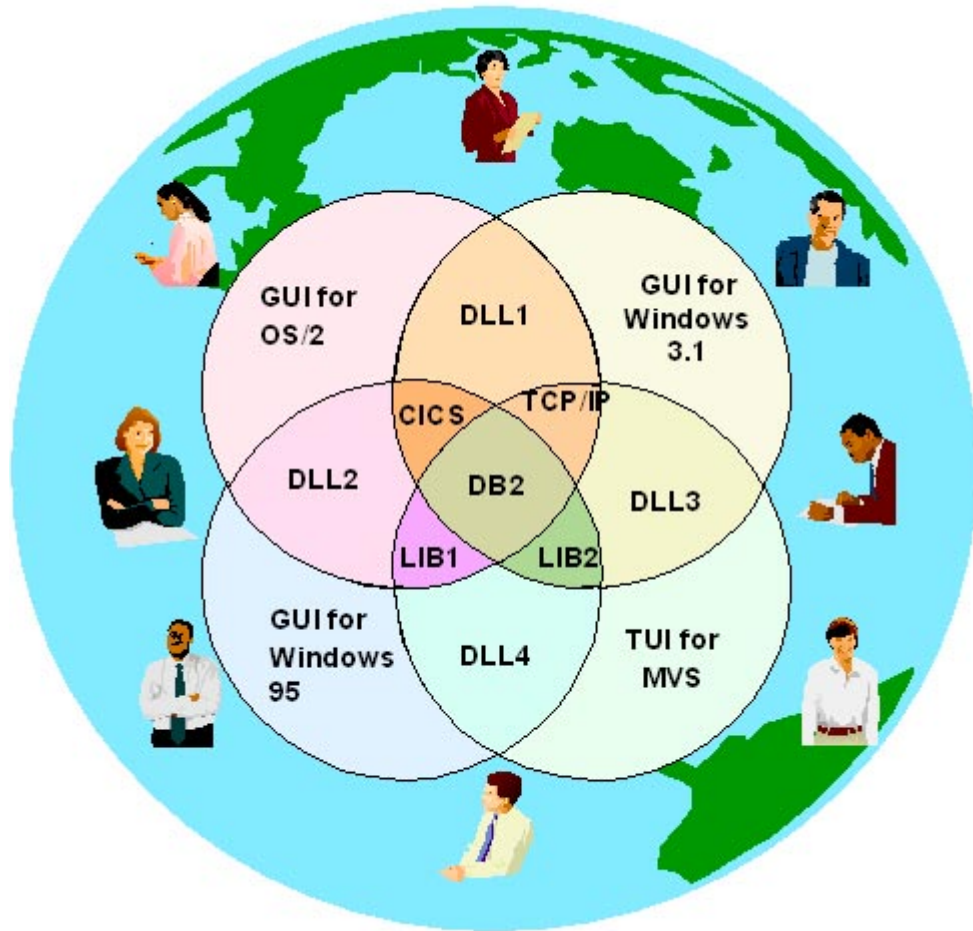


Figure 4. Anatomy of a Legacy Application

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.

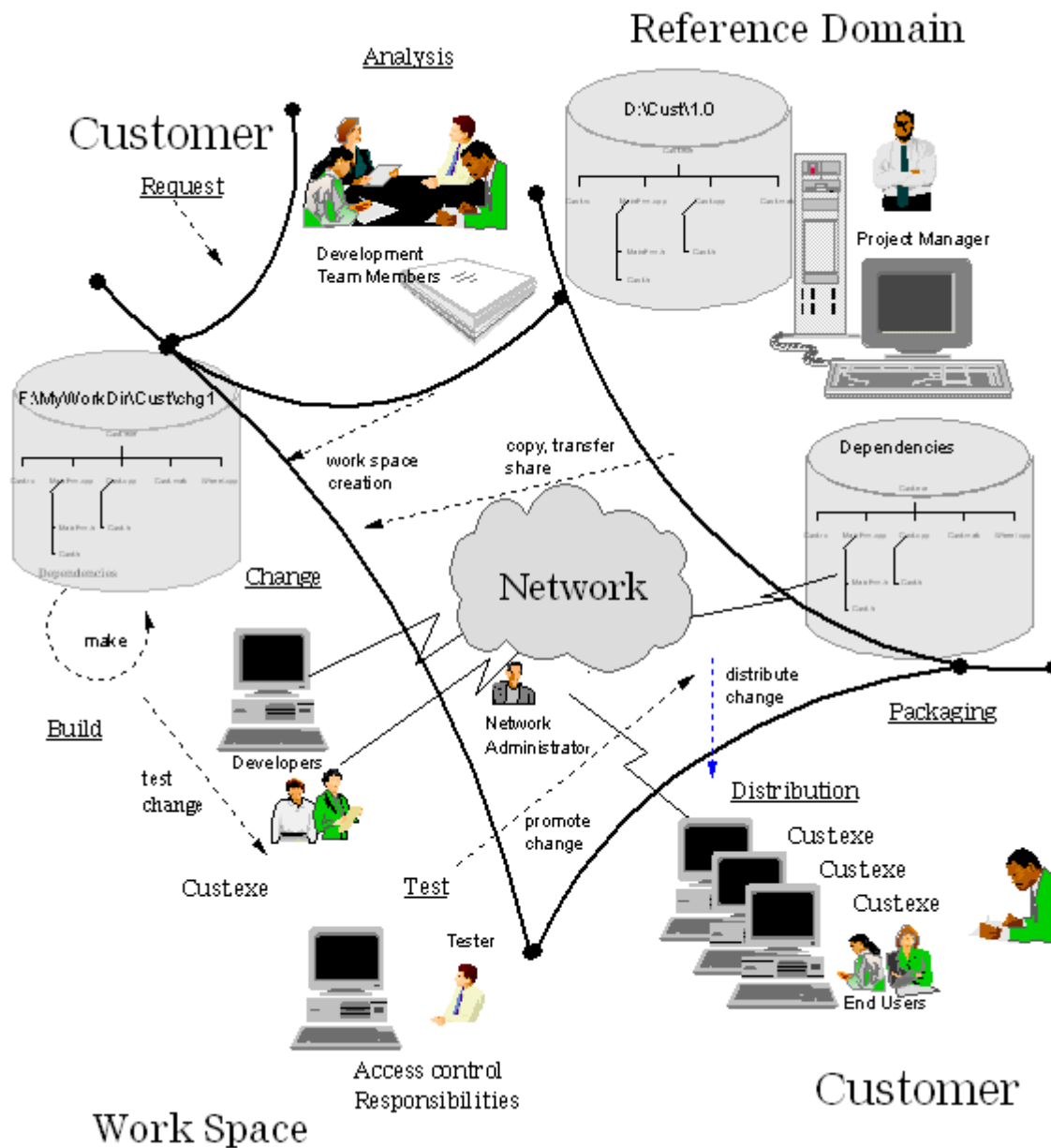


Figure 5. The Transparency of TeamConnection

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 *builders*. 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.



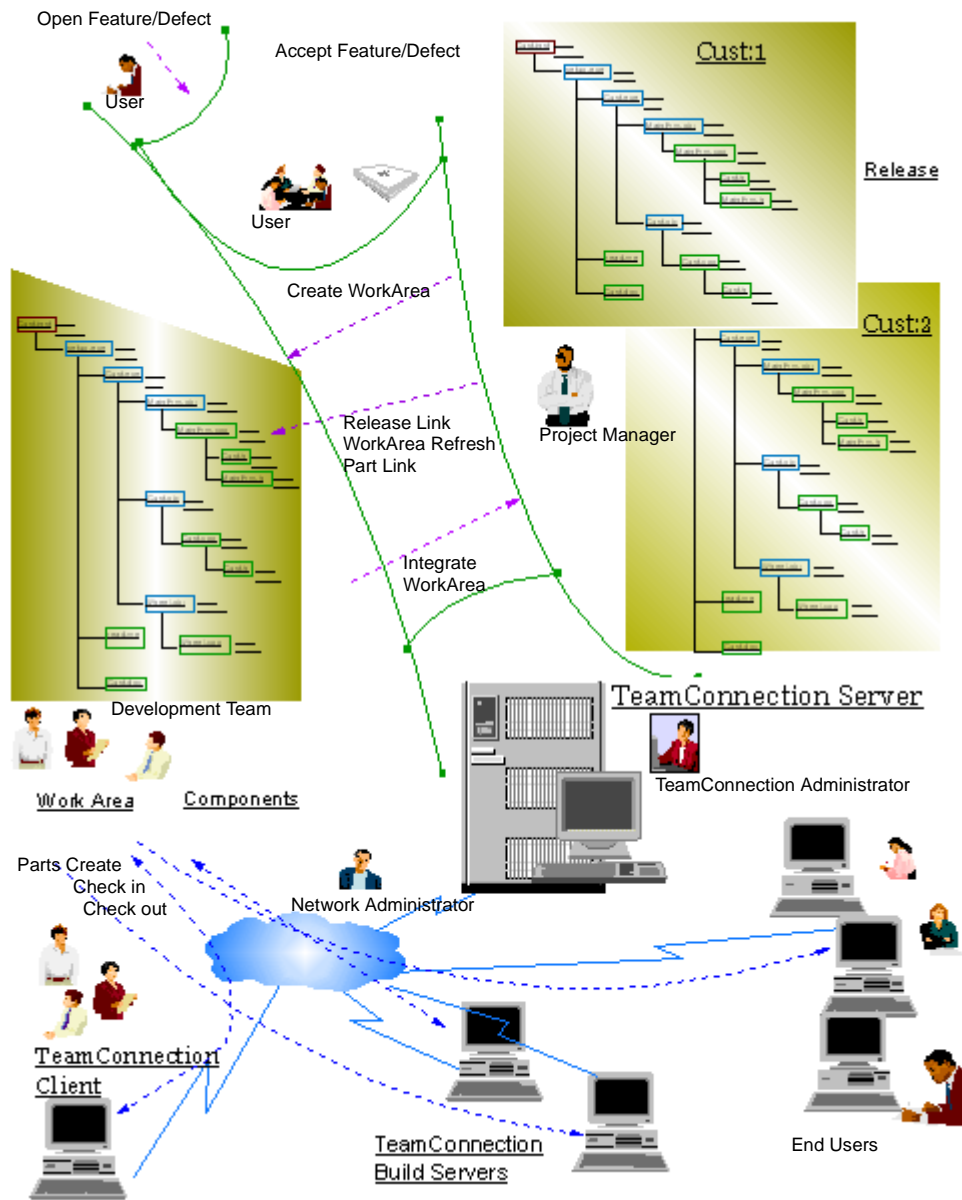


Figure 6. Inserting TeamConnection into Your Development Environment

## 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.

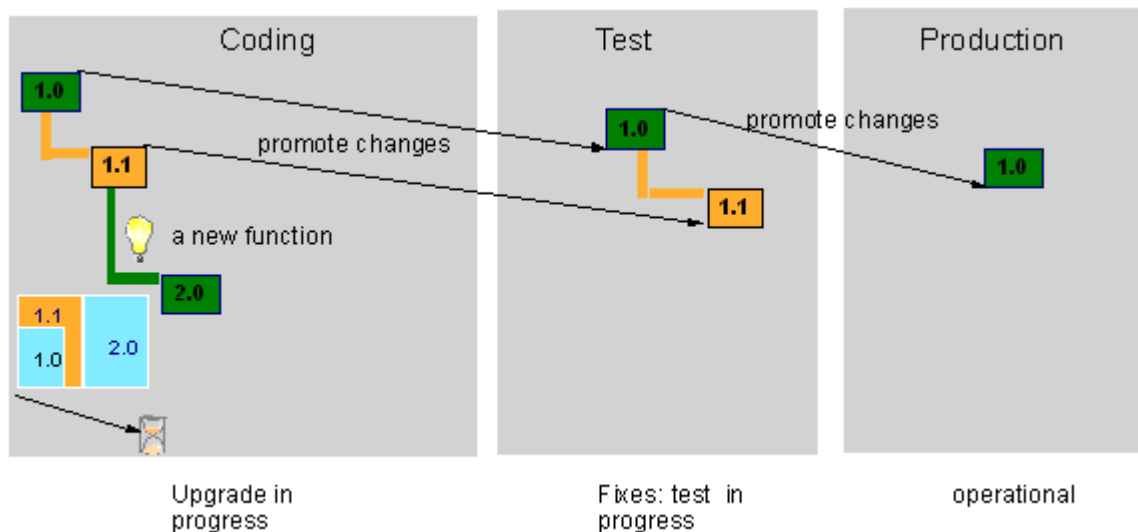


Figure 7. Project Status Before It Gets Crazy

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 `ls` {} \; ).
```

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.

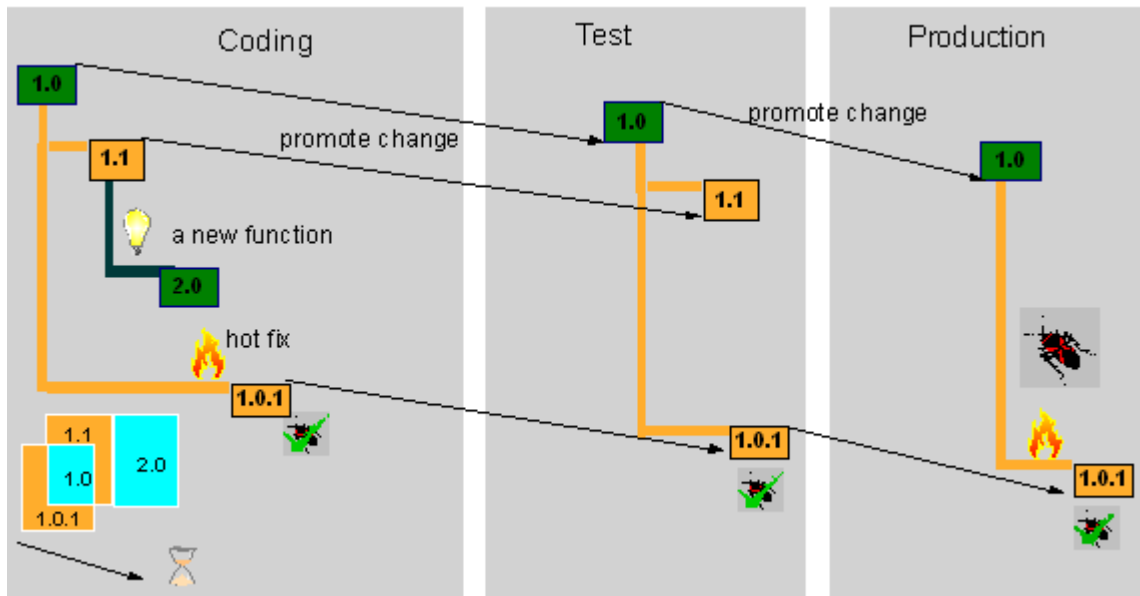


Figure 8. Project Status at the End of the Hot Fix Process

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.

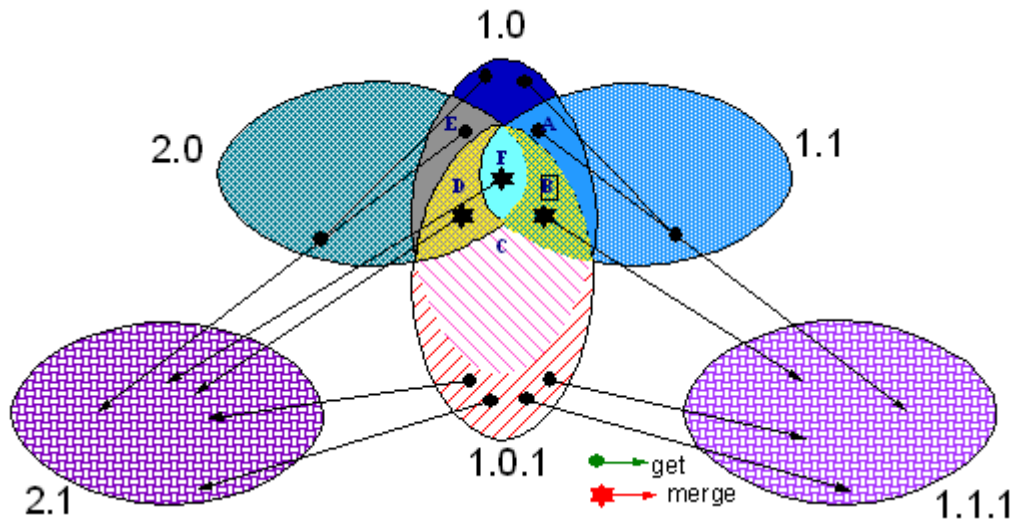


Figure 9. The Merge Nightmare

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.

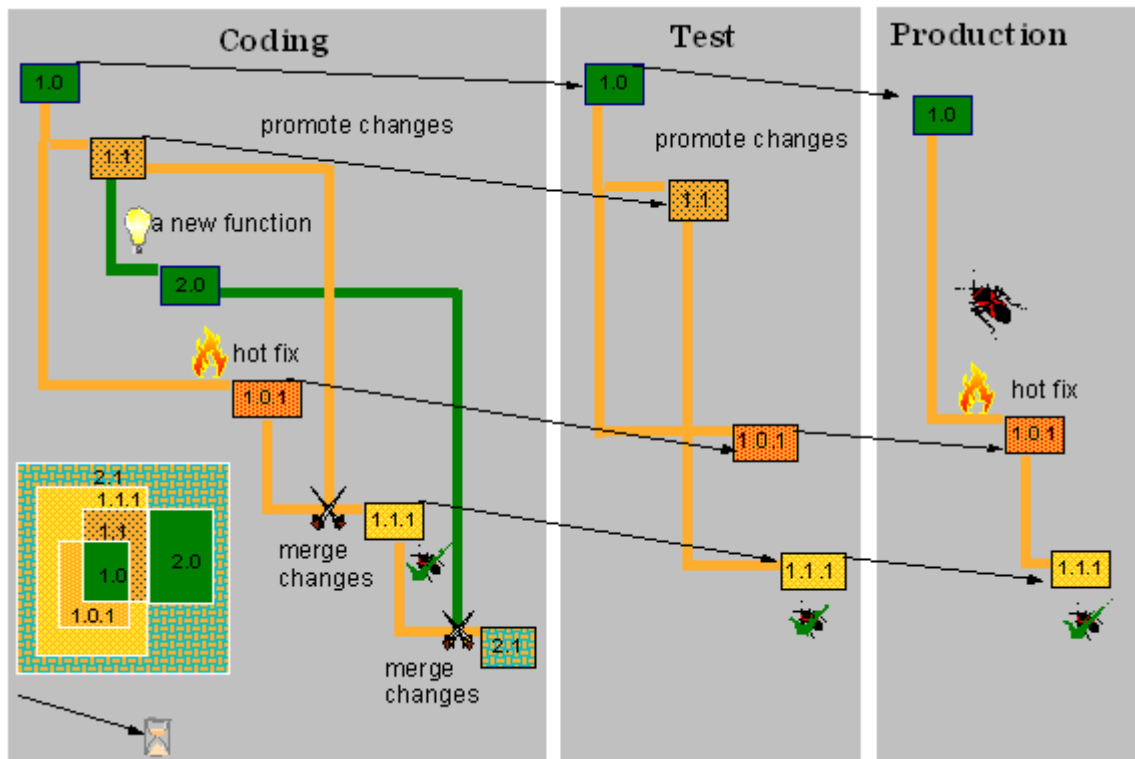


Figure 10. Getting Out of the Nightmare

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.

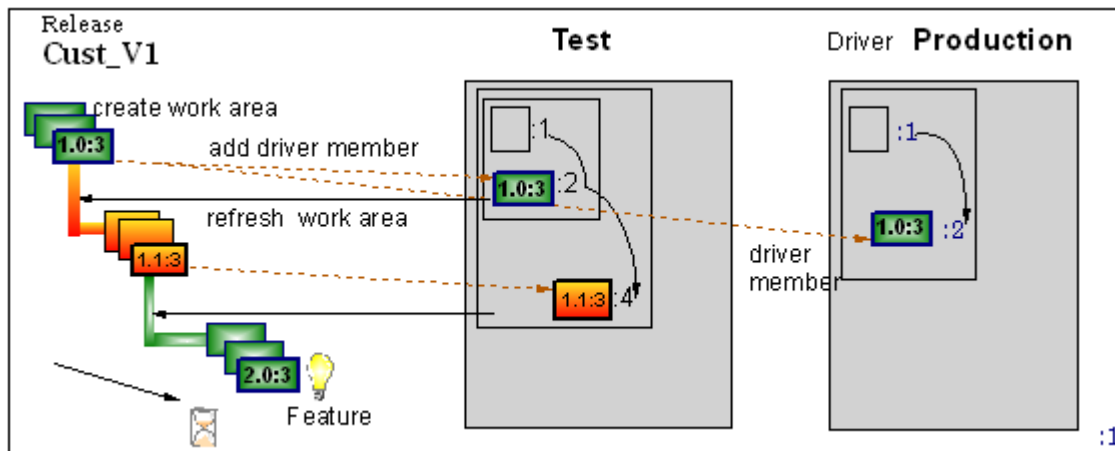


Figure 11. Project Status before Change Management Implementation.

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.

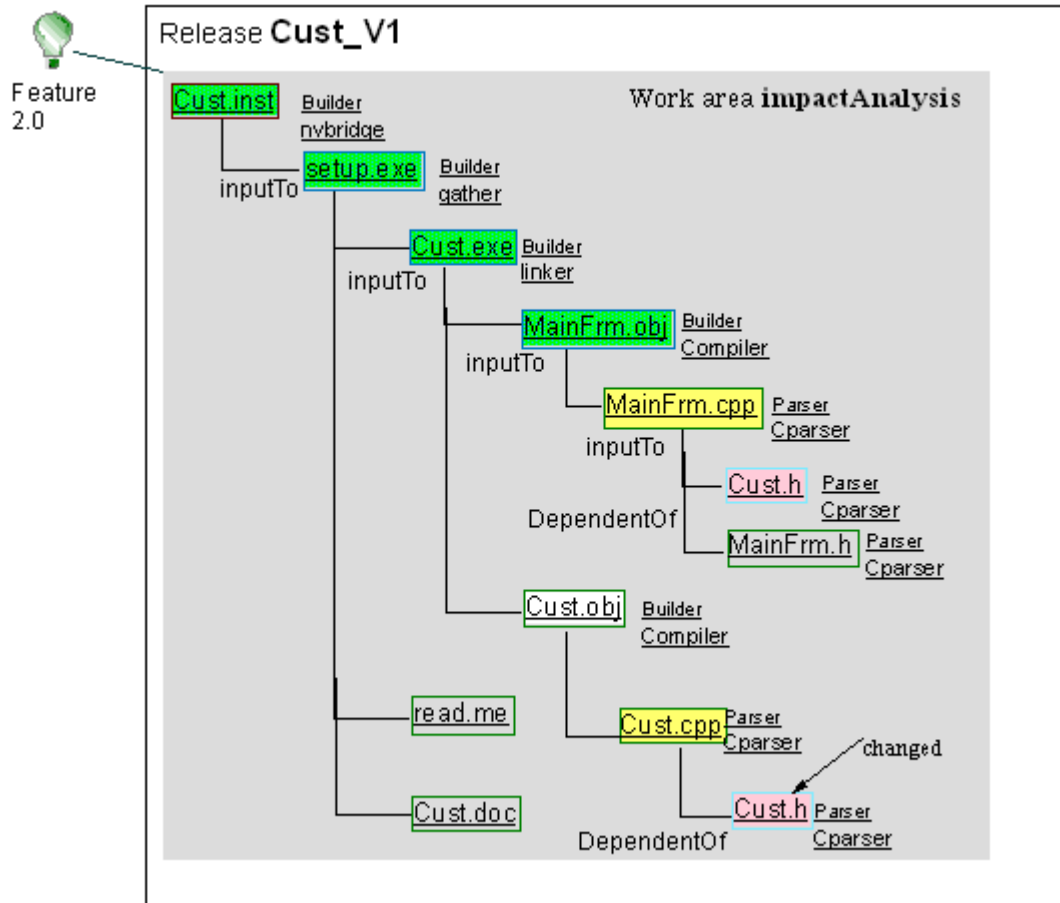


Figure 12. Local Impact Analysis TeamConnection Facilities

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.

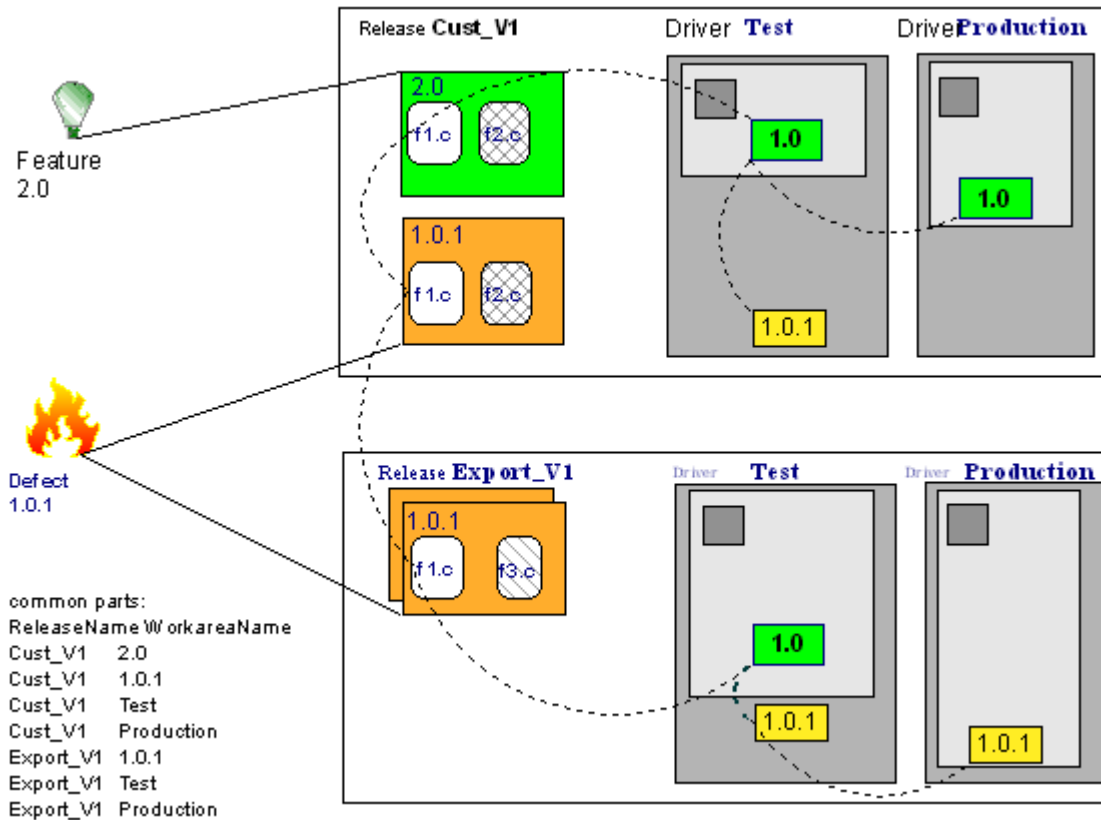


Figure 13. Cross-Impact Analysis TeamConnection Facilities

## 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.

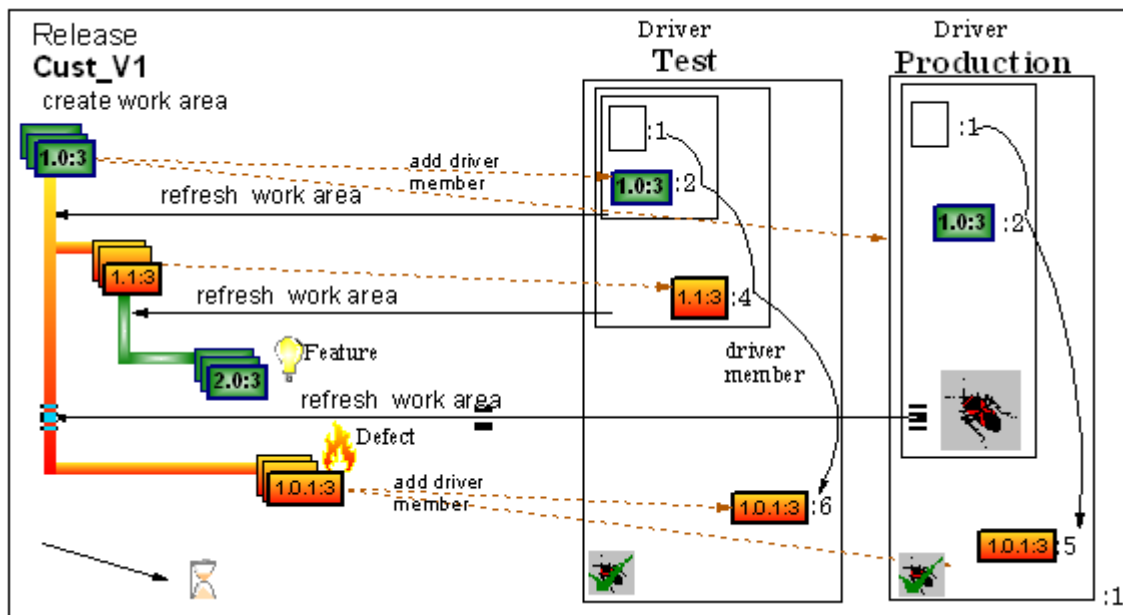


Figure 14. Hot Fix with TeamConnection

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.

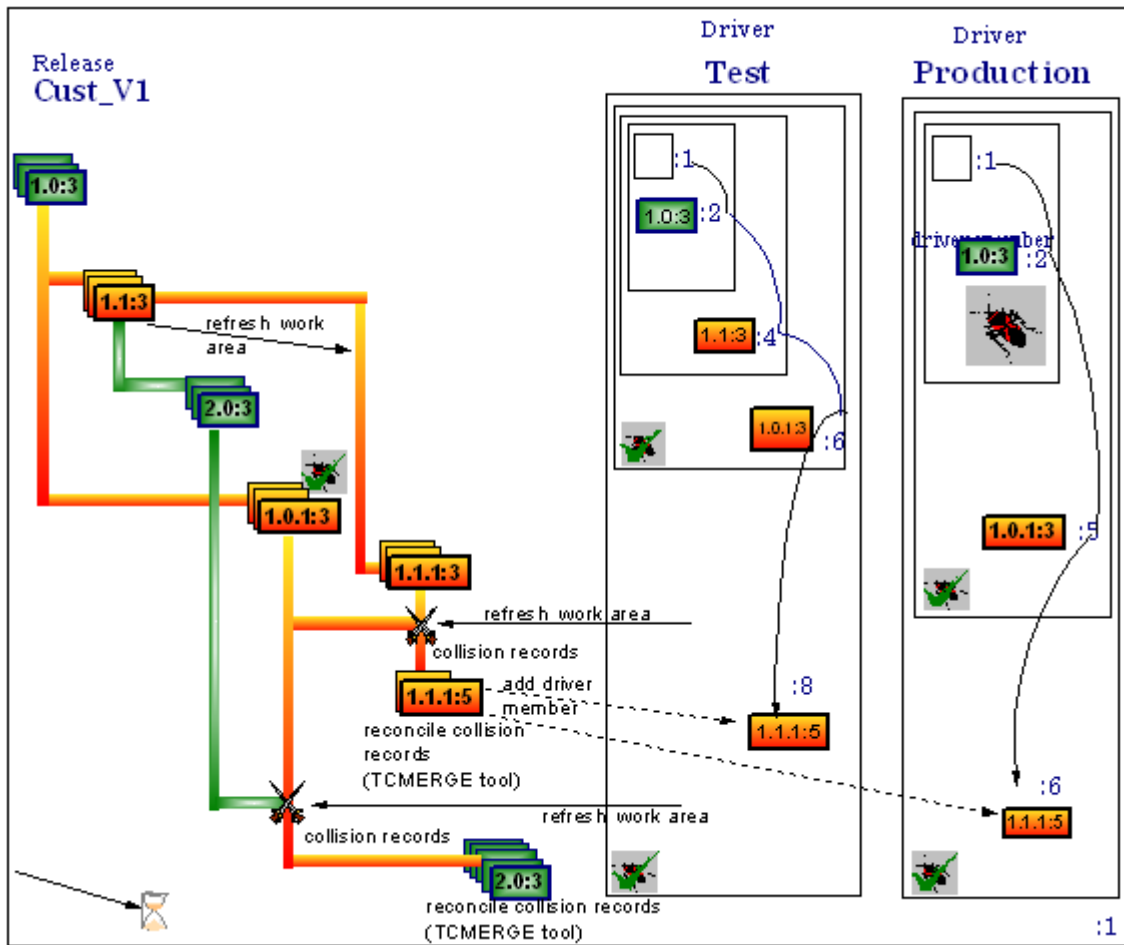


Figure 15. How to Propagate a Fix With TeamConnection

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 the book "SG24-2226-00, VisualAge TeamConnection Enterprise Server, From CMVC to TeamConnection Version 3" 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 Revealed" 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 TeamConnection 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	<b>Processor:</b> IBM RS/6000 with PowerPC architecture (recommended), such as 43P. PowerServer architecture can be used.
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> Any X11 graphics display supported by the RS/6000 workstation

	<b>Memory:</b> 128 MB minimum (256 MB recommended); more may be needed according to number of users and database size
	<p>Disk space:</p> <ul style="list-style-type: none"> <li>- 500 MB for operating system and prerequisites</li> <li>- 200 MB in the target file system for user data (1+ GB recommended)</li> <li>- 160 MB for TeamConnection server code</li> <li>- 65 MB for TeamConnection documentation</li> <li>- 200 MB for DB2</li> </ul> <p>- Amount recommended by operating system for swapper space</p> <p><b>Note:</b> 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.</p>
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	<b>CD-ROM drive:</b> A CD-ROM drive (internal or external) for installation of the product
Build server	<b>Processor:</b> IBM RS/6000 with PowerPC architecture (recommended), such as 43P. Power Server architecture can be used.
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> Any X11 graphics display supported by the RS/6000 workstation
	<b>Memory:</b> 64 MB memory minimum (128 MB recommended); more may be needed according to the compilers and linkers used
	<p>Disk space:</p> <ul style="list-style-type: none"> <li>- 500 MB for operating system and prerequisites</li> <li>- 60 MB for TeamConnection server code</li> </ul> <p>- Amount recommended by operating system for swapper space</p>
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	<b>CD-ROM drive:</b> 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	<b>Processor:</b> IBM RS/6000 with PowerPC architecture (recommended), such as 43P. PowerServer architecture can be used.
--------	---

	<b>Pointing device:</b> A mouse or other pointing device.
	<b>Monitor:</b> Any X11 graphics display supported by the RS/6000 workstation.
	<b>Memory:</b> 64 MB memory minimum.
	<b>Disk space:</b> – 300 MB for operating system and prerequisites – 60 MB for TeamConnection client code – Amount recommended by operating system for swapper space
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation.
	<b>CD-ROM drive:</b> 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 a114–19980430 <sup>2</sup> )
	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 Server

Family server	<b>Processor:</b> Any HP 9000 Series 700 or 800 workstation
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> Any X11 graphics display supported by the processor
	<b>Memory:</b> 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 <b>Note:</b> 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.
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	<b>CD-ROM drive:</b> A CD-ROM drive (internal or external) for installation of the product
Build server	<b>Processor:</b> Any HP 9000 Series 700 or 800 workstation
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> Any X11 graphics display supported by the processor
Build server (continued)	<b>Memory:</b> 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

	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	<b>CD-ROM drive:</b> 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	<b>Processor:</b> Any HP 9000 Series 700 or 800 workstation.
	<b>Pointing device:</b> A mouse or other pointing device.
	<b>Monitor:</b> Any X11 graphics display supported by the processor.
	<b>Memory:</b> 64 MB memory minimum.
	Disk space: <ul style="list-style-type: none"> <li>– 300 MB for operating system and prerequisites</li> <li>– 60 MB for TeamConnection client code</li> <li>– Amount recommended by operating system for swapper space</li> </ul>
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation.
	<b>CD-ROM drive:</b> 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: <ul style="list-style-type: none"> <li>– PHSS_10556</li> <li>– PHSS_10436</li> </ul>
	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.
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	<b>Processor:</b> SPARC or UltraSPARC workstation.
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> Any X11 graphics display supported by the processor
	<b>Memory:</b> 128 MB memory minimum (256 MB recommended); more may be needed according to number of users and database size
	<p>Disk space:</p> <ul style="list-style-type: none"> <li>– 500 MB for operating system and prerequisites</li> <li>– 200 MB in the target file system for user data (1+ GB recommended)</li> <li>– 200 MB for TeamConnection server code</li> <li>– 65 MB for TeamConnection documentation</li> <li>– 200 MB for DB2</li> <li>– 100 MB for temporary space</li> <li>– Amount recommended by operating system for swapper space</li> </ul> <p><b>Note:</b> 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.</p>

Family server (continued)	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	<b>CD-ROM drive:</b> A CD-ROM drive (internal or external) for installation of the product
Build server	<b>Processor:</b> SPARC or UltraSPARC workstation.
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> Any X11 graphics display supported by the processor
	<b>Memory:</b> 128 MB memory minimum (256 MB recommended); more may be needed according to the compilers and linkers used
	Disk space: <ul style="list-style-type: none"> <li>– 500 MB for operating system and prerequisites</li> <li>– 200 MB for TeamConnection server code</li> <li>– 100 MB for temporary space</li> <li>– Amount recommended by operating system for swapper space</li> </ul>
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation
	<b>CD-ROM drive:</b> 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	<b>Processor:</b> SPARC and UltraSPARC workstation.
	<b>Pointing device:</b> A mouse or other pointing device.
	<b>Monitor:</b> Any X11 graphics display supported by the processor.
	<b>Memory:</b> 64 MB memory minimum.
	<b>Disk space:</b> <ul style="list-style-type: none"> <li>– 300 MB for operating system and prerequisites</li> <li>– 60 MB for TeamConnection client code</li> <li>– Amount recommended by operating system for swapper space</li> </ul>
	<b>Communications support:</b> Any token-ring or Ethernet local area network (LAN) adapter card that supports TCP/IP and is supported by the workstation.
	<b>CD-ROM drive:</b> 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 <b>Note:</b> 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	<b>Processor:</b> 133 MHz Pentium-based processor or higher
---------------	---

	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> VGA or higher resolution with the appropriate adapter
	<b>Memory:</b> 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) <b>Note:</b> 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.
	<b>Communications support:</b> Network card supported by TCP/IP for OS/2
	<b>CD-ROM drive:</b> A CD-ROM drive (internal or external) for installation of the product
Build server	<b>Processor:</b> 133 MHz 486-based processor or higher
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> VGA or higher resolution with the appropriate adapter
	<b>Memory:</b> 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
	<b>Communications support:</b> Network card supported by TCP/IP for OS/2
Build server (continued)	<b>CD-ROM drive:</b> 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	<b>Processor:</b> 66 MHz 486-based processor or higher
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> VGA or higher resolution with the appropriate adapter

	<b>Memory:</b> 16 MB minimum
	Disk space: – 75 MB for operating system and prerequisites – 25 MB for TeamConnection client code – 32 MB for swapper space
	<b>Communications support:</b> Network card supported by TCP/IP for OS/2
	<b>CD-ROM drive:</b> 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

<b>Server</b>	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
<b>Client</b>	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
<b>Client</b> (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)
<b>Build server</b>	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	<b>Processor:</b> 133 MHz Pentium-based processor or higher (however, the PowerPC is not supported)
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> VGA or higher resolution with the appropriate adapter
	<b>Memory:</b> 64 MB memory minimum; more may be needed according to number of users and database size
Family server (continued)	<p><b>Disk space:</b></p> <ul style="list-style-type: none"> <li>–200 MB for operating system and prerequisites</li> <li>–200 MB for user data (1+ GB recommended)</li> <li>– 60 MB for TeamConnection server code</li> <li>–200 MB for DB2</li> <li>–128 MB for swapper space (150+ MB recommended)</li> </ul> <p><b>Note:</b> 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.</p>
	<b>Communications support:</b> Network card supported by TCP/IP for Windows NT
	<b>CD-ROM drive:</b> A CD-ROM drive (internal or external) for installation of the product

Build server	<b>Processor:</b> Any processor supported by the required version of Windows, except the PowerPC
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> VGA or higher resolution with the appropriate adapter
	<b>Memory:</b> 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
	<b>Communications support:</b> Network card supported by TCP/IP for Windows NT or Windows 95
	<b>CD-ROM drive:</b> 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	<b>Processor:</b> Any personal workstation supported by the operating system, except the PowerPC
	<b>Pointing device:</b> A mouse or other pointing device
	<b>Monitor:</b> VGA or higher resolution with the appropriate adapter
	<b>Memory:</b> 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
	<b>Communications support:</b> Any network card supported by the above workstation that supports TCP/IP
	<b>CD-ROM drive:</b> 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 NT** 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_seq:0. time=0. ms  
64 bytes from 1.23.456.78: icmp_seq:1. time=0. ms  
64 bytes from 1.23.456.78: icmp_seq:2. time=0. ms
```

For **Windows NT** 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 Revealed





---

## 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 tcmmerge.
  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>\db2files\fixes\readmew.db2
- OS/2: <cdrom>\db2files\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
<code>DB2DBDFT=tcfamily</code>	<code>DB2INSTANCE= db2inst1</code>
<code>TC_DBPATH=/home/tcfamily</code>	<code>DB2INSTANCE_HOME=/home/db2inst1</code>
<code>DB2_DBPATH=database</code>	<code>/home/db2inst1/db2inst1</code>
* Teamcd Server Daemon	

Table 18. Relationship between TeamConnection Codes and DB2 UDB Codes

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

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_05_00 # 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 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 Team-Connection 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
nfllocks	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
semmsns	256	512	1024
semmni	128	256	512
semmap	(2 + SEMMNI)	(2 + SEMMNI)	(2 + SEMMNI)
semnmu	256	512	1024
shmmax	67108864 (2)	134217728 (2)	268435456 (2)
shmseg	16	16	16
shmmni	300	300	300

Notes:

- Parameters msgmnb and msgmax must be set to 65535.
- 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/tl1list` -> 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_semns	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**: *cdarfs*.
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:  

```
sm &
```

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/c0t2d0 /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/c0t6d0s2 /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).

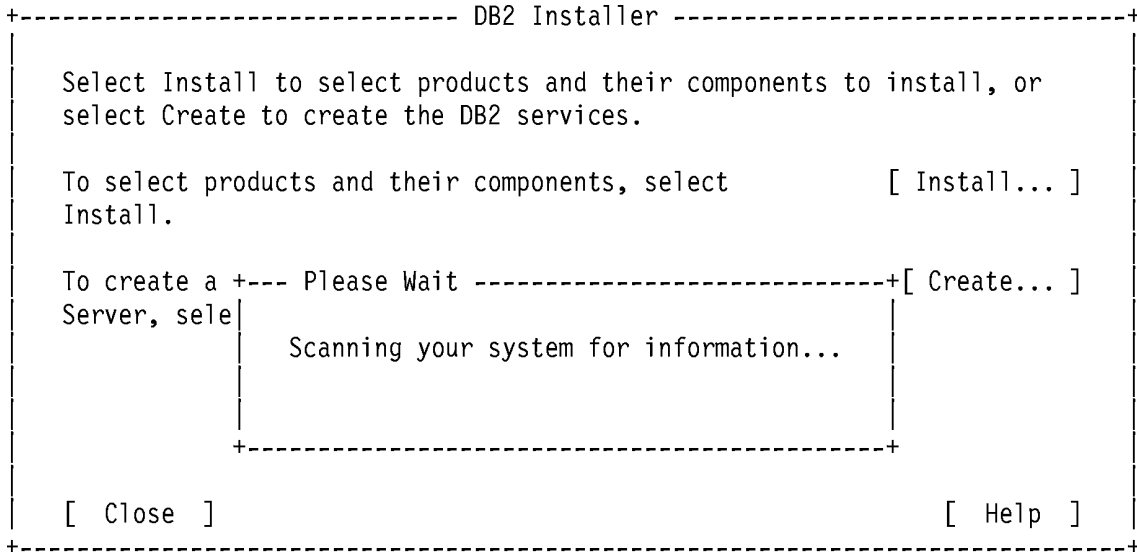


Figure 16. DB2 Installer Screen

- 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

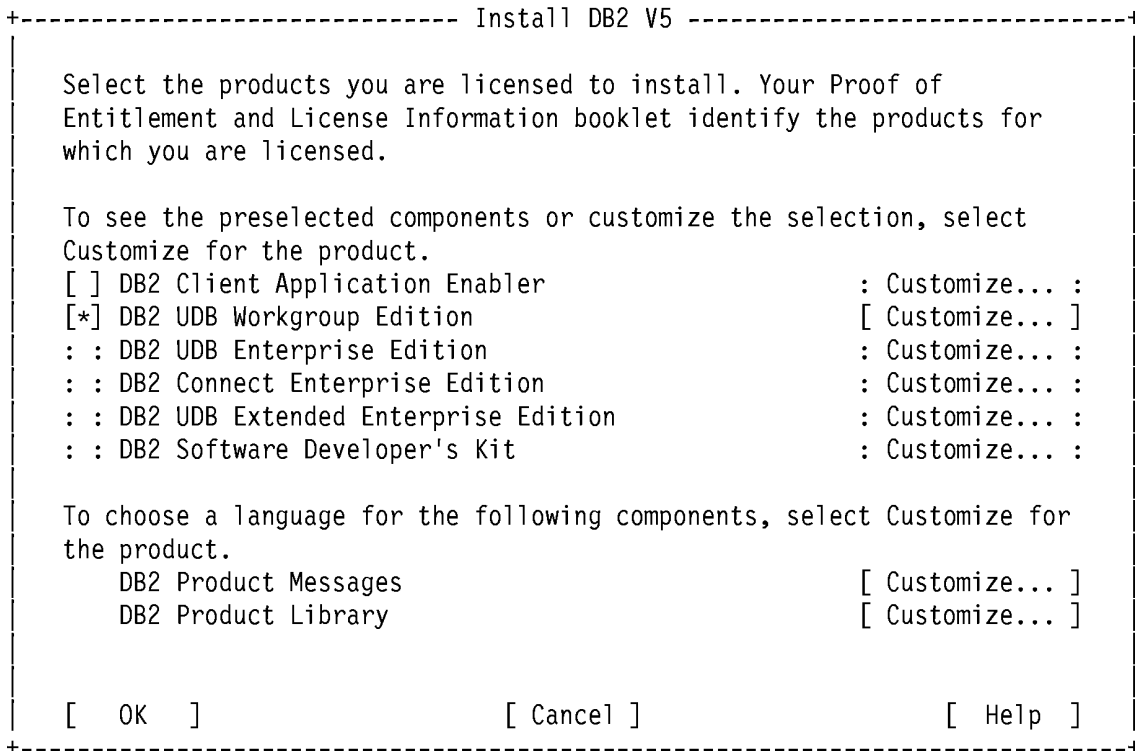


Figure 17. Install DB2 V5 Screen

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.

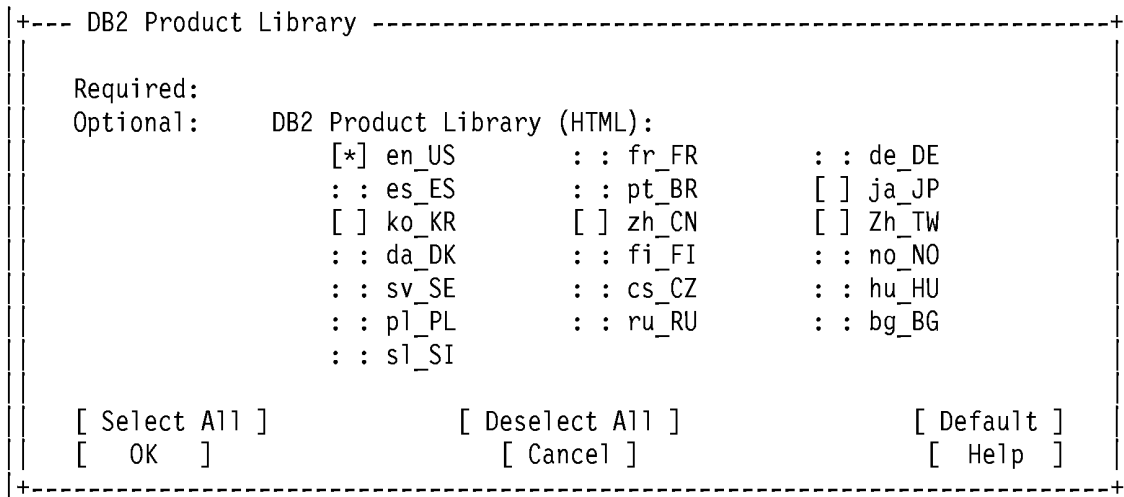


Figure 20. DB2 Product Library Screen

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 -acqX -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**.

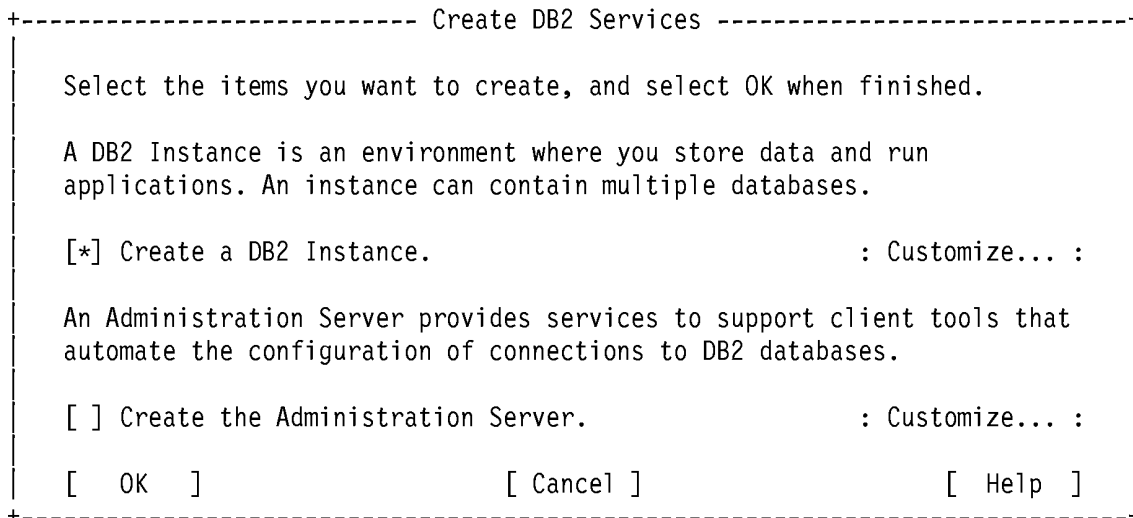


Figure 28. Create DB2 Services Screen, Snapshot 1

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).

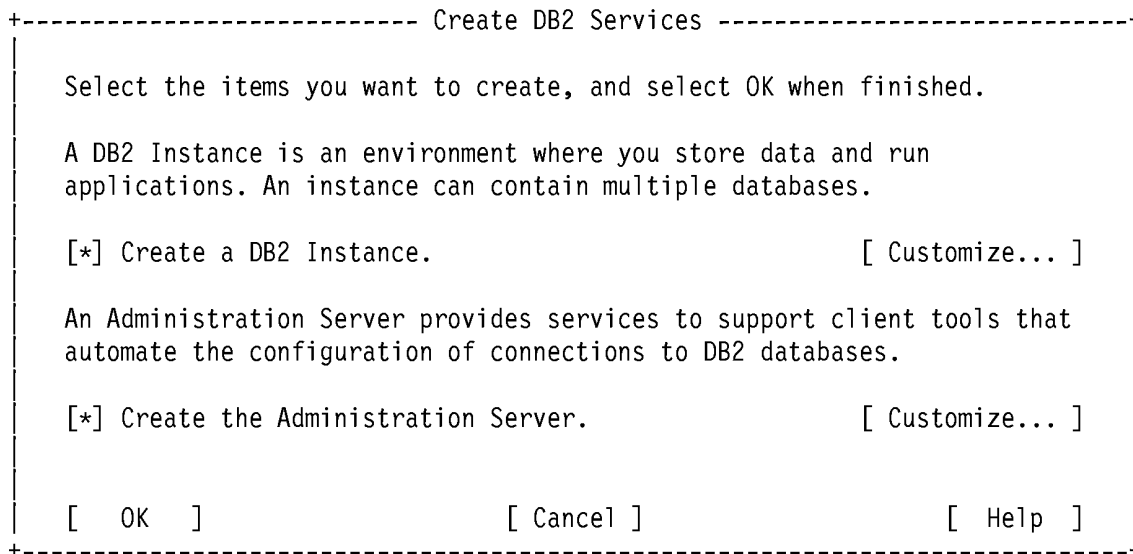


Figure 33. Create DB2 Services Screen, Snapshot 2.

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.





```

+----- 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.

```

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



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/hpux1đ/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:
  - Y Acrobat Reader for PDF files.
  - Y Java for tcadmin and tcmerge.
  - Y 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
<b>AIX</b>	/usr/teamc
<b>HP-UX</b>	/opt/teamc
<b>Solaris</b>	/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.

- Y 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.

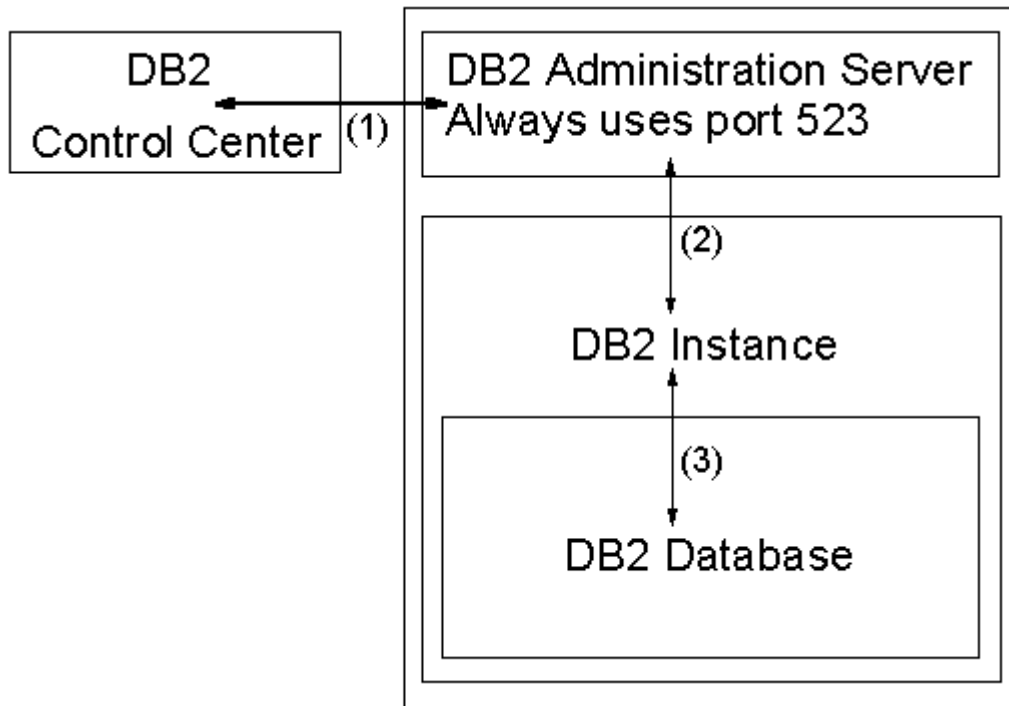


Figure 40. Relationship between Control Center and Administration Server

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 153 for 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/sql/lib/profile.env

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

## Windows NT

\HKEY\_LOCAL\_MACHINE\SOFTWARE\IBM\DB2\PROFILES\

---

### 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 221 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/sql/lib/db2profile
```

- **C shell.**

```
source $DB2INSTANCE_HOME/sql/lib/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, `db2system`, 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 221 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/sql/lib/db2profile
```
- **C shell.**  

```
source DB2INSTANCE_HOME/sql/lib/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. How to Build Applications



---

## Chapter 15. Basic Build Concepts

This chapter defines terms and briefly describes the TeamConnection pieces that work together in building an application. The TeamConnection build function does the following:

- It builds applications for platforms in addition to those it runs on. Currently, you can build applications using TeamConnection on the following platforms: AIX, HP-UX, Solaris, MVS, MVS/OE, OS/2, Windows NT, and Windows 95.
- Its graphical representation of the structure of an application makes it easier to visualize and change that structure.
- It lets you build an application using any number of machines working in parallel.
- Because it is fully integrated with TeamConnection's version control system, it ensures that the correct versions of parts are used in a build.
- It can not only work with parts that represent files, such as C source files, it can also work with parts that represent objects, such as VisualAge Generator applications.
- It can manage other steps related to software packaging and distribution.

---

### 15.1 The Physical Structure of the Build Function

Figure 41 shows the structure of the TeamConnection build function, including the build server.

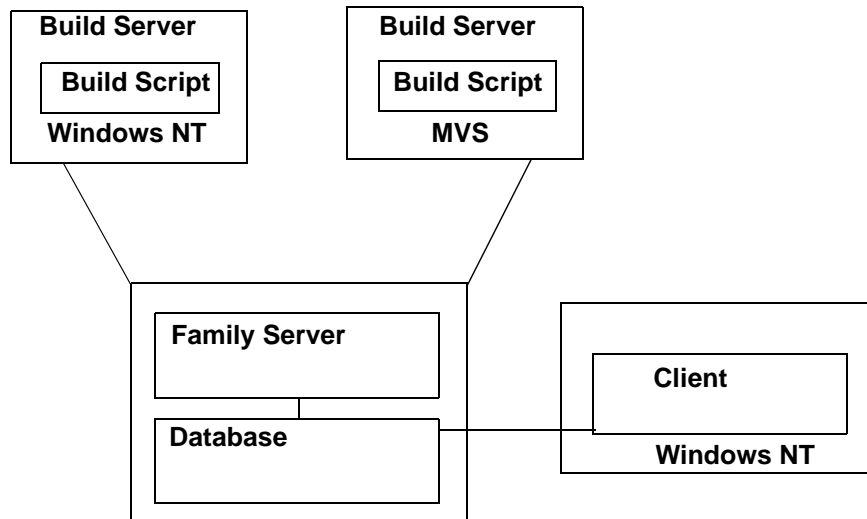


Figure 41. The Physical Structure of TeamConnection

Build servers are started by a TeamConnection administrator. For more information, see "Installing, Strating & Stopping Build Servers"

---

## 15.2 The Build Object Model

Figure 42 on page 51 shows the TeamConnection objects and events that constitute the build function, as illustrated in a sample application named *msgcat.exe*. This build object model is a conceptual model of the build function. When you use TeamConnection to define a build, you work with a build tree (a simplified graphical illustration of the build object model), which you can access through the TeamConnection GUI. This section explains the build objects and events represented in a build tree. See 15.5, "Working with a Build Tree" on page 163, for an explanation of build trees.

In TeamConnection, the build function is always described and discussed in terms of the final output of the build—that is, the product or executable file that the build produces. For the sample application shown in this illustration, *msgcat.exe* is the build output and appears at the top of the build object model and as the top branch of the build tree.

When you want to actually build the product, you request a build of *msgcat.exe*. TeamConnection uses the build tree that you define for this

product to determine which objects and build events it needs to generate the final output. The objects and events that TeamConnection uses for a build are discussed next.

### 15.2.1 TeamConnection Part

In TeamConnection, a *part* is an object produced or used during a build, containing any data produced or used by the build. For example, a part called *hello.c* contains the source code for the application called *msgcat*. A part can be a text or binary file, or an object such as a VisualAge Generator generic collector.

### 15.2.2 Build Event

A *build event* is an individual step in the build of an application, such as the compiling of *hello.c* into *hello.obj*. A specific build request typically contains many build events. For example, if you start a build of an entire application, TeamConnection initiates a build event for every compile and link operation.

Build requests are processed in a round-robin fashion for every Team-Connection family involved in a build. Build events are initiated in the order that they are received by each build machine involved in the build request, sometimes in parallel.

Build event processing is internal to TeamConnection; you cannot interact with these processes directly.

### 15.2.3 Builder

A *builder* is an object that can transform a build event's input parts into output parts by calling tools such as linkers or compilers. For example, one builder may know how to transform the input part *hello.c* into the output part *hello.obj*. A different builder may know how to transform *hello.obj* into *msgcat.exe*. Builders are associated with the parent, or output part, rather than with the child, or input.

### 15.2.4 Build Script

The *build script* is an object that a builder uses in transforming inputs to outputs; it is essentially a binding between TeamConnection and a transformation tool, such as a linker or compiler. In OS/2, Windows, UNIX, or MVS/OE environments, a build script is usually a command file, but it can be a string that calls the tool. In MVS, it is a file containing JCL-like instructions.

### 15.2.5 Parser

In TeamConnection, a *parser* is a tool that can read a source file and report back a list of dependencies of that source file. It frees a developer from having to identify all of the dependencies one part has on other parts to ensure that a complete build is performed. For example, a C parser can read a C source code file and report back a list of the files included by the source file or by any of the included files.

TeamConnection re verifies all parser dependencies. It functions as follows:

1. When the user creates or checks in the part, TeamConnection adds all parser dependencies that it can find.
2. During build, TeamConnection again checks all parser dependencies and updates the list as needed.

---

## 15.3 Parent-Child Relationships in a Build Tree

One relationship that is important to understand and distinguish is the relationship between parent and child parts in a build tree.

Although parent-child relationships usually imply that the parent part generates the child part, in a TeamConnection build, the opposite takes place. Because TeamConnection places the build output at the top of the tree, it refers to the build output as the *parent* and to the build input as the *child*.

A child part can be related to a parent part in one of three ways: it can be an input part, an output part, or a dependent part.

### 15.3.1 Input Parts

A part used as direct input to your build. An example of this is a C language source part. If you start a build and this part has changed, the changed part becomes part of the new build.

### 15.3.2 Output Parts

An *output part* is a generated output from a build, such as an OBJ or EXE part, or a part with no contents that serves as an organizer object. If you start a build and this part has changed, the changed part is included in the new build.

### 15.3.3 Dependent Parts

A *dependent part* is a part that is needed for the build operation to complete but that is not passed directly to the compiler. An example is an included part. If you start a build and this part has changed, it is the changed part that is included in the new build.

---

## 15.4 Generating Build Output

To understand how build output is generated, it may be easier to start at the bottom of the build object model and work your way up. In Figure 42 on page 163, `hello.h` and `bye.h` are C source files that are embedded in `hello.c` and `bye.c`, respectively. The parser, `parser1`, is able to read `hello.c` and `bye.c` to determine files they embed. This build object model contains three build events:

- The builder `compiler1` compiles `hello.c` into `hello.obj`.
- The builder `compiler1` compiles `bye.c` into `bye.obj`.
- The builder `linker1` links `hello.obj` and `bye.obj` into `msgcat.exe`

This build object model contains the following parent-child relationships:

- `msgcat.exe` is the parent of `hello.obj` and `bye.obj`.
- `hello.obj` is the parent of `hello.c`
- `bye.obj` is the parent of `bye.c`

You establish these parent-child relationships between parts when you create the parts in `TeamConnection`.

Before you can build `msgcat.exe`, for example, you need to create a place holder part for it and designate `linker1` as its builder. You then create place holder parts for `hello.obj` and `bye.obj` and designate `compiler1` as their builder and `msgcat.exe` as their parent.

---

## 15.5 Working with a Build Tree

Software developers must provide the information from which `TeamConnection` determines the build events that make up a build request. An application's build tree shows this information graphically.

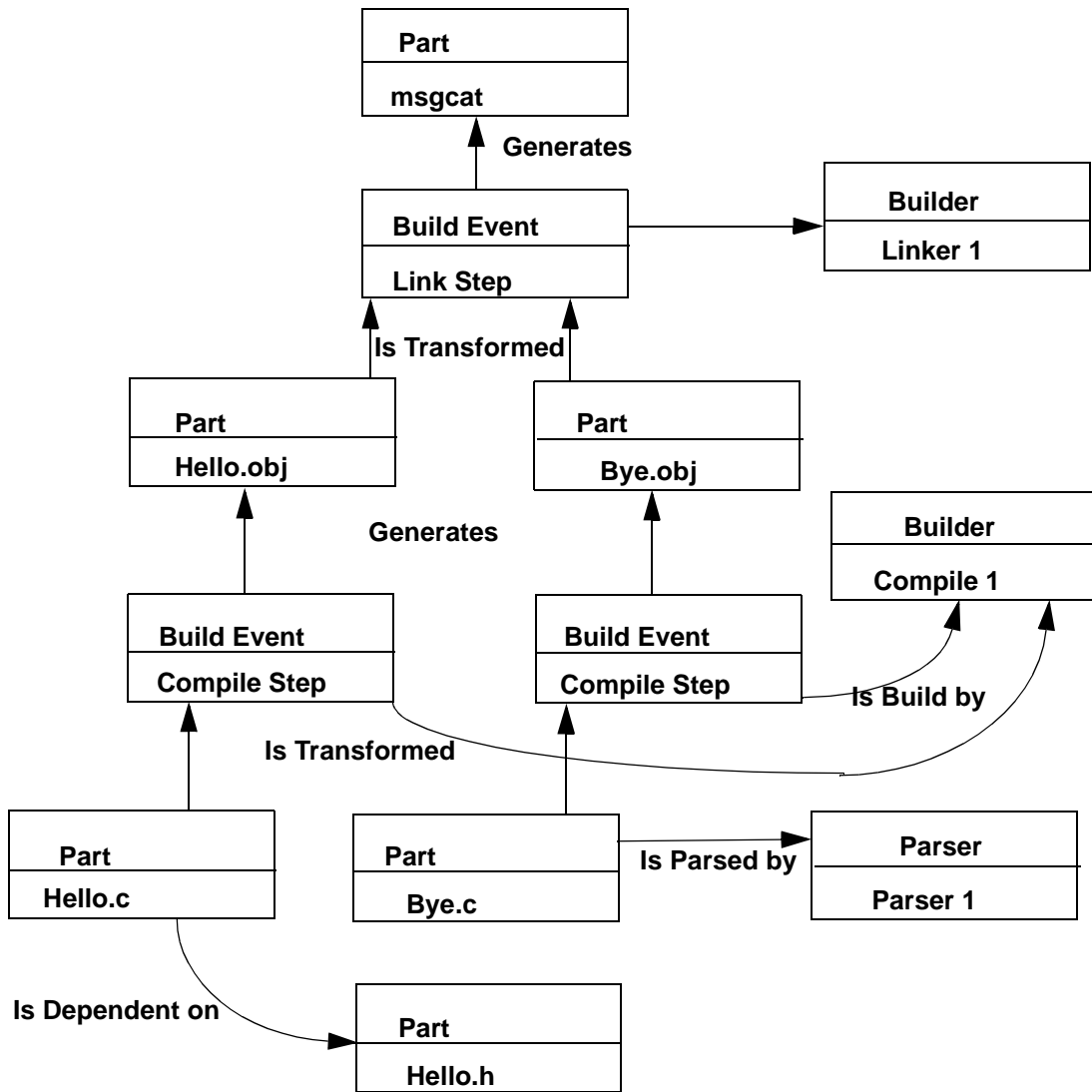
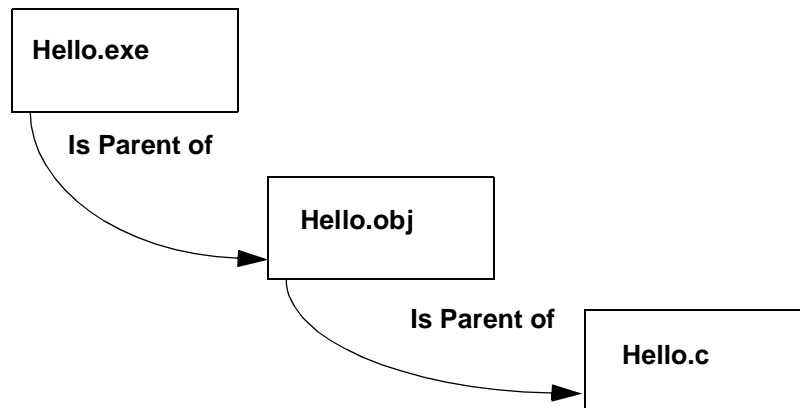


Figure 42. Sample Build Object Model for msgcat.exe

A build tree is a simplified version of the build object model, showing the dependence 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. Figure 43 shows a build tree for the hello application.





*Figure 43. The Build Tree for the hello Application*

In this simple application, `hello.c` is compiled to create `hello.obj`, which in turn is linked to create `hello.exe`. The build tree shows that `hello.exe` is the parent of `hello.obj`, which in turn is the parent of `hello.c`. To build the entire application, request to build `hello.exe`.

Just as the parts that make up an application are versioned, the relationships between these parts are versioned. Thus, more than one version of the build tree can exist. For example, Figure 44 shows two different versions of the same build tree.

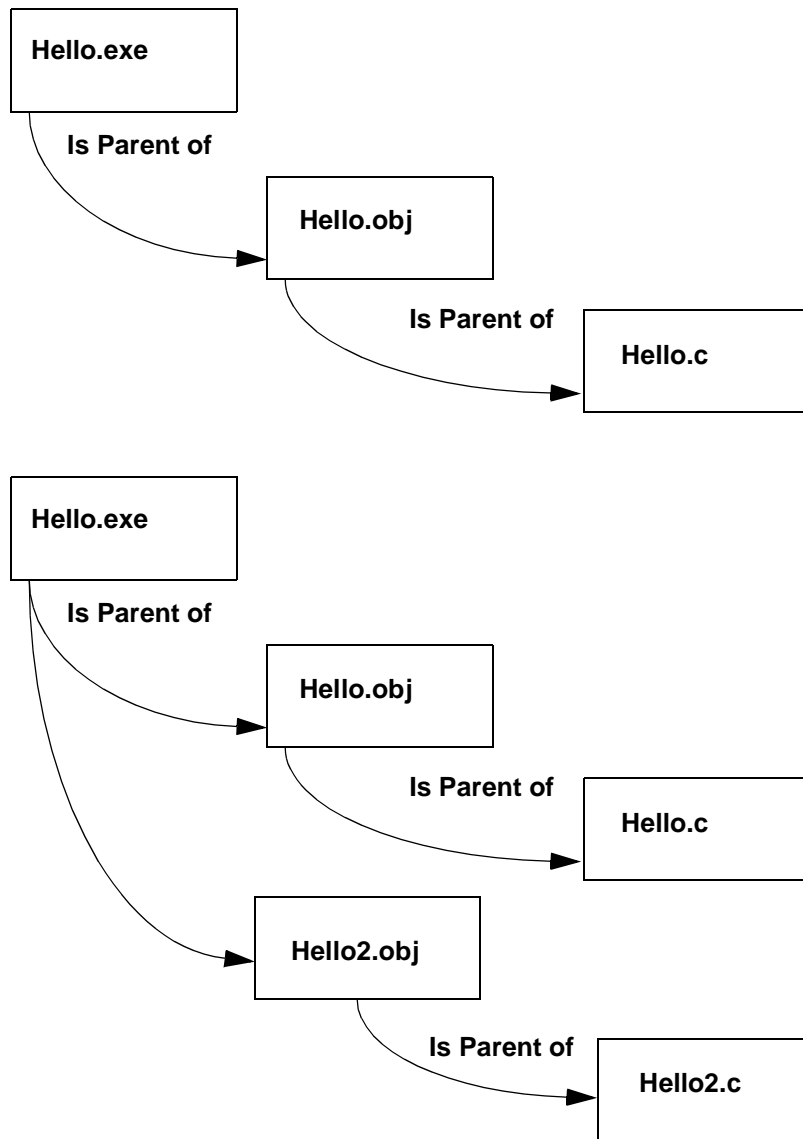


Figure 44. Two Versions of a Build Tree

---

## 15.6 Putting the Pieces Together

We hereunder lists the tasks involved in preparing for building an application and in actually building it. Usually an administrator does the preparation steps, but anyone with the proper authority can do so.

- Creating build startup files
- Starting build servers
- Stopping build servers
- Writing a build script
- Creating a builder
- Creating a parser
- Defining a build tree
- Starting a build
- Stopping a build
- Verifying the parts to be built



---

## Chapter 16. A Simple, Yet General Process

The life cycle of TeamConnection objects follows an orderly succession of states. These states include: Open, Size, Review, Working, Verify, and Closed. Team Connection tracks the state of objects. Defect reports are used to implement problem tracking associated with TeamConnection objects. Feature reports are used to follow the addition of new function or design enhancements to TeamConnection objects. However, tracking the state of objects is only part of the story.

You may be surprised at the way the TeamConnection build process works if you are used to more traditional library management systems such as CMVC. In such systems, you might check out or extract files from the repository and use a compiler on your local machine to build an executable from various source modules. You then test your executable file and, if it performs as expected, you check it back in to your repository.

TeamConnection has a slightly different philosophy. Your files (or parts in this case) are checked into TeamConnection and you then invoke what is known as a *builder*, which is most likely not even on your client or family server machine. The builder takes your parts and—perhaps with the help of a compiler and linker—it generates an executable file. This file exists as a *part* in the TeamConnection database. You must extract it into an appropriate environment in order to test it.

You can now envisage a sophisticated build environment comprising several build machines all working in concert to produce a product from parts created by a team of developers. You can even go one step further and build for multiple environments simultaneously.

---

### 16.1 The Flow

#### 16.1.1 Open State

When a TeamConnection user (originator) creates or originates a feature or identifies a defect, it is first in the Open state. When you open a defective part or start a feature, you must specify a TeamConnection component. The owner of that component is responsible for overseeing the work on that feature or defect until it is moved to the Closed state. The originator must verify that the implementation surrounding the defect or feature is in fact properly resolved

### **16.1.2 Returned State**

The component owner typically assigns the ownership of a defect or feature to a TeamConnection user who is directly responsible for the implementation. A defect or feature object in the Open, Design, Size or Review state can be returned by its owner to the originator. This may happen if it involves a problem that has been addressed in another defect fix or will be corrected in a feature of a new release

### **16.1.3 Canceled**

A defect-fix originator or a superuser can cancel an open or returned defect or feature. A canceled defect or feature remains inactive unless it is reopened by the originator.

### **16.1.4 Design**

Defects or features that follow the design size review process (DSR) are moved from the Open to the Design state. It is during this stage that the changes proposed for the feature or defect are designed. Some text summarizing the design is entered before the object can move to the Size state.

### **16.1.5 Size**

After design information is added, defects or features move to the Size state. It is here that a sizing record is created in order to allocate developer resources for the work required by the defect or feature.

### **16.1.6 Review**

The design text and sizing records are reviewed by the owner. Ideally, everything is acceptable and the act of accepting the feature or defect moves it into the Working state. If everything is not acceptable, the owner can return the defect or feature to the originator or move it back to the Design state for further refinement.

### **16.1.7 Working**

After the change (defect or feature) is accepted, it is in the Working state. If the dsrFeature or dsrDefect subprocesses are in effect, then a work area is automatically created for each release identified in the sizing records. Work areas can have several states or actions that apply

- Integrate
- Check
- Commit

- Test
- Complete
- Freeze
- Refresh
- Fix
- Cancel
- Create approval records
- Create fix records
- Add driver members
- Add corequisites
- Remove driver members
- Remove corequisites

### **16.1.8 Verify**

Defects and features are automatically moved to the Verify state when all of the associated work areas are integrated. If a release is specified when the defect or feature is created, then all work areas for the release must move to the Complete state before the defect or feature can be placed in the Verify state. When a defect or feature is accepted, then TeamConnection creates a verification record that allows the originator to *accept*, *reject*, or *abstain*. Once all verification records have been accepted, the change then moves to the Closed state. If a verification record is rejected, then the defect or feature is moved back to the Working state.

### **16.1.9 Testing**

When the work on a defect or feature is completed, it is subjected to various levels and complexity of testing to ensure that the associated objects are performing as required by the design. Depending upon the scope of the project, this can involve unit test, regression test, functional verification test, integration test, and so on.

### **16.1.10 Integration**

If the defect or feature survives the rigors of the testing process, it can then be integrated into the release.

### **16.1.11 Production**

Once the defect or feature has proven itself in the development and test environments, it must be packaged and made ready for its ultimate role in a production environment. Once again TeamConnection can step in and show its versatility and help you manage your packaging function.

### 16.1.12 Closed

The Closed state is the final state of a change, whether defect or feature. Once a defect or feature is closed, it cannot be reopened or moved back to the Working state.

---

## 16.2 Typical Evolution of a Build

### 16.2.1 Working with Features

During the normal course of business, the marketing organization is constantly evaluating comments and feedback from customers. After deliberation with the development staff, a decision is made as to which requests for new function in the software will be incorporated into the next or a subsequent release of the software.

Our customer has decided that the company cannot live without smaller fonts in the next release of their Java-based clock program called *Clock2.class*, which is the output of: `javac Clock2.java`. This code was originally developed by Sun Microsystems Javasoft Division. We downloaded it from the Internet at:

<http://www.javasoft.com/applets/jdk/1.0/index.html>

As we work through adding this new function, we will perform the following:

1. Create a feature.
2. Design, size, review, and accept the feature.
3. Create a work area.
4. Check out the affected part or parts.
5. Modify, build, and test part fixes (updates).
6. Integrate the work area.
7. Close the feature.

### 16.2.2 Opening a Feature

A new function is typically added to create a new release of software through the feature process. Features can be opened from the command line or the TeamConnection GUI interface. To open from the command line, type in the following:

```
teamc Feature -open -component RedBook5260 -remarks - -abstract "add
variable fonts as parameters to Clock2.class" -name feature3 -release
prototype -prefix s -verbose
```



Instead, you can open from the TeamConnection window shown in Figure 45.

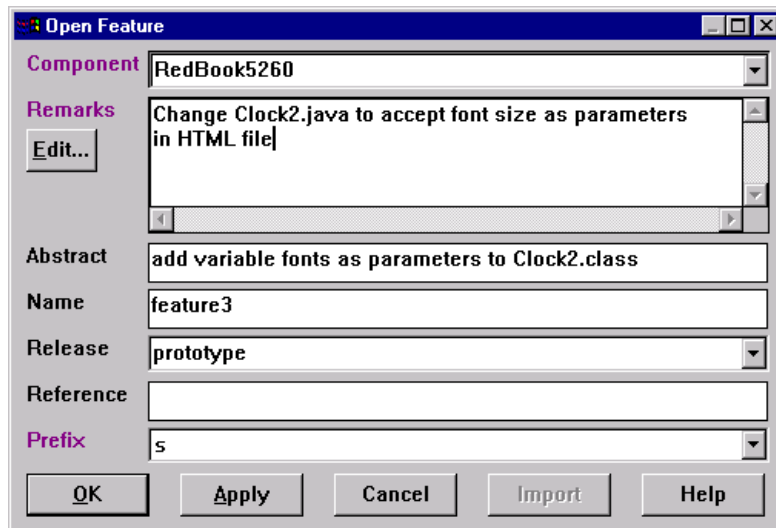


Figure 45. Dialog Screen for Open Feature

After you open your feature, you can view the information that it contains with the following command:

```
teamc Feature -long -view feature3 -verbose
```

As an alternative, TeamConnection offers Figure 46 , which shows the view feature information field. Enter `feature3` and click on **OK**.

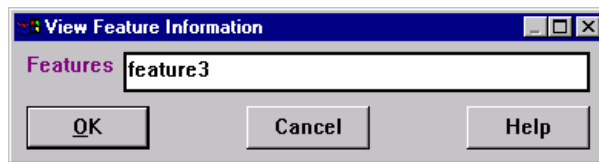


Figure 46. View Feature Information Field

### 16.2.3 Design – Size – Review Process

If the release follows the usual track process, then the feature follows the DSR process, alternatively known as *design - size -review*. A plan for the implementation of the feature needs to be agreed upon. We must determine what and how much programming resources need to be delegated to the implementation.

Figure 47 shows the tasks screen.

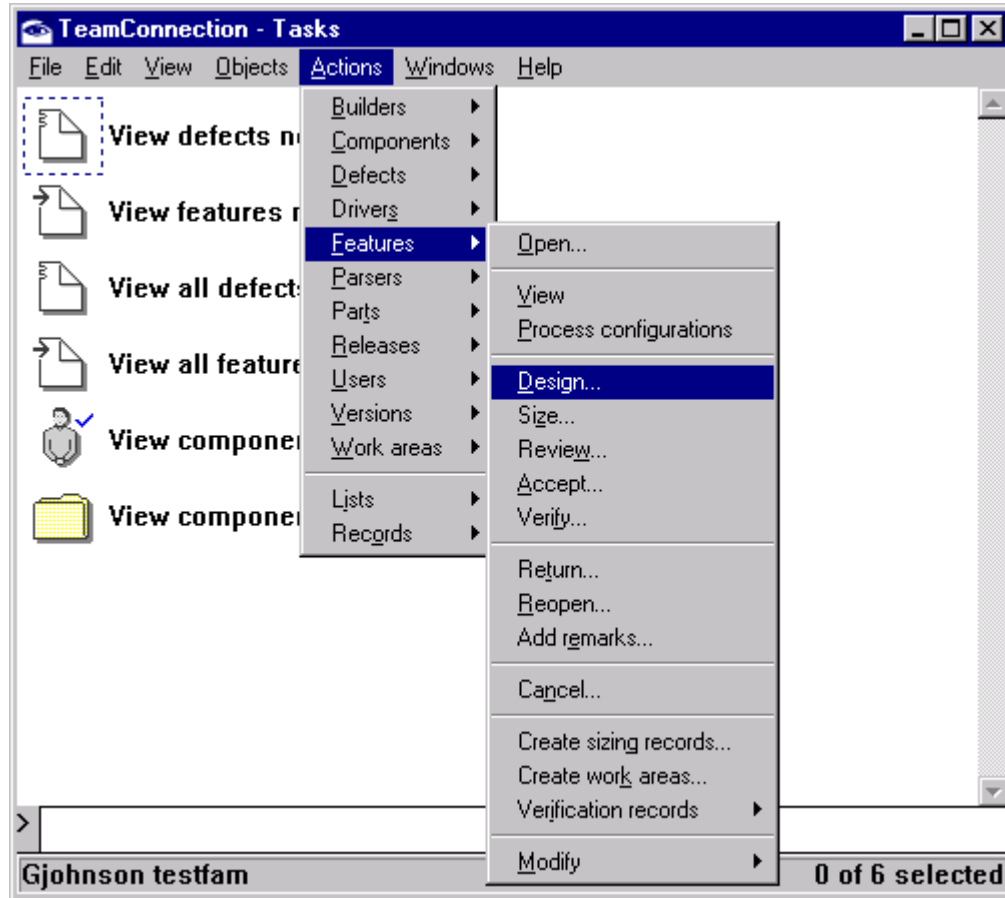


Figure 47. Screen Showing Design, Size, and Review Tasks

As you can see from Figure 47, **Design**, **Size**, and **Review** are conveniently located next to one another on the pull-down menu.

These are the commands:

```
teamc Feature -design feature3 -remarks - -verbose
teamc Feature -size feature3 -remarks - -verbose
teamc Feature -review feature3 -remarks - -verbose
```

After the feature is accepted, it is in the Working state and we may begin taking whatever actions with the parts are needed to effect the required changes. Acceptance takes the following:

```
teamc Feature -accept feature3 -answer new_function -remarks -
-verbose
```

#### 16.2.4 Create Work Area

We can create a work area where the feature's new function can be added to the release. To do so directly, type the following:

```
teamc WorkArea -create -name feature3 -defect feature3 -release
prototype -owner Gjohnson -verbose
```

As an alternative, Figure 48 shows the TeamConnection screen.

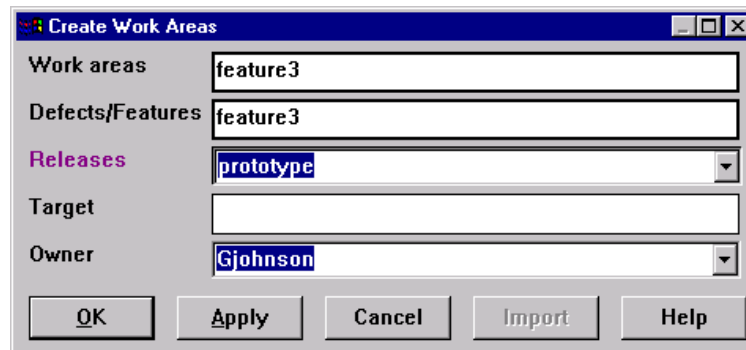


Figure 48. Create Work Areas Screen

#### 16.2.5 Modify, Build, and Test Modifications

We now run into a slight difficulty. The customer found a problem with Clock2.java that required immediate attention on the part of the developers. (See 16.3, "An Emergency Hot Fix" on page 179.) We might think that we could just check out Clock2.java and be merrily on our way, but our files have already been modified. What we need to do is refresh our work area from the work area where the code has changed:

To do so directly, type the following:

```
teamc Part -refresh Clock2.java -type TcPart -release release 2
-workarea feature3 -fromworkarea hotfix -verbose
```

Or, use the TeamConnection GUI. Figure 49 shows the dialog box for summoning the refresh.

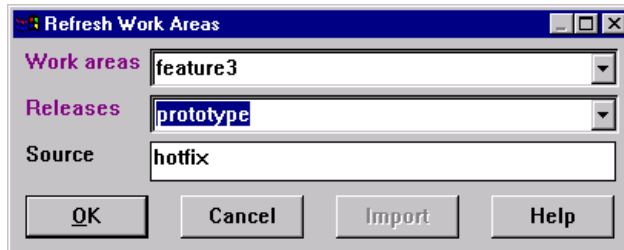


Figure 49. Refresh Work Area Dialog Fields

We can now begin our modifications. We can check out Clock2.java with the following command:

```
teamc Part -stdout -checkout Clock2.java -release prototype -workarea
feature3 -verbose
```

As an alternative, Figure 50 shows the dialog box to check out parts.

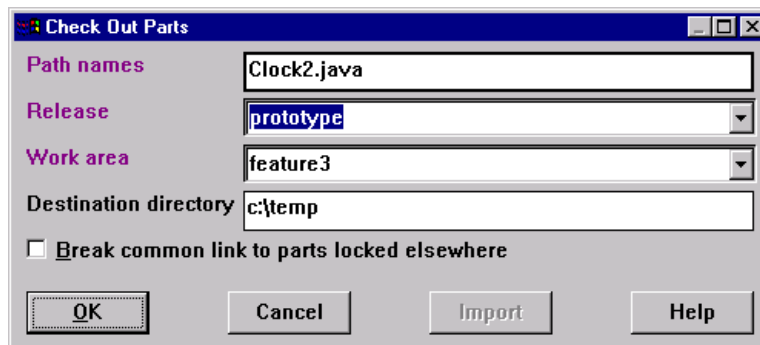
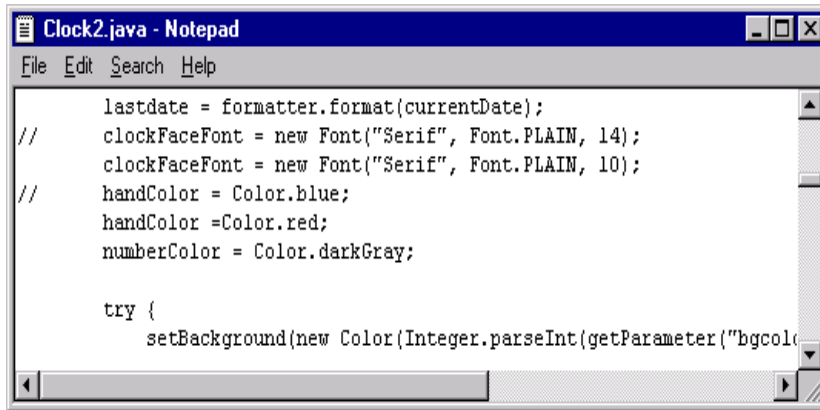


Figure 50. Check Out Parts Dialog Fields

You may have a better idea of where to place the file parts that need editing. We simply placed them in the C:\temp\ in order to make the appropriate changes. To do that directly, type the following:

```
notepad C:\TEMP\Clock2.java
```

To make the illustration of the changes straightforward, we have simply "commented out" the old line of code that uses the 14-point fonts. The new line using the 10-point fonts is added below (Figure 51). You can also see the next two lines of code that address the change in the clock hand color from blue to red as handled by the hot fix in 16.3, "An Emergency Hot Fix" on page 179.



```
lastdate = formatter.format(currentDate);
// clockFaceFont = new Font("Serif", Font.PLAIN, 14);
clockFaceFont = new Font("Serif", Font.PLAIN, 10);
// handColor = Color.blue;
handColor = Color.red;
numberColor = Color.darkGray;

try {
    setBackground(new Color(Integer.parseInt(getParameter("bgcol(
```

Figure 51. Notepad Showing the Changes

After we have made our changes, we check our parts back in. Enter the following command:

```
teamc Part -checkin Clock2.java -release prototype -workarea feature3
        -relative C:\TEMP -verbose
```

The alternative is shown in Figure 52 on page 174.

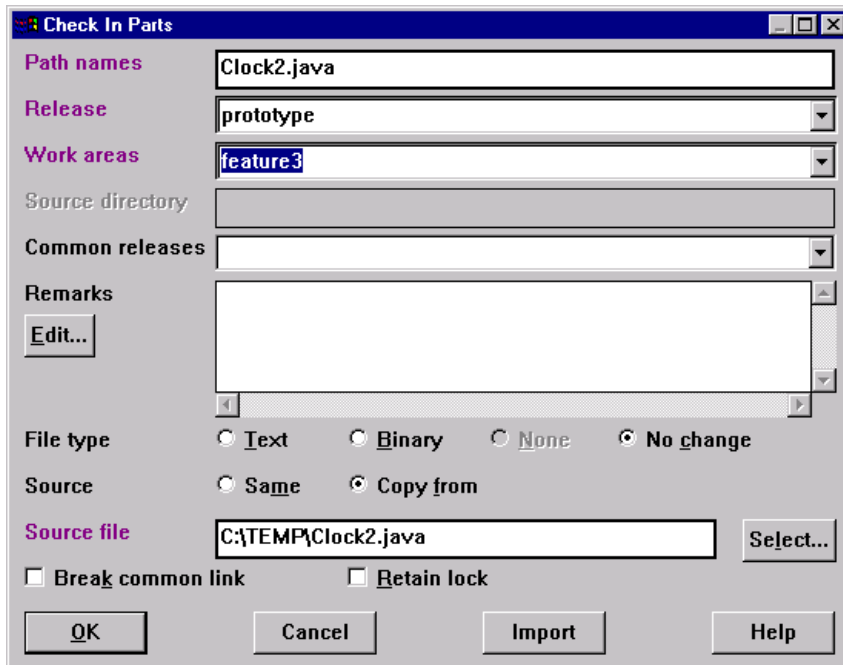


Figure 52. Check-in Parts Window with Dialog Fields

We can now build our Clock2.class from Clock2.java. To do it directly, type the following:

```
teamc Part -build Clock2.class -type TcPart -release prototype  
-workarea feature3 -pool NTpool -normal -verbose
```

The graphical alternative is shown in Figure 53 on page 175.

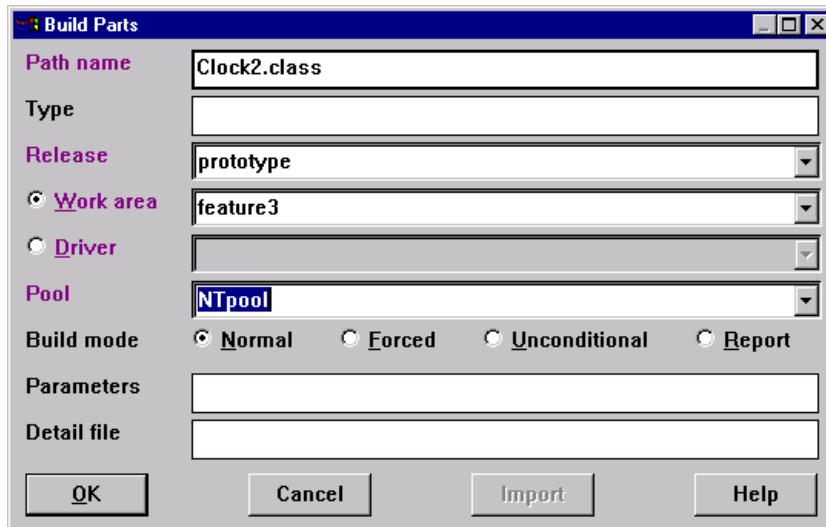


Figure 53. Build Parts Dialog Window

The build can be accomplished in another way that you may find more convenient if you have a complicated project and don't remember the details about all of the parts involved. From the GUI, select **Objects> Parts> BuildView**. The result is shown in Figure 54 on page 176.

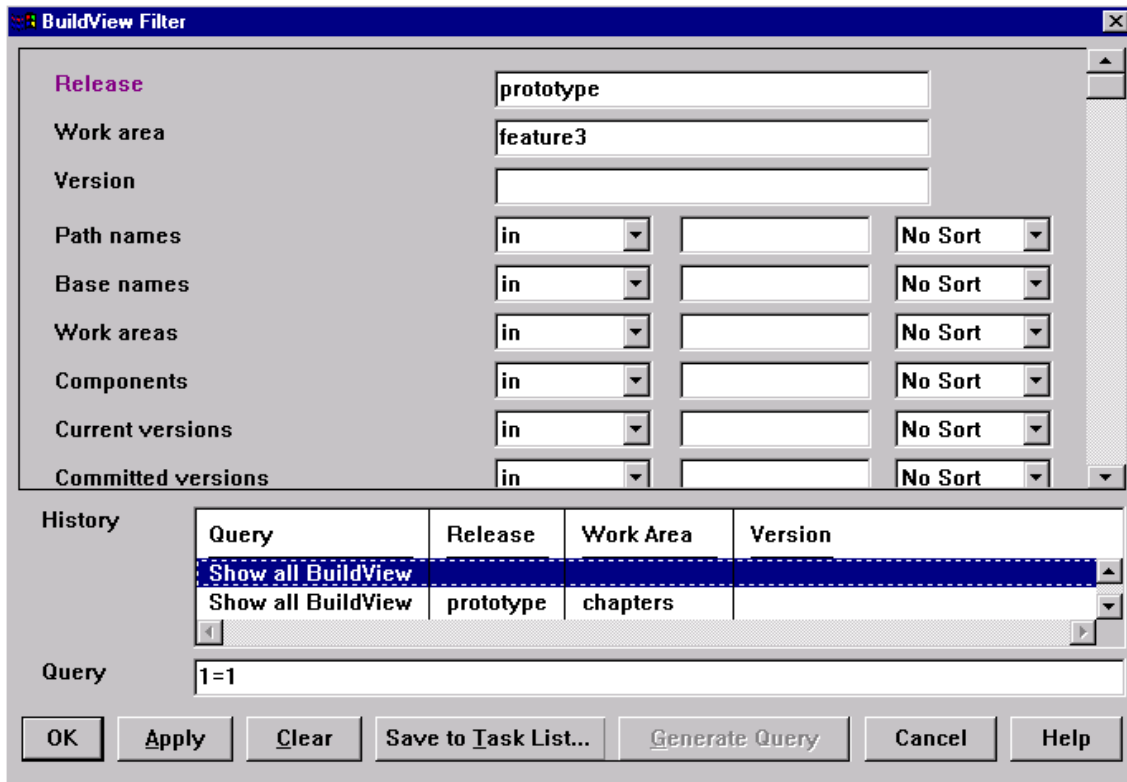


Figure 54. TeamConnection BuildView Screen

You can refine your query on this screen as your needs dictate. We have only one build view for this feature and work area, so we select **Show all BuildView**.

Click on **OK**. Figure 55 on page 177 shows the screen that results from filling in the dialog fields in Figure 54.



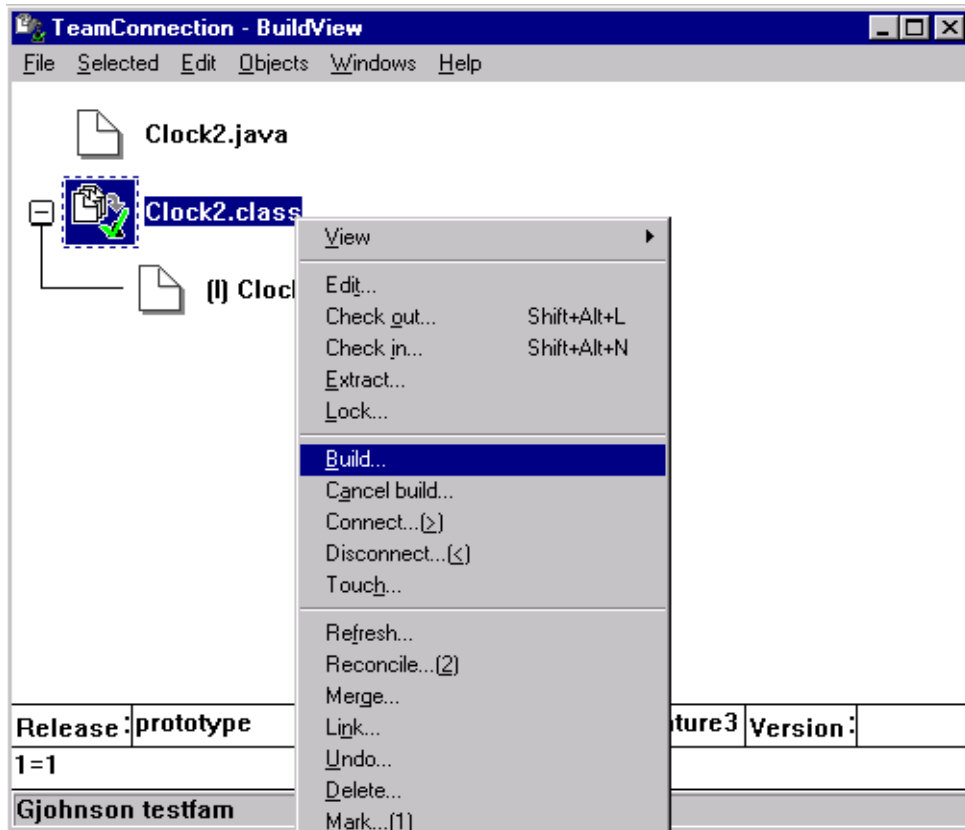


Figure 55. Parts View Screen

After you click on **OK**, the parts view screen with Clock2.java and Clock2.class appears. Click the right-hand mouse button on **Clock2.class**, select **Build** from the pull-down panel, and enter the information in the fields as above. You can examine the status of the build by looking at teamcbld.log:

```

98/10/26 16:22:18 --- Start of Job
-----
98/10/26 16:22:18 Version: testfam|prototype|feature3
98/10/26 16:22:18 Builder: c:\jdk1.1.7\bin\javaTC.exe
98/10/26 16:22:18 Output Files: Clock2.class
98/10/26 16:22:18 Input Files: Clock2.java
98/10/26 16:22:18 Command: c:\jdk1.1.7\bin\javaTC.exe
98/10/26 16:22:24 RC=1 (Expected: == 0) - Failed
98/10/26 16:22:24 Storing the build results + looking for more work

```

```
98/10/26 16:22:26 --- End of Job
```

-----

We see that our build has failed. Upon close inspection we see that `Clock2.class` needs to be modified to include a parameter of `Clock2.java`, after first refreshing it from the work area where the hot-fix work was being performed. Type the following:

```
teamc Part -refresh Clock2.class -type TPart -release prototype
-workarea feature3 -fromworkarea hotfix -verbose
teamc Part -modify Clock2.class -type TPart -release prototype
-workarea feature3 -parameters "Clock2.java" -verbose
teamc Part -build Clock2.class -type TPart -release prototype
-workarea feature3 -pool NTpool -normal -verbose
```

Then, upon inspection of the build log, we see:

```
98/10/26 16:33:45 --- Start of Job
-----
98/10/26 16:33:45 Version: testfam|prototype|feature3
98/10/26 16:33:45 Builder: c:\jdk1.1.7\bin\javaTC.exe
98/10/26 16:33:45 Output Files: Clock2.class
98/10/26 16:33:45 Input Files: Clock2.java
98/10/26 16:33:45 Command: c:\jdk1.1.7\bin\javaTC.exe Clock2.java
98/10/26 16:34:00 RC=0 (Expected: == 0) - Success
98/10/26 16:34:00 Storing the build results + looking for more work
98/10/26 16:34:03 --- End of Job
-----
```

After we have built our `Clock2.class` part, we can extract it from Team-Connection and test it with our favorite Java-enabled Web browser by wrapping it up in this simple HTML code which we call *clock.html*:

```
<html>
<HEAD>
<TITLE>A Clock (1.1)</TITLE>
</HEAD>
<BODY>
<H1>
A Clock (1.1)</H1>
<HR><APPLET codebase="." code="Clock2.class" width=170
height=150>alt="Your
browser understands the &lt;APPLET> tag but isn't running the applet,
for
some reason." Your browser is completely ignoring the &lt;APPLET>
tag!&nbsp;  </APPLET>
<HR><A HREF="Clock2.java">The source</A>.
</BODY>
```

</HTML>

From inside Netscape, the result looks like Figure 56.

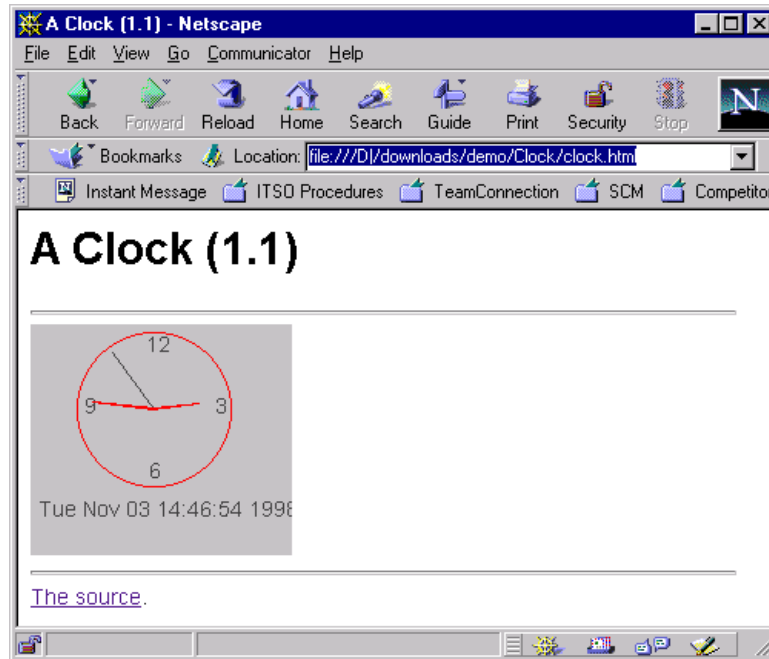


Figure 56. Result of Using the Web Browser

### 16.2.6 Building for Integration

After we are satisfied that our build is performing as designed, we are ready to integrate the work area with the release, then test, verify (and thereby close) the feature. Type in the following:

```
teamc WorkArea -integrate feature3 -release prototype -verbose

teamc Test -accept -workarea feature3 -release prototype -environment
NT -tester Gjohnson -verbose

teamc Feature -verify feature3 -verbose
```

---

### 16.3 An Emergency Hot Fix

This example has several steps, addressed in the following subsections.

### 16.3.1 Working with Defects

We first examine a situation in which your customer has reported a code defect in a production release of a Java applet called *Clock2.class*. Under certain circumstances, some of the colors displayed by the applet could not be distinguished by some customers, so we agreed to change them. To complete the fix on this code, we must do the following:

1. Create a defect record.
2. Accept the task of fixing the defect.
3. Create a work area.
4. Check out the affected parts.
5. Modify, build, and test the part fixes (updates).
6. Integrate the work area.
7. Close the defect fix.

Before we can begin our repair work, we have to take account of several assumptions. First, we already had a preliminary version of *Clock2.java* before we began using TeamConnection for development. This part was first created and checked into TeamConnection all in one step, using a command such as this:

```
teamc Part -create Clock2.java -release prototype -workarea chapters
-component RedBook5260 -text -relative d:\downloads\demo\Clock\
-builder java_comp -translation no -verbose
```

The graphical alternative is shown in Figure 57 on page 181, the **Create Parts** window.

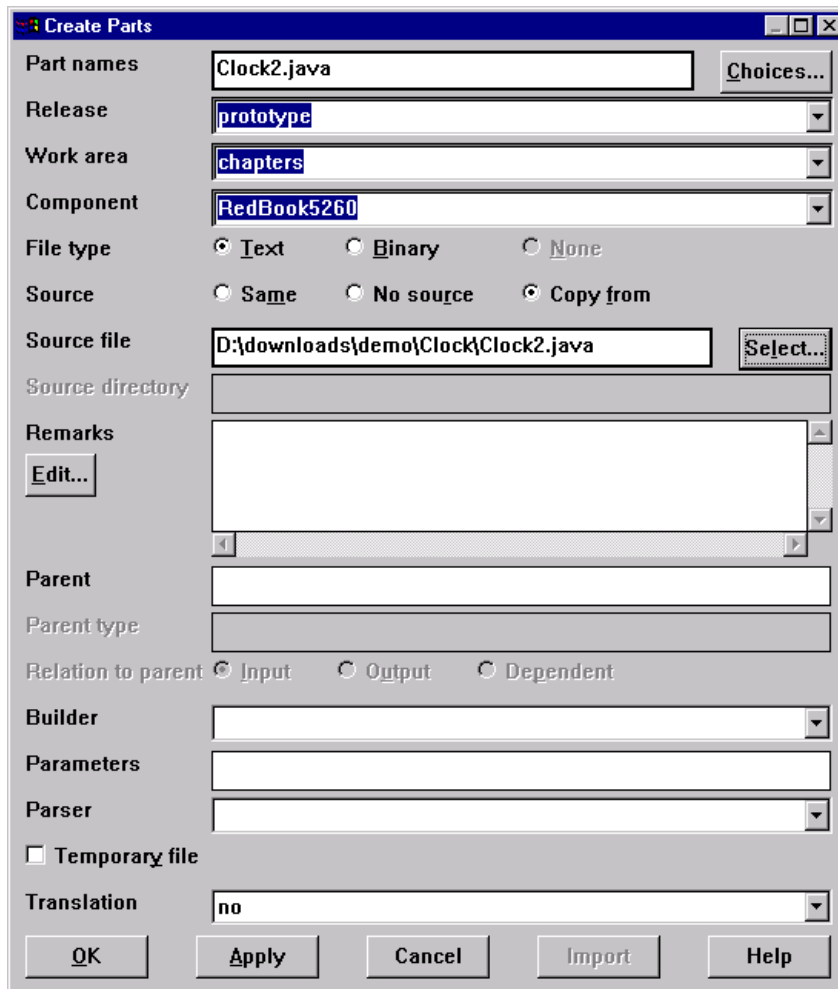


Figure 57. Create Parts Dialog Window

We also have to create a place holder for the .class file resulting from compilation with the Java compiler. To enter the command directly, type in the following:

```
teamc Part -create Clock2.class -release prototype -workarea chapters
        -component RedBook5260 -binary -empty -builder java_comp -translation
        no -verbose
```

The GUI alternative, Figure 58 on page 182, shows the **Create Parts** window we use when creating the Clock2.class from Clock2.java.

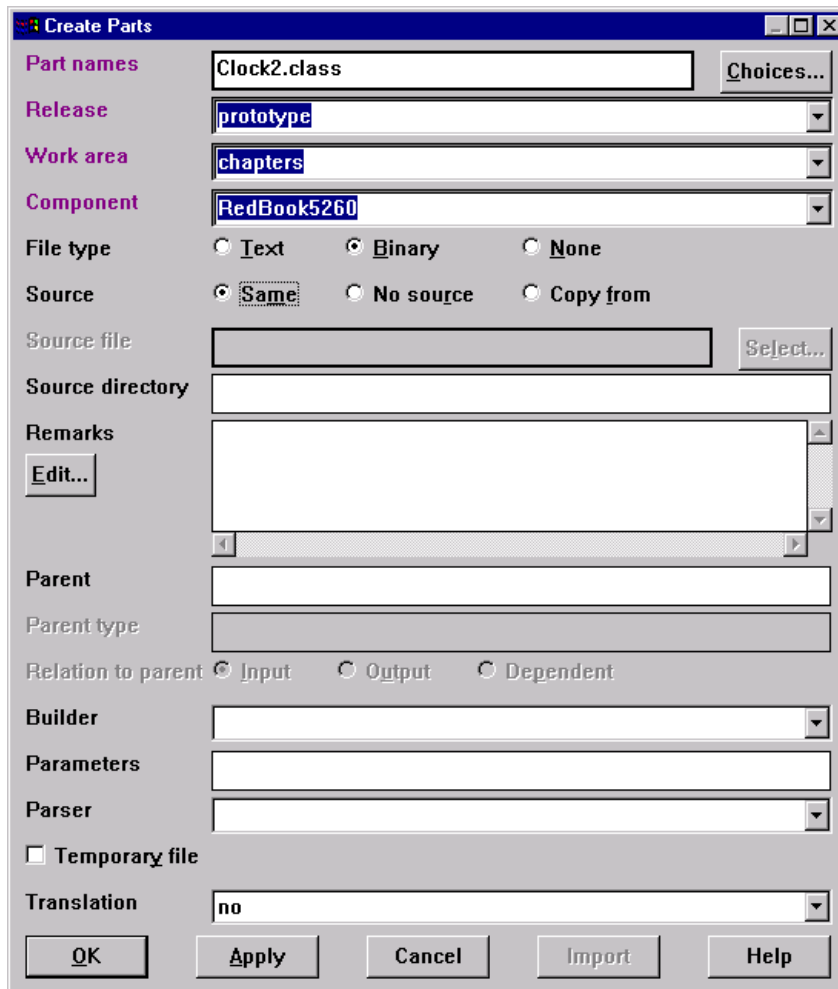


Figure 58. The Dialog Window that Creates the New Clock2.class Parts

After our Clock2.java and Clock2.class parts have been created, they must be connected from the perspective of TeamConnection. You can type the following:

```
teamc Part -connect Clock2.java -release prototype -workarea hotfix
-parent Clock2.class -input -verbose
```

The GUI alternative is shown in Figure 59..

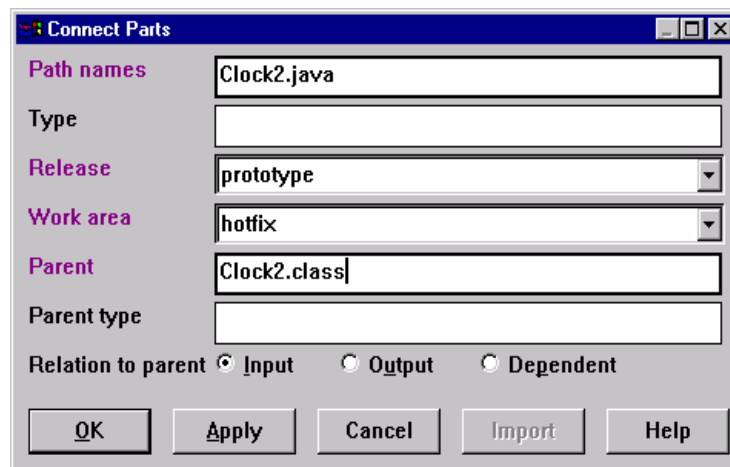


Figure 59. The Connect Parts Window, How Clock2.java Relates to Clock2.class

### 16.3.2 Opening a Defect Fix

We can open our defective component by typing in the following command:

```
teamc Defect -open -component RedBook5260 -remarks - -name 2 -release  
prototype -environment NT -prefix c -severity 1 -symptom design_wrong  
-phaseFound inspections -verbose
```

The TeamConnection GUI alternative is shown in Figure 60. Enter information in the fields as shown, and click on **OK**.

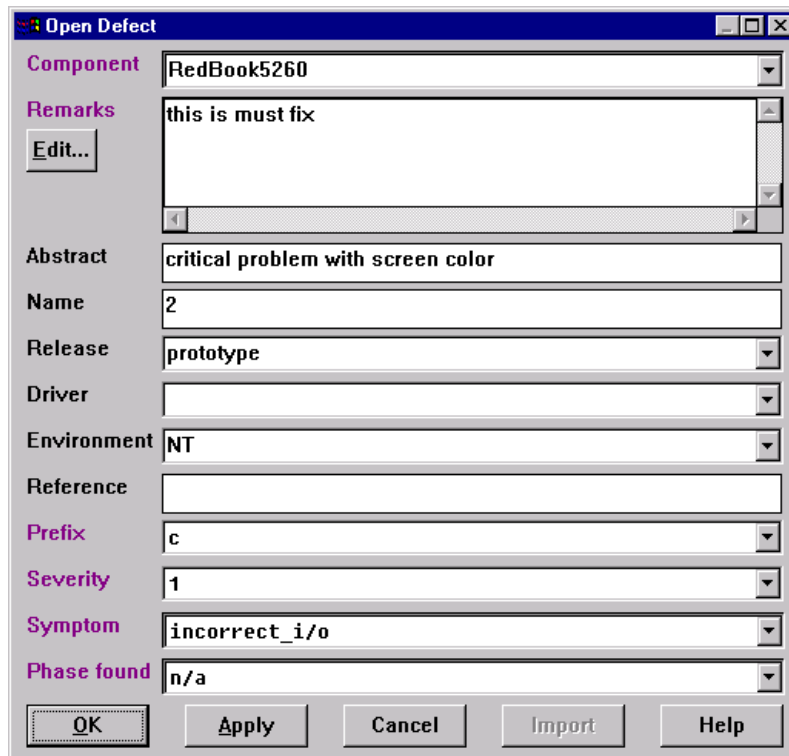


Figure 60. Open Defect Window Showing Fields with Entries

The defect must be accepted. To do so directly, type in the following:

```
teamc Defect -accept 2 -answer program_defect -remarks - -verbose
```

The alternative is the TeamConnection GUI shown in Figure 61 on page 185. Enter the information in the fields as shown and then click on **OK**.



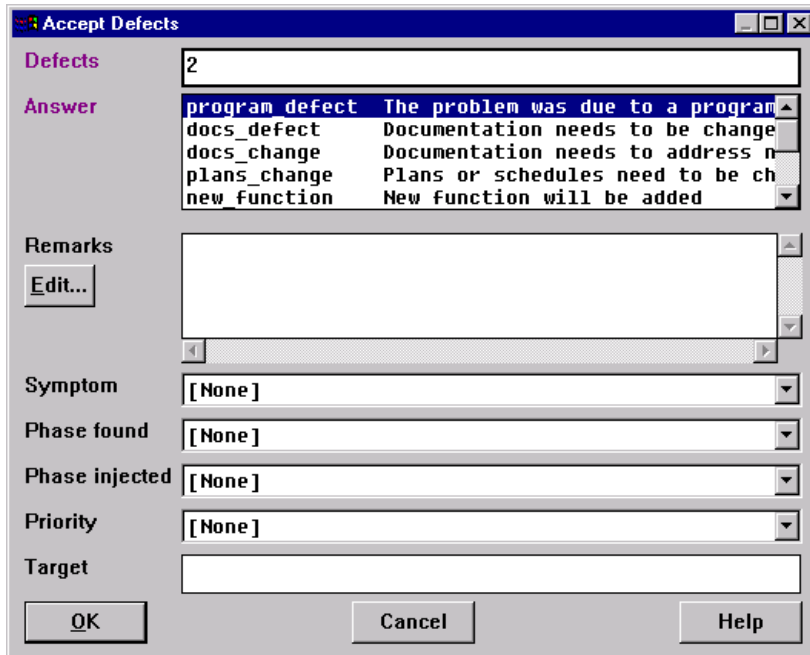


Figure 61. Accept Defects Window

### 16.3.3 Creating a Work Area

You can either type in the following:

```
teamc WorkArea -create -name hotfix -defect 2 -release prototype
-target 10/31/1998 -owner Gjohnson -verbose
```

or you can use the TeamConnection GUI shown in Figure 62 on page 186. Enter information in the fields as shown and click on **OK**.

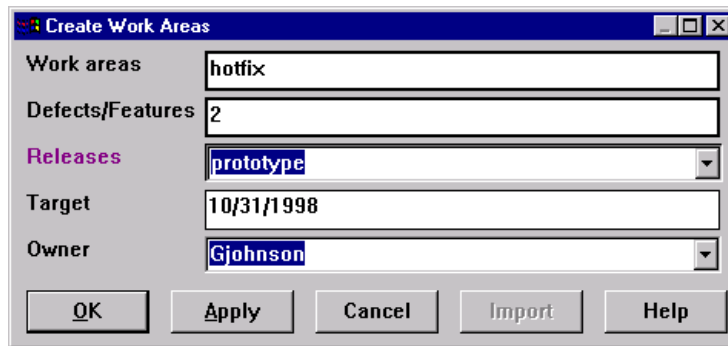


Figure 62. Create Work Areas Window

### 16.3.4 Parts Check-Out Process

The direct command is as follows:

```
teamc Part -checkout Clock2.java -release prototype -workarea hotfix
-verbose
```

The graphical alternative is shown in Figure 63. Enter information in the fields as shown, and click on **OK**.

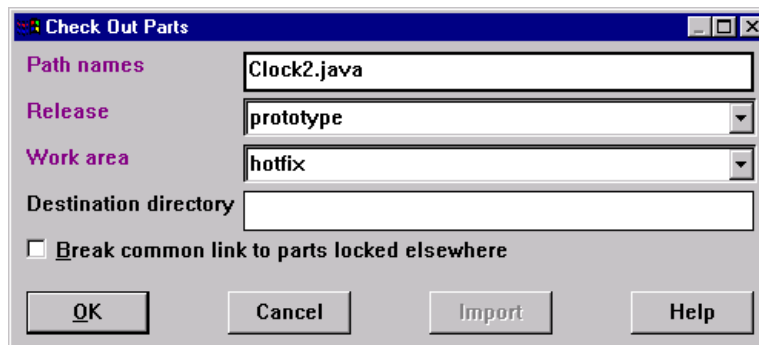


Figure 63. Check Out Parts Window

We are now ready to make the appropriate changes to the the colors within Clock2.java

### 16.3.5 Modify, Check-in, Build, Test Part Fixes (Updates)

Before we begin the build, the part must be checked back into Team-Connection.

To do it directly, type in the following:

```
teamc Part -checkin Clock2.java -release prototype -workarea hotfix  
-verbose
```

As an alternative, use the window shown in Figure 64.

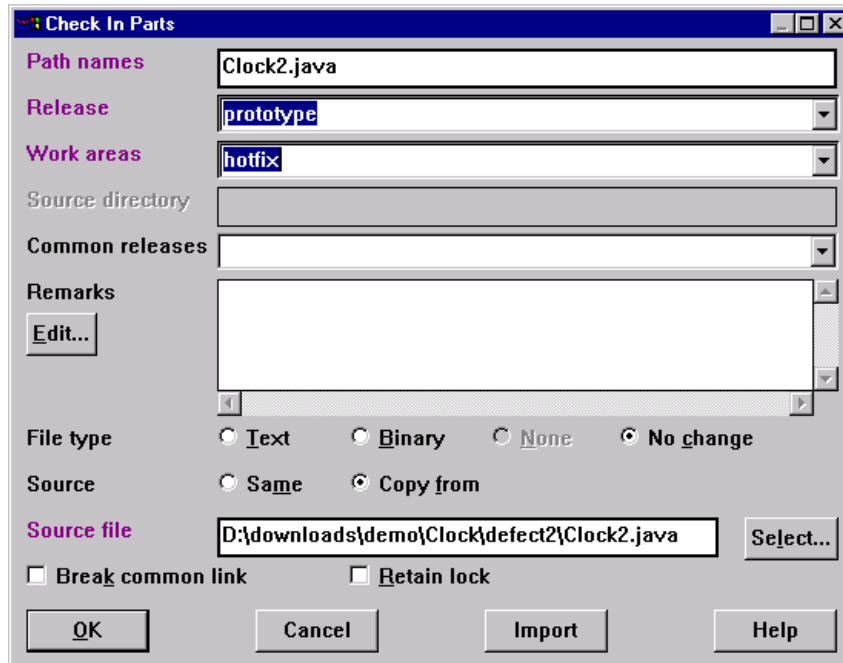


Figure 64. Check-in Parts Window

Note that the part is checked into the hotfix work area and is visible only from within this work area. Because the hotfix work area contains the Clock2.class from the production build, it does not reflect changes that were just made to Clock2.java

To build Clock2.class from Clock2.java make sure you have access to a running build server. (we started ours with this command:)

```
teamcbld -f testfam -e- NT -p NTpool
```

This build server was running on the same machine as the family server. We can however be even more flexible and place our build server on a separate NT machine. For illustration purposes, we tried this with the user Admin on the machine malawi.almaden.ibm.com, as follows:

```
teamcbld -e NT -p NTpool2 -f testfam@antimony.almaden.ibm.com.@9001 -b
Gjohnson
```

In order for this build server to establish communications with the testfam TeamConnection server, we were required to define a host record for the user Admin@malawi.almade.ibm.com. We did so by entering

```
teamc Host -create Admin@malawi.almaden.ibm.com -login Gjohnson
-verbose
```

For this particular example, we installed Sun's Java Development Kit (JDK) Version 1.1.7 for NT. This is available on the Internet at

<http://www.javasoft.com/products/jdk/1.1/>

on our machines running the build servers.

At this point, we can choose to type in the following:

```
teamc Builder -create java_comp -release prototype -script
\jdk1.1.7\bin\javaTC.exe -environment NT condition "==" -value 0 -binary
-verbose
```

or we can choose to use the graphical interface shown in Figure 65.

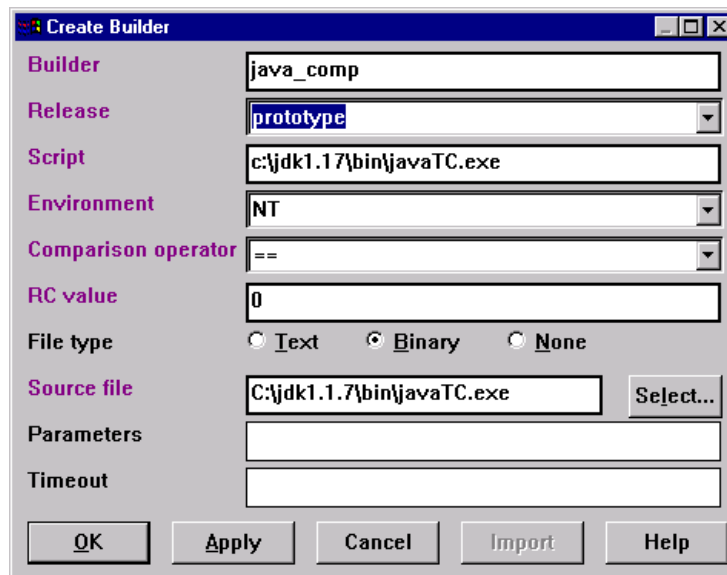


Figure 65. Create Builder Window with Relevant Fields Filled in

Note that when the builder is created, the contents of javaTC.exe are stored in the TeamConnection database. When the builder is invoked, TeamConnection creates a file named javaTC.exe and executes it.

Now, finally you can build your Clock2.class file with this command:

```
teamc Part -build Clock2.class -type TcPart -release prototype
-workarea hotfix -pool NTpool -force -verbose
```

If you would like to initiate a build on the secondary build server, simply reference NTpool2 as the build pool in your build command:

```
teamc Part -build Clock2.class -type TcPart -release prototype
-workarea hotfix -pool NTpool2 -force -verbose
```

The first build statement references a build server that happens to be on the same machine as our family server. The second build statement references a build server on a machine that is different from the TeamConnection family machine.

The alternative is to use the window shown in Figure 66.

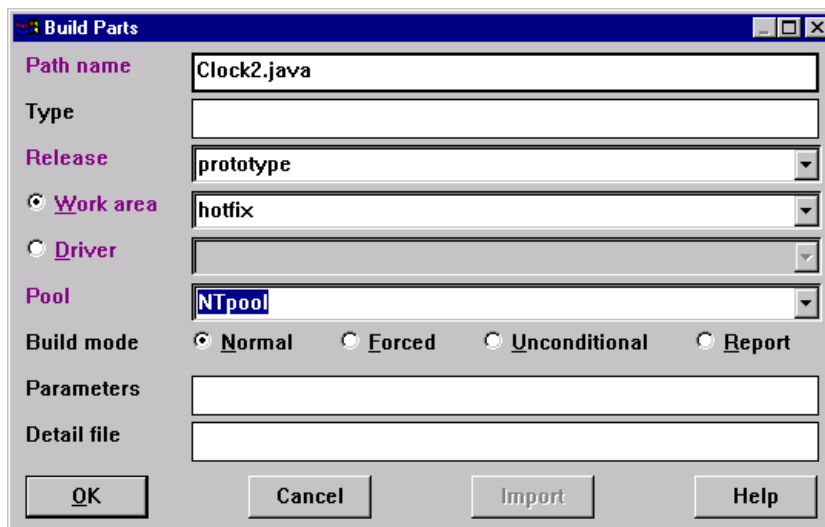


Figure 66. Build Parts Window with Relevant Fields Filled

After you complete your build, examine the contents of the teamcbuild.log. You should see an entry similar to the following:

```
98/10/21 10:36:08 --- Start of Job
-----
98/10/21 10:36:08 Version: testfam|prototype|defect2
98/10/21 10:36:08 Builder: c:\jdk1.1.7\bin\javaTC.exe
```

```
98/10/21 10:36:08 Output Files: Clock2.class
98/10/21 10:36:08 Input Files: Clock2.java
98/10/21 10:36:08 Command: c:\jdk1.1.7\bin\javaTC.exe Clock2.java
98/10/21 10:36:20 RC=0 (Expected: == 0) - Success
98/10/21 10:36:20 Storing the build results + looking for more work
98/10/21 10:36:23 --- End of Job
-----
```

It is important to take notice of the line in teamcbld.log

```
98/10/21 10:36:08 Command: c:\jdk1.1.7\bin\javaTC.exe Clock2.java
```

which contains the command for the Java compiler. The parameter Clock2.java is defined in the part definition of Clock2.class. This is equivalent to running the java compiler at the command line with

```
javac Clock2.java
```

The work area now contains the Clock2.class that reflects the changes made in Clock2.java. It may now be extracted and tested with your favorite Java-enabled Web browser, invoked by the following:

```
teamc Part -extract Clock2.class -release prototype
-workarea hotfix -verbose
```

### 16.3.6 Integrating the Work Area

To integrate the fix, type in the following:

```
teamc Fix -complete -workarea hotfix -release prototype
-component RedBook5260 -verbose
teamc WorkArea -integrate hotfix -release prototype -verbose
```

### 16.3.7 Resolving the Conflicts

In this type of scenario, where independent modifications need to be made to one or more parts, it can be useful to have a tool to help resolve the differences. The TCMERGE tool can be activated from the tasks section of the GUI shown in Figure 47 by:

```
Actions - Parts - Show -Differences
```

A Java-based dialog box appears in which you can indicate the locations of the particular parts whose differences need resolution.

---

## 16.4 Recapitulation

In this particular case, the Java class file is derived from a single source file, the Clock2.java file. The project to create this clock program was begun

before TeamConnection was used for development so Clock2.java was created and checked into TeamConnection from

```
d:\downloads\demo\clock\Clock2.java.
```

An empty binary file Clock2.java was created and checked into TeamConnection in a similar fashion. The Clock2.class was connected to Clock2.java with Clock2.class as the parent. In order to use TeamConnection to build a Java class file, the Sun JDK must be installed on a builder machine. This can be (but does not have to be) the same machine where the TeamConnection family resides and is called *NTpool* in our particular case.

The build server must be started prior to the triggering of the build process; it can be started with the `teamcbld` command. A builder must be created in TeamConnection (`java_comp` in our case) which references the location of the JDK java compiler (in our case `c:\jdk1.1.7\bin\javaTC.exe`).

The location of the Java compiler is placed in the **Script** field of the create or modify builder screen. The compiler returns a code of 0 after successful compilation so be sure to enter 0 in the **RC** field and == in the **Comparison** field.

One additional step is required before our build environment is suitable for constructing Clock2.class from Clock2.java. Make sure that Clock2.class includes Clock2.java as a parameter. Without that, our build process does not know where to find the source file for compilation.

We are now ready to use TeamConnection to build our Java class file. From the GUI tasks panel (Figure 47 on page 58), select **Objects, Parts, BuildView**, and query for your release and work area.

From the **BuildView** screen, Clock2.java and Clock2.class should be visible. Right-click the mouse on Clock2.class and click on **Build**, select **NTpool** as the pool, and watch for any error messages.

Your Clock2.class file is now stored in TeamConnection and can be extracted, tested, packaged, deployed, and so on. One important note: The source that is used for the build is the the one that is stored in the TeamConnection database, not the one on the local drive; you must remember to check it in.





---

## 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 TeamConnection 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 Team-Connection and a transform tool. The build script invokes the tool. Because of the way TeamConnection uses build scripts to invoke tools, Team-Connection 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++/](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. Naming Restrictions for DB2 User IDs

---

### B.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 C. Uninstalling DB2 UDB

This appendix shows you how to uninstall (remove) DB2 products.

---

### C.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.

#### C.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.

#### C.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.

#### C.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.

#### **C.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.

#### **C.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.

#### **C.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.

## C.1.7 Remove the DB2 Products

The following steps describe how you can remove DB2 products from UNIX operating systems.

### C.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
```

### C.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.

### C.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.

---

## C.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.

---

### C.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 Center**.
  - 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 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 230.

---

### 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 window, 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.Id* file. This section provides instructions for manually editing the *authorit.Id* file to create new authority groups or change information about existing authority groups. When you change the *authorit.Id* 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. Special Notices

This publication is intended to help customers and IBM technical representatives to understand and master the build process 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

them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following document contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples contain the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries

IBM ®  
AIX  
DB2  
DB2 Universal Database  
MVS  
OS/2  
TeamConnection  
VisualAge

The following terms are trademarks of other companies

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names may be trademarks or service marks of others.





---

## Appendix F. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

---

### F.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 239.

- SG24-4648 Introduction to the IBM Application Development Team Suite
- SG26-2008 Family Planning and Application Development

---

### F.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	

---

### F.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:

	<b>IBMMAIL</b>	<b>Internet</b>
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 Publications Customer Support P.O. Box 29570 Raleigh, NC 27626-0570 USA	IBM Publications 144-4th Avenue, S.W. Calgary, Alberta T2P 3N5 Canada	IBM Direct Services Sortemosevej 21 DK-3450 Allerød Denmark
--	--	--

- **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	<a href="http://www.redbooks.ibm.com">http://www.redbooks.ibm.com</a>
IBM Direct Publications Catalog	<a href="http://www.elink.ibm.link.ibm.com/pbl/pbl">http://www.elink.ibm.link.ibm.com/pbl/pbl</a>

### 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.





---

## 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 TeamConnection 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/WINDOWS 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.

**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.

**job queue.** A queue of build scopes. One job queue exists for each TeamConnection family.

**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.

**metadata.** In databases, data that describe data objects.

**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.

L

M

N

J

## 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 TeamConnection 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 administrator, 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 approximate 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 component 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 TeamConnection 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

<b>3GL</b>	Third Level Graphical Language	<b>UDF</b>	DB2 user-defined functions
<b>4GL</b>	Fourth Level Graphical Language	<b>Y2K</b>	Year 2000
<b>APA</b>	All Points Addressable		
<b>API</b>	Application Programming interface		
<b>CMSC</b>	Configuration Management and Software Control		
<b>CMVC</b>	International Technical Support Organization		
<b>DLL</b>	dynamic link library		
<b>ECU</b>	European Currency Unit		
<b>FTP</b>	File Transfer Protocol		
<b>GUI</b>	Graphical User Interface		
<b>HTML</b>	Hypertext Markup Language		
<b>IBM</b>	International Business Machines Corporation		
<b>IT</b>	Information Technology		
<b>ITSO</b>	International Technical Support Organization		
<b>JDK</b>	Java Developer's Kit		
<b>NLS</b>	National Language Support		
<b>OS</b>	Operating System		
<b>SCMS</b>	Software Configuration Management Services		
<b>SEI</b>	Software Engineering Institute		
<b>TUI</b>	Text User Interface		
<b>UDB</b>	Universal Database		





---

## Index

### Symbols

%etc% 54  
/etc/hosts 54  
/etc/services 54, 65  
/home/db2inst1 66  
/home/tcfamily 66  
/usr/lpp/db2\_05\_00 66  
/usr/teamc 66

### Numerics

3GL 14

### A

abbreviations 251  
accept defect 33  
accept version 36  
access  
    concurrent 5  
    distributed 5  
Acrobat Reader 39  
acronyms 251  
administrator 20  
    network 53  
    TeamConnection 53  
Adobe Acrobat Reader 39  
AIX 61  
alias 53, 55  
application  
    architecture 15  
    class hierarchy 23  
    configurations 4  
    consistency 11  
    corequisite 14  
    design 27  
    development  
        change management 9  
        configuration management 9  
    function 18  
    instance 12  
    loss of coherence 8  
    prerequisite 14  
    type 27  
    version 10, 12  
    versions 4  
assign defect 33

associated driver 37  
automatization 6  
    common development tasks 6

### B

backup 6  
backup workarea 36  
baseline 29  
bottom-up approach 9  
bug  
    defect 10  
    fix 24, 35  
build 16  
    (run a) 31  
    distributed 19, 20  
    include file 3, 14  
    integrated 6  
    parallel 20  
    part 18, 35  
    process 5, 12, 14, 28  
    repeatable 6  
    server 19  
    socket 55  
    status 30  
    tree 10, 29, 30  
builder 19, 28

### C

cancel\_work\_area 30  
CC clients 19  
change  
    manage 29  
    promote 34, 35  
    propagate 25  
    reconciliation 25  
change control 4, 6  
    process 5  
change management 3, 9, 28, 29  
change request 15  
    assign 15  
    attribute 15  
    build 16  
    create a workspace 16  
    distribute the change 16  
    form identifier 15  
    impact analysis 15

- monitoring the change 16
  - ownership field 15
  - promote change 16
- check-in 18, 30, 34, 36
- check-out 18, 30, 34, 36
- class hierarchy 23
- client/server 3, 4
- client/server technology 3
- CMVC 39
- coarse-grain 5
- collector 35
- collision record 36, 37
- common development tasks 6
- compilation
  - debug option 14
  - include file 3, 14
  - test phase 14
- component 18
  - reuse 3
- composite objects 10
- composite version 36
- concurrent access 5
- configuration 10
- configuration control 10
- configuration management 5, 9
- configuration management service 4
- consistency 11
- constraint checking 5
- controlling changes 4
- corequisite 14
- create
  - part 18
  - workarea 18, 33, 37
- cross-compiler 14
- cross-dependency 14, 23
- cross-impact analysis 23
- cross-reference 14, 23
- cycle
  - production 22

**D**

- daemon 39
- data
  - access control 14
  - cross-dependency 14, 23
  - TeamConnection part 10
- data constraints 6
- data exchange 4, 5
- data sharing 5
- DB2 5, 40, 71
  - administration server 65, 67
  - administration server instance 71
  - administrator server 62
  - control center 62, 65
  - fenced area 71
  - fenced user id 65, 67, 71
  - file system 74
  - fixes 63
  - general considerations 65
  - group id 71
  - installation image 63
  - installer 61, 71
  - instance 61, 65, 66, 67
    - parameter 61, 74
  - instances 62
  - parameter values 74
  - Performance SmartGuide 68
  - space used 73
  - tools 62
  - user id 71
  - user-defined functions 71
  - Workgroup Edition 71
- DB2 instance 71
- DB2 Personal Edition 40
- DB2 Quick Beginnings manual 40
- DB2 requirement 39
- DB2 UDB 40
- DB2 Workgroup Edition 40
- DB2\_DBPATH 68
- DB2\_HOME 67, 72
- DB2DBDFT 68
- DB2INSTANCE 67
- DB2INSTANCE\_HOME 67
- defect 10, 18, 30
  - accept 33
  - assign 33
  - open 33
  - software 8
- defect workarea 35
- delete
  - part 18, 34
- deliverable 15
- dependency 30
  - cross-impact 23
  - local impact 23
- dependency scanner 19
- DependentOf 30

- development cycle
  - coding phase 14
- development data 4
- development environment 21, 27
- development model 4
- distribute change 16
- distributed access 5
- distributed build 19, 20
- distributed computing 3
- distributed object 14, 23
- distribution 15
  - deliverable 15
  - package 15
- distribution support 6
- DLL 14, 23, 24
- DM/2 CC server 19
- document
  - template 14
- documentation
  - TeamConnection part 10
- driver 10, 18, 28, 29, 34
  - association 37
  - member 37
  - test 35
- dynamic link library 14, 23, 24

**E**

- ECU 10
- environment
  - development 21
  - development 27
  - production 28
  - scalable 6
  - test 28
- estimate work load 32
- etc/hosts 53
- etc/services 53
- European Currency Unit 10
- evolution
  - feature 10
- extensible information model 5
- extract version 36

**F**

- fake build 30
- feature 10, 18, 30
- fine-grain 4, 5
- fine-grained development objects 4

- fix
  - bug 24, 35
  - hot fix 24, 25, 33, 37
  - integration 37
  - propagate 23
  - propagation 11, 25, 35
  - status 37
- fix process 33
- FixPak U453783 40
- freeze workarea 34
- FTP 53
- function call 23

## G

- Gather tool 19

## H

- hardware requirement 39
- hardware requirements
  - AIX client 41
  - AIX server 40
  - HP-UX client 44
  - HP-UX server 43
  - OS/2 client 48
  - OS/2 server 47
  - Solaris client 46
  - Solaris server 45
  - Windows client 51
  - Windows server 50
- host 55
- hot fix 24, 25, 33, 37
- HP
  - sam 54
- HP-UX 61

## I

- impact analysis 4, 15, 23, 24, 25, 32
- include file 3, 14
- independent subsystem 14
- information model 5, 6
  - extensible architecture 5
  - schema evolution 6
- InputOf 30
- InputTo 30
- instance 12, 53
- integrate
  - fix 37

- state 37
- workarea 20
- integrated build 6
- integrated tools
  - data sharing 5
- integrated workarea 34, 37
- interface 14
- Internet Explorer 12
- IP address 55
- ISO 9000 3

## J

- Java Development Kit 39

## L

- LAN 3
  - topology 5
- language
  - 3GL 14
  - cross-reference 14
  - object oriented 14
- legacy application 13, 20
- link
  - part 18
  - release 18
- list of impacted parts 30
- local area network 3, 5
- local impact analysis 23, 29, 30
- loss of coherence 8

## M

- make file 24
- manage changes 3
- merge 25, 26, 27
  - change 11
  - collision record 36, 37
  - tool 36
- network
  - socket number 53
- migrating your database 39
- modify properties 34
- monitor change 16

## N

- name server 53, 54
- national language support 12
- Netscape 12, 39

- network 39
  - administrator 53
  - alias 53, 55
  - build socket 55
  - FTP 53
  - host command 55
  - name server 53
  - nslookup command 56
  - ping command 56
  - protocol 14, 65
  - tchost command 55
  - tchostw command 55
  - telnet 53

- NLS 12

- nondisruptive evolution 4
- none part 35
- nonregression test 24
- nslookup 56
- NVBridge 19

## O

- object
  - class 14
  - distributed 14
- object class 14
- object-oriented 3
  - database 5
  - information model 5
  - language 14
  - repository 6
  - repository services 4
- ObjectStore 73
  - uninstall 73
- open defect 33
- open information model 5
- operating system 14
- OS 14
- OS/2
  - DB2 installation 62
- OS/2 Warp 61
- out\_of\_date 30
- OutputOf 30

## P

- package 15, 20
- packaging 4, 6, 15
- parallel build 20
- parser 19, 28, 30

- dependency check 30
- part 10
  - build 18
  - check-in 18, 34
  - check-out 18, 34, 36
  - create 18
  - delete 18, 34
  - link 18
  - modify properties 34
  - recreate 34
  - rename 18
- part inconsistency 36
- PartFull 32
- parts
  - check-in 36
- ping 56
- prerequisite 14
- problem tracking 4, 6
- process 27
  - build 12, 14, 28
  - change control 5
  - fix 33
  - loss of coherence 8
  - propagation 36
  - reliable 6
  - version control 5
- production 14, 22
- production environment 28
- production version 27
- project management 4, 27
- project status 4
- promote change 34, 35
- propagate
  - change 25
  - fix 11, 23
- propagation process 36
- protocol 14, 39

## Q

- quality analysis 4

## R

- rebuild 4
- rebuild application 4
- reconcile version 36
- reconciliation 27
- recovery 6
- recreate part 34

- refresh 34
  - workarea 18
- refresh workarea 36, 37
- reject version 36
- related data 10
- relationship of TC families to DB2 instances 66
- release 10, 18, 28, 30
  - link 18
    - management 5
- reliable process 6
- rename
  - part 18
- repeatable build 6
- report 19
- report mode 30
- report mode "ON" 30
- reporting 6
- repository 4
  - services 5
  - technology 6
- request
  - change 15
- requirement
  - MVS build server 52
  - national language support 12
  - NLS 12
  - software 8
- reuse 3
- role-based access 4
- roles
  - authority 4
- roll back 34

## S

- sam 54
- secure repository 4
- SEI 3, 4
- smit 53
- socket 39
- software
  - configuration management 4
  - configuration management services 6
  - defect 8
  - development
    - bottom-up approach 9
    - top-down approach 9
  - requirement 8
- Software Engineering Institute 3, 4

- software requirement 39
- software requirements
  - AIX 42
  - HP-UX 44
  - OS/2 49
  - Solaris 47
  - Windows 51
- Solaris 61
- source file
  - TeamConnection part 10
- source of refresh 34
- standardized procedure 18
- static library 14, 24
- subsystem
  - interface 14
- successor 30
- sybsystem
  - independance 14
- system32/drivers/etc 54

## T

- TC\_DBPATH 67
- tchost 55
- tchostw 55
- tcinst.ksh 72
- TCMERGE 36
- TCP/IP 14, 39, 65
  - address 39, 55
  - alias 53, 55
  - name server 53, 54
  - OS/2 54
  - port 39
  - socket number 53
  - windows 54
- team
  - data sharing 5
  - programming environment 4
- TeamConnection
  - administrator 20, 53
  - automatization 6
  - backup 6
  - build 19, 20, 28, 30, 31, 35
  - build process 12
  - build server 19
  - build status 30
  - build tree 10, 29, 30
  - builder 19, 28
  - change control 6

- check-in 36
- check-out 36
- collector 35
- component 18
- configuration control 10
- configuration management 5
- data constraints 6
- defect 10, 18, 30, 33, 35
- distribution 6
- driver 10, 18, 28, 34, 35, 37
- fake build 30
- feature 10, 18, 30
- freeze 34
- Gather 19
- impact analysis 23, 24, 25
- information model 6
- integrated build 6
- merge 25, 26, 27, 36, 37
- none part 35
- object-oriented 6
- objects 7
- packaging 6, 20
- parser 19, 28, 30
- part 10
  - part check-in 30
  - part check-out 30
  - part inconsistency 36
  - part status 30
  - part\_build 30
  - PartFull 32
- problem tracking 6
- process 6, 27
- project management 27
- propagate change 25
- propagate fix 23
- recovery 6
- refresh 34, 37
- release 10, 18, 28, 30
- release management 5
- repeatable build 6
- report 19
- reporting function 6
- repository 6
- scalable environment 6
- scanner 19
- schema evolution 6
- SCMS 6
- software configuration management services 6
- standardized procedure 18

- touch part 30, 31
- transparency 17
- undo 34
- version control 5, 10
- work context 30
- workarea 18, 20, 29, 30, 33, 34
- workspace 18
- TeamConnection daemon 39
- TeamConnection Web site 39
- telnet 53
- temporary workarea 30, 31
- test database 14
- test driver 35
- test environment 28
- Tivoli
  - NVBridge 19
  - Tivoli/Courrier 19
- TME 10 19
- top-down approach 9
- touch part 30, 31
- touch\_part 30
- track 4
- transaction manager 14
- transparency 17

## U

- UDB 5, 40
  - Workgroup Edition 71
- UDF 71
- undo 34
- undo workarea 32, 34
- undo\_work\_area 30
- Universal Database 5
  - general considerations 61
- Universal Database Quick Beginnings manual 40
- UNIX
  - DB2 installation 61
  - file system 67
  - smit 53
  - tcinst.ksh 72
- update domain 16

## V

- version control 5, 10
- version management 5

## W

- web browser 12
- Windows NT 61
  - DB2 installation 62
- work context 30
- workarea 18, 29, 34
  - backup 36
  - cancel 30
  - create 18, 33, 37
  - defect 35
  - freeze 34
  - integrate 20, 34
  - refresh 18, 34, 36, 37
  - undo 30, 32, 34
- workarea vanishes 30
- workspace 16, 18

## Y

- Y2K 10
- Year-2000 10





---

# ITSO Redbook Evaluation

VisualAge TeamConnection Enterprise Server Keeping Your Build Process on a Short Leash  
SG24-5260-00

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](mailto: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!)**

---

---

---

---

