Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

By reading and using the thesis, the reader understands and agrees to the following terms:

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.

2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.

3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact libsys@polyu.edu.hk providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.
An Architecture for Web-Based
Distributed Product Data Management

by

LEONG Ka Kuan

A thesis submitted to
The Hong Kong Polytechnic University
in accordance with the regulations for the degree of
Master of Philosophy

Department of Industrial and Systems Engineering
The Hong Kong Polytechnic University

2002
ABSTRACT

Product data management (PDM) has been accepted as an important tool for the manufacturing industries. In recent years, more and more researches have been conducted in the development of PDM. Their research areas include system design, integration of object-oriented technology, data distribution, collaborative and distributed manufacturing working environment, security, and web-based integration. However, there are limitations on their researches. In particular, they cannot cater for PDM in distributed manufacturing environment. This thesis, thus, attempts to contribute a new PDM solution for the global manufacturing era.

In system design, development and implementation, a formal specification is necessary. However, there is no formal representation model for PDM system, therefore a graphical representation model is constructed to express the various scenarios of interactions between users and the PDM system. Statechart is then used to model the operations of PDM system. Statechart model bridges the current gap between requirements, scenarios, and the initial design specifications of PDM system. Afterproperly analyzing the PDM system, a new distributed PDM (DPDM) system is proposed. Both graphical representation and statechart models are constructed for the new DPDM system. New product data of DPDM and new system functions are then investigated to support product information flow in the new distributed environment.
A popular Object Modeling Technique (OMT) is used to model the object characteristic domain of product data. The proposed product data model explains the relationship between the product data and the various activities in the new DPDM system. It presents a product data modeling framework for DPDM system. The DPDM product data model includes object model, functional model and dynamic model.

Since the allocation of object in a distributed manufacturing environment is a critical issue, therefore object allocation objectives for DPDM are set to facilitate object distribution decision making. In addition, a comprehensive mathematical model is developed to minimize the DPDM system operation cost, response time, remote object request rate, and storage cost. Replication and migration techniques for product data object distribution in the new DPDM system are also discussed.

Since there are limitations in existing PDM system for the contemporary distributed and collaborative environment, a categorized stratified workspace architecture is proposed. The categorized features of the stratified workspaces can bring major benefits to security and version management. In addition, how stratified workspace can improve and increase both local and remote data sharing when integrated with web technologies is discussed.

In dealing with the tremendous amount of interactions between the DPDM system and the various users, the utmost importance is to ensure that all data are secured and all users are under controlled and managed. The prominent supporting features in DPDM system
such as user organization, workspace and security are thus analyzed. A mixed approach access model for the system is proposed. In this model, the two main classical access control schemas, the Lampson’s access matrix and the Bell-LaPadula’s (BLP) security labels, are adapted for DPDM purpose.

The implementation aspects of the new DPDM system are then highlighted. The limitations of the statechart modeling method, product data model, object-oriented technology, product data allocation, stratified workspace and system security in DPDM system are discussed. Future research directions are also proposed.

Finally, the thesis is concluded by summarizing the achievements developed for the proposed DPDM system.
ACKNOWLEDGEMENTS

I would like to express my deepest and sincere gratitude to my chief supervisor Dr. K. M. Yu for his patience, guidance and many insightful viewpoints throughout this research work. Without his help this research work could not be completed better. I would also like to thank my co-supervisor Prof. W. B. Lee for his invaluable advice and assistance throughout the course of this study.

I would like to thank Stuart Woolman, the person comes from the Smart Solutions Ltd. for his advice in distributed PDM system architecture and other Internet technologies application.

In addition, I would like to collectively thank many family members, friends and colleagues for their constant support and persistent encouragement.

Finally, but most importantly, I would like to express my love and appreciation to my parents who give me substantial care and taught me not only the value of education, but also the virtue of striving to achieve excellence in all these years.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>I</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>V</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>V</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>XVI</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>XXI</td>
</tr>
<tr>
<td><strong>CHAPTER 1  INTRODUCTION</strong></td>
<td></td>
</tr>
<tr>
<td>1.1. Background</td>
<td>1.1</td>
</tr>
<tr>
<td>1.2. Statement of Problem</td>
<td>1.3</td>
</tr>
<tr>
<td>1.3. Objectives of the Research</td>
<td>1.5</td>
</tr>
<tr>
<td>1.4. Outline of the Thesis</td>
<td>1.6</td>
</tr>
<tr>
<td><strong>CHAPTER 2  LITERATURE SURVEY</strong></td>
<td></td>
</tr>
<tr>
<td>2.1. Introduction to PDM</td>
<td>2.1</td>
</tr>
<tr>
<td>2.2. PDM System Design Approach</td>
<td>2.3</td>
</tr>
<tr>
<td>2.3. PDM System Shift to OO Approach</td>
<td>2.4</td>
</tr>
<tr>
<td>2.4. Distribution of Data</td>
<td>2.6</td>
</tr>
<tr>
<td>2.5. Workspace for Collaborative and Distributed Manufacturing</td>
<td>2.8</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CHAPTER 3  PDM OPERATIONS AND STATECHART

3.1. Use of Graphical Representation Method and Statecharts 3.1

3.2. Product Data of PDM 3.1

3.3. PDM Management Functionality 3.2

3.4. The Product Data Management Operation 3.3

3.5. Graphical Representation of PDM System Operation 3.5

3.5.1. Register a new product data object to a local PDM system 3.6

3.5.2. Release a product data object to a local PDM system 3.7

3.5.3. Check-out a product data object in a local PDM system 3.8

3.5.4. Check-in a product data object in a local PDM system 3.9

3.5.5. Check-out a product data object in a local PDM system 3.10

3.5.6. Check-in product data object in a local PDM system 3.11

3.5.7. Release a product data object in a local PDM system 3.12

3.5.8. New Release a product data object in a local PDM system 3.13

3.5.9. Check-in a modified product data object in a local PDM system 3.14

3.5.10. Check-out a modified product data object in a local PDM system 3.15

3.5.11. Check-in a re-modified product data object to a local PDM system 3.16

3.5.12. Release a modified product data object to a local PDM system 3.17

3.5.13. Obsolete a product data object in a local PDM system 3.18

3.5.14. Delete a product data object in a DPDM system 3.19
3.6. Statechart-A Visual Formalism for PDM System 3.20

3.6.1. The Statechart notation 3.20

3.6.2. Statecharts for PDM system 3.22

3.6.2.1. Register new product data object to a local PDM system 3.22

3.6.2.2. Release a new product data object to a local PDM system 3.23

3.6.2.3. Check-out a product data object in a local PDM system 3.23

3.6.2.4. Check-in a product data object in a local PDM system 3.24

3.6.2.5. Release a product data object in a local PDM system 3.24

3.6.2.6. New Release a released product data object in a local PDM system 3.25

3.6.2.7. Obsolete a product data object in a local PDM system 3.25

3.6.2.8. Delete a product data object in a local PDM system 3.26

3.6.2.9. Operations in a local PDM system 3.27

CHAPTER 4  DPDM OPERATIONS AND STATECHART REPRESENTATION

4.1. Introduction to DPDM 4.1

4.2. New Product Data of DPDM 4.2

4.3. DPDM Functionalities 4.2

4.4. New DPDM System Functions 4.4
4.5. The DPDM System Operations

4.6. Graphical Representation of DPDM System Operations and Data Objects

4.6.1. Register a new product data object to DPDM system

4.6.2. Release a new product data object to DPDM system

4.6.3. Release a new product data object to both local and remote sites

4.6.4. Retrieve product data object in a local site within a DPDM system

4.6.5. Retrieve a distributed product data object from remote site

4.6.6. Check-out product data object in local site for update

4.6.7. Update product data object in a local site

4.6.8. Update product data object in both local and remote sites

4.6.9. Update product data object in remote site

4.6.10. Obsolete a product data object

4.6.11. Delete a product data object

4.7. Statechart-A Visual Formalism for DPDM System

4.7.1. Register a new product data object to DPDM system

4.7.2. Release product data object in DPDM system

4.7.3. Release product data object to local and remote sites

4.7.4. Retrieve product data object from local site in DPDM system

4.7.5. DPDM Product data retrieval in local site
4.7.6. Retrieve a product data object from remote site in DPDM system

4.7.7. DPDM Product data retrieval in remote site

4.7.8. Update product data object in local site within DPDM system

4.7.9. Update product data object in remote site within DPDM system

4.7.10. DPDM Product data update in local site

4.7.11. DPDM Product data update in remote site

4.7.12. Obsolete product data object in DPDM system

4.7.13. Obsolete product data object in both local and remote sites

4.7.14. Delete product data object in DPDM system

4.7.15. Delete product data object in both local and remote sites

CHAPTER 5  OBJECT-ORIENTED STATIC DATA MODEL FOR DPDM SYSTEM

5.1. Introduction

5.2. Use of Object Model Technique (OMT)
   5.2.1. OMT object model notation
   5.2.2. OMT functional model notation
   5.2.3. OMT dynamic model notation

5.3. Object-oriented Product Data Model
   5.3.1. OMT for Product Data Object Model
5.3.1.1. Class 5.5
5.3.1.2. Object identity 5.5
5.3.1.3. Attribute 5.6
5.3.1.4. Object 5.7
5.3.1.5. Encapsulation 5.7
5.3.1.6. Complex object 5.8
5.3.1.7. Object association 5.9
5.3.1.8. Object inheritance 5.10

5.3.2. Functional model 5.11
5.3.3. Dynamic model 5.13

CHAPTER 6 PRODUCT DATA ALLOCATION IN DPDM SYSTEM

6.1. Migration and Replication Techniques for Product Data Object Distribution 6.1

6.2. Product Data Allocation Objectives for DPDM 6.4

6.3. Optimal Product Data Allocation Model 6.5

6.3.1. Minimization of DPDM system operation cost ($C_{operation}$) 6.7

6.3.2. Minimization of DPDM system response time ($T_{response}$) 6.10

6.3.3. Minimization of DPDM system remote object request rate ($R_{request}$) 6.13

6.3.4. Reduction of storage cost 6.14
CHAPTER 7  WORKSPACE STRATIFICATION FOR DPDM SYSTEM

7.1. Problems of Existing PDM System in Distributed and Collaborative Environment

7.1.1. PDM operation processing and data version control

7.1.2. Security control

7.1.3. Collaboration awareness

7.1.4. Synchronous distributed collaboration

7.2. Shared Workspaces

7.3. Workspace-based PDM System Environment

7.4. Workspace System Category and Architecture

7.4.1. Categorization of PDM workspace

7.4.2. User classification

7.4.3. Application levels in PDM system

7.4.4. Shared workspace hierarchy and web-based model

7.5. Solutions Provided by the Workspace-based PDM System

7.5.1. PDM operation processing and data version control

7.5.2. Workspace security control

7.5.3. Workspace collaboration awareness

7.5.4. Synchronous distributed collaborative environment

7.6. MultiNet in a DPDM System

CHAPTER 8  SECURITY MODEL FOR PDM/DPDM SYSTEM

8.1. Requirements
8.1.1. Scenario 1: Project-based
8.1.2. Scenario 2: Intra-organization PDM user environment
8.1.3. Scenario 3: DPDM user environment
8.1.4. Scenarios summary
8.2. DPDM System and Security
8.2.1. Workspace-oriented integration classification
8.2.2. Workspace domain
8.2.3. Project team organization structure
8.2.4. MultiNet user
8.3. Stratified Workspace Schema
8.4. Access Control Schema for Workspace-oriented DPDM System
8.4.1. UOSD matrix-based access control
8.4.2. BLP label-based mandatory access control
8.5. Mixed Approach to DPDM Access Control
8.5.1. Well control access to the workspace-oriented DPDM system
8.5.2. Product data protection

CHAPTER 9 IMPLEMENTATION OF WEB-BASED DPDM SYSTEM
9.1. OODBMS and Web-based DPDM System Architecture
9.2. Java and Three-Tier Client/Server Technologies
9.2.1. Java-based networking application
9.2.2. Java remote method invocation (RMI)
9.2.3. Java database connectivity (JDBC)
9.2.4. Three-tier client/server approach 9.6

9.2.5. Three-tier client/server model operation 9.8

9.3. Implementation of DPDM Product Data Model 9.9

9.4. Stratified Workspace in DPDM System 9.10

9.4.1. Personal workspace 9.10

9.4.2. Project team workspace 9.12

9.4.3. Corporate workspace 9.14

9.4.4. Public workspace 9.16

9.4.5. Scenario of MultiNet architecture 9.17

9.4.6. Event message flow between DPDM workspaces 9.18

9.5. Web-based DPDM System Software Development 9.21

9.5.1. DPDM system browser 9.21

9.5.2. DPDM system functional modules 9.22

9.5.2.1. DPDM system security module 9.22

9.5.2.2. DPDM stratified workspace module 9.25

9.5.2.3. DPDM project management module 9.30

9.5.2.4. DPDM product data distribution module 9.32

9.5.3. DPDM system database schema 9.37

CHAPTER 10 DISCUSSION

10.1. Statechart Modeling Method for DPDM Operations 10.1

10.2. Product Data Model and Object-oriented Technology 10.1

10.3. Product Data Allocation in DPDM System 10.2
A.16. DpdmLoginPool.asp  A.32
A.17. DpdmLogOut.asp  A.36
A.18. DpdmSite.asp  A.37
A.19. find_project.asp  A.38
A.20. findData BySite.asp  A.45
A.21. FindSiteByPID.asp  A.47
A.22. Login_check.asp  A.50
A.23. login_check.asp  A.52
A.24. obj_detail.asp  A.54
A.25. objectUST.asp  A.56
A.26. registerFrm.asp  A.60
A.27. registerOp.asp  A.62
A.28. SelectDistData.asp  A.64
A.29. SelectDistributeObj.asp  A.66
List of Figures

Figure 2.1 Product data volume increase rapidly in product life cycle 2.2
Figure 2.2 IMAN’s open architecture [Unigraphics 2000] 2.4
Figure 2.3 Summary of key steps in PDM system implementation 2.12
Figure 2.4 Conventional and PDM integrated concurrent processing 2.13
Figure 2.5 Web and PDM logical architecture 2.14
Figure 2.6 Mapping between PDM and Web’s three-tier architecture 2.14
Figure 2.7 Web-browser based PDM interface [Mendel 1999] 2.15
Figure 3.1 Product Data Document Management Operations 3.5
Figure 3.2 PDM Register operation on a new object 3.6
Figure 3.3 PDM Release operation on a new object 3.7
Figure 3.4 PDM Check-Out operation on a Checked-In object and update object version 3.9
Figure 3.5 PDM Checked-In operation on Checked-Out object 3.9
Figure 3.6 PDM Check-Out operation on a Checked-In object and assign a new object version 3.10
Figure 3.7 PDM Check-In operation on a Checked-Out object with new assigned object version 3.11
Figure 3.8 PDM Release operation on a Checked-In object and update the object release version 3.12
Figure 3.9 PDM New Release operation on released object and assigning a new object version 3.13
Figure 3.10 PDM Check-In operation on a Checked-Out object with the new assigned version 3.14
Figure 3.11 PDM Check-Out operation on a Checked-In object and assigning a new object version 3.15
Figure 3.12 PDM Check-In operation on a Checked-Out object with the new assigned object version 3.16
Figure 3.13 PDM Release operation on a Checked-In object and update the object release version 3.17
Figure 3.14 PDM Obsolete operation on a released object and changing the object state 3.18
Figure 3.15 PDM Delete operation on an object 3.19
Figure 3.16 Simple state transition 3.20
Figure 3.17 Sate transition with action 3.21
Figure 3.18 A condition connector 3.21
Figure 3.19 Register a new product data to PDM system database and vault 3.22
Figure 3.20 Release a new product data 3.22
Figure 3.21  Check-Out a Checked-In product data for modification or update  3.23
Figure 3.22  Check-In the Checked-Out product data after modification or update  3.23
Figure 3.23  Release a Checked-In product data  3.24
Figure 3.24  Check-Out a released product data for modification or update  3.24
Figure 3.25  Obsolete a Released product data  3.25
Figure 3.26  Delete product data from PDM system database and vault  3.26
Figure 3.27  Overall operations and state transitions in a PDM system  3.27
Figure 4.1  DPDM Register operation on a newly created object and assigning object version in a local site  4.6
Figure 4.2  DPDM Release operation on a newly created object in a local site  4.7
Figure 4.3  DPDM Release operation on a new object to release it to a local and remote sites  4.8
Figure 4.4  DPDM Retrieval operation on an object in a local site  4.9
Figure 4.5  DPDM Retrieval operation on an object located in remote site  4.10
Figure 4.6  DPDM Update operation on an object with new object version  4.11
Figure 4.7  DPDM multi-update operation on a Released object in a local site  4.13
Figure 4.8  DPDM Update operation on a local object located in both local and remote sites  4.14
Figure 4.9  DPDM Update operation on a remote object located in remote site  4.15
Figure 4.10  DPDM Obsolete operation on an object  4.16
Figure 4.11  DPDM Delete operation on an object  4.17
Figure 4.12  Register a new product data to a DPDM system database  4.18
Figure 4.13  Release a Checked-In product data  4.18
Figure 4.14  DPDM system states of Release processing in local and remote sites  4.19
Figure 4.15  Retrieve Released product data in local site  4.19
Figure 4.16  DPDM system states of retrieval processing in local site  4.20
Figure 4.17  Retrieve Released product data from remote site  4.20
Figure 4.18  DPDM system state of retrieval processing in remote site  4.21
Figure 4.19  Update product data in local site  4.22
Figure 4.20  Update product data in remote site  4.22
Figure 4.21  DPDM system states of update processing in local site  4.23
Figure 4.22  DPDM system states of update processing in remote site  4.24
Figure 4.23  Obsolete Released product data  4.25
Figure 4.24  DPDM system states of Obsolete processing in both local and remote  4.26
Figure 4.25  Delete product data  

Figure 4.26  DPDM system states of delete processing in both local and remote sites  

Figure 5.1  Class  

Figure 5.2  Association  

Figure 5.3  Multiplicity of association  

Figure 5.4  Generalization (Inheritance)  

Figure 5.5  Aggregation  

Figure 5.6  Data flows between processes in the functional model  

Figure 5.7  State diagram in the dynamic model  

Figure 5.8  Object class for a product data model  

Figure 5.9  Encapsulation of an object  

Figure 5.10  An example of complex object in manufacturing domain  

Figure 5.11  Object Association  

Figure 5.12  Object inheritance  

Figure 5.13  Data flow diagram for DPDMS  

Figure 5.14  Distributed product development cycle under DPDM system environment  

Figure 5.15  Inter data flow of the distribution activity under DPDM system environment  

Figure 5.16  State transition diagram in DPDMS  

Figure 5.17  State transition diagram of distribution activity  

Figure 6.1  Run time PDM system  

Figure 6.2  DPDM system operation cost  

Figure 6.3  DPDM system response time  

Figure 6.4  Object allocation in a DPDM system  

Figure 6.5  DPDM system remote object request  

Figure 7.1  Information sharing in conventional production system  

Figure 7.2  Information sharing in a PDM integrated production system  

Figure 7.3  Information sharing in a workspace based PDM system  

Figure 7.4  Workspace based user classification  

Figure 7.5  Class hierarchy of PDM user  

Figure 7.6  Application levels in a DPDM system  

XVIII
Figure 7.7 Basic workspace category hierarchy of DPDM system
Figure 7.8 Web-based Product data and information flow model of DPDM system
Figure 7.9 Summary of application level based workspace category
Figure 7.10 Sensitivity levels of product data
Figure 7.11 Association of users/permissions
Figure 7.12 (a) and (b) illustrate the association of user/workspace and workspace/permission
Figure 7.13 Association with workspace for each user, and associations with Workspace for each permission
Figure 8.1 Interaction association between DPDM system object classes
Figure 8.2 Summary for DPDM operations of each user type
Figure 8.3 User classes in a DPDM system
Figure 8.4 User hierarchy in a DPDM system
Figure 8.5 Workspace security level in DPDM system
Figure 8.6 Workspace-oriented DPDM system mandatory access control
Figure 9.1 OODBMS in a DPDM system architecture
Figure 9.2 DPDM system work flow
Figure 9.3 BOM in traditional database
Figure 9.4 BOM in OODB
Figure 9.5 (a) and (b) Client/Server Concept
Figure 9.6 (a) HK-server and China-client and (b) China-server and HK-client three-tier architecture
Figure 9.7 Personal Workspace boundary
Figure 9.8 Data objects are created in Personal workspace
Figure 9.9 Data structure
Figure 9.10 Object inherit attributes from their parent classes
Figure 9.11 Project Team Workspace boundary
Figure 9.12 Created object will be registered to the PDM database in Project team Workspace
Figure 9.13 Classification and data structure in a DPDM database
Figure 9.14 Object inheritance in a data structure
Figure 9.15 Workspaces support version management
Figure 9.16 Corporate Workspace boundary
Figure 9.17 Corporate workspace is used for Data sharing between corporate Users
<table>
<thead>
<tr>
<th>Figure 9.18</th>
<th>Public Workspace boundary</th>
<th>9.17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 9.19</td>
<td>Viewing product data by a browser</td>
<td>9.18</td>
</tr>
<tr>
<td>Figure 9.20</td>
<td>Interaction between Public workspace and outside company</td>
<td>9.18</td>
</tr>
<tr>
<td>Figure 9.21</td>
<td>Enterprise Multi-Net network architecture</td>
<td>9.19</td>
</tr>
<tr>
<td>Figure 9.22</td>
<td>Event message flow between shared workspaces</td>
<td>9.20</td>
</tr>
<tr>
<td>Figure 9.23</td>
<td>DPDM System browser</td>
<td>9.22</td>
</tr>
<tr>
<td>Figure 9.24</td>
<td>DPDM System user authentication</td>
<td>9.23</td>
</tr>
<tr>
<td>Figure 9.25</td>
<td>DPDM System logout operation</td>
<td>9.24</td>
</tr>
<tr>
<td>Figure 9.26</td>
<td>DPDM System view log function</td>
<td>9.25</td>
</tr>
<tr>
<td>Figure 9.27</td>
<td>DPDM Workspace System menu</td>
<td>9.25</td>
</tr>
<tr>
<td>Figure 9.28</td>
<td>View Login Pool function of DPDM system</td>
<td>9.26</td>
</tr>
<tr>
<td>Figure 9.29</td>
<td>DPDM Workspace Modules menu</td>
<td>9.27</td>
</tr>
<tr>
<td>Figure 9.30</td>
<td>Project Team Workspace</td>
<td>9.28</td>
</tr>
<tr>
<td>Figure 9.31</td>
<td>Object Detail Information from Project Team Workspace</td>
<td>9.29</td>
</tr>
<tr>
<td>Figure 9.32</td>
<td>Registration Operation in Project Team Workspace</td>
<td>9.29</td>
</tr>
<tr>
<td>Figure 9.33</td>
<td>DPDM system product data viewer</td>
<td>9.30</td>
</tr>
<tr>
<td>Figure 9.34</td>
<td>DPDM project management module</td>
<td>9.31</td>
</tr>
<tr>
<td>Figure 9.35</td>
<td>Create new project in DPDM system</td>
<td>9.32</td>
</tr>
<tr>
<td>Figure 9.36</td>
<td>Distribution module of DPDM system</td>
<td>9.33</td>
</tr>
<tr>
<td>Figure 9.37</td>
<td>Select product data and distribute to remote site in DPDM system</td>
<td>9.34</td>
</tr>
<tr>
<td>Figure 9.38</td>
<td>Remote site information</td>
<td>9.35</td>
</tr>
<tr>
<td>Figure 9.39</td>
<td>Distributed product data in a remote site</td>
<td>9.36</td>
</tr>
<tr>
<td>Figure 9.40</td>
<td>Distributed product data in a remote site information</td>
<td>9.37</td>
</tr>
<tr>
<td>Figure 9.41</td>
<td>DPDM system database schema</td>
<td>9.38</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.1</td>
<td>General grouping of product data</td>
<td>6.3</td>
</tr>
<tr>
<td>6.2</td>
<td>Retrievals and updates of product data</td>
<td>6.3</td>
</tr>
</tbody>
</table>
CHAPTER 1  INTRODUCTION

1.1. Background

Manufacturing is important to human beings. After the industrial revolution in the eighteenth century, manufacturing generally refers to utilizing machines and human labour to produce products from raw materials [Sephanou 92]. Since human beings will increasingly need to consume products. It is predictable that manufacturing will remain as important as it has been.

As noted by most people, increasing competition exists in the manufacturing industry. To be successful in today’s highly competitive and global manufacturing environment, a company must be able to supply and support the products that customers require, at the time required by the customer. These requirements put tremendous pressure on the engineering function to improve product quality and reduce lead times. One way to do this is to increase the productivity of individual engineering activities. Another way is to improve the coordination between activities. Both approaches must be followed if maximum results are to be achieved. A new system is required to support both approaches. Currently, most companies would claim to have engineering data more or less under control. They recognize, though, that this state of affairs may not last for long. There are several different factors that will make data more difficult to control. One of these factors is the rapidly increasing amount of data produced by computers. Another is the increasing customization of products. Whereas in the past there may only have been one set of product definition data for a given product, in the future a product may have a different set of product definition data for each customer. There will be a wider range of
products, as well as more versions of each product. Information will have to be maintained on the exact product configuration used by each customer and on all the different assemblies used to make up the product in its different configurations.

Furthermore, due to the advances in transportation and communications, plus the changes in international political and business environments. Competition arises not only domestically but internationally as well. It is not a trivial issue to investigate the causes and effects of global competition. There are many aspects in surviving and thriving in global competition. One of the most important aspects is to improve productivity [Liou 1994]. Productivity improvement resides in many factors, such as lower cost, better quality and reliability, shorter time to market, faster delivery, etc. All the above factors can be realized through advances in manufacturing technology [Sephanou 92].

To improve the productivity, a manufacturing system must have efficient material flow, cash flow, control signal flow and information flow. Among the flows, information flow seems to be the most important since other flows depend on it. On the other hand, in today’s Internet era, information control has played an important role for all fields of the industries.

In today’s manufacturing environment, people have realized that how to manage the product data, distribute it to the people who need it at the right time is critical to the success of a company. Today, product data management (PDM) has been realized as an important tool for the industries. PDM is an enabling technology that helps work group, department, division, or enterprise to manage product data and the development process throughout the product life cycle,
from inception, through development and production to dispatch and beyond. Due to the
marketing strategy and for lowering the production cost, many companies have distributed their
production systems to different locations. Especially in Southern China, in the past decade,
Hong Kong manufacturers have moved their production systems to Mainland China, but they
retain their main offices in Hong Kong (HK). A phenomenon of "production in the backyard
and marketing at the front gate" is formed.

In recent years, centralized data center service is available in the market. However, it is suitable
for the storage of business data, for example the customer information, inventory data, etc. It is
not suitable for product data. It is because product data includes the meta data and physical
product data (e.g. CAD file). For today's distributed manufacturing environment, it is not a good
and efficient way for distributed users to transfer a large amount of product data from the
centralized data center frequently. On the other hand, a data center for data storage is unlike a
PDM system, it cannot provide PDM function for the user to perform on the product data.

Therefore, a new product data management system is required. Apart from having all the
advantages of existing PDM systems, the new system provides the following advantages:

- It can break down the geographical boundaries between distributed sites since it manages
  the distributed product data through a computer network, the Internet.

- It can distribute product data to remote sites and this operation is under controlled by a new
  product data allocation model.

- It can improve distributed collaborative activities since it manages and facilitates
  distributed product data sharing under security control.
- It can improve system performance and efficiency by employing new advanced technology, object-oriented database.
- It can improve system security to protect product data by the new security model.

1.2. Statement of Problem

A PDM system can at least provides the following advantages:

1. Data sharing in different stages of product development
2. Implementing concurrent or simultaneous engineering
3. Achieving and maintaining regulatory compliance or certification (i.e. ISO 9000)
4. Improving integration between engineering and manufacturing
5. Improving product quality
6. Elimination of paper-based documentation
7. Shortening lead time to market

Although existing PDM systems can cater for traditional needs, there are still limitations and problems that have to be tackled in the real life distributed and collaborative manufacturing engineering arena. The problems can be summarized as the following:

(1) The existing PDM system is for local application. It manages and improves data sharing within a local company. In a distributed manufacturing environment, distributed manufacturing system requires a local PDM system to manage their own local data. Since two PDM systems cannot communicate with each other, distributed standalone PDM systems cannot share the product data in a consistent manner and will lead to a variety of problems in product development and production.
(2) There is no formal model for representing and developing PDM system.

(3) Most existing PDM systems use relational database. In product development, different types of product data are generated and the data objects are usually very complicated. The traditional relational database system is not efficient to deal with the high variety of data types and the complexity of data objects.

(4) In a distributed environment, which data should be distributed to a remote site? Which methods and technologies should be used? It is not an easy task to make the right decision to allocate the right data to the right site. However, correct data allocation is important to the performance of the system. It also affects the performance of the distribution production system.

(5) In a collaborative and distributed environment, handling of voluminous amount of data, consistency of product data version, high collaborative awareness and support of synchronization of distributed collaboration are some of the major factors for a successful distributed manufacturing and PDM system. However, the existing PDM systems are just for local use or standalone application. It is not enough for today's collaborative application, especially when it involves a multitude of operations and stages.

(6) PDM system has made product data a valuable and available commodity for many different kinds of computing applications in production. However, there are fears that this product data will lead to new security risks, and to the invasion of the PDM system.
It is because all people in the manufacturing enterprise will somehow get in touch with the PDM system. Dealing with this tremendous amount of interactions between the system and the various users, the utmost importance is to ensure that all data are secured and all users are under controlled and managed.

1.3. Objectives of the Research

In order to solve the problems stated above, this research is to develop a new product data management system for today’s distributed manufacturing environment. The main objective of this research is:

**To develop an architecture for web-based Distributed Product Data Management System.**

The objective is achieved by the following investigations:

- Analyzing and constructing a formal operation model for the new system.
- Developing a new product data model.
- Developing a new product data allocation model.
- Developing a new system security model.
1.4. Outline of the Thesis

Chapter 2: it provides some background and literature review which is relevant to the research work in this project. These include PDM system architecture and framework, implementation and application, product design application, web integration and MRP integration. The future trends to enhance the application of PDM system are also discussed.

Chapter 3: it gives an introduction to PDM system, its need and its PDM functionality. Statechart is employed to model the operation of the PDM system.

Chapter 4: it gives an introduction to DPDM system, its need, its functionality and the system architecture. Statechart is employed to model the operation of the DPDM system.

Chapter 5: it presents an object-oriented product data model for a DPDM system. The model is developed by the OMT method. It also discusses the advantages of the object-oriented product data model.

Chapter 6: it presents an object-oriented product data allocation model in the DPDM system. It discusses the product data distribution objectives and their quantitative analysis. An OODBMS and web-based DPDM system architecture is also presented.

Chapter 7: it defines the stratified workspace for a DPDM system. Explanation of its realization in categorization, user classification, and different application levels in PDM/DPDM system is given in detail. It also defines the MultiNet architecture.

Chapter 8: it presents a security model for DPDM system. In particular, the workspace-oriented schema and the Access control schema. It also explains how a mixed approach is suitable for the DPDM system.
Chapter 9: it highlights the implementation results of the proposed models and theories.

Chapter 10 and 11: it discusses limitations and the potential future work development of DPDM.

Chapter 12: it concludes the thesis.
CHAPTER 2 LITERATURE SURVEY

Product data management (PDM) is one of the most promising technologies to enter the industrial arena. More and more researches have been conducted in the development of it. This chapter aims to review researches in the development of PDM for collaboration in distributed environment.

2.1. Introduction to PDM

The need for data management was raised in the early 1980s due to the increasing growth of "islands of automation" within an organization. The subject of computerized data management started to be addressed in the mid to late 1980s [Harris 1996]. The term PDM was first introduced by Johnson, CIMdata [Johnson 1988].

In product development, product data increases rapidly during development cycle, Figure 2.2. Therefore, people have to pay more attention on the product data management. In the early state of PDM system implementation, PDM system grew out of the need to manage CAD related data. Later, solid modeling concept is discussed for its special relationship to the PDM system [Johnson 1986]. In 1983, Boeing, an aerospace company was aware of the need for better control of the product data created in CAD/CAM environment [Atkinson 1990]. The company decided to implement a PDM system for multiple processes.
Initially, PDM is used to manage and control the volumes of electronic media that were created by engineering design automation. Later in the late 1980s, several software companies realized the need and associated business opportunity. They introduced the first generation commercial PDM systems. These include many of the traditional PDM systems such as IBM’s ProductManager, SDRC’s Metaphase, Sherpa’s SherpaWORKS, Computervision’s Optegra, EDS/Unigraphics’ IMAN, Adra’s Matrix, Eigner’s CADIM/EDB, CoCreate’s Workmanager, Auto-trol’s Centra 2000, Workgroup Technology’s CMS, and several others newer vendors like Agile, Consensus, Smart Solutions, and Right Angle. Now, PDM is a proven way to speed up workflow, improve communication, and facilitate process change.
2.2. PDM System Design Approach

In PDM system design, user-driven system approach is proposed. In this approach, PDM user must satisfy the requirements of individual with different job functions and requirements. Also, the interrelationship between data, retrieving method and integration of the PDM into the product development process are of concern [Chris 1995]. This approach has been applied to PDM design for an agile manufacturing pilot project and achieve a much faster design-to-product process for a precision electromechanical surety device. In the view of process-based PDM system, PDM system is applied for two common processes: the development and the delivery processes [Hannu 1996]. Later, more elements, such as the people, information, applications and processes are discussed to be managed electronically by the integration of PDM workflow management system [Ramanathan 1996, Izuchukwu 1996]. More about the integration of workflow management function to PDM system is investigated [Puttre 1994, Kovacs 1998, Chlebus 1998]. A philosophy for the support of product data throughout the full development and production lifecycle is proposed. Apart from integrating workflow, people of Technical University of Wroclaw have concerned more about PDM system integration. A PDM-CAX integrated system is proposed in concurrent design of manufacturing processes.

Many researches have been performed on the function development of local PDM system. However, there is no formal system nor representation model for PDM system. System model is important for system development, it not only provides a graphical picture of the system, but also guides the development in the right track and avoids problems in the early stage of the development.

2.3
2.3. PDM System Shift to OO Approach

For the development of the PDM system architecture, an open one is proposed [Zhang 1995]. In this architecture, product data definition model, and a set of components of the architecture is discussed, especially the principle and mechanism of one component, the object-oriented database management system (OODBMS) GH-EODB. One PDM vendors, EDS/Unigraphics, provides a PDM product system IMAN. It is modular and object-oriented. It also provides a comprehensive framework to support the entire product life cycle, Figure 2.1.

![Diagram of IMAN's open architecture](image)

Figure 2.2 IMAN's open architecture [Unigraphics 2000]

Nowadays, system architecture and design are shifted from the conventional approach to object-oriented approach. Rumbaugh's OMT (object modeling technique) has been adopted for the system design. Using the object-oriented paradigm, an object-based data model for the PDM
system is proposed [Kim 1997]. However, there is a limitation in the scope of the modeling which can only be used to manage drawings, parts, and product structure.

To satisfy more requirements of real application, an extendible and general PDM system framework model is discussed [Kim 1998]. Object-oriented technology is applied to construct the generalized object model. However, since the framework is based-on RDBMS instead of OODBMS. The constructed object model cannot be mapped into RDB.

In a data management system, data model is important for the system design. A semantic product data model was presented by Shaw, Susan and Pennington [Martin 1996]. Their model supports product design and manufacturing. In their paper, they described the characteristics for semantic data models. The capabilities of a prototype product description system are described in relation to the same characteristics. Additional data modeling characteristic such as parameterization and data sharing have been introduced to support engineering design.

Based on the semantic data model in [Martin 1996], Stadlbauer proposed a product data model for design support using functional skeletons [Sadlbauer 1992]. He integrated a new feature called functional skeletons into a product data model for the design process. This feature represents the main functional flows in a product and allows the efficient storage of designs as well as the generation of verified products. A data model called EDM (Engineering Data Model) has been introduced by Eastman, Bond and Chase [Pahny 1998]. Their model is for representing design and engineering information, which defines a small set of structures capable of depicting a wide range of semantics necessary for engineering design. These structures allow

2.5
the definition of specific product models that are equivalent to database schemas. Another product data model was introduced by Gabbert and Wehner [Martino 1998]. The paper presented some fundamentals of product data modeling and related aspects of design and product data analysis. The model can provide improvement in productivity and product quality by improving product data exchange.

In the published papers, there is no object-oriented product data model for distributed product data management nor any complete data model for existing product data management system. No functional, dynamic nor object model of the product data model has been investigated, most of the product data models simply describes the product design aspect. In product development, product design is just one of the many stages. A product model should be applicable to all stages of product development and can deal with all product related data. On the other hand, the existing data models are localized and developed for local Computer Integrated Manufacturing (CIM) system. They do not well support the distributed manufacturing environment. From a global view and for real time distributed manufacturing, a new product data management system and a comprehensive product data model is required.

2.4. Distribution of Data

Some papers have been published for file allocation. Chu [Chu 1976] proposed an approach which reports the cost performance trade-off of the classes of centralized directory system, local directory system and distributed directory system in star and distributed networks. The paper is concerned with the allocation of directories in a distributed database. Lee proposed a method to
determine a compromise design for file allocation via a wide area telecommunication network [Lee 1994]. This compromised approach is likely to be better in representing the system designers' goal setting behavior under a multiple design criteria situation. Another approach about file allocation is introduced by Dowdy and Foster [Dowdy 1982]. They evaluate advantages and weakness of the various models discussed. The perspective implies that further study is required to improve the file allocation problem. Salvatore [Salvatore 1995] proposed an approach of distributed database design which emphasizes on allocating data and operation to nodes in a computer communication network. In this approach, both data fragments and operations are allocated to nodes using a mathematical modeling approach. Another design approach is introduced by Cheng [Cheng 1993] who proposed a method to design distributed object databases. Some of the distributed features are identified and how earlier research in distributed database may or may not be applied to distributed object databases are pointed out. A different kind of object distribution approach for concurrent object-oriented programming is proposed by Antonio [Nebro 1999]. This approach is designed to take the advantages of both concurrent programming and object-oriented programming. It implements distributed objects (programs) in a reasonable and efficient way. This approach is based on the use of a protocol that allows objects to be migrated and replicated. The basic idea is to reduce the overhead of remote invocations among objects by making them local.

No distributed product data allocation model has been proposed for distributed manufacturing environment. Most of the existing approaches are in traditional database architecture and using fragmentation technique. Some approaches may not be applicable in a real-life environment, due to their hard computational complexity and the great number of parameters which have to be
estimated [Ceri 1982]. Therefore, a new product data distribution model is required. The new model should be designed not only for data distribution and also help to achieve well distributed product data management.

2.5. Workspace for Collaborative and Distributed Manufacturing Environment

PDM has been the recognized way for managing product design data in manufacturing and engineering companies for almost a decade. Complex design projects demand more than just engineering tools. They also demand automated ways of coordinating teams and tracking projects to produce a concurrent engineering environment [Williams 1991]. Product data management automates the design process all the way from the drawing office to the shop floor in a much more sophisticated way than CAD and electronic document management (EDM) system [Goodwin 1995].

Design product right the first time has become one of the most important goals of manufacturers. They recognize that to remove the obstacles that stand in their way of design phase, they need to push quality forward into the design cycle, rather than add quality after the manufacturing is done [Deitz 1995]. PDM system ensures that the appropriate engineers and managers in all areas review design approaches early, when changes are easily and inexpensively made. PDM system can make it easier for design and manufacturing engineers to collaborate on project and allow quality-assurance engineers to identify areas for improvement.
To ensure a right product design in manufacturing, it requires to increase the linkages of product data to other stages of the design process by the integration of PDM system with other CAx tools [Wiebe 1997, Stamps 1990]. PDM combines with CAx to enhance a creative and innovative design process, and improve quality and time to market through information exchange. PDM system can be used effectively as an information link between departments.

Some papers have discussed the application of workspace in a cooperative working environment. Shared workspace is discussed in the study of small group design activity [Tang 1993]. It focuses on the use of a workspace by team working on the product development stage of conceptual design. This study emphasizes on the activities of the listing, drawing, and gesturing that occurs in a shared working environment. Although it has mentioned the storage of data, it does not talk about the management operation and versioning of data. Basic support for cooperative work (BSCW) system has been introduced [Bentley 1997]. It supports cooperation through the shared workspace which is used as small repositories. Users can upload and download data to and from them. This system allows information sharing to help users to coordinate their own work. Another paper introduces workspace in concurrent design process [Kung 1999]. Two types of workspaces, personal and global are mentioned in brief. Both workspaces are set up for data sharing during the design processes.

Workspace is indispensable for product development. It is important for users to work out their jobs, it establishes the linkage between each CAx system, it helps the right people to get the right data effectively and securely. Existing workspace systems are not categorized nor well defined. They mainly exist for data sharing. In a collaborative working system, well managed
workspaces can bring distinct advantages, especially when the collaborative system involves various operation and changing stages. A new workspace system should thus be designed for all product developers.

2.6. System Security

The first attempt of computer system protection is proposed by Dennis and van Hon in 1966 [Van 1966]. Later, a model of protection mechanisms in computing system is presented by Harrison and Ruzzo in 1976 [Harrison 1976]. This model offers a classification of certain features of protection systems and provides a framework for investigating the features, which cause an undesirable safety problem to the system. A military security control is described by Landwehr in 1981 [Landwehr 1981]. It reviews the need for formal security models, describes briefly the structure and operation of the security control. It also suggests possible directions for future models. Risk analysis is generally not considered as an important task in the implementation of computer security at major organization. Badenhorst and Eloff [Badenhorst 1990] addresses the issue of risk analysis in view of an overall information security plan. The method is the first to indicate the position of a risk analysis study within the security plan. It also shows how to use the analysis result to determine countermeasures. Security requirement and access control is described by Leonhardt and Magee in 1988 [Leonhardt 1981]. It outlines the salient features of a location service to support a location model and discusses the access control methods for location information service system.
No product data management security model was proposed for existing product data management system. Therefore, a formal security model is required for the new distributed product data manage system to protect the valuable data.

2.7. Web-integrated PDM System Implementation

The task of implementing PDM system is going to be a tough job. Several people share the successes and challenges of their companies’ PDM implementations. They discuss the keys to success such as obtaining top-line support, knowing the process, forming a strong cross-functional team, getting help from the PDM vendor, etc [Siegel 1995]. They also point out the steps of implementation. Figure 2.3 summarizes the key steps in PDM system implementation. Implementing a PDM system can benefit the whole enterprise business and achieve the business objectives, one of the most concerned objectives is reducing product introduction time [Obank 1995, Bowman 1996]. The goals in the automotive industry today are time-to-market, quality, and cost. To achieve these goals, automotive companies are integrating PDM with other CAD/CAM technologies such as industrial design, solid modeling, design, etc. These integrated technologies make concurrent engineering (CE) works [Kempfer 1995, Chen 1998]. Figure 2.4 illustrates the conventional and PDM embedded concurrent manufacturing processing.
Today, more and more web-enabled PDM-based application will be provided in the market [Tinham 1998]. PDM is going to be integrated with web technologies to enhance its capabilities and performance [Chu 1999, Miller 1999]. It is because both of them have similar logical architecture as shown in Figure 2.5. There is also a close conformity between PDM and the web (Figure 2.6).
The increased number of users accessing PDM through the Internet technologies raises questions of system performance for vendors to consider. Vendors and users have learned much about the relative performance of browser-based PDM access versus client-based PDM access [Mendel 1999]. Many vendors have realized the impact of the Internet technologies and start to re-architecting their systems to improve performance. The processing burden now assumed by the client in the client/server system is shifted to the server in a browser-based system. Browsers (e.g., Netscape Navigator and Microsoft Internet Explorer) are incredibly inexpensive, simple to understand, run across all platforms (e.g., UNIX workstation, PCs and Macintoshes), and capable of supporting all users. Figure 2.7 shows a Web-browser based PDM interface.
Figure 2.5 Web and PDM logical architecture

Figure 2.6 Mapping between PDM and Web’s three-tier architecture
Figure 2.7 Web-browser based PDM interface [Mendel 1999]

People have realized the importance to integrate web technology with product data management. Although some managerial implementation key steps and logical architecture were discussed, there is does not have any detail physical layout nor network setup for implementation. A system should thus be designed, which is compatible with all web communication protocols as well as all common commercial browsers.
CHAPTER 3  PDM OPERATIONS AND STATECHART REPRESENTATION

3.1. Use of Graphical Representation Method and Statecharts

Development of product data management system is inherently complex. In this chapter and the next chapter, graphical representation method and statechart will be employed to model the product data management system. Graphical representation method expresses the scenarios. The method is capable of automatically generating design approximations, interactions among user and system. Statechart is a highly structured and economical description language for specifying the behavior of a system. The definition of the formal semantics of statechart has proved to be extremely challenging due to the richness of the language, and the fact that it is a visual language. Statechart support for the transition between requirements and design is important for a product data management system development process. It bridges the current gap between requirements, scenarios, and initial design specifications.

3.2. Product Data of PDM

According to Port and Pikosz [Port 1997, Pikosz 1996], product related data refer mainly to product name, product part number (product identification), drawings, material specifications, dimension requirement, quality specification, test result, log size, production schedules, product data version and date of release, special tooling (e.g. jig and fixture), mould design, project engineering in charge, cost spreadsheets, while process data includes engineering release,
engineering change information management and other workflow related to the process information.

3.3. PDM Functionality

According to [Cornelissen 1995], a contemporary PDM system contains the following main management functions:

*Structure Management*

- Classification: can be made through group-related documents, parts or data with the same features
- Document Structure: manage the complex interrelationships of documents and data
- Product Structure Management: the bill of materials (BOM) describes the complete product structure with relationships such as options, versions, and effectivities
- Configuration Management: manage the complex product structures

*Retrieval Management*

- Searching: used to create database query
- Viewing: view the image of the file

*Release Management*

- Authorization: document or data can be safely locked by password or other authorization tools
- Sign-off: control the change requests and change implementation
Status Control: describes a condition of a document such as Checked-out, Checked-in, etc.

**Change Management**

- Change Process: the responsibilities of person in the change process
- Engineering Change Order: the essential management forms in the change process
- Mark-up and Redlining: redline the change of document in different colors or layers to specify the change
- Version Control: provide the means to structure the change process

**Workflow Management**

- Process Management: the system deals with project overview
- Planning: a list of activities which can have dependencies among them and can be related to documents

**3.4. The Product Data Management Operation**

The main PDM operations contain the following [Smart Solutions 1998],

- Register: to begin a revision process, a document must first be registered
- Check-out: in order to make change to a file, the document must be checked out of the PDM system database
- Check-in: after a file has been checked out and modified, it must be placed back into the database
- Obsolete: removing an obsolete document to the obsolete section of database, the obsolete object cannot be changed or released
- Release: move the document to the next phase of the product development

Figure 3.1 illustrates the various product data management operations. When a user wants to modify an existing product data document. The user should address the system through a user interface, and search for the data. His/her request for the data will initiate a retrieve process in the PDM system that will query and extract the requested data from the database. The data is then presented in the user interface. Having seen the data, the user makes change request which initiates a change process that will check-out the required data from the PDM system database. The user is now provided with the ability to change the data in an editing application. For all other users that checked-out the same data, they will be notified with a “read-only” status. When the changes have been finished, the user should check-in the modified data to release it back to the system database. The PDM system will give the recheck-in document a new version number automatically. The version can also be assigned by the user manually. The data is now available for application and all read/write operations by any authorized users.
3.5. Graphical Representation of PDM System Operations

In this section, PDM operations and state transitions of product data object are represented graphically. The detail exposition of individual PDM operation allows us to visualize and verify the correctness of each information flow transition. A new product data object is created with a “New” object state. The object state will be changed after performing various PDM operations on it.
3.5.1. *Register a new product data object to a local PDM system*

After a product data object is created, PDM user registers it to a PDM system. The registered product data can be accessed by other PDM users. Any operation on the object will be managed by the PDM system.

![Diagram of PDM Server Database and PDM Client Database](image)

Figure 3.2 PDM Register operation on a new object
3.5.2. *Release a product data object to a local PDM system*

A new object can also be directly released to PDM by a user who must have privilege to perform the operation.

![Diagram of PDM Release operation on a new object](image)

Figure 3.3 PDM Release operation on a new object
3.5.3. **Check-out a product data object in a local PDM system**

A checked-in product data object can be checked-out by system user for modification. The object state and version of the object are updated automatically by the system.

![Diagram](image)

**Figure 3.4** PDM Check-Out operation on a Checked-In object and update object version
3.5.4. Check-in a product data object in a local PDM system

After modification of the product data object, the object should be checked-in to the PDM system for other users to retrieve, update, release and perform other operations.

![Diagram](image)

Figure 3.5 PDM Checked-In operation on Checked-Out object
3.5.5. *Check-out a product data object in a local PDM system*

A modified and checked-in can be check-out again by a user for further modification. New object version and object state will be updated by the system.

![Diagram showing PDM Check-Out operation](image)

**Figure 3.6** PDM Check-Out operation on a Checked-In object and assign a new object version
3.5.6. Check-in product data object in a local PDM system

After modification, the data object is checked-in to the PDM system with a new content and object state.

Figure 3.7 PDM Check-In operation on a Checked-Out object with new assigned object version
3.5.7. *Release a product data object in a local PDM system*

Having been modified, the object is released and the object version and state is updated.

Figure 3.8 PDM Release operation on a Checked-In object and update the object release version
3.5.8. *New Release a product data object in a local PDM system*

New Release operation is employed for updating a released object. It updates the object version and state.

Figure 3.9 PDM New Release operation on released object and assigning a new object version
3.5.9. *Check-in a modified product data object in a local PDM system*

Checked-in operation is employed to update the object state and content to the PDM system.

![Diagram of PDM Check-In operation]

Figure 3.10 PDM Check-In operation on a Checked-Out object with the new assigned version
3.5.10. **Check-out a modified product data object in a local PDM system**

A modified object is checked-out again for further modification. The object version and state is updated.

![Diagram of PDM Server Database and PDM Client Database (PDM Workspace) showing check-out process]

Figure 3.11  PDM Check-Out operation on a Checked-In object and assigning a new object version
3.5.11. Check-in a re-modified product data object to a local PDM system

After further modification, the object is checked-in to the PDM system. It allows the other user to access or check-out.

![Diagram showing PDM Check-In process]

Figure 3.12 PDM Check-In operation on a Checked-Out object with the new assigned object version
3.5.12. **Release a modified product data object to a local PDM system**

Having finished modification, the object is released to update its version and content by an authorized person.

![Diagram of PDM Release operation on a Checked-In object and update the object release version](image)

Figure 3.13 PDM Release operation on a Checked-In object and update the object release version
3.5.13. *Obsolete a product data object in a local PDM system*

Obsolete operation is performed to freeze the object. This prevents all people from accessing the object.

![Diagram](image)

Figure 3.14 PDM Obsolete operation on a released object and changing the object state
3.5.14. *Delete a product data object in a DPDM system*

Delete operation is employed to remove an object from the PDM system.

![Diagram of PDM delete operation](image)

Figure 3.15 PDM Delete operation on an object
3.6. Statechart - A Visual Formalism for PDM System Model

For a long time, people have understood the appropriateness of using the event-state approach to specify complex systems. Despite the appropriateness of the approach, they have lacked a notation that avoids large numbers of states and event arrows and one that is rich enough to allow modularity and abstraction. The lack of these basic properties has hindered the serious use of the event-state approach in the specification of large systems. Recognizing the limitations of state transition diagrams (STDs), David Harel proposed an extension to the STDs [Harel 1998] called statecharts. Statecharts are important because they provide a very rich and expressive notation that allows complex system to be specified concisely and at different levels of abstraction. Statecharts are also part of some object-oriented methodologies such as Object Modeling Technique (OMT) and Unified Modeling Language (UML) [Horrocks 1999].

3.6.1. The Statechart Notation

A statechart is simply a network of states and events. Figure 3.16 shows a simple transition from one state to another. This is interpreted as: “if event E1 occurs in state S1 and condition C1 is true at the time, then make the transition to state S2”.

![Figure 3.16 Simple state transition](image)
Figure 3.17 shows a state transition with action. If, in a system, state S1 and event E1 occurs, the system performs the action A1 and the state is transformed to state S2.

![State Transition Diagram](image)

**Figure 3.17 State transition with action**

A way of using conditions to guard state transitions is to employ the condition connector. An arrow enters the connector, labeled with the triggering event, and the connector may have several exit arrows, each labeled with a condition enclosed in square brackets and optionally also with an action. Figure 3.18 shows a condition connector.

![Condition Connector Diagram](image)

**Figure 3.18 A condition connector**
3.6.2. Statecharts for PDM system

In this section, Statechart is employed to model the dynamic view of PDM system. It is a behavioral modeling tool for Concurrent and Real-Time Systems. It is a form of state transition diagram (STD) that allows for nesting, orthogonality (concurrency) and broadcasting, thus reducing the complexity of many STDs. It also emphasizes flow of control among interacting objects.

3.6.2.1. Register new product data object to a local PDM system.

To register a product data object, a Register event is triggered. If the product data exists, the object will be migrated to PDM system database and its state will be updated to Checked-In.

![Statechart Diagram]

Figure 3.19 Register a new product data to PDM system database and vault. Corresponding graphical representation is shown in Figure 3.2.
3.6.2.2. Release a new product data object to a local PDM system

To release a product data object, a Release event is triggered. If the user has the right authority, the object state will be updated to Released.

![Diagram](image)

Figure 3.20 Release a new product data. Corresponding graphical representation is shown in Figure 3.3.

3.6.2.3. Check-out a product data object in a local PDM system

To check out a product data object, a Check-Out event is triggered, the object state will be updated to Checked-Out.

![Diagram](image)

Figure 3.21 Check-Out a Checked-In product data for modification or update. Corresponding graphical representation is shown in Figure 3.4.
3.6.2.4. Check-in a product data object in a local PDM system

To check in a product data object, a Check-In event is triggered, the object state will be updated to Checked-In.

![Diagram](PDM_STATES)

Figure 3.22 Check-In the Checked-Out product data after modification or update. Corresponding graphical representation is shown in Figures 3.5, 3.7, 3.10, 3.12.

3.6.2.5. Release a product data object in a local PDM system

To release a Checked-In product data object, a Release event is triggered. If the user has the right authority, the object state will be updated to Released.

![Diagram](PDM_STATES)

Figure 3.23 Release a Checked-In product data. Corresponding graphical representation is shown in Figures 3.8, 3.13.
3.6.2.6. *New Release a released product data object in a local PDM system*

To check out a Released product data object, a New-Release event is triggered, the object state will be updated to Checked-Out.

![Diagram of PDM_STATES](image)

Figure 3.24 Check-Out a released product data for modification or update. Corresponding graphical representation is shown in Figures 3.6, 3.11.

3.6.2.7. *Obsolete a product data object in a local PDM system*

To obsolete a product data object, an Obsolete event is triggered and the object state will be updated to Obsolete.

![Diagram of PDM_STATES](image)

Figure 3.25 Obsolete a Released product data. Corresponding graphical representation is shown in Figure 3.14.
3.6.2.8. *Delete a product data object in a local PDM system*

To delete a product data, a Delete event is triggered and the object will be deleted.

![Diagram showing PDM states with deletion event](image)

**Figure 3.26** Delete product data from PDM system database and vault. Corresponding graphical representation is shown in Figure 3.15.
3.6.2.9. *Operations in a local PDM system*

To perform PDM operations in a PDM system, the system state change is shown in Figure 3.27.

![Diagram of PDM system states](image)

**Figure 3.27** Overall operations and state transitions in a PDM system
CHAPTER 4  DPDM OPERATIONS AND STATECHART REPRESENTATION

Good product data management is very important, especially when the design house and production department are located in different geographic regions. Poor PDM easily causes the use of outdated and wrong product data, which finally results in problems in product quality. Poor PDM also causes higher scrap rate, late delivery, increase in production cost, decrease in sales and finally lost of order. Non-distributed PDM system cannot manage the distributed product data in remote factories. DPDM is required, so that they can manage the product data in a synchronized fashion, and ensure a smooth product development process.

4.1. Introduction to DPDM

DPDM system should consist of all PDM functions in order to manage the product data locally. In addition, remote product data should be managed transparently to remote users. DPDM should not only provide data management, but also a channel for data transfer. DPDM should break down geographical barriers, let the authorized people to modify data remotely and provide data when and where people need it for better decision-making. DPDM will be implemented over the Internet. Today, the Internet is the most important computer network. It is a network of networks, that is, computer systems all over the world are linked together so that data can be passed between them. It has the ability to link up a virtually unlimited number of databases on a global scale.
4.2. New Product Data of DPDM

In a DPDM system, all product data object should have the "site" attribute to store the distributed information of the data object. The information allows distribution operation to be performed. This object property guarantees the operation (e.g., update, delete, etc.) can be propagated to all distributed data. It enables the system to achieve the distributed product data management.

4.3. DPDM Functionalities

DPDM functions are similar to those in a local PDM system. Some of the management work will only be done on the server-side of the DPDM system. This can reduce duplicate work and network traffic load. However, some significant PDM functions mentioned in the following must be retained for managing the necessary engineering data.

Structure Management

In a distributed production plant, it is important to well manage the complex document interrelationships and have a complete product structure. Therefore, DPDM Structure Management emphasizes on Document Structure and Product Structure Management. The other functions such as Classification will only work on the server-side of the DPDM system. There is no need to reclassify the product data in the distributed client.
Retrieval Management

Searching and Viewing functions must be provided in the whole DPDM system. People should be allowed to search and view product data from both the local PDM database and the remote server-side master database.

Release Management

Authorization function is important in DPDM system. It protects the important data from unauthorized modification. Other functions like Sign-off and Status Control can be carried out in the server-side of the DPDM system. It is because these two functions will be used mainly in the product design or pre-production stage. The release authorization should only be granted to the super user and the project leader.

Change Management

Version or engineering changes will always occur in production. The DPDM system of a distributed production environment should thus provide the function of Engineering Change Order, Version Control, Change Process, Mark-up and Redlining.

Workflow Management

Process Management and Planning play an important role in product data management and production. A good process management and planning should be done by distributed PDM system to ensure a smooth production.
4.4. New DPDM System Functions

*Product Data Distribution*

Distribution is a key function for a DPDM system to be implemented in a distributed manufacturing environment. The function is based on the replication and migration techniques. Authorized DPDM users can apply the function to distribute the right product data to the right remote site.

*User authentication*

Since DPDM system is integrated with the web technology and implemented on the open computer network, the Internet, DPDM system users will include the project team members, corporate users, vendors, suppliers and public users. Therefore, user authentication is the most important function to protect the valuable product data. It prevents product data from unauthorized access. The system security function should be developed based on a security model for DPDM.

4.5. The DPDM System Operations

The DPDM operates more or less like the PDM system. DPDM operation is carried out remotely through the Internet. Since product data is in the form of electronic file, all product data documents will be treated as data objects. If the DPDM system user wants to request a remote product data object, he/she should first make communication with the distributed PDM system. Remote PDM database connectivity should then be set up. The request will then be
processed via the Internet. After the required object has been transferred from the distributed PDM system, the requester can view the file. If the user wants to modify the product data, he/she can check-out the document from the remote PDM system via the network. Then he/she can modify the document if he/she is authorized to do so. After modification, the user should check-in the object back to the remote PDM system through the Internet again.

4.6. Graphical Representation of DPDM System Operations and Data Objects

The DPDM operations and the object state transitions will be shown in this section. The operation not only work on local site data object, but also on remote site data object.

4.6.1. Register a new product data object to DPDM system

![Diagram of DPDM Register operation on a newly created object and assigning object version in a local site]

Figure 4.1 DPDM Register operation on a newly created object and assigning object version in a local site
4.6.2. Release a new product data object to DPDM system

![Diagram of DPDM system and release operation]

Figure 4.2 DPDM Release operation on a newly created object in a local site
4.6.3. **Release a new product data object to both local and remote sites**

Figure 4.3 DPDM Release operation on a new object to both local and remote sites
4.6.4. Retrieve product data object in a local site within a DPDM system

Figure 4.4 DPDM Retrieval operation on an object in a local site
4.6.5. Retrieve a distributed product data object from remote site

![Diagram showing the retrieval process of a distributed product data object from a remote site.](image)

Figure 4.5 DPDM Retrieval operation on an object located in remote site
4.6.6. Check-out product data object in local site for update

![Diagram of DPDM Update operation on an object with new object version](image)

Figure 4.6 DPDM Update operation on an object with new object version

4.10
4.6.7. Update product data object in a local site

(Figure to be Cont'd)
Figure 4.7 DPDM multi-Update operation on a Released object in a local site
4.6.8. **Update product data object in both local and remote sites**

![Diagram of DPDM Update operation on a local object located in both local and remote sites](image)

Figure 4.8 DPDM Update operation on a local object located in both local and remote sites
4.6.9. Update product data object in remote site

Figure 4.9 DPDM Update operation on a remote object located in remote site
4.6.10. **Obsolete a product data object**

![Diagram](image)

**Site** \( j = 1 \)

**DPDM Server Database**

<table>
<thead>
<tr>
<th>Id</th>
<th>Version</th>
<th>State</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.0</td>
<td>Released</td>
<td>CT</td>
</tr>
<tr>
<td>1</td>
<td>1.1</td>
<td>Released</td>
<td>CT</td>
</tr>
<tr>
<td>1</td>
<td>1.1</td>
<td>Obsolete</td>
<td>CT</td>
</tr>
</tbody>
</table>

**DPDM Client Database**

**Site** \( j = 1 \)

Figure 4.10 DPDM Obsolete operation on an object
4.6.11. Delete a product data object

Figure 4.11 DPDM Delete operation on an object
4.7. Statechart - a visual formalism for DPDM system

In this section, state chart technique is employed to model the various DPDM system operations.

4.7.1. Register a new product data object to DPDM system

To register a product data object, a Register event is triggered. If the product data exists, the object will be migrated to DPDM system database and its state will be updated to Checked-In.

![Diagram](image)

Figure 4.12 Register a new product data to a DPDM system database. Corresponding graphical representation is shown in Figure 4.1.

4.7.2. Release product data object in DPDM system

To release a Checked-In product data object, a Release event is triggered. If the user has been granted the authority, the object state will be updated to Released.

![Diagram](image)

Figure 4.13 Release a Checked-In product data. Corresponding graphical representation is shown in Figure 4.2
4.7.2. Release product data object to both local and remote sites

To perform local and remote release operation in a DPDM system, the DPDM system state change is shown in Figure 4.14.

![Diagram of DPDM system states](image)

Figure 4.14 DPDM system states of Release processing in local and remote sites. Corresponding graphical representation is shown in Figure 4.3.

4.7.3. Retrieve product data object from a local site in DPDM system

To retrieve a product data object in a local site, the local Retrieve event is triggered. If the object exists, the object will be checked out and its state will be updated to Checked-Out.
4.7.5. DPDM Product data retrieval in a local site

To perform local retrieval operation in a DPDM system, the system state change is shown in Figure 4.16.
4.7.5. Retrieve product object data from a remote site in DPDM system

To retrieve a product data object from a remote site, a remote Retrieve event is triggered. If the object is exists, the object state will be updated.

Figure 4.17 Retrieve Released product data from remote site. Corresponding graphical representation is shown in Figure 4.5.

4.7.7. DPDM Product data retrieval in a remote site

To perform remote retrieval operation in a DPDM system, the system state change is shown in Figure 4.18.
Figure 4.18 DPDM system states of retrieval processing in remote site. Corresponding graphical representation is shown in Figure 4.5.
4.7.8. Update product data object in a local site within DPDM system

To update a released product data object in a local site, a local update event is triggered. If the object exists and is not being updated by another, the Check-Out operation will be performed to check out the object for update. Further update can be done by performing Check-In/Check-Out operations. Having finished the modification, the object will be released and its state will be updated to Released again.

![Statechart Diagram]

Figure 4.19 Update product data in local site. Corresponding graphical representation is shown in Figures 4.6. and 4.7
4.7.9. Update product data object in a remote site within DPDM system

To update a remote product data object, remote update event is triggered, the operation and object state change is similar as mentioned in section 4.7.8

Figure 4.20 Update product data in remote site. Corresponding graphical representation is shown in Figure 4.8
4.7.10. DPDM Product data update in a local site

To perform local update operation in a DPDM system, the system state change is shown in Figure 4.21.

![Diagram of DPDM system states](image)

Figure 4.21 DPDM system states of update processing in local site Corresponding graphical representation is shown in Figure 4.7
4.7.10. DPDM Product data update process in a remote site

To perform remote update operation in a DPDM system, the system state change is shown in Figure 4.22.

Figure 4.22 DPDM system states of update processing in remote site. Corresponding graphical representation is shown in Figures 4.8 and 4.9.
4.7.11. Obsolete product data object in DPDM system

To obsolete a product data object, an Obsolete event is triggered. If the object exists and is not being locked by another, the object state will be updated to Obsolete.

![Statechart Diagram]

Figure 4.23 Obsolete Released product data. Corresponding graphical representation is shown in Figure 4.10.

4.7.13. Obsolete product data object in both local and remote sites

To perform local and remote Obsolete operation in a DPDM system, the DPDM system state change is shown in Figure 4.24.
Figure 4.24 DPDM system states of Obsolete processing in both local and remote sites. Corresponding graphical representation is shown in Figure 4.10.
4.7.14. *Delete product data object in DPDM system*

To delete a product data object, a Delete event is triggered. If the object is not in use, the object will be deleted.

Figure 4.25 Delete product data. Corresponding graphical representation is shown in Figure 4.11.
4.7.15. *Delete product data object in both local and remote sites*

To perform local and remote delete operation in a DPDM system, the DPDM system state change is shown in Figure 4.26.

---

**Figure 4.26** DPDM system states of delete processing in both local and remote sites. Corresponding graphical representation is shown in Figure 4.11.
CHAPTER 5  OBJECT-ORIENTED PRODUCT DATA
MODEL FOR DPDM SYSTEM

5.1. Introduction

In developing a DPDM system, a product data model is required. The product data model will explain the relationship among the product data and the various activities in a DPDM system. In product development, different types of product data are generated and the data objects are usually very complicated. Despite of some semantic product data models [Shaw 1989, Sadlbauer 1992], an object-oriented product data model is required. This kind of model can make use of object-oriented technology for DPDM system development. The primary purpose of using object-oriented model is to provide a sufficient description and specification to enable developers to deploy and build the system. An object-oriented model is a representation of the structure and behavior of a system. It is the blueprint that serves as the specification for the system developer who will build and maintain the system. It also helps to map the system design phase to actual implementation phase. This chapter presents a product data modeling framework for DPDM system and shows how the inter-relationship between product data object and the various system activities. Object Modeling Technique (OMT) is used to model the object characteristic domain of product data. The object model, functional model and dynamic model are presented to provide a framework for object-oriented product data model.
5.2. Use of Object Model Technique (OMT)

A number of object-oriented modeling methods have already existed [Linag 1998, Booch 1994]. They differ mainly in their expressive notation and design steps. One of the most popular object-oriented system development techniques is the OMT [Martin 1996]. It is primarily used by system developers supporting full life-cycle development, and targeting object-oriented implementations. Because of its simple core notation, OMT has proven easy to understand, to draw, and to use. The object-oriented paradigm using the OMT, spans the entire development process, so there is no need to transform from one type of model to another.

5.2.1. OMT object model notation

The object model captures the static structure of the objects within the system, relationships between the objects, and the attributes and operations that characterize each class of objects. The followings are the key notation of the object model.

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Attributes</th>
<th>Operation</th>
</tr>
</thead>
</table>

Figure 5.1 Class
Figure 5.2 Association

Figure 5.3 Multiplicity of association

Figure 5.4 Generalization (Inheritance)

Figure 5.5 Aggregation
5.2.2. OMT functional model notation

The functional model describes computations within a system. The functional model is represented by data flow diagram. A data flow diagram is a graph showing the flow of data values from their sources in objects through processes that transform them to their destinations in other objects, Figure 5.6.

![Data flow diagram](image)

Figure 5.6 Data flows between processes in the functional model

5.2.3. OMT dynamic model notation

The dynamic model is graphically represented with state diagrams, Figure 5.7.

![State diagram](image)

Figure 5.7 State diagram in the dynamic model

5.3. Object-oriented Product Data Model

The appearance of product data in a DPDM system contains the essential characteristics of object-oriented technology. In this research study, we apply OMT to describe these properties.
5.3.1. **OMT for Product Data Object Model**

OMT method can provide the developers different views of the system. They include object model, functional model and dynamic model. Object model is the fundamental model of OMT methodology. It is concerned with static structure aspects of the system i.e. class, objects, object identity, attributes, association, operation, inheritance, complex object and encapsulation.

5.3.1.1. **Class**

Class is used to define the class of objects. It contains a set of objects with common structure and behavior. Class is used to abstract product data. For example, a product may consist of a number of assemblies. Then ASSEMBLY is the abstracted form of assembly1, assembly2, ..., assemblyN. The assembly1, ... , assemblyN are particular instances of the general concept of ASSEMBLY.

5.3.1.2. **Object identity**

In our product data modeling, some of the candidate classes of objects are identified by identifiers as ASSEMBLY, DRAWING, CUSTOMER, PRODUCT, PARTS, MARKETING, etc. The identifiers contrast with the traditional database system where values of the attributes defined as primary keys are used to identify the objects. According to OMT method notation, object class is consisted of class name, object attributes and object operations (Figure 5.1). The object classes for product data modeling are shown in Figure 5.8.
### Attribute

Attributes are used to describe the state of an object. It consists of two parts, name and value. The attribute value is itself an instance object. Take an assembly object as an example, the attribute name and its value can be:
assembly No : 0051,
assembly name : cylinder1,
designer : John,
material : aluminum alloy1,
...

5.3.1.4. Object

We can loosely define an object as something which a user is interested in. An object represents a self-contained unit, entity, or aspect of interest of the modeled application domain. It is uniquely identified by an object identifier. It has encapsulated properties, data and operations.

5.3.1.5. Encapsulation

An important feature of an object is that in an object-oriented data model, classes are used to describe a collection of objects having the same properties. Encapsulation is used to introduce abstract data type property of a class. The property implies that an object can only be directly altered or accessed by operations associated with the object. It also means that an object has a public interface and a private (user transparent) implementation. Figure 5.9 shows an encapsulation of an object. Product data is “inside” the object and one cannot obtain the data directly. The check-out operation can only be performed by specific “check-out” method after receiving a message triggered by the requester.
5.3.1.6. Complex object

Complex object always appears in CAD/CAM. It consists of an entity that is composed hierarchically of several subentities that themselves may be composed of subentities via a hierarchical relationship. Complex object in manufacturing domain is complex due to two reasons. The first reason is that an object is often defined in terms of a number of components, or many other objects. As shown in Figure 5.10, an example of complex object, an engine, is defined in term of a number of pistons, cylinders and other parts. Actually, the cylinder head is itself a complex object and is composed of a number of rockers, washers, and other parts. The second reason is the heterogeneity aspects of objects in some class or group. It is because in some operations, the operating parameters are different even for the same operation.
Figure 5.10 An example of complex object in manufacturing domain

5.3.1.7. Object association

Having identifying the object in data modeling, we need to identify the functional structural relationship among them. Association is a semantic link between classes. It denotes a semantic dependency but does not state the dependency direction and relating way of the classes [Booch 1994]. In OMT graphic notation (Figures 5.2, 5.3, 5.5), an association is represented by the line between classes, Figure 5.11.

Figure 5.11 Object Association
5.3.1.8. Object inheritance

Object inheritance is another class relationship to be used in developing a product data model. Inheritance enables one to request an operation on a member of a class, even though the instances belong to a subclass. It also supports polymorphism, an object-oriented characteristic that allows one to develop a system where different objects respond differently to the same message. The abstraction of inheritance allows the sharing of attributes, operation or behavior among classes. It creates a hierarchy of abstraction. Through generalization method, the common attributes and operations are added to superclass where all the common aspects are inherited by its subclasses. Figure 5.12 shows how object inheritance simplifies the object classes according to OMT notation (Figure 5.4). Object inheritance also helps code/object reusability and provides a better structure for the product data model.

![Object Inheritance Diagram](image)

Figure 5.12 Object inheritance
5.3.2. Functional Model

The functional model is used to specify the meaning of activities and data flow in an object model. DPDM system is used to control and manage all data generated by different processes and the data flow in the product life cycle. Managing the product data in DPDM system involves some activities. Figure 5.13 provides an overall view and shows how DPDMS manages and controls a product data object [Martin 1996].

![Data flow diagram for DPDMS](image)

Figure 5.13 Data flow diagram for DPDMS

Within a DPDM implemented manufacturing system, product data flows in different activities. In Chapter 4, graphical model has been employed to represent the DPDM operations between the user and the DPDM system. In this section, functional model will describe the relationship between activities in the application level, especially the *Distribution* activity (object), the new DPDM function. Figure 5.14 shows the outer data flow diagram of a product development cycle in a distributed manufacturing system.
Figure 5.14 Distributed product development cycle under DPDM system environment

Figure 5.15 shows the inner data flow for the Distribution activity. Actually, the data flows in the diagram corresponds to attribute values in the objects represented by the bubbles.

Figure 5.15 Inner data flow of the distribution activity under DPDM system environment
5.3.3. Dynamic Model

Dynamic model concerns with the temporal relationships between objects, their changes over time and sequences of operations [Martin 1996, Rumbaugh 1991]. State diagram is used to represent the key concept of this model. Figure 5.16 illustrates an overall view of the state transitions in DPDM system.

Figure 5.16 State transition diagram of DPDMS

In Chapter 4, statechart has been employed to model the state transition of object when performing different DPDM operations. This section describes the state transition when the activities occur, especially the Distribution activity. Figure 5.15 shows what is happening in the distribution activity. The state transition diagram in Figure 5.17 shows the temporal and dynamical aspects of the system when they happen.
Figure 5.17 State transition diagram of the distribution activity

From the functional and dynamic model, we have a complete picture of the object-oriented product data model.
CHAPTER 6  PRODUCT DATA ALLOCATION IN A DPDM SYSTEM

In this chapter, a distributed object-oriented database architecture is proposed for the DPDM system use. In the object-oriented database, all real-world entities are modeled as objects. Product data object is the most logical unit of distribution in a DPDM system. Replication and migration techniques can be used to distribute the object efficiently. Since the allocation of object in a distributed manufacturing environment is a critical issue in a DPDM system, an optimal object allocation policy is needed. In this chapter, object allocation objectives for DPDM are set to facilitate object replication and migration decision making. A comprehensive mathematical modeling approach is developed for product data object distribution within a DPDM system. The proposed model is newly derived with reference to the distributed object model [Nebro 1999].

6.1. Migration and Replication Techniques for Product Data Object Distribution

In a PDM system, database and electronic vault are the cores of the system (Figure 6.1). PDM system database stores the meta-data of physical product data. It contains the data object identity (OID), pointers to product data, relationship between product data, product structure relationship and administrative data. An electronic vault is a secure area used to store the physical product data.
In a DPDM system, both meta-data and the application data are distributed to remote site together. For simplicity, we define a product data object to be a combination of meta-data and physical product data. Indeed, product data and its meta-data come hand in hand. In an object-oriented database management system (OODBMS), these objects are the most logical units of distribution. It is rarely necessary to fragment an object for distribution. To increase object availability and performance, we apply replication and migration techniques. Replication makes copies of the same object in different sites and migration moves an object from one site to another [Nebro 1999]. However, the decision to make migration or replication of an object will be a complex task. Therefore, we summarize and analyze the necessity and retrieval frequency of product data as shown in Tables 6.1 and 6.2.
<table>
<thead>
<tr>
<th>Group</th>
<th>Product data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering</td>
<td>Drawing, quality specification, manufacturing processes, operation parameters,</td>
</tr>
<tr>
<td></td>
<td>material specification, testing method, etc.</td>
</tr>
<tr>
<td>Production</td>
<td>WIP report, scrap rate, shipping schedule, inventory, etc.</td>
</tr>
<tr>
<td>Marketing</td>
<td>Weekly sales, market share, forecasting, customer information, etc.</td>
</tr>
</tbody>
</table>

Table 6.1 General grouping of product data

<table>
<thead>
<tr>
<th>Group / data object</th>
<th>Retrievals</th>
<th>Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering / drawing</td>
<td>Average</td>
<td>Rare</td>
</tr>
<tr>
<td>Production / WIP report</td>
<td>Frequent</td>
<td>Frequent</td>
</tr>
<tr>
<td>Marketing / Sales report</td>
<td>Average</td>
<td>Frequent</td>
</tr>
</tbody>
</table>

Table 6.2 Retrievals and updates of product data

From the tables, production data is often retrieved by production engineer and other people in the remote production plant. Therefore, the availability of such data is important to ensure a smooth production. On the other hand, since the approved and released engineering data is seldom updated in a short production period. So the engineering data can be replicated and distributed to remote site’s PDM system when needed.
Since the production and marketing data group have to be updated frequently, replication is not a good way for them. To achieve our allocation objectives, they should be allocated at a central position. In our case, migration techniques should be applied to production data allocation. For instance, the data is migrated to China since it is more relevance to be updated by the production people there. This will then decrease the update access cost and increase the update availability. On the contrary, the marketing data should be kept in Hong Kong’s central office. It is because the data is more frequently updated by the marketing people there.

6.2. Product Data Allocation Objectives for DPDM

According to the previous section, we must set some objectives for data allocation model design. Objective setting is important for both system model design and human management practice. In product development, a lot of product data is required for each stage of product life cycle. How to manage the great amount of product data is not an easy task. In a distributed manufacturing environment, product data management is more difficult and a DPDM system is indispensable. In order to build a DPDM system, to distribute product data in an efficient way and ensure an optimum data allocation model, we set the following product data distribution objectives for one to make reasonable and correct decision.

- **Minimize DPDM system retrieval and update operation cost**: Maximization of locality of application can minimize the communication and operation cost by reducing the amount of product data transfer between sites.
- **Increase production system concurrency:** Replicated product data is distributed geographically and stored in different sites. This increases data and resource sharing, and achieve concurrent engineering in production.

- **Minimize remote object request rate in a DPDM system:** Distributing product data to local site PDM system can reduce remote request rate and increase product data availability. This allows the retrievals to be processed locally even when there is a crash of database in other remote PDM system.

- **Reduce storage cost:** Only the necessary product data is required to be stored in each remote PDM system.

### 6.3. Optimal Product Data Allocation Model

In a DPDM system, how to minimize the operation cost, system response time and remote object request rate are the most critical performance measures.

The following parameters are used in the allocation model:

\[ n \]  
\[ j, k \]  
\[ \phi_j \]  
\[ m \]  
\[ i \]  
\[ \sigma_i \]  
\[ r_{ij} \]

- total number of site
- site index, \( j, k \in [1, n] \),
- the maximum available mass storage space of site \( j \) [megabyte, MB]
- total number of object
- object index, \( i \in [1, m] \),
- size of object \( i \) [MB]
- number of retrieval request for object \( i \) at site \( j \)
$u_{ij}$ number of update request for object $i$ at site $j$

$C_{local}$ unit cost of local access [$$/request]

$C_{remote}$ unit cost of remote access [$$/request]

$s_{ij}$ the storage cost of object $i$ at site $j$

$\delta$ average queuing delay of transferring an object [unit time]

$\mu$ communication channel transferring capacity [MB/unit time]

$\omega$ total number of object retrieval

$R$ total number of retrieval request

$U$ total number of update request

$\gamma_j$ reliability of site $j$

$x''_{ij}$ object allocation switching variable; $x''_{ij}=1$, when object $i$ is allocated in site $j$, otherwise, $x''_{ij}=0$.

$x'_{ijk}$ retrieval request routing switching variable, $x'_{ijk}=1$ when object $i$ is requested by site $j$ but has not been allocated in it, and the request must be routed to a remote site $k$. If object $i$ is located in site $j$, no routing is required, $x'_{ijk}=0$.

The storage capacity constraints on each site $j$ are:

$$\sum_{i=1}^{m} \sigma_i, x''_{ij} \leq \phi_j$$
In a DPDM system, the distributed data objects are allocated in at least one site and can be redundantly allocated at more than one site. The switching variables will thus have the properties:

\[ \sum_{j=1}^{n} x_{ij}^a \geq 1 \text{ and } \]

\[ x_{ij}^a + x_{ij}^r = 1 \]

6.3.1. Minimization of DPDM system operation cost \((C_{\text{operation}})\)

The operation cost consists of local retrieval, local update, remote retrieval, remote update, and storage cost. The retrieval and update operations in PDM system include Register, Check-in, Check-out, Approve, New Release, and Freeze. Figure 6.2 illustrates the system operation cost of a DPDM system.

![Diagram of DPDM system operation cost](image)

**Figure 6.2** DPDM system operation cost

6.7
When a data object $i$ is retrieved or updated from a local site $j$ (i.e. $x^a_{ij} = 1$, $x^r_{jk} = 0$, and $x^s_{ik} = 0$ may or may not equal to 1), the local retrieval cost can be represented as

$$x^a_{ij} r_{ij} C_{local}$$

the local update cost of object $i$ in site $j$ as

$$x^a_{ij} u_{ij} C_{local}$$

and the local storage cost of object $i$ in site $j$ as

$$x^a_{ij} s_{ij}$$

The total local retrieval, update and storage cost for $i \in [1, m]$ becomes

$$\sum_{i=1}^{m} \left( x^a_{ij} r_{ij} C_{local} + x^a_{ij} u_{ij} C_{local} + x^a_{ij} s_{ij} \right)$$

$$= \sum_{i=1}^{m} \left[ x^a_{ij} \left( r_{ij} C_{local} + u_{ij} C_{local} + s_{ij} \right) \right]$$

When a site $j$ requests or updates an object $i$ which is not allocated in itself, $x^a_{ij} = 0$, and $x^s_{ik} = 1$ which implies $x^r_{jk} = 1$ since the request has to be routed to a remote site, $k$. In other words, the remote retrieval cost can be represented as

$$x^r_{jk} r_{ij} C_{remote}$$

the remote update cost of object $i$ via site $j$ and $k$ is

$$x^r_{jk} u_{ij} C_{remote}$$

and the remote storage cost of an object $i$ in site $k$ is

$$x^r_{jk} s_{ik}$$
The total remote retrieval, update and storage cost for \( i \in [1, m] \) and \( j, k \in [1, n] \) is

\[
\sum_{i=1}^{m} \left( \sum_{k \neq j}^{n} x_{jk}^r r_{ij} C_{remote} + x_{jk}^u u_{ij} C_{remote} + x_{jk}^s s_{ik} \right)
\]

\[
= \sum_{i=1}^{m} \sum_{k \neq j}^{n} x_{jk}^r \left( r_{ij} C_{remote} + u_{ij} C_{remote} + s_{ik} \right)
\]

The total local and remote retrieval, update and storage costs of a DPDM system can be represented as the following:

**Local retrieval, update and storage cost is**

\[
\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}^l \left( r_{ij} C_{local} + u_{ij} C_{local} + s_{ij} \right)
\]

**Remote retrieval, update and storage cost is**

\[
\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k \neq j}^{n} x_{jk}^r \left( r_{ij} C_{remote} + u_{ij} C_{remote} + s_{ik} \right)
\]

The total DPDM system operation cost can be represented as

\[
C_{operation} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij}^l \left( r_{ij} C_{local} + u_{ij} C_{local} + s_{ij} \right) + \sum_{k \neq j}^{n} x_{jk}^r \left( r_{ij} C_{remote} + u_{ij} C_{remote} + s_{ik} \right)
\]

The above formula is used to calculate the total DPDM system operation cost. It is the cost to be minimized.
6.3.2. Minimization of DPDM system response time ($T_{\text{response}}$)

The response time is the time required for object retrieval or update operation within the system. We assume that there is enough logical data transfer channel and each has a processing capacity $\mu$. Figure 6.3 illustrates the DPDM system response time partitioning.

![Diagram of DPDM system response time]

Figure 6.3 DPDM system response time

Figure 6.4 shows the different cases of object allocation in local and remote sites within a DPDM system.
In a DPDM system, the object transferring time of an object can be expressed as

\[
\frac{\sigma_i}{\mu}
\]

For Cases 1 and 2 in Figure 6.4, object \( i \) is allocated in site \( j \), so that \( x^a_j = 1 \). On the other hand, since no request routing is required, \( x^a_j = 0 \). Therefore, the local object retrieval time is

\[
\left( \delta + \frac{\sigma_i}{\mu} \right) \times \left( r_y x^a_j \right)
\]

Since the transferring capacity \( \mu \) of a local site is assumed to be very large, the object transferring time is then negligible and the local retrieval time is approximated to its local site queuing delay within a DPDM system.
For Case 3 in Figure 6.4, object \( i \) has not been allocated to site \( j \), so that \( x^u_{ij} = 0 \) and \( x^v_{ik} = 1 \).

Object \( i \) is allocated in a remote site \( k \), the request has to be routed to site \( k \) from \( j \) and therefore, \( x^r_{jk} = 1 \). The time spent in remote object retrieval between site \( j \) and \( k \) is

\[
\left( \delta + \frac{\sigma_i}{\mu} \right) \times \left( r_{ij} x^r_{jk} \right)
\]

The object retrieval time in a DPDM system can be represented as

\[
\left( \delta + \frac{\sigma_i}{\mu} \right) \times \left( r_{ij} x^u_{ij} + r_{ij} x^r_{jk} \right)
\]

\[
= \left( \delta + \frac{\sigma_i}{\mu} \right) \times \left[ r_{ij} \left( x^u_{ij} + x^r_{jk} \right) \right]
\]

The total object retrieval time is

\[
\sum_{i=1}^{m} \left( \delta + \frac{\sigma_i}{\mu} \right) \times \left[ r_{ij} \left( x^u_{ij} + x^r_{jk} \right) \right]
\]

The average system response time for object retrieval within a DPDM system, i.e. can be represented as

\[
T_{\text{response}} = \frac{1}{\omega} \sum_{i=1}^{m} \left( \delta + \frac{\sigma_i}{\mu} \right) \times \left[ r_{ij} \left( x^u_{ij} + x^r_{jk} \right) \right]
\]

where \( \omega \geq m \), since the same object may be retrieved for more than once.

The above formula is used to calculate the average DPDM system object retrieval response time. It is also the time to be minimized.
6.3.3. Minimization of DPDM system remote object request rate \( (R_{request}) \)

An object request consists of retrieval and update request. In a DPDM system, local request is unavoidable, unless the data is not wanted. However, the remote object request will bring a profound effect to the system performance. Therefore, remote object request rate should be reduced to increase system performance and efficiency. Minimization of remote object request rate allows a requested object to be successfully retrieved or updated within a minimum definite time interval in a DPDM system. Figure 6.5 illustrates remote object request in a DPDM system.

![Diagram of DPDM system remote object request]

Figure 6.5 DPDM system remote object request

In formulating the retrieval request rate of an object in a site, it can be expressed in the ratio of retrieval request of an object and the total retrieval request in the system.

\[
\frac{r_{it}}{R} x_{ijk}^r
\]

where \( x_{ijk}^r = 1 \), remote retrieval rate is resulted. Otherwise, \( x_{ijk}^r = 0 \), the requested object is placed on the local site and there is no need for remote retrieval.
Total remote object retrieval request rate for $i \in [1,m]$ and $j \in [1,n]$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \left( \frac{r_{ij}}{R} x_{ijk}^{r} \right)$$

The update request rate can be represented as

$$\frac{u_{ij}}{U} x_{ijk}^{r}$$

Since any data object update request has to be transferred to all sites containing that objects [Lee 1993]. Therefore, total remote object update request rate for $i \in [1,m]$ and $j \in [1,n]$ is

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \left[ \frac{u_{ij}}{U} x_{ijk}^{r} \right]$$

The total remote object request rate in a DPDM system can be represented as

$$R_{\text{request}} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ijk}^{r} \left( \frac{r_{ij}}{R} + \frac{u_{ij}}{U} \right)$$

The above formula is used to calculate the total remote object request rate in DPDM system. It is to be minimized.

### 6.3.4 Reduction of storage cost

In a DPDM system, to achieve the product data distribution objective of increasing data object availability and sharing, the product data must be available in the local PDM system. On the other hand, we must minimize the remote retrieval and update cost in order to operate the DPDM
system. To achieve this objective, we must control the storage of product data in each local site. In a DPDM system, it is important to keep the product data objects to be consistent with its replicas. Therefore, any update of a product data object must be accompanied with the propagation of the update to other replicas, which are located in other sites. For the above product data object distribution objectives, we can achieve them by storing the necessary product data object or its replicas in the site which needs it. In other words, DPDM system provides the necessary data for each local site only. As a result, it can reduce the total storage cost of the system since each local or remote site in the DPDM system do not need to have all data objects in their local PDM systems.
CHAPTER 7  STRATIFIED WORKSPACE FOR DPDM SYSTEM USER

Today, more and more manufacturers have allocated their production systems to different locations, a distributed and collaborative environment is formed naturally. In a collaborative environment, increase amount of data, consistency of data version control, data security control, collaborative awareness and support of synchronization of distributed collaboration are some of the major factors for a successful PDM system. However, the existing PDM systems are just for local use and standalone application. It is not enough for today’s collaborative and distributed application. To enhance the PDM system, a stratified workspace system is investigated to integrate with it. In this chapter, a categorized stratified workspace architecture is introduced. This architecture is defined on the application levels of DPDM system. The categorized features of the stratified workspaces can bring major benefits to security and version management. Stratified workspace is good for managing the processes of operations and stage change. When integrated with web technologies, stratified workspace can also improve and increase both local and remote data sharing. It results in transforming the system from passive data repository to a more active one for collaborative work.
7.1. Problems of Existing PDM System in Distributed and Collaborative Environment

7.1.1. PDM operation processing and data version control

In conventional manufacturing system, one may possibly conduct data processing in a unsuitable area. For example, one should modify the design data in a specific design area. However, he/she finishes the modification of data in the application area. The lack of control of working area and operation process will cause production problem, especially product data management problem of controlling or maintaining the data consistency and version.

7.1.2. User Security control

Conventional manufacturing system does not well define a working area for manipulation and storage of data. Therefore, user can access the product data from different area. This results in access control problem in product data management. For some confidential and sensitive data, it will cause more harmful effect to the enterprise.

7.1.3. Collaboration awareness

In preparing, manipulating and editing product data, conventional system does not provide a specific area for the project team members to express their ideas and opinions. This lowers the collaboration awareness especially in a distributed environment, and results in decrease of collaborative efficiency of project team and the reliability of data.

7.2
7.1.4. Synchronous distributed collaboration

Conventional collaborative system are usually localized, project team members are forced to work face-to-face. On the other hand, more and more manufacturers have relocated their production system in remote locations. Some specific working areas are required for the remote activities. The support of synchronous remote collaboration is significant for complete enterprise DPDM system.

Thus the problem being addressed in this chapter is the lack of well defined data management workspaces. These workspaces are some specific areas for product data management operations. They should provide collaborative and secure working area for project team collaboration. Furthermore, the workspace should support remote collaboration within a corporate DPDM system when integrated with the web technology.

7.2. Shared Workspaces

A DPDM shared workspace system is a collaborative working area for DPDM users. It is a system for product data object storage, update and retrieval but extended with features to support collaborative data sharing. Users can achieve storage, retrieval and update activities, and can work concurrently in the workspace system. DPDM system manages the embedded workspaces system for different projects. It also manages the workspace users since the project team members may involve in several projects and can be a user of several workspaces.
7.3. Workspace-based PDM System Environment

DPDM workspaces allow accessibility from different computer platforms, information sharing and operation in heterogeneous environments. Therefore, they are oriented around computers which are linked by a computer network. Shared workspaces can compose of software of CAD/CAM system, word processor, PDM, and other application software. The capability of shared workspace is determined by its composition. The composition also plays a major role in offering and controlling workspace activities.

Providing a shared workspace is especially important when considering how to support remote collaboration [Tang 1993]. To construct a web-based distributed collaborative workspace environment, information sharing is necessary. With the explosive growth of the Internet, shared workspace can be integrated with the web technologies. Also, based on the web technologies, Intranet can be set up for local internal workspace communication. Making the advantages of the global addressing system, network protocol, distributed technology and local client/server architecture, remote workspace architecture can be achieved and allow communication with each other via the Internet.

7.4. Workspace System Category and Architecture

7.4.1 Categorization of PDM workspace

In the proposed DPDM system, the shared workspace system is partitioned into four workspaces.
(1) **Personal workspace:** a private and independent working area for a user. However, it can also communicate with the Project team workspace to share information. Objects being performed in this workspace cannot be updated by other user. User can create a new object in this workspace. Also, user can check out the object from the Project team workspace to this workspace for update operation.

(2) **Project team workspace:** the master workspace for a project team. The object shareability is greater than in the Personal workspace. All team members can access the data objects, retrieve and update them. A project leader performs the *Release* operation. This operation logically migrates or maps the object to a corporate workspace.

(3) **Corporate workspace:** a workspace opens to the enterprise internally. This workspace has greater shareability than Project team workspace. Every corporate user can access the released objects. However, they can read the object only. Project team must share the objects to other corporate users at critical stage of product development.

(4) **Public workspace:** a workspace opens to public externally. The mapped object in this workspace is managed and controlled by the system manager and the project leader. Apart from corporate users, the workspace users can be vendors, sub-contractors, customers and the general public. To connect the public workspace with the outside companies, an Extranet is suitable for close interaction and communication with the business partners. The data objects in this workspace have the maximum shareability. However, the users can only retrieve and read the objects.
PDM system helps an enterprise to effectively manage their product information sharing, the major challenge facing by today's manufacturers. Figures 7.1 and 7.2 show the conventional and PDM integrated information sharing between different functional groups throughout the entire organization. To advance a PDM system, categorized workspace is integrated for various benefits. Figure 7.3 shows the workspace based DPDM system and information sharing. The classified workspace can facilitate the information sharing management within the system by categorized information sharing. It also strengthens the security of the system with category specific security control.

Figure 7.1 Information sharing in conventional production system
Figure 7.2  Information sharing in a PDM integrated production system

Figure 7.3  Information sharing in a workspace based DPDM system
7.4.2 User classification

In an enterprise, most people need to come into contact with the product data and they have to use the service provided by the DPDM system. Each person requires different data and may perform different DPDM operations on the data object. To manage the usage of product data, the product data users are classified according to the workspace category as shown in Figure 7.4. The user classes are Project team users (e.g. product designers, project leader, etc.), Corporate users (e.g. operators, process engineers, sale engineers, etc.), outsiders (e.g. customers, vendors, etc.). Figure 7.5 illustrates DPDM user class hierarchy and the operation inheritance from their parent classes.

Project team users: they can perform retrieval, storage and update DPDM operations in Personal, Project team and Public workspaces. They create and manage the product data in a DPDM system. A DPDM system super user can perform all DPDM operations and access all workspaces.

Corporate users: they are Intranet users, can perform retrieval operation in Corporate and Public workspaces to retrieve the product data which is needed in production or other business activities.

Outsiders: they are Extranet and Internet users, can perform retrieval operation only in public workspace to retrieve the insensitive product data.
Figure 7.4 Workspace based user classification
Figure 7.5 Class hierarchy of PDM user

7.4.3 Application levels in DPDM system

DPDM workspace architecture is based on the DPDM system application levels. Figure 7.6 illustrates the application levels in DPDM environment. Application level is categorized into four distinct levels. They are Private, Project team, Corporate and Public levels.
Figure 7.6 Application levels in a DPDM system
7.4.4. *Shared workspace hierarchy and web-based model*

Figure 7.7 shows the basic hierarchy of DPDM workspace and illustrates the configuration of each workspace.

1. In Personal workspace: DPDM client integrated with other application tools will be offered for the users to create or update object individually.

2. In Project team workspace: DPDM client and server applications will be offered for the project team. Project team members use the DPDM operation provided by DPDM client application to manage the registered data which are stored in the DPDM server. Project leader will control the “Release” process.

3. In Corporate workspace: DPDM client and browser applications will be offered for the corporate users to retrieve and browse the data object.

4. In Public workspace: Browser application is supported for outsiders to browse the limited data object over the Internet.
Figure 7.7 Basic workspace category hierarchy of DPDM system

Figure 7.8 illustrates the web-based DPDM model, such that its construction is based on the DPDM application levels. It also indicates the object replication and migration flow between each workspace. The users inside a workspace can perform the DPDM operation of the resident workspace. Figure 7.9 shows the integration of the application levels, the workspace hierarchy, workspace user and operations of the shared workspace based architecture of DPDM system.
Figure 7.8 Web-based product data and information flow model of DPDM system
<table>
<thead>
<tr>
<th>Application levels</th>
<th>Workspace hierarchy</th>
<th>Workspace User</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public</td>
<td><img src="image" alt="Diagram" /></td>
<td>Outside companies: - Sub-contractors - Vendors - Customers - Suppliers - Max. shareability</td>
<td>Retrieve</td>
</tr>
<tr>
<td>Corporate</td>
<td><img src="image" alt="Diagram" /></td>
<td>- All corporate user have access permission - High shareability - Read operation only</td>
<td>Retrieve</td>
</tr>
<tr>
<td>Project Team</td>
<td><img src="image" alt="Diagram" /></td>
<td>- All team member have access permission - Leader perform Release operation - Medium shareability</td>
<td>Retrieve and Update</td>
</tr>
<tr>
<td>Private</td>
<td><img src="image" alt="Diagram" /></td>
<td>- Min. shareability - Private access permission</td>
<td>Retrieve and Update</td>
</tr>
</tbody>
</table>

Figure 7.9 Summary of application level based workspace category
7.5. Solutions Provided by the Workspace-based DPDM System

In a DPDM system, various activities of storage, retrieval and update are achieved by performing the appropriate operations. The operations are Registration, Retrieval request, Update request, Check-Out, Check-In, Release, and etc. They are conducted in different stages and working areas. Since many people are involved and will perform different operations to achieve the DPDM activities, there is a need to confine them within specific workspaces where the users have the permission to perform the operations. Shared workspace system can solve the problems in following aspects:

7.5.1. DPDM operation processing and data version control

In a production facility, many information are generated by different operation and they need to be monitored and controlled. Similarly in a DPDM system, DPDM operations are performed in the product data life cycle. A DPDM process control is used to monitor and control DPDM operations. To facilitate the DPDM process control, a categorized workspace system is required as it can provide a specific DPDM operation control and the latest information. DPDM process control is based on the product data life cycle workflow. Workspace can return the current object state, current object version, and current workflow information of the product data object. The process control not only monitors the data object information, but also keeps track of the version. During product data life cycle, many versions are released and put into production. The product data information returned from each workspace is crucial for data version management. It not only affects single data version synchronization, but also the complete data structure.
Version control is a task to manage the version of product data object and configuration when the objects are requested to be modified. When an object is created in a personal workspace, workspace user invokes the Register operation

Reg obj

to migrate the object to DPDM system database and to create a new version for the object simultaneously (initial version number is $v$). If a project team member wants to modify the object, he/she invokes the Check-Out operation

C-O obj

This operation replicates the object to a local operating system database.

To prevent overlapping modifications, the object being modified cannot be checked out by other. It is required to request an update lock (\(\text{-L}\)) from lock manager before checking out the object for modification.

C-O -L obj

Now the user can perform update operation

Ud obj
After finishing modification, the user invokes the Check-In operation to check back the object to the DPDM database.

C-I obj

A pre-release version number is written to the object, the new version is \(v.n_{c-i}\). Where \(n_{c-i}\) is the number of Check-In operation invoked.

For instance, if a user wants other users in the same workspace to edit the object, he/she can check in the object by retaining the lock.

C-I-\_L obj

Retaining the update lock also prevent other workspace users from modifying the same object. Therefore, if other person outside the workspace tries to check out the same object for update, C-O will alert the users with the message:

C-O error: object is being modified by <workspace user>

If other users in the same workspace wants to modify the object. He/She can invoke the C-O and C-I operations to check out and check in the object respectively. The lock is shifted to and retained by the user who is in the same workspace. Every C-I operation will trigger the system to assign a new pre-release number.
\[ v_r, n_{c-i} \]

where \( v_r \) and \( n_{c-i} \) are the release and pre-release version number respectively.

7.5.2. **Workspace security control**

Data security is the protection of data against intentional or unintentional threats using computer-based controls. In DPDM system, the security control should protect the product data in a DPDM system against unauthorized access, disclosure, modification, and etc. One of the strength of shared workspace system is its ability to control access right. All project team members of a shared workspace have all access right which are granted based on classification of the users. It is important for a company to protect proprietary product secrets from competitors. The DPDM system should prevent unauthorized users from accessing the company's sensitive data. To increase the protection of all data, they need to be categorized first. The levels of product data sensitivity can be characterized as Top secret, Secret, Confidential, Classified and Unclassified. They are shown in Figure 7.10.
Figure 7.10 Sensitivity levels of product data

Categorization of product data is based on the data sensitivity, data value, business risk and other criteria. Each category is available in a specific workspace and is protected by the workspace security control. Workspace not only maintains the category and availability of data, it also maintains the product data integrity. Integrity is the protection of data from intentional or accidental unauthorized modification. It depends on the workspace access control to ensure that any modifications are correctly applied on the data object.

In a collaborative DPDM system, product data will be shared between corporate users or even outsiders. Therefore, authorization is indispensable for protecting the data. Each user is granted permissions to conduct operations on data objects. A large number of inter-corporate users and
an unlimited number of outsider will result in an unlimited number of user/permission pairs which increases the difficulty and risk in user access right management.

For the above mentioned problem, workspace based access control (WBAC) is proposed for the collaborative DPDM system. There are two types of authorization association. The first is static association, the association between workspaces and permissions. When permissions are well authorized to the workspaces, the association does not need to change even a new project is launched. The second is dynamic association, the association between workspaces and users. Dynamic association can be broken or created when user is added or removed from the workspace.

We can express the associations of authorizing permission to user as an ordered pair which consists of user (U) as the left element and permission (P) as the right element.

(U,P)

The number of association between users and permissions is the product

|U|*|P|
Figure 7.11 Association of user/permission

where |U| and |P| are the number of users in U and permissions in P respectively, and |U|, |P| > 2. Indeed, it is reasonable that there are more than 2 employees in an organization or production corporate. Otherwise, there is no need for access control.

In a workspace based DPDM system, each workspace is authorized a class of permission, so that the permission class $P^c$ can represent the workspace (W) and the association between user and workspace can be expressed as

$$(U^c, P^c)$$

where $P^c$ represents workspace.
The number of user/workspace association is $|U^c|$, and the workspace/permission association is $|P^c|$

Figure 7.12 (a) and (b) show the association of user/workspace and workspace/permission

Figure 7.12 illustrates that the number of association of user/workspace and workspace/permission of each user in user class $U$ and each permission in permission class $P$ is $|U^c| + |P^c|$

Figure 7.13 Association with workspace for each user, and associations with workspace for each permission
In a workspace based environment, WBAC reduces the number of associations since

$$|U_i| + |P_i| < |U_i| \cdot |P_i|$$

where $|U_i|$, $|P_i|$, $|U|$, $|P| > 2$, which are real life case in most corporate. The decrease in associations lowers the prone to error and administrative cost. The advantages provided by WBAC are more obvious when the number of individuals ($n_{ind}$) increases in an organization.

In particular,

$$\sum_{i=1}^{n_{ind}}(|U_i| + |P_i|) < \sum_{i=1}^{n_{ind}}(|U_i| \cdot |P_i|)$$

7.5.3. Workspace collaboration awareness

We define collaboration awareness as the understanding of the potential for collaboration with colleagues and their up-to-the-moment actions, ideas, and how they interact within workspaces. Nowadays, most projects are too large to locate in a single site. It is obvious that the separation is the obstacle to the communication between local and remote sites. There is an explicit need for shared and collaborative workspaces for remote collaboration. The stratified workspace makes the distant work collaborators to be co-located not only in space but also in time. Since they can express their idea synchronously with the remote co-workers via a computer network.

In a shared workspace, it is apparent that the awareness of other local and remote collaborators will played an important role in the fluidity of collaboration, and is looked on as a way of reducing awkwardness of distributed collaboration. Workspace collaborative awareness can
brings more efficient interaction, less error-prone, improve information exchange and dissemination in a distributed system.

7.5.4. Synchronous distributed collaborative environment

Integrating with the World Wide Web (W3) collaborative technologies, DPDM shared workspace system can be conceived as a means of supporting the work of widely-distributed product data management. The digital information transmitted over the Internet can represent many different types of information including simple text, drawings, graphics, sounds, video, and etc. In order to move this information accurately and reliably, a number of different standard protocols are used to ensure that all information is transferred in the most appropriate manner. By conforming to these standards, every computer in the Internet is assured to have the ability to interpret all the available information. The integration of web technologies not only facilitates data exchange, but also collaborative data sharing over the Internet. This extends the Web from a passive medium into an active cooperation tool.

7.6. MultiNet in a DPDM system

To implement workspace based DPDM system, a collaborative workspace must be provided for the DPDM users to interact effectively. Therefore, it is necessary to employ the Multi-Net technologies to link up all collaborative workspace in both local and remote sites. The Multi-Net includes the Internet, Extranet and Intranet.
Internet: The Internet is a network of networks, that is unlimited number of computer system on a global scale. It is free to use and is not owned by anybody entirely. Internet provides the communication highway for the DPDM system.

Intranet: An Intranet is a private application of the same internetworking technology, software, and applications within a private network. It is contained within an enterprise. It may consist of many interlinked local area networks. It may or may not include connections through one or more gateways to the outside Internet. The main purpose of the Intranet is to share company information and computing resources among employees. An Intranet uses TCP/IP, HTTP, and other Internet protocols. In general, it looks like a private version of the Internet.

Extranet: An Extranet is extending the Internet to an enterprise’s business partners. It is a use of Internet/Intranet technology and the public telecommunication system to share information with other business partners, suppliers, customers, consultants, and etc. An Extranet can be viewed as part of a company’s Intranet behind a firewall extended to outsiders by user authentication, an Extranet allows limited access for outsiders by granting privilege to pre-specified persons or groups through the use of passwords, making it possible to control the access to sensitive data. Typically, an Extranet will link the Intranet of distributed organizations for the purpose of conducting business. An Extranet requires security and firewall server management.
CHAPTER 8 SECURITY MODEL FOR DPDM SYSTEM

Distributed product data management (DPDM) systems can make product data a valuable and available commodity for many different kinds of computing applications in production. However, these also pose new security risks. It is because all people in a manufacturing enterprise will somehow get in touch with the DPDM system. Dealing with these tremendous amount of interaction between the system and various users, the utmost importance is to ensure that all data are secured and all users are under controlled and managed. Nowadays, security of PDM systems is of great concern to individuals and organizations, though for different reasons. Manufacturing organizations are mostly interested in the security and secrecy of product data. These concern and interest have driven the development of computer security technology.

The spectrum of applications for PDM system is wide. It ranges from business office document management systems to transportation company and computer-supported cooperative work (CSCW) system. As a result, the system will often become repositories for potentially sensitive users and corporate product data. To leave this data unprotected for everybody to see is clearly undesirable. Further, product data will be used to guarantee a smooth production, and directly or indirectly as import for decision-making processes throughout the product lifecycle. Hence, the security of product data is important. It is concluded that product data needs to be protected against unauthorized access, disclosure and modification. However, the exact level of protection varies widely from context to context. Different user groups will also have different requirements in
secrecy and security. Therefore, in this chapter, focus will be given on developing a model for specifying security. The model will be suited to address the requirements of DPDM systems.

This chapter discusses security requirements faced by a DPDM system in different organizational contexts. It is argued that access control requires a user management, workspace and security model to specify. The prominent supporting features of the system such as user organization, workspace and security are outlined. A mixed approach access model for the system is proposed. In this model, user management and two main classical access control methods, the Lampson’s access matrix and the Bell-LaPadula’s (BLP) security labels, are analyzed and adapted to application with multiple system users and product data in order to be applicable to workspace-oriented DPDM system.

8.1. Requirements

In this section, three usage scenarios for a DPDM system will be discussed. These scenarios highlight different deployment environments and the resulting sets of requirements. Special concern are put on DPDM user management and the balance between security imposed by the system (mandatory security), and security specified by individuals (discretionary security).
8.1.1. Scenario 1: Project-based DPDM system and User Management

In manufacturing companies today, whatever its size, project based approach is employed for most manufacturing system. Therefore, project management is required to manage and control the complex and diverse activities of the projects, in spite of all risks or dangers.

When a manufacturing project is conducted, three objectives should be achieved. They are the quality, time and budget. To achieve the quality objective requires competence in engineering and design, but this must be complemented by a good and reliable product data management system to provide the right data. To achieve the time for completion objective, no shortage of information and other resources must be ensured. Obviously, a DPDM system is required to manage and speed up sharing of information and resources. To achieve the budget objective, good product quality and short time for completion are required and they can be complemented by the DPDM system. A DPDM system provides the right product data at the right time by improving data sharing between people and department. This not only reduces the rework cost, rebuild cost and scrap cost, but also allows the project to start and finish on time. A project not started on time can hardly be expected to finish on time. This will then cause an increase in expenditure.

Therefore, the application of sophisticated DPDM computer system is important to the practice of effective project management. In order to achieve the three project objectives with the help of DPDM system implementation, we need to consider the organization of system user in the project. The organization should ensure a clear authority and user
responsibility. Moreover, the user organization will help to control and manage the system user architecture.

8.1.2. Scenario 2: Intra-organization PDM user environment

Within a manufacturing organization, there is often a need for product data in real-time. For example, production operator often uses PDM systems to efficiently get the right data for daily production. Further example of the use of a PDM system within an organization includes computer-supported collaborative work and communication across departments. The common theme in this scenario is that acquisition, management, and use of product data should be ultimately controlled by a PDM project team and a local PDM system. This case is referred to as the intra-organizational scenario. Here, the security policy is set by the organization for the PDM system. Typically, local discretion is permitted only within the bounds defined by the organizational policy. Users and data objects are under controlled within the system.

As far as security within a PDM system is concerned, there are two basic requirements: for individual user and for users belonging to the same group. For example, the system may allow only a certain group of people to access the product data. However, all product data should be visible to the Project leader.

Presently, most commercial PDM system falls into the intra-organization category. However, the provision of a distributed product data management will require a more general and distributed PDM approach.
8.1.3. Scenario 3: DPDM user environment

In contrast to the well-controlled, relatively closed intra-organization PDM environment described earlier, there is another scenario of DPDM for the common distributed manufacturing environment. DPDM users may request PDM operation remotely and their access to the DPDM system are strictly controlled.

The DPDM scenario represents an open system in two ways. Firstly, the Extranet user layer, where the users in this layer are well known to the system. Secondly, the Internet user layer, where the users are unknown to the DPDM system. The problem here will be mainly on DPDM user remote access control and management.

It is expected that the DPDM system is obliged to implement certain generic security policies, such as not allow modification by unauthorized Extranet and Internet users. In addition, the Project leader may need to specify his/her own security policy. For example, Internet user can browse the product outlook or picture but not the detail drawings. These policies would presumably be applicable to all DPDM users. Generic authorization constraints are especially important since the users should now be controlled in both local and remote access.

8.1.4. Scenarios summary

In the above scenarios, security is the main concern. In scenario 1, management of the PDM user is the critical consideration, since improper user management will violate the system security. In scenario 2, the main concern is to protect product data of intra-
organization need from unauthorized local access. In scenario 3, distributed PDM user is the main concern. To cater for all scenarios, a security model should include user management, user access control and workspace management. Furthermore, the system needs to be protected from false product data and the integrity of product data should be maintained against improper modifications.

In the remainder of this chapter, focus will be put on models for the specification of security that are applicable to the three scenarios.

8.2. DPDM System and Security

In a manufacturing company, nearly all corporation individuals require product data for various purpose of their works. Therefore, they have to interact with the DPDM system. The interaction between the classified users and the DPDM system can be specified by their class associations. This is illustrated with an object-oriented model as shown in Figure 8.1. For handling and controlling the tremendous amount of interaction, security control is important to protect the system and the classified data. As to a computer data management system, the security aspect is probably the most important. In a DPDM system, a security model is indispensable to achieve the fundamental security objectives of product data confidentiality, integrity, availability and also the operation tractability.

*Confidentiality:* protects the data from unauthorized disclosure of product data. It allows the data to be accessed only by the authorized users.
**Integrity:** prevents the data from unauthorized modification or deletion as well as creation of data.

**Availability:** provides the product data or allows data access when desired by authorized users.

**Tractability:** records when and who has accessed the data object, how and what operation has been performed on it.

![Diagram](image)

Figure 8.1 Interaction association between DPDM system object classes

In a DPDM system, various users will interact with the system. This is not restricted to intra-organizational users, but also includes users of the outside world such as Extranet and Internet users. Each type of user should have their own access rights. In a DPDM system, a large volume of documents is generated during the product life cycle, such as
product specifications, engineering models and drawings, process plans, manufacturing schedules and instructions, operation and maintenance manuals, customer feedback, etc. Each product data has different degree of sensitivity. For different levels of access right and data sensitivity, different access control methods are more desirable.

The security of the proposed DPDM system depends primarily on the workspace system management and access control. The abstraction is the workspace domain system described in early section. A workspace system will offer the following basic functions:

- Controls user access to product data and provides secure working environments for the system users.
- Provides a secure storage environments for product data with different sensitivity levels.
- Avoids unauthorized disclosure, destruction or corruption of sensitive product data.

The difficulty in specifying a security model over these three functions is to control the various access right of system user to product data with various security levels. Since the proposed classifications are based on workspace, the access control functions are maintained in a consistent manner. As far as the stratification of the workspace system is concerned, both local and distributed access will be considered.

In the following sections, discussion will be on workspace dependent security model for DPDM system with the above specified functions.
8.2.1. Workspace-oriented integration classification

In the proposed organization structure, user and product data classification are workspace-oriented which simplifies the organizational structure. In this conceptual framework, the system is allowed to manage the working environment, system user and the data object in an easy and a secure way. It is because classified workspace protects the system environment from unauthorized access. The classified users are granted with different authority to perform different operations on the classified data objects which are protected by the stratified workspaces. So the workspace integrated classification framework provides a highly secure operational environment for the DPDMS system. Figure 8.2 summarizes the allowable operations performed on the product data objects by different user classes in their respective workspaces.

<table>
<thead>
<tr>
<th>Super User</th>
<th>Project Leader</th>
<th>Project Member</th>
<th>Intranet User</th>
<th>Extranet User</th>
<th>Internet User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open acc.</td>
<td>Open acc.</td>
<td>Register</td>
<td>Retrieval request</td>
<td>View obj</td>
<td>View obj</td>
</tr>
<tr>
<td>Delete acc.</td>
<td>Delete acc.</td>
<td>Retrieval request</td>
<td>Check-out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read log</td>
<td>Grant authority</td>
<td>Check-out terminate</td>
<td>Update request</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grant authority</td>
<td>Retrieve request</td>
<td>Update obj</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open project</td>
<td>Check-out</td>
<td>content</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Close project</td>
<td>Retrieval terminate</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Register</td>
<td>Update request</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retrieval request</td>
<td>Update obj</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Check-out</td>
<td>Approve</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Retrieve</td>
<td>Obsolete obj</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>request</td>
<td>Delete obj</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Close</td>
<td>View obj</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>project</td>
<td>Red lining</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 8.2 Summary for DPDMS operations of each user type
8.2.2. Workspace domain

Following the approach of management domains, a workspace domain is proposed as the workspace-oriented stratifying mechanism for product data. Workspace domain is explicitly defined and bound to well-defined working areas. Workspace domain is associated with product data object. The following convention is used to name the different workspace domain

[<Abstract workspace>] | [<PDM Workspace>] @
[<Workspace>| <Product data object>]

Example:

Intranet @ / Workspace : DWG1001

Explanation:

Drawing object DWG1001 is in the Intranet workspace

Workspace domain provides a named representation for specific working area in the DPDM system. When a data object is Checked-In or Approved to a working area, it will become its workspace data object and is controlled and protected by the security pertinent to the workspace. The set of workspace domain in the system is referred to as the workspace domain space. Although product data state may change in DPDM workflow over the product life cycle, the structure of the workspace domain space does not change. In the following sections, discussion will be on how effective management policies over the workspace domain space can be specified using only that workspace domain with a fixed stratification.
8.2.3 Project team organization structure

In most existing PDM systems, they provide individual, user groups or roles concept in their system [Miller 1997]. However, they do not provide a definite supervision and user management organization structure for a DPDM system.

To facilitate the management of various user types as well as to make the system more secure, a Project Team organization structure is proposed here for a DPDM system. A project team is set up to carry out all the work processes of a project. A project team is specially assembled for achieving the project tasks. The Project leader and the Super user are in direct command, with complete authority for directing the team members. Project leader grants team members with the necessary authority to perform operations on product data within the system. As various tasks of the project are finished, the team size can be gradually reduced until it is finally disbanded when the project is completed. This organization framework is flexible and will draw company resources when only necessary.

DPDM system manages all common and secret product data. The project team organization structure can provide a secret and confidential environment for the system users to work on the product data. Establishment of project team greatly helps the system to contain all product data within closed and secured boundaries. Project team provides a high security environment when product data has to be confined to those working on the project by enclosing the project work in secure areas, and by restricting the total number of users who need to know about the work.
8.2.4. MultiNet user

In existing PDM systems, they only consider local PDM users. In a product lifecycle, both the internal corporate PDM users and the external people are involved. In the proposed user organization structure, apart from considering the internal users of Super user, Project leaders, and Project team members, the MultiNet users of Intranet, Extranet and Internet users are also considered. Intranet user is other corporate user outside the project team, such as corporate CEO who is not in the design project team. Extranet user is a close business partner, such as the subcontractor or supplier. The Internet user is a general public user, such as a virtual customer. Figure 8.3 illustrates the proposed system user classes in an object-oriented manner. Figure 8.4 shows the proposed user hierarchy in the DPDM system. The dashed lines indicate users that are outside the corporate.
Figure 8.3 User classes in a DPDM system

Figure 8.4 User hierarchy in a DPDM system
8.3. Stratified Workspace Schema

Most existing PDM systems provide a single workspace for all users. Access controls are provided to maintain data security based on users or user groups authorization. In a monolithic workspace system, the users see all the objects but can only perform the authorized operation. For example, they can read the data objects but cannot update them in a specific project. The security control is relatively loose. To provide a more secure and easily manageable environment, all users should be classified in the organization and should have their own workspaces. As to product data, the security sensitivity will vary from one to another. It is proposed to classify product data into a stratified workspace system. It is because a stratified workspace system can provide different security levels for different product data. For example, competitor companies may want to access product related data of their rivals (e.g. the marketing plans and trade secrets) in order to gain a sales advantage, these kinds of data will then be classified into a higher security level workspace. Moreover, a user can only access product data of his/her own workspace. Figure 8.5 illustrates the stratified user workspaces in the proposed DPDM system.
8.4. Access Control Schema for Workspace-oriented DPDM System

Traditionally, access control is achieved by either mandatory or discretionary method. Both approaches are based on the user and product data object (subject-object) paradigm. In mandatory access control, a DPDM user is allowed a read access or a write access into a product data object if certain axioms over the security labels of the DPDM user and product data object are satisfied. When using discretionary method, access controls and access matrix with possible DPDM_User-DPDM_Operation-DPDM_Site-Product_Data (UOSD) combinations is constructed. Access right of a DPDM user to perform an
operation on a product data object is granted if the corresponding combination of user, operation, site and product data is a member of a certain access matrix.

In the next two sections, how user management can cope with the DPDM security requirements will be explained.

8.4.1. \textit{UOSD matrix-based access control}

In DPDM domain-based framework, the rules of UOSD matrix control policies are specified in the form of,

\begin{verbatim}
<DPDM user scope>
{Authorized DPDM operations}
<DPDM Site>
<Product data object scope>
\end{verbatim}

Semantically, this policy means that a user from \texttt{<DPDM user scope>} performs \texttt{Authorized DPDM operations} on a product data object from \texttt{<product data object scope>} which is allocated in \texttt{<DPDM site>}.

Example for the policy:

John may Check-Out DWG1001 which is located at Intranet workspace @/ Local site
The policy can also be expressed as

John may Check-Out Intranet workspace @/Local site contains DWG1001

Both of the above expressions specify the authorization that John is allowed to login the Intranet workspace and perform Check-Out operation in a local PDM site. To express the policy in a User-Operation-Site-Data paradigm, additional constraints are added to explicitly specify the required policy in a canonical form:

John \{Check-Out (DATA OBJECT)\} Intranet workspace @/ Local site WHEN DATA OBJECT = DWG1001

This expression specifies that John is authorized to perform the Check-Out (DATA OBJECT) on Intranet Workspace @/Local site when DATA OBJECT equals DWG1001. In this policy, the DPDM operation is Check-Out (DATA OBJECT) and the drawing data DWG1001 is the implicit product data object.

In a DPDM system, user can perform the operation on more than one object at the same time. For example, a user can browse the data object, DWG1001 and DOC1012 concurrently.
John {View} DWG1001,
DOC1012, Intranet workspace @/ Local site

For this situation, multi-target policies of the following form is proposed:

<DPDM User> {DPDM operation}
<Product Data 1>...<Product Data n>

The semantics of such a policy is that a user is authorized to perform operation over
multiple objects consisting of data 1 to data n.

On the other hand, more than one user can perform the same operation on the same data
object at the same time. For example:

John, Tom {View} DWG1001,
Intranet workspace @/Local site

The policy can be expressed as:

<DPDM User 1>...<DPDM User n> {DPDM operation}
<Product Data 1>...<Product Data n>

For completeness sake, a multi-user, multi-data object situation can be expressed as:

<DPDM User 1>...<DPDM User n> {DPDM operation}
<Product Data 1>...<Product Data n>
Multi-user authorization schema describes authorization for multiple DPDM users to perform the same operation together. For example, users John and Tom can perform the View operation on the same object at the same time. A set of policies over users $u$ and product data object $d$ corresponds to a $u+d$ dimensional access matrix. For example:

John, Tom {View} DWG1001, DOC1012

This policy specifies that John and Tom may browse the drawing object DWG1001 and document object DOC1012 together.

For the distributed users, their access authorizations are also controlled by the UOSD matrix. An example for a distributed user access policy is:

Peter may Check-Out DWG1001 that is located at Intranet Workspace @/ Remote site

The policy can also be expressed as

Peter may Check-Out Intranet workspace @/ Remote site contains DWG1001
The constrained policy can also be expressed as

\[
\text{Peter \{Check-Out (DATA OBJECT)\}} \\
\text{Intranet workspace @/ Remote site WHEN DATA OBJECT = DWG1001}
\]

This policy specifies that \text{Peter} is authorized to perform the \text{Check-Out} operation to check-out the implicit product data object in a remote site. Similarly, this policy can also be applied to the multiple user and data object matrix.

### 8.4.2. BLP label-based mandatory access control

Mandatory access control in the workspace-oriented domain framework can be implemented by assigning security labels to object domains. All objects within a domain inherit the domain’s label. If a user is a member of multiple domains, he/she inherits the least upper bound of all his/her parents’ labels. Access is granted whenever the security labels for a user satisfy a certain set of axioms.

Analogous to the matrix-based access control, the user and product data object pair contains the information to decide whether access to a certain workspace is allowed or not. The axioms of the label-based method must also cater for multiple users and multiple data objects policy described in the matrix-based control method.

The security labels being assigned to the domains of classified DPDM users, stratified workspaces and classified product data contains two attributes. One of them is the
sensitivity level (S), which is totally ordered. The other is classification (C), which is partially ordered by subset inclusion. Based on these two attributes, equivalence relationships are defined over the set of values for each attribute. Due to partially or totally ordered of the attribute values, dominant relationship is established. For example, \( u_1 \) dominates \( u_2 \) can be expressed as

\[
\text{Dominate} (u_1, u_2) \iff (Su_1 \geq Su_2) \land (Cu_1 \supseteq Cu_2)
\]

To develop a secure DPDM system, a security model should be constructed and the purpose of this model is to precisely define the desired behavior of the system. Here, the Bell and LaPadula Model (BLP) is adapted and extended [White 1995]. The BLP method has three properties, Simple security property, \(*\)-property, and Tranquillity principle. Since these properties do not completely fit the actual needs of providing a flexible and secure DPDM system, the properties are modified and new property is added to suit the system security needs. For the sake of brevity, the object’s name is used to refer to the object’s label in the dominate predicates. In the DPDM security model, the BLP simple security property is applied in a similar manner.

The level of the DPDM user must be at least the level of the product data object if the mode of access allows the product data object to be Read.
**Axiom 1:** user \( u \) may read data object \( d \) only if dominates \((U, D)\)

\[
\begin{array}{c}
U \\
\xrightarrow{\text{read}}
\end{array}
\rightarrow
\begin{array}{c}
D
\end{array}
\]

iff dominate \((U_i, D_j) \land i = j\)

where \( i \) and \( j \) are project identifiers, \( i = j \) indicates that both user and data object come from the same project.

In order to increase system flexibility in the DPDM system security model, dynamic change of user level in downward direction is allowed. Therefore, a user at a higher level can behave as a user of lower level. This means that user at higher level can perform his/her authorized operations on objects of other lower workspace level. For example, a Project leader can perform Check-In operation on the checked-out object in any other lower user workspace level such as the Project team member, Intranet user, Extranet user, or Internet user workspace. This is the second property of the system security model, *Dynamic user leveling*.

*The level of DPDM user can be transferred to a lower workspace level.*
Axiom 2: user $u$ may write or update data object $D'$ only if dominate $(U, U')$ and dominate $(U', D')$

$$U_i \xrightarrow{\text{write}} D_j$$

iff dominate $(U_i, U'_i) \land$ dominate $(U', D') \land i = j$

where $U$ and $U'$ represent different user classifications.

In order to avoid unauthorized disclosure of product data when updating product data object in a DPDM system, all operations must be strictly under controlled by the system security model through the third DPDM security model property, *Exceptional *-property

Except Super user ($U^S$) and Project leader ($U^{PL}$), the level of the product data must be at least the level of the DPDM user if the mode of access allows the product data to Write.
**Axiom 3:** user $U$ may write or append data object $D$ only if dominates $(U, D)$

\[ U_i \xrightarrow{\text{write}} D_j \]

iff Dominates $(U_i, D_j) \land i = j$

**Axiom 4:** Super user and Project leader may write data object if dominate $(U, D) \lor$ dominate $(D, U)$

\[ U_i^S \xrightarrow{\text{write}} D_j \]

iff dominate $(U_i^S, D_j) \lor$ dominate $(D_j, U_i^S)$ and

\[ i \stackrel{\text{dominate}}{\rightarrow}_j \]

\[ U_i^{PL} \xrightarrow{\text{write}} D_j \]

iff dominate $(U_i^{PL}, D_j) \lor$ dominate $(D_j, U_i^{PL})$ and $i = j$

This write-down policy is strictly concerned with the disclosure of product data. For example, a Project team member cannot update (write) object to a lower security level such as the Intranet workspace.

Based on the Dynamic user leveling property, Super user and Project leader can act as other users. In other words, they can login different workspace and perform the Check-In or Approve operation to write data object down to lower level workspaces. For example, when an object is well prepared in the Project member workspace and is ready to be
approved to Extranet workspace, the Project leader will login the workspace and perform the Approve operation. After that, the object is written down to the Extranet workspace with an object state Released. Before being approved by the Project leader, operation performed on the data object may change its state and/or its security level. In a DPDM system, it is important to maintain the security level of all released products data object. Product data object should not flow to lower level from higher workspace level. To prevent the potential disclosure of data, the fourth property, Released tranquillity principle is proposed,

*Any operation performed on any Released data object by a Project member or other lower level user cannot change the object security level.*

**Axiom 5:** The released object state change but the security level remains unchanged under any read or write operation during the DPDM workflow life cycle.

Figure 8.6 summarizes all properties of the proposed mandatory access control method in a DPDM system.
Figure 8.6 Workspace-oriented DPDM system mandatory access control

The above axioms control the data flows between workspaces securely. For decision on whether access should be granted to Project team member and other lower level users, a set of axioms has to be defined over users, workspaces and product data objects.
The security labels $U$, $W$, and $D$ are defined for user, workspace, and product data object respectively. The policy can then be expressed as follows:

$U$ may see $D$ at $W$ only if

dominate $(U, W)$ and

dominate $(U, D)$.

Instead of verifying the dominate relationship for each label in turn, all object labels can be compounded into a single workspace label. The workspace label’s level is intended to be greater than or equal to any of the individual workspace DPDM user labels. This notion corresponds to the mathematical concept of a least upper bound (l.u.b) $lub(U)$ over a set $U$ partially ordered by the dominate relationship. The l.u.b. over a set of labels is defined as follows:

$$ub(U,x) \iff (\forall y \in U) (\text{dominate}(x, y))$$

$$(lub(U)=x) \iff ub(U,x) \land \lnot(\exists z)(ub(U,z) \land \text{dominate} \ x, z) \land \lnot (x=z))$$

The upper bound of set $U$, $ub(U,x)$ in this definition is a predicate that is true if $x$ is an upper bound of set $U$.

Using the l.u.b. of the participants’ security labels, the policy can be expressed as

$U$ may see $D$ at $W$ only if

dominate $(U, lub\{W, D\})$
More generally, the least upper bound of a set of attribute tuples can be computed as the tuple of the least upper bounds of the individual attribute values:

\[ \text{lub}(\{(a_1, \ldots, a_n)_A, (b_1, \ldots, b_n)_B\}) = (\text{lub}_1\{(a_1, b_1)\}, \ldots, \text{lub}_n\{(a_n, b_n)\})_{A\cup B} \]

Note that the l.u.b. for each set of attribute values operates over the partially ordered set specific to that attribute. Axioms over operations with multiple users can be defined analogously. The workspace label of all system users should be less than or equal to the individual labels. Therefore, the greatest lower bound (g.l.b) of the user labels can be defined for the workspace label.

Consider the following policy with \(D\) and \(U\) defined as sets:

\[ U \text{ may read } D \text{ if dominates } (\text{glb} (U), \text{lub} (D)) \]

This policy permits information flow or operation performing from a classification \(U\) to a classification \(D\) if the least classified element of \(U\) still dominates the highest classified member of \(D\). This indicates that the mandatory security model can be applied to multiple users or multiple data objects with various authority or sensitivity. Hence, mandatory method is applicable to the problem of specifying security for DPDM system.
8.5. Mixed Approach to DPDM Access Control

In the proposed workspace-oriented DPDM system, user and product data are classified according to workspace stratification. Different workspace users are granted with different access right based on the workspace security level. Since there are various security levels in the system, a single monolithic access control scheme cannot work well in all circumstances. Therefore, a mixed approach of access control is used to take advantage of both access control methods.

8.5.1. Well control access to the workspace-oriented DPDM system

From the two earlier mentioned access control methods, the label-based mandatory access control is found suitable for the DPDM system, especially for Project team member or lower level workspace users. In a DPDM system, authority is centralized to the Super user and the Project leader. It is a simple framework for system security. Obeying the specified properties and axioms, mandatory access control approach is suitable to provide a simple and highly efficient framework for product data protection.

However, the Super user and the Project leader will sometimes selectively allow or restrict access for some users to write up or to read necessary and sensitive product data in their workspaces. For example, the Project leader may selectively allow some Project team members to access and write up to his/her workspace via the UOSD access matrix. This protects the sensitive product data in his workspace as well as the lower level user's privacy. This policy can be expressed as:
Project team member, Tom
/Register} DWG1002, Project
leader workspace @/Local site

This policy specifies that Tom is a Project team member workspace user, and is
authorized to access and perform Register operation to write up data object to a higher
level workspace. This policy has also been shown in Figure 8.6. It safeguards sensitive
data both inside and outside the workspace borders. On the other hand, the Super user
and the Project leader should have the authority to perform operation across workspaces.
Especially the Project leader, he/she must perform the Approve operation to release and
write data down to different workspace. Other operations such as Delete and Obsolete
can also be performed under the control of UOSD access matrix relationship.

8.5.2. Product data protection

Under the mixed approach access method, data confidentiality is strictly protected by the
modified BLP access properties. They disallow any unauthorized access or write down
to product data. Data integrity is achieved by both the modified BLP access properties
and the UOSD access matrix relationship. They prevent unauthorized modification and
deletion of product data. Data availability is achieved by the UOSD matrix since it
allows selected authorized users to Check-In and Approve product data across
workspaces to ensure other DPDM system user to get the right data at the right time.
Operation tractability is maintained by the Log object as depicted in Figure 8.1. The log
object access is specially controlled by the UOSD matrix. It guarantees no common user
can access, modify or delete the log object. As a result, the mixed approach access method can achieve all the DPDM system security objectives.
CHAPTER 9  IMPLEMENTATION OF WEB-BASED DPDM SYSTEM

In this chapter, the web-based implementation issues of DPDM will be discussed. Object-oriented database management system (OODBMS), Object-oriented programming language, Java, and three-tier client/server approach are proposed for the product data model implementation. Stratified workspace architecture and network layout for the new DPDM system are also suggested. DPDM operations, product data allocation and security are demonstrated by using the web-based DPDM system program.

9.1. OODBMS and Web-based DPDM System Architecture

An example of an OODBMS based DPDM system architecture is shown in Figure 9.1. In this architecture, distributed database approach is employed so that each site has an OODBMS and all are treated equal. Data objects in the Hong Kong site can be distributed to the China site or across other location through the Internet. In production, the required data can be requested from the local electronic vault via the OODBMS. This reduces the need to transfer data through the network. This architecture is better than the simple client/server architecture where single point of failure and bottleneck appear in the server will cause a profound effect on the whole system.

In Figure 9.2, it shows the implementation of user authentication, stratified workspace, access and privilege control in DPDM system. When a project team member or a corporate user wants to perform any operation on product data in DPDM system, the user is required to login the
system. Having login successfully, the user is directed to his/her specific workspace. The user can perform the DPDM operations which had been granted to the workspace. If the user is going to perform an operation which had not been granted to the current workspace, authorization checking process will be triggered based on the security model. After finished all the operation, the user can logout to exist the DPDM system.

![Diagram of OODBMS in a DPDM system architecture](image)

**Figure 9.1** OODBMS in a DPDM system architecture
In our architecture, Internet is used to provide the communication highway for the DPDM system implementation. Moreover, object-oriented database (OODB) is employed. An OODB can retrieve complex structures much more quickly since the object structures are stored directly to reference other objects. The OODB does not need to reconstruct those relationships by searching and joining through a number of tables. Figure 9.32 shows a bill of materials (BOM) of a product. A traditional database represents a product by a series of entries in a table which
lists out its assemblies and components. However, each assembly or component is defined by another series of entries in other tables until the most elementary parts in the product were reached. Therefore, retrieving BOM from a traditional database is time consuming since the system must search the database each time by tracing link from one assembly to the next.

![Diagram of BOM in traditional database]

Figure 9.3 BOM in traditional database

Figure 9.4 illustrates a BOM in an object-oriented database. An OODB stores product data objects with the links to its components represented as direct references. This structure eliminates the search and join operations to find the matching values since it can go to the component objects directly.
Most of today’s major PDM systems use relational database management system (RDBMS) to store their meta-data [Pahny 1998]. Though it is suitable for business application, this is not enough for application in engineering area, especially in CAD/CAM environment. So the potential of OODBMS application in next generation PDM system is high. Object-oriented database management system (OODBMS) provides the following advantages:

- OODBMSes provide an edge in the support for complex objects. Conventional database simply lacks the ability to provide efficient applications in CAD/CAM environment
- OODBMSes permit the extensibility to abstract data types, thereby increasing the database functionality
- Versioning for specialized applications such as CAD/CAM
• Capabilities for asynchronous operation, independent and simultaneous access to the database

• Compatibility to distributed operation

• Object structures correspond closely to real life product assembly and organization structures

9.2. Java and Three-Tier Client/Server Technologies

9.2.1. Java-based networking application

To achieve DPDM, we have to tackle two main problems. The first is on how to provide communication between two local PDM systems. The second is on how to establish database connectivity within the distributed PDM systems. In the market, there are more than one programming tools available for solving the two problems. In this project, the Java programming language is used because of its various advantages. Java is a general purpose object-oriented programming language. It contains a collection of familiar constructs and features from programming language such as C, C++, object-C, SmallTalk, and Common Lisp. In addition, Java adds security, has a simple programming paradigm, and is client/server based. Any platform with a Java interpreter can run Java applet and applications without any need to recompile. Moreover, Java’s network-aware functionality, general robustness, secure, easy to use, easy to understand, cross-platform portability and automatically downloadable on a network make it the ideal environment for demanding, distributed client/server application. In the DPDM system, the distribution function is achieved by Java Remote Method Invocation (RMI) and Java Database Connectivity (JDBC) Application Programming Interface (API). RMI and JDBC are used to solve the first and second problems mentioned above respectively [Jackson 1997].
9.2.2. Java remote method invocation (RMI)

The Java Remote Method Invocation (RMI) mechanism provides the means to invoke methods remotely. In practice, this ability enables the development of distributed applications, since the invoked method object sits on a distributed computer. By using Java RMI, applications can communicate and execute across multiple PDM systems on a network. Since RMI is centered around Java, it can take a natural, direct, and fully powered approach to provide one with security and portability to distributed computing.

9.2.3. Java database connectivity (JDBC)

In the proposed DPDM model, data transfer has to be provided between databases of remote PDM systems. Distributed databases need to be interconnected and is done by JDBC. JDBC is a standard defined by Sun Microsystems to allow database access from Java applets and applications. JDBC consists of a set of classes and interfaces written in the Java programming language. It was developed to have a standard, Open Database Connectivity (ODBC) like interface. It allows Java applets and applications to access different database management systems (DBMSs) through the same interface, in much the same way as ODBC, but requires no end-user configuration of data sources. Thereby, an easier to deploy and maintain middleware solution is delivered.

9.2.4. Three-tier client/server approach

In the proposed DPDM model, the communication is bi-directional. Both the HK side and the China side can act either as a server or as a client. It is important to realize that the client and server labels last in one single method call only. When a computer runs the Java program to call a remote method it is the client for that call. When the computer hosting the object (remote method) that processes the call it becomes the server for that call. The roles of server and client can change. This concept is shown in Figure 9.5 (a) and (b).
Figure 9.5 (a) and (b) Client/Server Concept

Instead of a single client/server architecture, a three-tier client/Server architecture is employed to implement the proposed DPDM model. Figure 9.6 (a) and (b) illustrate the three-tier RMI and JDBC application architectures. In the three-tier model, a third server is employed to handle requests from the client and then pass them off to the distributed server inside the HK PDM system. The third server is installed in HK and China to act as a proxy for all PDM client requests. All client requests for database are routed through the proxy server, thereby creating a more secure environment for DPDM databases since the databases will not be exposed to the open world directly.

Figure 9.6 (a) HK-server and China-client and (b) China-server and HK-client three-tier architecture
9.2.5. Three-tier client/server model operation

In a Java program, it is natural to implement the request and the product data as objects. Using the RMI, they can be transported as objects between the client and the proxy server, JDBC provides the database connection function for processing the objects.

When one side (client) of the DPDM system requests product data from the other side (server), the client can invoke method call on object (e.g. search) that resides in the server. The request object is transported to the proxy server across the Internet. The proxy server then routes the request object to the system server. The request object calls a remote method search which has the request as its parameter. The search method returns an object to the proxy server and then to the client. The client can view the product data object with the PDM system interface.

By integrating various powerful features of the Java programming language, the entire DPDM system becomes more robust, secure and efficient. Java technology plays an important role in the distributed function of the DPDM system. On the other hand, the local PDM software can provide product data retrieval and modification control throughout the product life cycle. In summary, wherever the headquarter and branch offices are distributed in different geographic locations, the proposed Java-based DPDM system is likely to let them share, exchange, manage, and distribute the product information and to support concurrent engineering.

9.3. Implementation of DPDM Product Data Model

In DPDM implementation, suitable tool is an important aspect. The cores of DPDM system are PDM software system, database management system, programming language and industrial
standard. Most PDM software products have been reviewed and evaluated by CIMdata [Miller 1997] on their functions, user environment, operation environment, training and support, pricing, etc.

In implementing DPDM system across heterogeneous systems, it will involve the communication with a variety of programs, platforms and operation systems. Therefore, the Common Object Request Broker Architecture (CORBA) industrial standard is employed to add distributed communication capability to DPDM system. CORBA provides a distributed object infrastructure for the model and allows applications to extend their reach across networks and operation systems. It interoperates among the DPDM system and lets applications to communicate with one another within the DPDM system no matter where they are located. On the other hand, an object oriented programming language Java will also be used. Java is a versatile language and possesses inherent network programming features which makes it an ideal networking and distributed programming language. It provides the product data model a portable object infrastructure that works on every major operation system. In the proposed network-centric model, Java is used as the first step toward creating an object-based DPDM system. Employing the intersection of CORBA's network transparency and Java's implementation transparency features, it leads to an efficient and robust distributed product data management system.
9.4. Stratified Workspace in DPDM System

9.4.1. Personal workspace

Personal Workspace is implemented in a local area network (LAN) inside a company as shown in Figure 9.7. Product data is created in the client terminals.

![Diagram showing Personal Workspace boundary]

After creating a data object in the Personal workspace (Figure 9.8), the object owner will register it to the DPDM database which is located in the Project team workspace. To construct a data structure, classification is necessary during the registration operation. The object will be registered to their own classes (Figure 9.9). The data object is the instance and inherits the attributes from the class as shown in Figure 9.10. For example, the object identity, version, state, etc. After the registration operation, values of object ID, version number, object state and other attributes are assigned to the new object.
Figure 9.8 Data objects are created in Personal workspace

Figure 9.9 Data structure

Figure 9.10 Objects inherit attributes from their parent classes
9.4.2. Project team workspace

Project Team Workspace is also implemented in a LAN connected to a DPDM backbone. Figure 9.11 shows the architecture of the LAN boundary of the Project Team Workspace.

![Diagram of Project Team Workspace boundary](image)

Figure 9.11 Project Team Workspace boundary

The Project team workspace is a platform which provides primary function for supporting cooperative work for DPDM system, independent of the computing platform. This shared workspace is accessible to members of the project team for analyzing and manipulating data object and also managing its version.

Receiving new objects from Personal workspace (Figure 9.12), a data structure is now formed and the abstract concept module of relationship between object is created in the project team workspace (Figure 9.13). It can contain different data types such as documents, engineering drawings, solid parts, images, URL links to other web or FTP server of remote sites, etc. Objects will inherit all attributes from their parent classes as shown in Figure 9.14.
Figure 9.12  Created object will be registered to the DPDM database in Project team workspace

Figure 9.13  Classification and data structure in a DPDM database

Figure 9.14  Object inheritance in a data structure

After verification of the new object, project team leader invokes the Release DPDM operation to release the object. For updating a released object, Personal or Project team workspace users can make a request by performing Update request operation to lock the object. If the request is successful, they can Check-Out the required object to their working area by replication. A new
object version will be assigned to the replicated object. Having updated the object, user Check-In the object back to the DPDM database. Project team leader will verify it and perform Release operation to release the object and its lock. Then a new release version will be assigned to the released object. For every DPDM operation, versioning function will be triggered to keep track of the object version history (Figure 9.15).

![Diagram of version management and workspaces](image)

**Figure 9.15 Workspaces support version management**

### 9.4.3. Corporate workspace

Corporate Workspace is implemented in the enterprise Intranet. All other departmental systems are linked to enterprise Intranet backbone. Figure 9.16 shows the Intranet architecture.
When an object is approved to a Released state, it will be logically migrated to the corporate workspace as shown in Figure 9.17. Indeed, the corporate users are granted the access right to access released objects which are stored in the DPDM system database. Corporate user can only perform the retrieval operation to view the object. They can retrieve and view the object by using a DPDM client application or a browser.
9.4.4. Public workspace

Public Workspace is the part of the network system of DPDM system that can link to the outside world via the Internet. Therefore, a web server is required to provide web service or request. Firewall is used to prevent unauthorized user or hacker to access and damage the enterprise Intranet. Figure 9.18 shows the configuration of Public Workspace.

![Diagram](image-url)

**Figure 9.18 Public Workspace boundary**

After an object is decided to be published to the outsiders such as the vendors, sub-contractors and customers. The object will be mapped to the Public workspace. Public workspace users can view the object content simply through the Extranet by using a browser similar to the one shown in Figure 9.19. Only retrieval operation is allowed in this workspace. Figure 9.20 shows the interaction between the corporate and outside company.
9.4.5. Scenario of MultiNet architecture

Figure 9.21 illustrates the Intranet, Extranet and Internet network architecture of an enterprise. PDM and other functional systems are constructed on the enterprise Intranet backbone. As a result, other corporate users can collaborate within the PDM system via the network. Computer linkages of each shared workspace are integrated into the PDM system. Connecting to external
selected vendors and strategic customers can be achieved through Extranet by different protocols, for example the DCOM, IIOP, and other available protocols.

![Diagram of Enterprise Multi-Net network architecture](image)

**Figure 9.21** Enterprise Multi-Net network architecture

**9.4.6. Event message flow between DPDM workspaces**

In a DPDM system, a lot of operations will be performed by different users on the objects. The operations are event-based and triggered by the event message flow as shown in Figure 9.22.
The interaction of these event messages is explained below.

1. Project team member in the Personal workspace is asked to create a data object.
2. Personal workspace user stores the created object in his/her local database.
3. Project team leader invokes the Release operation to release the object.
4. The object is migrated to the PDM database.
5. Object update control lock is created and available for request.

6, 7, 7* Object is mapped to Corporate and Public workspace to share with other users. Note the appointed shared object is for Extranet user.

8. Retrieval is requested (e.g. by a Corporate workspace user).
(9) Replicate the object and invokes Check-Out operation to retrieve the object.

(10) Receive the replicated object content and read the object content.

(11) Update is requested (e.g. by a Project Team workspace user).

(12) Obtain the update lock from the primary site lock-manager. If this operation fails, no update operation is allowed.

(13) User invokes Check-Out operation, the system will automatically replicate and migrate the replicated object content to user’s own OS database.

(14) Replicated object content is migrated to local OS database.

(15) Object is being updated.

(16) Updated object is migrated to PDM database by invoking Check-In operation for further modification or release.

(17) Checked-In object is migrated to local OS database again for further update by invoking the Check-Out operation.

(18) Object is migrated to PDM database again and waits for Release approval or further update.

(19) Object is released again.

(20) Re-released object is stored in PDM database with a new release version number.

(21) Update lock is released for other user request.

(22,23, 23*) New released object is mapped to the Corporate and Public workspace. Note the appointed shared object is for Extranet user.
9.5. Web-Based DPDM System Software Development

In DPDM System development, there are three critical system components: System Browser, System Functional Modules, and System Database.

9.5.1. DPDM system browser

The browser supports Internet communication protocol (e.g. http). It provides the interface for system users to browse the product data and perform their DPDM operations through the Internet. Figure 9.23 shows the DPDM System Browser.

Figure 9.23  DPDM System browser
9.5.2. **DPDM system functional modules**

In this section, three DPDM system functional modules will be discussed for the implementation of the stratified workspace, security model, and distribution theories.

9.5.2.1. **DPDM system security module**

Figure 9.24 shows the security module, which checks whether the user is a valid user or not. It also checks what the user type is, for example, the Project Team, Intranet User, etc. Different user type will direct the user to his/her authorized workspace(s).

![Figure 9.24 DPDM System user authentication](image-url)
After using the DPDM system, user should logout the system as shown Figure 9.25. It will clear the user information from the login pool.

Figure 9.25  DPDM system logout operation

Authorized DPDM system user, project leader and super user can use the View System Log function to trace who and what operation had been performed on the product data. Figure 2.26 and Figure 2.27 show the View System Log operation.
Figure 9.26  View System Log function of DPDM system

Figure 9.27  Detail information of each product data object
Authorized user can also check who had login the DPDM system by using the View Login Pool function. Figure 9.28 shows the function.

![Figure 9.28 View Login Pool function of DPDM system](image)

9.5.2.2. **DPDM stratified workspace module**

Figure 9.29 shows the workspace system module main menu. The Workspace System control will direct the user to the DPDM workspace environment as shown in Figure 9.30.
Figure 9.29 DPDM Workspace Modules menu
Authorized user access his/her workspace and can view the classified product data. They can perform the DPDM operation in the workspace.

![Project Team Workspace](image)

Figure 9.30 Project Team Workspace
Workspace user can retrieve the detail information of product data. Figure 9.31 shows the modification history and information of product data.

Figure 9.31 Object Detail Information from Project Team Workspace

When DPDM Registration operation is performed, meta data of product data is required to update DPDM System database. The data entry form is shown in Figure 9.32.

Figure 9.32 Registration Operation in Project Team Workspace
Figure 9.33 shows a DPDM system product data viewer which allows system user to view the classified product data.

Figure 9.33  DPDM system product data viewer
9.5.2.3. **DPDM project management module**

Figure 9.34 shows the DPDM system project module. The module is used for project management and project team maintenance. Authorized user can use the module function to create new project, form project team, allocate users to different workspaces to grant them system access privilege.

![Figure 9.34 DPDM project management module](image-url)
Figure 9.35 shows data entry form for creating a new project in DPDM system.

Figure 9.35  Create new project in DPDM system
9.5.2.4. **DPDM product data distribution module**

Figure 9.36 shows the DPDM system Distribution module. The module allows the authorized user to select and distribute product data to remote site. It also allows the authorized user to gain the information of the distributed site and product data.

![Distribution module of DPDM system](image)

Figure 9.36  Distribution module of DPDM system
Figure 9.37 shows the DPDM system distribution function, which allows authorized user to select product data to be distributed to remote site.

Figure 9.37 Select product data and distribute to remote site in DPDM system
Figure 9.38 shows the remote sites information within a DPDM system

![DPDM System Distributed Site Information](image)

**Figure 9.38** Remote site information
Figure 9.39 shows how user can find the information of distributed product data.

![Information of Distributed Product Data object](image)

<table>
<thead>
<tr>
<th>Object ID</th>
<th>Create date</th>
<th>Created By</th>
<th>Object version</th>
<th>Object State</th>
</tr>
</thead>
<tbody>
<tr>
<td>MTR001</td>
<td>11/1/99</td>
<td>Michael</td>
<td>1.7</td>
<td>Checked-In</td>
</tr>
<tr>
<td>MTR003</td>
<td>10/10/00</td>
<td>Peter</td>
<td>1</td>
<td>Checked-Out</td>
</tr>
<tr>
<td>MTR003</td>
<td>12/12/00</td>
<td>Michael</td>
<td>1</td>
<td>Released</td>
</tr>
<tr>
<td>MTR014</td>
<td>5/5/00</td>
<td>Peter</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>MTR015</td>
<td>4/4/00</td>
<td>Peter</td>
<td>1</td>
<td>Checked-out</td>
</tr>
<tr>
<td>3mm_cap.prt</td>
<td>12/12/00</td>
<td>Tom</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>back_case.prt</td>
<td>1/1/01</td>
<td>Joe</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>BlackWheel.prt</td>
<td>2/2/01</td>
<td>Michael</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Board_dry</td>
<td>1/1/01</td>
<td>Tom</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Board.prt</td>
<td>11/11/00</td>
<td>John</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Chip.prt</td>
<td>12/12/00</td>
<td>Michael</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Circuit_asm</td>
<td>9/9/00</td>
<td>Joe</td>
<td>1</td>
<td>NULL</td>
</tr>
</tbody>
</table>

Figure 9.39  Distributed product data in a remote site
Figure 9.40 shows the function to find remote site information by project code or product data identity number.

Figure 9.40  Distributed product data in a remote site information
9.5.3. **DPDM system database schema**

Database is another core DPDM system component. Figure 9.41 shows the DPDM system database schema. In the demonstration system, relational database schema is implemented in the RDBMS, Microsoft SQL Server.

![Diagram of DPDM system database schema](image)

Figure 9.41  DPDM system database schema
CHAPTER 10 DISCUSSION

10.1. Statechart Modeling Method for DPDM Operations

Statechart modeling technique was employed in modeling the operations of DPDM system. Statecharts give an explicit and a formal definition of the system behavior. A big disadvantage of them is that one has to define all possible states of the system. Whilst this is all right for small systems, it may break down in larger systems as there is an exponential growth in the number of states. This state explosion problem leads to state transition diagrams becoming far too complex for practical use. Although other object modeling methods may be investigated. For example, separate class is defined from a system to achieve a simple comprehensible state transition diagram. However, it is difficult to visualize the behavior of the whole system from a large number of diagrams of individual classes.

10.2. Product Data Model and Object-oriented Technology

In implementing the object-oriented product data model, an object-oriented database management system is required. However, OODBMS is a relatively new technology and its development is not yet fully mature. The use of OODBs requires an educational change among the database maintainers and requires a corporate commitment to formal training in the proper use of the OODB features. Most programmers understand RDBMS but they do not understand OODBMS. In addition, most of the OODBMS vendors are small companies. Although OODB
based DPDM system could benefit from higher performance on complex product data, one may wonder whether this risk is worthwhile or not.

10.3. Product Data Allocation in DPDM System

It is not an easy job to decide which product data should be distributed to remote sites. Therefore, data analysis must be performed and set the distributed objectives. In the proposed product data distribution model, some factors have not been considered. For example, the probability of network collapsing may affect the accuracy of the model. Data acquisition, storage and distribution of product data in a DPDM system are not cheap. As a distributed system contains many different kinds of hardware and software over many locations working together in cooperative manner, the networking technology to be used should comply with the same protocol.

10.4. MultiNet Workspace user management in DPDM system

The effective communication in a DPDM system is the key to achieve successful outcome. MultiNet workspace is developed to integrate communication environment for both internal and external communications. Apart from managing the product data and user, the workspace can also help to manage other corporate resources, for example, application software, web resource, supplier information, etc. A further consideration is to integrate the MultiNet workspace with the Internet conferencing and video-conferencing.
10.5. DPDM System security

Explosive growth in user of information systems for all kinds of applications in all parts of life has made the provision of proper security essential. Security of DPDM system is an important matter because the DPDM system itself often crosses geographical boundaries and it gives rise to the most effective data sharing. It manages all product related data in all remote sites and guarantees smooth operation of distributed production system. Therefore, apart from access right and authority granting, the DPDM system must also support protection from extreme environmental events. Extreme environmental events include earthquake, fire, flood, and excessive heat and humidity.
CHAPTER 11  FUTURE RESEARCH

In this thesis, theories of a DPDM system had been developed. However, further investigation is needed to fully develop and implement the proposed theories in the following areas:

(1) Object-oriented database was proposed as one of the core component in DPDM system. Further research on the integration of object-oriented database with traditional database-based CAD/CAM system and CIM system is essential for smooth transition.

(2) Product data allocation of DPDM system relies on the Internet as its underlying distribution infrastructure. The Internet is an open network. To further enhance the product data allocation model, research can be done by considering more factors, for example probability of network collapsing and data loss, human mistake, etc.

(3) Research on stratified workspace to support multimedia can be performed as video conferencing can further encourage the sharing of idea and the awareness of collaboration.

(4) In a DPDM system, product data will be distributed to more than one remote site due to some specific requirement. Further research can be done on developing the ability to automatically configure the remote site system to obtain product data from the best connection it can find.
(5) DPDM system product data is transmitted via the Internet. Research on data encryption should be investigated and policies can then be set to decide whether the product data need to be encrypted or not.

(6) For DPDM system security, further research can be done on technology to prevent intentional attacks on the system and virus which will distort or destroy product data and normal system function.
CHAPTER 12 CONCLUSION

In the light of the need for a DPDM system support for today’s distributed manufacturing environment, theories and framework for DPDM system were presented and had the achievement as follows:

*DPDM system theories achievement:*

A new web-based DPDM system architecture was developed. This architecture can be integrated with web and networking technologies. It includes:

- Investigated the model for both PDM and DPDM systems, a new formal model of DPDM system was developed with statechart.
- Introduced a new object-oriented product data model for the development of DPDM system in object-oriented manner.
- Developed a product data allocation model for DPDM system.
- Proposed a new stratified workspace architecture for DPDM system.
- Proposed a new security model for DPDM system.

*DPDM system implementation achievement:*

- Based on the statechart model, a DPDM system for demonstration was developed to implement the DPDM operations.
- DPDM system was implemented on the web. Networking technologies such as Java was used in developing the distribution application of the system.
- Stratified workspace system was implemented in the DPDM system.
Mixed approach security model was implemented in the DPDM system

**DPDM system Implementation Constrains:**

- Object-oriented database system had not been implemented in the DPDM system as the technology is neither mature nor common to user. On the other hand, not many vendors provide OODBMS product. Therefore, traditional relational database system is used to implement the DPDM system.

- Integration of DPDM system with CAx system has not been implemented, since the integration requires the CAx developer to provide the Dynamic Link Library (DLL) and Application Programming Interface (API).

- Resistance to changing the methods, processes, and systems that a company uses to conduct business is a natural occurrence. Management must be actively committed to the success of the project because some employee in the organization may not support the new system implementation. Employees may resist change for a number of reasons, such as, they had familiarized the existing applications, fear of the unknown, they fear that something may go wrong during the transition and they may lose their job. Employees should be persuaded to embrace system changes by understanding the benefits of the new system.
REFERENCES

[Ames 1997]
The Integration Challenge p. 214-20.

[Apers, 1988]
Apers, P. M. G.
Data allocation in distributed database systems.

[Atkinson 1990]
Atkinson, D.; Glasscock, A.
An implementation of a product data management system. Proceeding of the 1990 ASME International computer in engineering conference and exposition.

[Badenhorst 1990]
Badenhorst, K.P., Eloff, J.H.P.
Computer Security Methodology: Risk Analysis and Project Definition.
[Baker 1997]

Baker, R. H.
Extranet, the complete sourcebook.

[Bentley 1997]


Basic support for cooperative work on the World Wide Web.

[Booch 1994]

Booch, G.

[Bowman 1995]

Bowman, I.

Design and production in harmony.

[Bowman 1996]

Bowman, I.

Can PDM win the paper battle in your company.

Ref.2
[Cai 1998]

Cai, W.; Lee, G. S.

*File allocation with balanced response time in a distributed multi-server information system.*

Information and Software Technology, 40, p. 27-35.

[Ceri 1982]

Ceri, S.; Martella, G.; Pelagatti, G.

*Optimal file allocation in a computer network: a solution method based on the Knapsack Problem.*

Computer Networks, 6, p. 345-357.

[Chen 1997]

Chen, Y. M.

*Development of a computer-aided concurrent net shape product and process development environment.*


[Chen 1998]

Chen, Y. M.; Tsao, T. H.

*A structured methodology for implementing engineering data management.*

[Cheng 1999]

Cheng, F. T.; Shen, E.; Deng, J. Y.; Nguyen, K.

*Development of a system framework for the computer-integrated manufacturing execution system: a distributed object-oriented approach.*


[Cheng 1993]

Cheng, W. K.

*Distributed object database management systems.*


[Chlebus 1998]

Chlebus, E.; Cholewa, M.; Dudzik, R.; Kozera, M.

*CAX application for process oriented concurrent design.*


[Chris 1995]

Chris, F.; Rodema, A. M.


Ref. 4
[Chu 1976]
Chu, W. W.

Performance of file directory systems for data bases in star and distributed networks.
AFIPS Conference Proceedings, p. 577-587.

[Chu 1999]
Chu, X.; Fan, Y.

Product data management based on Web technology.

[CIMdata 1997]
CIMdata

Product Data Management: The Definition and Introduction to concepts, Benefits, and Terminology, Fourth Edition.
CIMdata, Inc.

[Cmstat 2000]
Cmstat.

New PDM Apps are more capable at managing Complex data relationships.
http://www.pdmic.com/

Ref.5
[Cornelissen 1995]
Cornelissen, M., Kok, A.; Mandemaker, D.
PDM Selection Guide.

[Covill 1998]
Covill, R. J.
Implementing extranets, the Internet as a virtual private network.

[Deitz 1995]
Deitz, D.
Designing for quality control.

[Deitz 1997]
Deitz, D.
Re-engineering virtual prototypes.

[Dorfman 1994]
Dorfman, J.; MacKrell, J.; Miller, Ed.; Romzick, P.
PDM Implementation Survey Report.
CIMdata, Inc.

[Dowdy 1982]
Dowdy, L. W.; Foster, D. V.
Comparative models of the file assignment problem.

[Dwyer 1999]
Dwyer, J.
Whither workflow? A PDM or ERP paradigm.

[Dwyer 1999]
Dwyer, J.
Getting to market faster the PDM way.

[Elmasri 1997]
Elmasri, R.; Navathe, S. B.
[Goodwin 1995]
Goodwin, B.

*Doing away with the factory paperchase.*

[Gregory 1996]
Gregory, A.

*A prescription for manufacturing's ills.*

[Han 1998]
Han, J. H.; Requicha, A. A. G.

*Modeler-independent feature recognition in a distributed environment.*

[Hannu 1996]
Hannu, P.; Olli, P.; Reijo, S.

*Process-based view of product data management.*

[Harel 1998]
Harel, D.; Politi, M.

Modeling Reactive System with Statecharts.

[Harris 1996]

Harris, S.B.


[iMAN 2000]

http://www.ugsolutions.com/products/iman/

[Izuchukwu 1996]

Izuchukwu, J. I.


[Jasnoc 1996]

Jasnoc, U.; Haas, S.

[Johnson 1986]

Johnson, R. H.

Product data management-with solid modeling.


[Johnson 1988]

Johnson, R. H.

Commercial product data management systems: a survey on needs and systems. In managing engineering data: emerging issues.


[Kato 1999]

Kato, H.; Kanai, N.; Mizoguchi, N.; Ueda, M.

Scattered database access-concept and implementation.


[Kempfer 1995]

Kempfer, L.

Roadmap for success.

[Kim 1998]
Kim, J. A.; Kim, J. H.; Park, N.
*Development of PDM framework and customization environment.*

[Kim1997]
Kim, S. H.; Oh, T. H.; Park, J. Y.
*The object-oriented modeling for product data management (PDM).*
Computer Applications in Production and Engineering. IFIP TC5 International Conference on Computer Applications in Production and Engineering (CAPE'97) p. 33-46.

[Kovacs 1998]
Kovacs, Z.; Le Goff, J. M.; McClatchey, R.
*Support for product data from design to production.*

[Kung 1999]
Kung, H.K.; Du, T.C.T.; Weng, C. H.
*Applying object-oriented database technologies in concurrent design processes.*
[Lee 1996]
Lee, H.; Jang, G.

*File and workload allocation on a local multi-access computer network: incorporating local processing and communication overhead.*


[Lee 1994]
Lee, H.; Shi, Y.

*Allocation data files over a wide area network: Goal setting and compromise design.*

Information and Management, 26, p. 85-93.

[Lee 1993]
Lee, H.

*An evaluation method for availability of a distributed database management system.*

Information and Management, 24, p. 203-208.

[Leonhardt 1998]
Leonhardt, U., Magee, J.

*Security Considerations for Distributed Location Service.*


[Linag 1998]
Linag, J.; Chang, T. Y.; Chan, C. M.
An object-oriented database management system for computer-aided design of tall building.


[Lindeman 1994]
Lindeman, D. D.; Moore, B.

PDM: an enabling technology for integrated product development.

[Low 1996]
Low, G. C.; Sellers, B. H.

Incorporation of distributed computing concerns into object-oriented methodologies.

[MacKrell 1997]
MacKrell, J.

CIMdata, Inc.

[Madria 1998]
Madria, S. K.
Handling of mutual conflicts in distributed databases using timestamps.


[Mainwaring 1999]
Mainwaring, J.
PDM tools: a necessity?

[Martin 1996]
Martin, L.
Succeeding with the Booch and OMT methods: A practical approach.
Addison-Wesley, 1996.

[Martino 1998]
Martino, T. D.
Design and engineering process integration through a multiple view intermediate modeller in a distributed object-oriented system environment.

[Matsushita 1999]
Matsushita, Y.
[McIntosh 1995]
McIntosh, K. G.
Engineering Data Management.
McGraw-Hill.

[McIntosh 1995]
McIntosh, K. G.
Engineering Data Management.
McGraw-Hill.

[Mendel 1999]
Mendel, A.

*PDM and the Internet.*

[Miller 1996]
Miller, E.

*Linking PDM and MRP.*

[Miller 1999]
Miller, E.

*PDM for real collaborative working.*

[Miller 2000]
Miller, E.

PDM e-business takes off.
http://www.cimdata.com/

[Miller 1997]
Miller, E.; Macckrell, J.; Mendel, A.

CIMdata, Inc.

[Nebro 1999]
Nebro, J. N.; Pimentel, E.; Troya, J.

Distributed object: An approach based on replication and migration.

[Obank 1995]
Obank, A.; Leaney, P.; Roberts, S.

Data management within a manufacturing organization.
[Ozsu 1991]
Ozsu, M. T.; Alduriesz, P.
Principles of Distributed Database Systems.
Prentice-Hall, Inc.

[Pahny 1998]
Pahny, F; Senin, N.; Wallace, D.
Distribution modeling and evaluation of product design problems.

[Parker 1997]
Parker, K.
Interface of things to come.

[PDM II 2000]
PDM II
Executive Summary Product Development Management II (PDM II).
http://www.clearlake.ibm.com

[Peck 1997]
Peck, B.
Integration & management issues in a web-based, PDM, ERP, & parts management system in PDM Conference’97, Apr 23-25, Los Angeles, California.

CIMdata.

[Pikosz 1996]
Pikosz, P.; Malmqvist, J.
Possibilities and Limitations when using PDM system to support the product development.
http://www.mvd.chalmers.se/~pikosz/research/NordDesign96/pm96.fm.html.

[Port 1997]
Port, S.; MacKrell, J.
CIMdata, Inc.

[Puttre 1994]
Puttre, M.
Taking control of the desktop.

[Ramanathan 1996]
Ramanathan, J.
Process improvement and data management.

[Rumbaugh 1991]
Rumbaugh, J; Blaha, M; Premerlani, W; Eddy, F.; Lorensen, W.
Object-oriented modeling and design.

[Sackett 1998]
Sackett, P. J.; Bryan, M. G.
Framework for the development of a product data management strategy.

[Sadlbauer 1992]
Sadlbauer, H.
A product data model for design support using functional skeletons, Engineering data management: key to integrated product development: proceedings of the 1992 ASME.

[Saha 1994]
Saha, D.; Mukherjee, A.
An optimal file allocation Policy in a networked database management system.

[Salvatore 1995]
Salvatore, T. M.; Rho, S.
Allocating data and operations to nodes in distributed database design.
ACM Transaction on Database Systems, 7(2), 305-317.

[Shapiro 1996]
Shapiro, D.; Tauber, M. J.; Traunmüller, R.
The Design of Computer Supported Cooperative Work and Groupware Systems.

[Shaw 1989]
Shaw, N. K; Suan Bloor, M.; Pennington, A.
Product data models.

[Siegel 1995]
Siegel, J.
Hands-on product data management.

[Smith 1994]
Smith, P.; Kearny, L.
Creating Workspaces Where People Can Think.

[Stamps 1990]
Stamps, D.
Survival software for manufacturing.

[Susaki 1998]
Susaki, S.; Muramoto, T.
Information sharing system on the WWW with interactive communication.

[Tang, 1993]
Tang, J. C.
Toward an understanding of the use of shared workspaces by design teams.
UMI, 1993.

[Tinham 1998]
Tinham, B.
Achieving more in a lot less time.

Ref. 21
[Turbide 1995]
Turbide, D. A.

*MRP II still number one.*


[Unigraphics 2000]

http://www.ugsolutions.com/products/unigraphics/

[Wiebe 1997]
Wiebe, E. N.

*Adding agility to CAD: Integrating product data management tools into an organization.*


[Williams 1991]
Williams, T.

*Software tools integrate the management of complex design projects.*


[Windchill 2000]
Windchill

*Windchill Executive white Paper: leveraging the internet for collaborative product commerce.*

Parametric Technology Corporation.

Ref.22
[Zhang 1995]

Zhang, X.; Wang, T.; Wan, L.; Zhou, J.

*Open architecture and implementation for product data management.*

import java.awt.*;
import java.awt.event.*;

public class DPDMSetup extends Dialog{
    public String loginName, loginPassword,
    public String remoteHost, remotePath, localPath;
    Label lbName, lbPass, lbRemotePath, lbLocalPath, lbRemoteHost;
    TextField tbName, tbPass, tbRemotePath, tbLocalPath, tbRemoteHost;
    Button cmdOK, cmdCancel;
    static Dialog setupDialog;
    static Frame parentFrame;
    final int cmdHeight=30, cmdWidth=80;
    final int tbHeight=30, lbHeight=30;
    public DPDMSetup(Frame f) {
        super(f, "System Setup");
        parentFrame = f;
    }
    public DPDMSetup(Frame f, String title, boolean m) {
        super(f, title, m);
        parentFrame = f;
    }
    public DPDMSetup(Frame f, boolean m) {
        super(f, "System Setup", m);
        parentFrame = f;
    }
    public void setDefault(String loginname, String loginpassword, String remotehost,
    String localpath, String remotepath) {
        loginName = new String(loginName);
        loginPassword = new String(loginPassword);
        localPath = new String(localPath);
        remotePath = new String(remotePath);
        remoteHost = new String(remoteHost);
        tbName.setText(loginName);
        tbPass.setText(loginPassword);
        tbRemoteHost.setText(remoteHost);
        tbRemotePath.setText(remotePath);
        tbLocalPath.setText(localPath);
    }
    public void init(){
        try {
            frameWidth=510, frameHeight=5;
            setBackground(Color.lightGray);
            catch (Exception E) { }
            Panel p = new Panel();
            p.setLayout(null);
            lbName = new Label("User Name");
            tbName = new TextField("", 20);
            lbPass = new Label("Password");
            tbPass = new TextField("", 20);
            tbPass.setEchoChar("");
    }
lbRemoteHost = new Label("Remote Site");
tbRemoteHost = new TextField("", 200);

lbRemotePath = new Label("Remote Allocation Path");
tbRemotePath = new TextField("", 200);

lbLocalPath = new Label("Local Allocation Path");
tbLocalPath = new TextField("", 200);

cmdOK = new Button("OK");
cmdCancel = new Button("Cancel");

p.add(lbName);
p.add(tbName);
p.add(lbPass);
p.add(tbPass);
p.add(lbRemoteHost);
p.add(tbRemoteHost);
p.add(lbRemotePath);
p.add(tbRemotePath);
p.add(lbLocalPath);
p.add(tbLocalPath);
p.add(cmdOK);
p.add(cmdCancel);

lbName reshape(3, frameHeight, 00, lbHeight);
tbName reshape(150, frameHeight, 200, tbHeight);
frameHeight += tbHeight;

lbPass reshape(3, frameHeight, 00, lbHeight);
tbPass reshape(150, frameHeight, 200, tbHeight);
frameHeight += tbHeight;

frameHeight += 20;
lbRemoteHost reshape(3, frameHeight, 130, lbHeight);
tbRemoteHost reshape(150, frameHeight, 250, tbHeight);
frameHeight += tbHeight;

lbRemotePath reshape(3, frameHeight, 130, lbHeight);
tbRemotePath reshape(150, frameHeight, 250, tbHeight);
frameHeight += tbHeight;

lbLocalPath reshape(3, frameHeight, 130, lbHeight);
tbLocalPath reshape(150, frameHeight, 250, tbHeight);
frameHeight += tbHeight;

frameHeight += 20;
cmdOK reshape(frameWidth/2 - 5 - cmdWidth, frameHeight, cmdWidth, cmdHeight);
cmdCancel reshape(frameWidth/2 + 5, frameHeight, cmdWidth, cmdHeight);
frameHeight += cmdHeight;

frameHeight += 10;
p.resize(frameWidth, frameHeight);
add(p);

frameHeight += 30;
resize(frameWidth, frameHeight);

Dimension d = parentFrame.size();
move((d.width - frameWidth)/2 + parentFrame.getLocation().x, (d.height - frameHeight)/2 + parentFrame.getLocation().y);
setResizable(false);
}

public boolean action(Event e, Object o) {
try{
Button b = (Button)e.target;
String lbl = B.getLabel();

if(lbl == "OK")
{
    loginName = tbName.getText();
    loginPassword = tbPass.getText();
    localPath = tblLocalPath.getText();
    remotePath = tbRemotePath.getText();
    remoteHost = tbRemoteHost.getText();
    hide();
} else if(lbl == "Cancel")
{
    loginName = "";
    loginPassword = "";
    localPath = "";
    remotePath = "";
    remoteHost = "";
    hide();
}

} catch(Exception E) {}

return false;
}
import java.io.*;
import sun.net.*;
import sun.net.*;

public class DPDMMFTP {
    FtpClient ftp;

    public DPDMMFTP(String hostName, String loginName, String loginPassword) throws IOException {
        ftp = new FtpClient(hostName);
        ftp.login(loginName, loginPassword);
        ftp.binary();
    }

    public void GetFile(String filename) {
        try {
            telnet_read_to_file(filename);
        } catch (IOException e) {
            System.out.println("Can't get file : " + e);
        }
    }

    public void PutFile(String LocalFile, String RemoteFile) throws IOException {
        int c;
        long i, fsize;
        RandomAccessFile f;
        TelnetOutputStream t;

        t = ftp.put(RemoteFile);
        try {
            f = new RandomAccessFile(LocalFile, "r");
            fsize = f.length();

            for (i=0; i<fsize; i++) {
                c = f.readUnsingedByte();
                t.write((byte) c);
            }

            f.close();
            t.close();
        } catch (IOException e) {
            System.out.println("Can't open local file : " + e);
        }
    }

    public void CloseConnection() {
        try {
            ftp.closeServer();
        } catch (IOException e) {
            System.out.println("Can't close server : " + e);
        }
    }

    String read_line() throws IOException {
        DataInputStream s = new DataInputStream(System.in);
        return s.readLine();
    }

    void telnet_read() throws IOException {
        int c;

        TelnetInputStream t = ftp.list();
        System.out.println("start listing");
    }
while((c = t.read()) >= 0) {
    System.out.print("" + (char) c);
}
t.close();

// Get file from remote host
void telnet_read_to_file(String filename) throws IOException {
    int c;
    int count = 0;

    TelnetInputStream t = ftp.get(filename);
    RandomAccessFile f = new RandomAccessFile(filename, "rw");

    while((c = t.read()) >= 0){
        System.out.print("" + (char) c);
        f.writeByte((byte) c);
    }
    f.close();
t.close();
}

// Put file to remote host
void telnet_write_to_file(String filename) throws IOException {
    int c;
    int count = 0;
    long i, fsize;

    TelnetOutputStream t = ftp.put(filename);
    RandomAccessFile f = new RandomAccessFile(filename, "r");

    fsize = f.length();

    for(i=0;i<fsize;i++) {
        c = f.readUnsignedByte();
        //
        System.out.print("" + (char) c);
        t.write((byte) c);
    }
    f.close();
t.close();
}

void get_file() throws IOException {
while(true) {
    System.out.println("\nEnter filename ");
    System.out.println("Press ESC+Enter to exit");
    String string = read_line();
    if(string.charAt(0) == (char) 27)
        return;
    try {
        if(string.startsWith("cd ")) {
            System.out.println("Change directory to <" + string.substring(3) + ">");
            ftp.cd(string.substring(3));
            telnet_read();
        } else {
            System.out.println("Getting file <" + string + ">");
            telnet_read_to_file(string);
            break;
        }
    } catch(FileNotFoundException e) {
        System.out.println("File not found");
    }
}
import java.awt.*;
import java.awt.event.*;
import java.io.*;
import DPDMSSetup;
import DPDMTFTP;
import DialogBox;

public class Distribution extends Frame implements WindowListener, ActionListener, Runnable {
    static final String default_loginname = "97981546r";
    static final String default_remotepath = "public_html";
    static final String default_localpath = "C:\\DPDM\distribute";
    static final String default_remotehost = "www.acad.polyu.edu.hk";
    static final int detect_timeout = 5000;
    static final String ProgramTitle = "Distribution";
    static final int Win_Width = 600, Win_Height = 400;
    static final char UNIX_file_separator = '/';

    static DPDMTFTP ftp;
    static Thread DPDMDistribute;
    static Distribution mainFrame;

    static String loginName, loginPassword, localPath, remotePath, remoteHost;
    Menu mnuFile, mnuHelp;
    MenuItem miExit, miSetup, miAbout;
    MenuBar mainMenu;

    public Distribution(String s) {
        super(s);
        addWindowListener(this);
    }

    public void start() {
        DPDMDistribute = new Thread(this);
        DPDMDistribute.start();
    }

    public void stop() {
        DPDMDistribute.stop();
    }

    public void run() {
        while (true) {
            OnWantSendFile();

            try {
                Thread.currentThread().sleep(detect_timeout);
            } catch (InterruptedException e) {} 
        }
    }

    static void MoveFile(String src, String dest) throws IOException {
        int c;
        long i, fsize;

        RandomAccessFile rf = new RandomAccessFile(src, "r");
        RandomAccessFile wf = new RandomAccessFile(dest, "rw");

        fsize = rf.length();
        for(i=0; i<fsize; i++) {
        
        
        }

        A.6
c = rf.readUnsignedByte();
wf.write((byte) c);
}
rf.close();
wf.close();

File fl = new File(src);
fl.delete();
}

static boolean OnSendFile(String filename) {
    try {
        ftp = new DerbyFTP(remoteHost, loginName, loginPassword);
        try {
            ftp.PutFile(localPath + File.separator + filename, remotePath + UNIX_file_separator + filename);
        } catch (IOException e) {
            new DialogBox(mainFrame, ProgramTitle, "Can't distribute object to remote site.\n\nNo permission or invalid path name", 1);
            ftp.CloseConnection();
            return false;
        }
        ftp.CloseConnection();
        } catch (IOException e) {
            new DialogBox(mainFrame, ProgramTitle, "Can't login to remote site.\nIncorrect password or Login name.", 1);
            return false;
    }

    // move the file to the Sent directory
    /*
    int c;
    long i, fsize;
    try {
        RandomAccessFile rf = new RandomAccessFile(localPath + File.separator + filename, "r");
        RandomAccessFile wf = new RandomAccessFile(localPath + File.separator + "sent" + File.separator + filename, "rw");
        fsize = rf.length();
        for(i=0;i<fsize;i++) {
            c = rf.readUnsignedByte();
            wf.write((byte) c);
        }
        rf.close();
        wf.close();
    } catch (IOException e) {
        new DialogBox(mainFrame, ProgramTitle, "Can't move the file to " + localPath + File.separator + "sent", 1);
        return false;
    }*/
    try {
        MoveFile(localPath + File.separator + filename, localPath + File.separator + "Distributed" + File.separator + filename);
    } catch (IOException e) {
        new DialogBox(mainFrame, ProgramTitle, "Can't move the file to " + localPath + File.separator + "Distributed", 1);
        return false;
    }
    System.out.println("Distributed : " + localPath + File.separator + filename);
    return true;
}
static void OnExit() {
    DialogBox a = new DialogBox(mainFrame, ProgramTitle, "Exit DPDM distributed application?", 3);
    if(a.answer == "Yes") {
        new DialogBox(mainFrame, ProgramTitle, "Thank you for using DPDM distributed application!", 1);
        System.exit(0);
    }
}

static int OnWantSendFile() {
    int filesent = 0;
    int nfile = 0;
    String Distributed = new String(""");
    File f = new File(localPath + File.separator + "x");
    f = new File(f.getAbsolutePath());
    String[] flist = f.list();
    // Check how many file need to send first
    if(flist != null) {
        for(int i = 0; i < fлист.length; i++) {
            File f1 = new File(localPath + File.separator + fлист[i]);
            if(f1.isFile()) {
                nfile++;
            }
        }
    }
}

if(nfile > 0 && flist != null) {
    DialogBox a = new DialogBox(mainFrame, ProgramTitle, "Distribute objects to remote site?", 3);
    if(a.answer == "Yes") {
        for(int i = 0; i < fлист.length; i++) {
            File f1 = new File(localPath + File.separator + fлист[i]);
            File f2 = new File(localPath + File.separator + "Distributed" + fлист[i]);
            if(f1.isFile()) {
                if(f2.exists()) {
                    DialogBox al = new DialogBox(mainFrame, ProgramTitle, "File distributed before, distribute again?", 3);
                    if(al.answer == "Yes") {
                        if(OnSendFile(fлист[i]) == true) {
                            filesent++;
                            Distributed = Distributed + "\n"
                            + fлист[i];
                        } else {
                            return 1;
                        }
                    } else {
                        if(OnSendFile(fлист[i]) == true) {
                            filesent++;
                            Distributed = Distributed + "\n"
                            + fлист[i];
                        } else {
                            return 1;
                        }
                    }
                } else {
                    for(int i = 0; i < fлист.length; i++) {
                        File f1 = new File(localPath + File.separator + fлист[i]);
                        if(f1.isFile()) {
                            try {
                                Distributed = Distributed + fлист[i];
                            } catch (Exception e) {
                                // Handle exception
                            }
                        }
                    }
                }
            }
        }
    }
}

A.8
MoveFile(localPath + File.separator + flist[i], localPath + 
File.separator + "Not_Distributed" + File.separator + flist[i]);

        ) catch (IOException e) {
            new DiabloBox(mainFrame, ProgramTitle, "Can't move the file to " 
+ localPath + File.separator + "Not_Distributed", 1);
            return 1;
        }
    }

    if(fileSent > 0) {
        new DiabloBox(mainFrame, ProgramTitle, "The following object(s) was/were 
distributed to remote site\nand allocated to local path " + localPath + File.separator + 
"Distributed directory\n" + Distributed, 1);
    }
    return fileSent;

    static void OnSetup(Frame f) {
        int wantExit=0;

        DPDMSetup setup = new DPDMSetup(f, "DPDM System Distribution Setup", 
true);

        setup.init();
        setup.setDefault(loginName, loginPassword, remoteHost, localPath,
remotePath);

        setup.show();

        loginName = (setup.loginName);
        loginPassword = (setup.loginPassword);
        localPath = (setup.localPath);
        remotePath = (setup.remotePath);
        remoteHost = (setup.remoteHost);
        setup.dispose();
    }

    static void OnAbout() {
        new DiabloBox(mainFrame, ProgramTitle, ProgramTitle + " Version 1.01", 1);
    }

    static void OnIntroduction() {
        new DiabloBox(mainFrame, ProgramTitle, "Hello, this is a introduction\n\n" + 
"send line\n" + 
"3rd line"
, 1);}

    public static void main(String argv[]) {
        mainFrame = new Distribution("DPDM System Setup");
        mainFrame.init();
        mainFrame.show();

        loginName = new String(default_loginname);
        loginPassword = new String("");
        localPath = new String(default_localpath);
        remotePath = new String(default_remotepath);
        remoteHost = new String(default_remotehost);

        OnSetup(mainFrame);

        mainFrame.start();
    }

    public void init()
```java
{  
    setBackground(Color.gray);
    mainMenu = new MenuBar();

    mnuFile = new Menu("File");
    mnuFile.add("Setup");
    mnuFile.addSeparator();
    mnuFile.add("Send File");
    mnuFile.addSeparator();
    mnuFile.add("Exit");
    mainMenu.add(mnuFile);

    mnuHelp = new Menu("Help");
    mnuHelp.add("Introduction");
    mnuHelp.add("About");
    mainMenu.add(mnuHelp);

    setMenuBar(mainMenu);

    Dimension d = Toolkit.getDefaultToolkit().getScreenSize();
    resize(Win_Width, Win_Height);
    move((d.width - Win_Width)/2, (d.height - Win_Height)/2);

    mnuFile.addActionListener(this);
    mnuHelp.addActionListener(this);
}

public void windowClosing(WindowEvent e) {
    OnExit();
}

public void windowClosed(WindowEvent e) {
}

public void windowDeiconified(WindowEvent e) {
}

public void windowIconified(WindowEvent e) {
}

public void windowOpened(WindowEvent e) {
}

public void windowDeactivated(WindowEvent e) {
}

public void windowActivated(WindowEvent e) {
}

public void actionPerformed(ActionEvent evt) {
    String label = evt.getActionCommand();

    if(label == "Exit")
        OnExit();
    else if(label == "About")
        OnAbout();
    else if(label == "Introduction")
        OnIntroduction();
    else if(label == "Send File") {
        if(OnWantSendFile() == 0) {
            new DialogBox(mainFrame, ProgramTitle, "No object ready to distribute", 1);
        }
    }
    else if(label == "Setup")
        OnSetup(mainFrame);
}
```
import java.awt.*;
import java.awt.event.*;

public class DialogBox extends Dialog{
    public String answer;
    Button cmd1, cmd2;
    static Frame parentFrame;

    public DialogBox(Frame f, String Title, String message, int boxtype) {
        super(f, Title, true);
        parentFrame = f;

        try {
            setBackground(Color.lightGray);
        } catch (Exception E) { }

        Panel p = new Panel();
        p.setLayout(null);

        int myHeight = 20;
        int myWidth = 400;

        while (message.length() > 0) {
            int ix = message.indexOf('
');
            String line;
            if (ix >= 0) {
                line = message.substring(0, ix);
                message = message.substring(ix+1);
            } else {
                line = message;
                message = "";
            }
            Label l = new Label(line);
            p.add(l);
            l.reshape(10, myHeight, myWidth, 20);
            myHeight += 20;
        }

        switch(boxtype) {
        case 1:
            cmd1 = new Button("OK");
            break;
        case 2:
            cmd1 = new Button("OK");
            cmd2 = new Button("Cancel");
            break;
        case 3:
            cmd1 = new Button("Yes");
            cmd2 = new Button("No");
            break;
        }

        myHeight += 30;
        int cmdWidth = 80;
        int cmdHeight = 30;

        if(boxtype == 1) {
            p.add(cmd1);
            cmd1.reshape((myWidth - 80) / 2, myHeight, cmdWidth, cmdHeight);
        } else { 
            p.add(cmd1);
            p.add(cmd2);
        }
cmd1.reshape(myWidth/2 - 5 - cmdWidth, myHeight, cmdWidth, cmdHeight);
cmd2.reshape(myWidth/2 + 5, myHeight, cmdWidth, cmdHeight);
}

myHeight += 50;
p.resize(myWidth, myHeight);
add(p);
resize(myWidth, myHeight+10);
Dimension d = parentFrame.size();
move((d.width - myWidth)/2 + parentFrame.getLocation().x, (d.height - myHeight)/2 + parentFrame.getLocation().y);
setResizable(false);

show();
}

public boolean action(Event e, Object o) {
try{
    Button b = (Button)e.target;
    String lbl = b.getLabel();
    answer = lbl;
    hide();
} catch(Exception E) {}

return false;
}
'**************************
'// createProject.asp //
'**************************

<% @Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgcolor="black">
<P>&nbsp;</P>
<% projectCode = Request("projectCode")
projectName = Request("projectName")
productCode = Request("productCode")
productName = Request("productName")
startDate = Request("startDate")
finishedDate = Request("finishedDate")
newProject = Request("newProject")
projectDescription = Request("projectDescription")

if newProject="" then
  conn=dsn=dpdm_db; uid=dpdm_user;pwd=
  sql="insert dpdm_project values('"&projectCodes"', '"&projectName"', '"&startDate"', '"&finishedDate"', '"&productCodes"', '"&productName"', '"&projectDescription"')"
  set rs=server.CreateObject("adodb.recordset")
  rs.Open(sql, conn)
  Response.Write("<table border='3' width='350' background='img/background.gif'>
<tr><td align='center' height='50'>
&lt;font size='3' face='arial' color='navy'&gt;
Project had been created succefully!
Response.Write("&lt;/font&gt;
</td></tr></table")
else

</%

<table bgcolor="#bc4e99" border="5" cellspacing="0" cellpadding="0" width="600">
<tr>
<td align="center" height='50'>
&lt;font size='5' face='arial' color='navy'&gt;&lt;b&gt;New Project&lt;/b&gt;&lt;/font&gt;
</td></tr>
</table>

<form method="post" action="createProject.asp" name="newProject">
<table><tr height='35'>
&lt;td color='navy'&gt;&lt;input type='text' name='projectCode' size='15' maxlength='20'&gt;

&lt;td width='50'&gt;
&lt;br&gt;&lt;/td>

&lt;td color='navy'&gt;Project Name:&lt;/font&gt;

&lt;td&gt;&lt;input type='text' name='productName' size='15' maxlength='20'&gt;

</form>

A. 13
<table>
<thead>
<tr>
<th><strong>Product Code:</strong></th>
<th>&lt;input type=&quot;text&quot; name=&quot;productCode&quot; size=&quot;15&quot; maxlength=&quot;20&quot; /&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project Name:</strong></td>
<td>&lt;input type=&quot;text&quot; name=&quot;productName&quot; size=&quot;15&quot; maxlength=&quot;20&quot; / &gt;</td>
</tr>
<tr>
<td><strong>StartDate:</strong></td>
<td>&lt;input type=&quot;text&quot; name=&quot;startDate&quot; size=&quot;15&quot; maxlength=&quot;20&quot; / &gt;</td>
</tr>
<tr>
<td><strong>Finished Date:</strong></td>
<td>&lt;input type=&quot;text&quot; name=&quot;finishedDate&quot; size=&quot;15&quot; maxlength=&quot;20&quot; / &gt;</td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>&lt;textarea cols=&quot;48&quot; rows=&quot;5&quot; name=&quot;projectDescription&quot; /&gt;</td>
</tr>
</tbody>
</table>

A. 14
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<!DOCTYPE BODY background="img/background.gif" bgColor=Gold-->
<BODY bgColor=White topmargin="0" leftmargin="0" rightmargin="0" cellspacing="0"
cellpadding="0" background="img/background.gif">
<center>
<table width="210">
<tr align="center" bgcolor="#C03CFF" width="100%">
<form method="post" action="dataList.asp" id=form1 name=form1>
<td height="35" width="100" align="left">
<font face="arial" size="3" color=Blue>Project ID : </font>

<input type="text" name="find_input" size="8"/>
<input type="submit" name="go" value="Go">
</form>
</td>
</tr>
</table>
</center>

workspace=session("myuserClass")

'myFind_input=Request("myFind_input")
'Response.Write("workspace = ")
'Response.Write(workspace)

find_input=Request("find_input")

'Response.Write("find_input = ")
'Response.Write(find_input)

class_1="Document"
class_2="Drawing"
class_3="Solid Part"

conn=dsn-dpdm db:uid=dpdm_user;pwd="

'sql_cl1=select * from dpdm_data where project_id="&find_input&" and
data_class="class_1" and workspace="workspace&"
'sql_c2=select * from dpdm_data where project_id="&find_input&" and
data_class="class_2" and workspace="workspace&"
'sql_c3=select * from dpdm_data where project_id="&find_input&" and
data_class="class_3" and workspace="workspace&"

sql_cl1=select * from dpdm_data where project_id="&find_input&" and
data_class="class_1" and workspace="workspace&"
sql_c2=select * from dpdm_data where project_id="&find_input&" and
data_class="class_2" and workspace="workspace&"
sql_c3=select * from dpdm_data where project_id="&find_input&" and
data_class="class_3" and workspace="workspace&"

set rs_cl1=server.CreateObject("adodb.recordset")
rs_cl1.open(sql_cl1).conn
set rs_c2=server.CreateObject("adodb.recordset")
rs_c2.open(sql_c2).conn
set rs_c3=server.CreateObject("adodb.recordset")
rs_c3.open(sql_c3).conn

if not rs_cl1.eof then
'Response.Write("<br>Hello world! hello world!"
"</br>
</table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif">

A. 16
<tr><td bgcolor=DeepSkyBlue>

| rs_c1.MoveNext
| wend
| end if
|
|--table--

| if not rs_c2.eof then
|
|--table border="1" width="590"-->
<tr><td bgcolor=DeepSkyBlue>

| rs_c2.MoveNext
| wend
| end if
|
|--table--

| if not rs_c3.eof then
|
|--table border="1" width="590"-->
<tr><td bgcolor=DeepSkyBlue>

| rs_c3.MoveNext

A.17
wend
end if
%
</table>
</center>
</BODY>
</HTML>
<%@ Language=VBScript %>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=big5">
</head>
<body bgcolor="#c9ff" topmargin="0" leftmargin="0">
<center>
<table width="100%">
<tr align="center">
<td height="35">
<font face="arial" size="5" color=Navy><b>DPDM System Product Data Viewer</b></font>
</td>
</tr>
</table>
</center>
</body>
</html>
```vbnet
<%@ Language=VBScript %>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=big5">
</head>
<body bgcolor="#FFFFFF">
<center>
<% 
    objectId=Request("obj_id")
    Response.Write("objectId = ")
    Response.Write(objectId)
%>
<br>
<img src="img/%-objectId%.gif">
</center>
</body>
</html>
```
<%@ Language=VBScript %>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=big5">
</head>
<body background="img/background.gif">
<center>
<p>&nbsp;</p>
<table>
<tr>
<td align="center">
<font face="arial" size="5" color=OrangeRed><b>Welcome to DPDM System <br>Product Data Viewer</b></font>
</td>
</tr>
</table>
</center>
</body>
</html>
```vbscript
'// distribute.asp //
'***************

<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>

<P>&nbsp;</P>
<$
    content=Request("content")
    'Response.Write("content = ")
    'Response.Write(content)
    object=content
    set fs=server.CreateObject("scripting.fileSystemObject")
    fs.CopyFile
    "e:\Inetpub\wwwroot\dpdmSystem\dpdmVault\"&object&"","e:\Inetpub\wwwroot\dpdmSystem\distributedObject\"&objects"
    fs.CopyFile
    "e:\Inetpub\wwwroot\dpdmSystem\dpdmVault\"&object&"","e:\Inetpub\wwwroot\dpdmSystem\distribution\"&object&"
$>

<font color="blue" size="5" face="arial">Data object has been distributed!</font>
</BODY>
</HTML>
```
'// dpdm_login.asp //
***************

<% Language=VBScript %>
<html>
<head>
<meta name="-generator" content="Microsoft Visual Studio 6.0">
</head>
<body bgcolor="#000080">
<center>
<form method="post" action="login_check.asp">
<table align="center" width="250" height="120">
<tr>
<td align="center" name="dpdm_login">
<font face="arial" size="3" color="black">
Please enter your User name and Password
</font>
<br>
<table border="5" bgcolor="#ffdead" width="400" height="200">
<tr>
<td align="center" width="150">
<input type="text" name="usr_name" size="15" max_length="15">
</td>
</tr>
<tr align="center">
<td align="center">
<input type="password" name="usr_pwd" size="15" max_length="15">
</td>
</tr>
<tr align="center">
<td align="center">
<input type="submit" name="submit" value="Submit">
</td>&nbsp;&nbsp;</tr>
<tr align="center">
<td align="center">
<input type="reset" name="reset" value="Reset">
</td>
</tr>
</table>
</font>
</td>
</tr>
</table>
</form>
</center>
</body>
</html>
<%@ Language=VBScript %>
<% Response.Buffer=true%>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>

myUserName = session("myUserName")
myPassword = session("myPassword")
myUserClass = session("myUserClass")
myRealName = session("myRealName")

' Response.Write("myUserName = ")
' Response.Write(myUserName)
' Response.Write("<br>myPassword = ")
' Response.Write(myPassword)
' Response.Write("<br>myUserClass = ")
' Response.Write(myUserClass)
' Response.Write("<br>myRealName = ")
' Response.Write(myRealName)

dpdm_op=Request("dpdm_op")
' Response.Write("<br>dpdm_op = ")
' Response.Write(dpdm_op)

ws_class=Request("ws_class")
obj_id=Request("obj_id")
find_input=request("find_input")
obj_version=Request("obj_version")
modifiedDate=date() & " - " & time()

' Response.Write("<br>ws_class = ")
' Response.Write(ws_class)
' Response.Write("<br>obj_id = ")
' Response.Write(obj_id)
' Response.Write("<br>modifiedDate = ")
' Response.Write(modifiedDate)

conn="dsn=dpdm_db;uid=dpdm_user;pwd="
sql_op="update dpdm_data set state='&dpdm_op&', modified_by='"&myRealName&", last_modified_date='"&modifiedDate&", parent_version='"&obj_version&" where obj_id='"&obj_id&"

set rs_sql_op=server.CreateObject("adodb.recordset")

if dpdm_op="Register" then
Response.Redirect "register Frm.asp"
end if

if dpdm_op="Checked-In" then
rs_sql_op.Open(sql_op, conn
else if dpdm_op="Checked-Out" then
sql_out="select * from dpdm_data where obj_id='"&obj_id&"
set rs_out=server.CreateObject("adodb.recordset")
rs_out.Open(sql_out, conn

obj_id = rs_out("obj_id")
obj_id = rs_out("obj_id")
version = rs_out("version")
Response.Write("version = ")
Response.write(version)
newVersion = version + 0.1
Response.write("newVersion = ")
Response.write(newVersion)
state = rs_out("state")
content = rs_out("content")
site = rs_out("site")
create_date = rs_out("create_date")
created_by = rs_out("created_by")
approved_by = rs_out("approved_by")
project_id = rs_out("project_id")
modified_by = rs_out("modified_by")
last_modified_date = rs_out("last_modified_date")
parent_version = rs_out("parent_version")
description = rs_out("description")
user_class = rs_out("user_class")
data_class = rs_out("data_class")

sql_insert="insert into dpdm data values("&obj_id&", "&newVersion&", "&state&", "&content&", "&site&", "&create_date&", "&created_by&", "&approved_by&", "&project_id&", "&modified_by&", "&last_modified_date&", "&parent_version&", "&description&", "&user_class&", "&data_class&")"
sql_out2="update dpdm data set version="&newVersion&", state="&dpdm_op&", modified_by="&myRealName&", last_modified_date="&modifiedDate&", parent_version="&obj_version&" where obj_id="&obj_id&"

set rs_insert=Server.CreateObject("adodb.recordset")
rs_insert.open(sql_insert), conn
set rs_out2=server.CreateObject("adodb.recordset")
rs_out2.Open(sql_out2), conn
else if dpdm_op="Distribute" then
sql_dist="select content from dpdm data where obj_id="&obj_id&"
set rs_dist=server.CreateObject("adodb.recordset")
rs_dist.Open(sql_dist), conn
content=rs_dist("content")
Response.Write("content = ")
Response.Write(content)
Response.Redirect("distribute.asp?content="&content&"
else if dpdm_op="Released" then
newVersion=Cint(newVersion)
sql_rel=select version from dpdm data where obj_id="&obj_id&"
set rs_sql_rel=server.CreateObject("adodb.recordset")
rs_sql_rel.Open(sql_rel), conn
newVersion=Request("rs_sql_rel")

Response.Write("newVersion = ")
Response.Write(newVersion)
newRelVersion=Cint(newVersion)+1
Response.Write("<hr>newVersion = ")
Response.Write(newRelVersion)
sql_rel=update dpdm data set version="&newRelVersion&", state="&dpdm_op&", approved_by="&myRealName&", last_modified_date="&modifiedDate&" where obj_id="&obj_id&"
set rs_rel=server.CreateObject("adodb.recordset")
rs_rel.Open(sql_rel), conn

end if
end if
end if
end if

Response.Redirect "find_project.asp?find_input="$find_input

%>

</BODY>
</HTML>
<%@ Language=VBScript %> 
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY background="img/background.gif">
<center>
<%yes=Request("yes")%>
<P>&nbsp;</P>
<P>&nbsp;</P>
<P>&nbsp;</P>
<%if yes="" then%>
<font size="6" face="arial" color=OrangeRed><b>Thank you for using DPDM On Line System!</b></font>
<%else%>
<font size="6" face="arial" color=OrangeRed><b>Welcome to DPDM On Line System!</b></font>
<%end if%>
</center>
</BODY>
</HTML>
```html
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgColor=black>
<center>
<table cellspacing="1" border="5" width="95%" cellpadding="0">
  <tr align="center" bgcolor="#c9c9ff">
    <td height="40">
      Information of Product Data</td>
  </tr>
</table>
</center>

<table border="5" cellspacing="1" cellpadding="1" width="95%" background="img/background.gif">
  <tr>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >Object ID</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >Object version</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >DPDM Operation</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >User</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >Time Stamp</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >Project ID</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial" >Site ID</font>
    </td>
  </tr>
  <tr>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial">%rs!"obj_id"%</font>
    </td>
    <td bgcolor=NavajoWhite>
      <font size="3" face="varial">%rs!"obj_version"%</font>
    </td>
    </tr>
</table>

A. 28
```
<table>
<thead>
<tr>
<th>operation</th>
<th>user_realName</th>
<th>timeStamp</th>
<th>project_id</th>
<th>site_id</th>
</tr>
</thead>
</table>

```html
<% rs.MoveNext %>
end if
```

```html
</table>
</center>
</BODY>
</HTML>
```
```html
<!-- Language=VBScript -->

<% @Language=VBScript %>
<%@ LANGUAGE=VBScript %>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgcolor=black>
<center>
<table width="60%" border="5" cellpadding="0" cellspacing="0">
<tr align="center" bgcolor="#cac9ff">
<td height="40">
<font size="3" face="arial" color="navy"><b>DPDM System Log</b></font>
Information of Product Data</td>
</tr>
<tr align="center" bgcolor="#c9e9ff">
<form method="post" action="dpdmLogH.asp" id=form1 name=form1>
<td height="35">
<font size="3" face="arial" color="navy">Product Data Object ID</font>
</td>
</tr>
<tr align="center" bgcolor="#c9e9ff">
<td size="12"><input type="text" name="objectID" value=""></td>
</tr>
<tr align="center" bgcolor="#c9e9ff">
<td><input type="submit" name="go" value="Go"></td>
</tr>
</table>
</center>
</BODY>
</HTML>

<% objectID=Request("objectID")
'Response.Write("objectID = ")
'Response.Write(objectID)
conn="dsn=dpdm_db;uid=dpdm_user;pwd="
if objectID="" then
sql="select distinct(obj_id), project_id, site_id from dpdm_log"
else
sql="select distinct(obj_id), project_id, site_id from dpdm_log where obj_id='"&objectID&"'"'
end if
set rs=server.CreateObject("adodb.recordset")
rs.open(sql,conn)
if not rs.eof then
%
<table border="5" cellspacing="1" cellpadding="1" width="60%"
background="img/background.gif" topmargin="0">
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial" >Object ID</font></td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial" >Project ID</font></td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial" >Site ID</font></td>
</tr>
</table>
%
while not rs.EOF
%
<form method="post" action="dpdmLogDetail.asp" name="dpdm_log">
<tr>
<td size="3" face="varial" ><a href="dpdmLogDetail.asp?objectID=<%rs("obj_id")%>"%&%<font size="3" face="varial" >&%rs("obj_id")%></font></a>
</tr>
</form>
A 30
```
<td>
  <font size="3" face="varial">
    <% rs("site_id") %>
  </font>
</td>

<% rs.MoveNext %>
    wend
  end if
%
</table>
</center>
</BODY>
</HTML>
```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta name="GENERATOR" content="Microsoft Visual Studio 6.0">
</head>
<body background="img/background.gif" bgcolor="Gold">
<center>

<table cellspacing="1" border="5" width="95%" cellpadding="0" bgcolor="#cac9ff">
<tr><td height="80" align="middle" align="center">
<font size="5" face="arial" color="Navy"><b>DPDM System Distribution Operation</b></font>
</td></tr>
<tr>
<form method="post" action="SelectDistributeObj.asp" id="form1" name="form1">
<td height="35">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<input type="text" name="projectCode" size="12">
<input type="submit" name="go" value="Retrieve Data"
style="width=120">
</td></tr>
</form>
</table>

`myFind_input=Request("myFind_input")`
`Response.Write("myFind_input = ")`
`Response.Write(myFind_input)`

find=Request("find")
projectCode=Request("projectCode")

`Response.Write("find_input = ")`
`Response.Write(find_input)

class_1="Document"
class_2="Drawing"
class_3="Solid Part"
conn="dsn=dpdm_db;uid=dpdm_user;pwd=

sql_c1="select * from dpdm_data where project_id='"&projectCodes"' and data_class='"&class_1"' and state='released'"
sql_c2="select * from dpdm_data where project_id='"&projectCodes"' and data_class='"&class_2"' and state='released'"
sql_c3="select * from dpdm_data where project_id='"&projectCodes"' and data_class='"&class_3"' and state='released'"
```
set rs_c1=server.CreateObject("adodb.recordset")
rs_c1.open(sql_c1),conn
set rs_c2=server.CreateObject("adodb.recordset")
rs_c2.open(sql_c2),conn
set rs_c3=server.CreateObject("adodb.recordset")
rs_c3.open(sql_c3),conn

if projectCode="" then
%

<form method="post" action="distributeObj.asp" id=form1 name=form1>
</form>
</td>
</tr></table>

<table border="5" cellspacing="1" cellpadding="1" width="95%" bgcolor="#ffffff" border="1">
<tr>
<td bgcolor="NavajoWhite" width="120">
<font size="3" face="varial" >Object Class</font>
</td>
<td bgcolor="NavajoWhite">
<font size="3" face="varial" >Object ID</font>
</td>
<td bgcolor="NavajoWhite width="15%">
<font size="3" face="varial" >Object version</font>
</td>
<td bgcolor="NavajoWhite">
<font size="3" face="varial" >Object Description</font>
</td>
</tr>
</table>

<%
if not rs_c1.eof then
%

<tr border="5" cellspacing="1" cellpadding="1" width="95%" bgcolor="#ffffff" border="1">
<td bgcolor="DeepSkyBlue" width="120">
 &nbsp;&nbsp;&nbsp;&nbsp;<font size="4" face="arial" color="blue">cb</font>
</td>
<td bgcolor="#ffffff" width="120">
<%=class_1%></td>
</tr>
</table>

<%
while not rs_c1.EOF
%

<tr border="5" cellspacing="1" cellpadding="1" width="95%" bgcolor="#ffffff" border="1">
<td align="right">
&nbsp;
</td>
<td bgcolor="#ffffff" width="120">
<input type="checkbox" name="objectId"
value="&lt;%=rs_c1("obj_id")%>"
<font size="3" face="varial" &%26 %23 rs_c1("obj_id")%></font>
<input type="checkbox" name="objectId" value="<%= rs_c3("obj_id")%>">
</td>
<td><font size="3" face="varial" >&lt;%= rs_c3("obj_id")%&gt;&lt;/font&gt;</td>
<td><font size="3" face="varial" >&lt;%= rs_c3("version")%&gt;&lt;/font&gt;</td>
<td><font size="3" face="varial" >&lt;%= rs_c3("description")%&gt;&lt;/font</td>
</tr>

&gt;
ra_c3.MoveNext
wend
end if
</table>
</table>
<tr>
<td><input type="submit" name="distributeObj" value="Distribute"/></td>
</tr>
<tr>
<td><input type="reset" name="reset" value="Re-selected"/></td>
</tr>
<tr>
<td><input type="button" name="cancel" value="Cancel" onclick="javascript:history.go(-1)"/></td>
</tr>
</table>
</form>
</center>
</BODY>
</HTML>

A. 35
<%@ Language=VBScript %>
<% Response.Buffer=true %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgcolor="black">
<center>
<table width="400" border="5" cellpadding="1" cellspacing="0" background="img/background.gif">
<tr>
<td align="center">
<table valign="middle">
<tr height="100">
<td align="center">
<form method="post" action="dpdmLogOut.asp" name="logout">
<tr height="60">
<td align="center">
<input type="submit" name="yes" value="Yes" style="width=100">
<input type="button" name="no" value="No" style="width=100">
</td>
</tr>
</form>
</td>
</tr>
</table>
</td>
</tr>
</table>
</center>
<%}
yes=Request("yes")
if yes="" then
myUserName=""
myPassword=""
myUserClass=""
myRealName=""
session("myUserName")=myUserName
session("myPassword")=myPassword
session("myUserClass")=myUserClass
session("myRealName")=myRealName
Response.Redirect"dpdmBlank.asp?yes=&yes&"
end if
%
</center>
</BODY>
</HTML>
<%- Language=VBSscript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgcolor="00000000" border="0" rightmargin="0" topmargin="0">
<center>
<% conn="dsn=dpdm_db;uid=dpdm_user;pwd="
sql="select * from dpdm_site"
set rs=server.CreateObject("adodb.recordset")
rs.open(sql),conn
%
<table width="95%" border="5" cellspacing="1" cellpadding="1">
<tr align="center" bgcolor="#bc9ff">
<form method="post" action="find_project.asp" id=form1 name=form1>
<td height="35">
<font face="arial" size="4" color=Navy><b>Distributed Site Information</b></font>
</td>
</form>
</tr></table>
<table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif">
<tr>
<td bgcolor=Navy>
<font size="4" face="arial" color="white">Site ID</font>
</td>
</tr>
</table>
<table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif">
<tr>
<td bgcolor=Navy>
<font size="4" face="arial" color="white">Site Name</font>
</td>
</tr>
</table>
<table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif">
<tr>
<td bgcolor=Navy>
<font size="4" face="arial" color="white">Site Location</font>
</td>
</tr>
</table>
<table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif">
<tr>
<td bgcolor=Navy>
<font size="4" face="arial" color="white">Site Information</font>
</td>
</tr>
</table>
<%
while not rs.eof
%
<tr>
<td><font size="3" face="arial" color="#f54300"><%=rs("site_id")%></font></td>
</tr>
<tr>
<td><font size="3" face="arial" color="#f54300"><%=rs("site_name")%></font></td>
</tr>
<tr>
<td><font size="3" face="arial" color="#f54300"><%=rs("location")%></font></td>
</tr>
<tr>
<td><font size="3" face="arial" color="#f54300"><%=rs("site_info")%></font></td>
</tr>
</table>
</center>
</BODY>
</HTML>
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<body background="img/background.gif" bgColor=Gold-->
<body bgColor=black>
<center>
<table cellspacing="1" border="5" width="95%" cellpadding="0">
<tr>
<td>
<table width="100%" cellspacing="0" border="1" cellpadding="0" bgcolor="#cac9ff">
<tr align="center"><td height="40" <font size="5" face='arial'>
select case cint(workspace)
 case 1
 Response.Write("Project Team Workspace")
 case 2
 Response.Write("Project Team Workspace")
 case 3
 Response.Write("Personal Workspace")
 case 4
 Response.Write("Corporate Workspace")
 case 5
 Response.Write("Public Workspace")
end select
 Response.Write(</td></tr></table>
</font>
</td>
</tr>
<form method="post" action="find_project.asp"
 id=form1 name=form1
 value="project_code">Find project by project code
 <input type="radio" name="find"
 value="project_id">Find data object by object ID
 <input type="radio" name="find_input"
 size="12">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
 <input type="text" name="find_input"
 value="Go">
</form>
</td>
</tr>
</table>
</center>
</body>
</HTML>

find=Request("find")
find_input=Request("find_input")
'Response.Write("find_input = ")
'Response.Write(find_input)

class_1="Document"
class_2="Drawing"
class_3="Solid Part"

conn="dsn=dpdm_db;uid=dpdm_user;pwd="

sql_c1="select * from dpdm_data where project_id='"&find_input&"' and
data_class='"&class_1&"' and workspace='"&workspace&"'
sql_c2="select * from dpdm_data where project_id='"&find_input&"' and
data_class='"&class_2&"' and workspace='"&workspace&"'
sql_c3="select * from dpdm_data where project_id='"&find_input&"' and
data_class='"&class_3&"' and workspace='"&workspace&"'

set rs_c1=server.CreateObject("adodb.recordset")
rs_c1.open(sql_c1),conn
set rs_c2=server.CreateObject("adodb.recordset")
rs_c2.open(sql_c2),conn
set rs_c3=server.CreateObject("adodb.recordset")
rs_c3.open(sql_c3),conn

if not rs_c1.eof then

	<tr>
	<td bgcolor=DeepSkyBlue>
	&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&n

A.39
<form method="post" action="dpdm_op.asp" name="ws_class%i">
<tr>
<td align="center" valign="middle">
<a href="obj_detail.asp?obj_id=%=rs_cl('obj_id')%">
<font size="3" face="varial" >%=rs_cl('obj_id')%</font></a>
</td>
<td align="center" valign="middle">
<font size="3" face="varial" >%=rs_cl('create_date')%</font>
</td>
<td align="center" valign="middle">
<font size="3" face="varial" >%=rs_cl('created_by')%</font>
</td>
<td align="center" valign="middle">
<font size="3" face="varial" >%=rs_cl('version')%</font>
</td>
<td align="center" valign="middle">
<font size="3" face="varial" >%=rs_cl('state')%</font>
</td>
</tr>
</form>

if workspace<4 then

<select name="dpdm_op" style="HEIGHT: 22px" onchange="ws_class%i.submit()">
<option value="" selected>DPDM Operation</option>
<option value="Register">Register</option>
<option value="Checked-In">Check-IN</option>
<option value="Checked-Out">Check-Out</option>
<option value="newRelease">New Release</option>
<option value="Deleted">Delete</option>
<option value="Obsolete">Obsolete</option>
<option value="Distribute">Distribute</option>
</select>

<input type="hidden" name="ws_class" value="%=class_1%">
<input type="hidden" name="obj_id" value="%=rs_cl('obj_id')%">
<input type="hidden" name="obj_version" value="%=rs_cl('version')%">
</form>

end if

rs_cl.MoveNext
wend
end if

<!--/table-->

A.40
if not rs_c2.eof then

<table border="1" width="590">
  <tr>
    <td bgcolor=DeepSkyBlue>&nbsp;&nbsp;&nbsp;&nbsp;&lt;font size="4" face="arial" color="blue">&lt;b>&lt;%=class_2%&gt;&lt;/b&gt;&lt;/font&gt;&nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
    <td colspan="6"&nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  </tr>
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &nbsp;&nbsp;&nbsp;&nbsp;&lt;/td&gt;
  &n...
if workspace<4 then
%
<select name="dpdm_op" style="HEIGHT: 22px;"
onChange="ws_class+=i".submit();>
  <% response.write "<option value=' Register'>Register"
response.write "<option value=' Checked-In'>Check-In"
response.write "<option value=' Checked-Out'>Check-Out"
response.write "<option value=' New Release'>New Release"
if workspace=1 then
  response.write "<option value=' Released'>Release"
response.write "<option value=' Obsolete'>Obsolete"
response.write "<option value=' Delete'>Delete"
else
  response.write "<option value=' Distribute'>Distribute"
end if
  Response.Write "</option>
response.write "</select>
%
</td>
</tr>
<input type="hidden" name="ws_class" value="%"/>
<input type="hidden" name="obj_id" value="%"
<input type="hidden" name="obj_version" value="%"/>
</form>
<% end if %>

rs_c2.MoveNext
wend
end if

<!--/table--><%
if not rs_c3.eof then
%
<!--table border=1 width=590-->
<tr><td bgcolor=DeepSkyBlue>
  &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbs...
<input type="hidden" name="obj_id" value="<%=rs_c3("obj_id")%>">
<input type="hidden" name="obj_version" value="<%=rs_c3("version")%>">
<input type="hidden" name="find_input" value="<%=find_input%>">
</form>
<% end if %>
</tr>
<% rs_c3.MoveNext wend end if %>
</table>
</center>
</BODY>
</HTML>
'***********************
// findData BySite.asp //
'*******************************

<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" CONTENT="Microsoft Visual Studio 6.0">
</HEAD>
<!-BODY background="img/background.gif" bgColor=Gold-->
<BODY bgColor=black>
<center>

<font size="5" color="#191970" face="arial">
<table cellspacing="1" border="5" width="95%" cellpadding="0">
<tr align="center" bgcolor="#cac9ff">
<td align="center" height="55">
<font face="arial" size="4" color=Navy><br>Information of Distributed Product Data object<br></font>
</td>
</tr>
<tr align="center" bgcolor="#cac9ff">
<form method="post" action="findDataBySite.asp" id=form1 name=form1>
<td height="35">
<font face="arial" size="3" color=Navy>Find the distributed data in site:

<input type="text" name="remoteSite" size="12"&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
<input type="submit" name="go" value="Go">
</td>
</form>
</table>
</tr>

<% remoteSite=Request("remoteSite")
'find_input=Request("find_input")
'Response.Write("find_input = ")
'Response.Write(find_input)
conn="dsn=dpdm_db;uid=dpdm_user;pwd="
sql="select * from dpdm_data where site_id='&remoteSite&'" order by project_id"
set rs=server.CreateObject("adodb.recordset")
rs.open(sql,conn)
if not rs.eof then
%
<table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif">
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Object ID</font>
</td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Create date</font>
</td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Created By</font>
</td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Object version</font>
</td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Object State</font>
</td>
</tr>
</table>
A. 45
<% while not rs.EOF %>

<tr>
<td><a href="obj_detail.asp?obj_id Nº=rs("obj_id")">$</a>\</td>
<td><font size="3" face="varial" face="varial" face="varial" face="varial" face="varial">$=rs("obj_id")$</font></td>
<td><font size="3" face="varial" face="varial" face="varial" face="varial" face="varial">$=rs("create_date")$</font></td>
<td><font size="3" face="varial" face="varial" face="varial" face="varial" face="varial">$=rs("created_by")$</font></td>
<td><font size="3" face="varial" face="varial" face="varial" face="varial" face="varial">$=rs("version")$</font></td>
<td><% if isnull(rs("state")) then
    objState = &nbsp;
else
    objState = rs("state")
end if
%
<font size="3" face="varial" face="varial" face="varial" face="varial" face="varial">$=objState$</font></td>
</tr>

<% rs.MoveNext
wend
end if
%>
</table>
</center>
</BODY>
</HTML>
<table cellspacing="1" border="5" width="95%" cellPadding="0">
  <tr align="center" bgcolor="#cccccc" td height="55">
    <font face="arial" size="4" color="Navy">Remote Site Information</font>
    of Distributed Product Data object</b></font>
  </tr>
  <tr align="center" bgcolor="#cccccc">
    <form method="post" action="findSiteByPID.asp" id=form1 name=form1>
      <input type="radio" name="find" value="project_id">Find project by
      project code
      <input type="radio" name="find" value="product_id">Find data object
      by object ID&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
      <input type="text" name="findinput" size="12">
      <input type="submit" name="go" value="Go">
    </form>
  </tr>
</table>

find=Request("find")
findinput=Request("findinput")
Response.Write("find_input = ")
Response.Write(find_input)

conn="dsn=dpdm_db;uid=dpdm_user;pwd=")
if find="project_id" then
sql1="select * from dpdm_site where site_id in (select distinct(site_id) from dpdm_data
where project_id ="findinputs")"
set rsl=server.CreateObject("adodb.recordset")
rsl.open(sql1,conn
end if
if find="project_id" then
%
</table cellspacing="1" border="5" cellPadding="0" width="95%" background="img/background.gif">
<tr>
  <td bgcolor=Navy>
    <font size="4" face="arial" color="white">Site ID</font>
  </td>
</tr>
<tr>
  <td bgcolor=Navy>
    <font size="4" face="arial" color="white">Site Name</font>
  </td>
</tr>
<tr>
  <td bgcolor=Navy>
    <font size="4" face="arial" color="white">Site Location</font>
  </td>
</tr>

A. 47
<table>
<thead>
<tr>
<th>Site Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>$site_id$</td>
</tr>
<tr>
<td>$site_name$</td>
</tr>
<tr>
<td>$location$</td>
</tr>
</tbody>
</table>

```
<% while not rs1.eof %>
<tr><font size="3" face="arial" color="white">Site ID</font></tr>
<tr><font size="4" face="arial" color="white">Site Name</font></tr>
<tr><font size="4" face="arial" color="white">Site Location</font></tr>
<tr><font size="4" face="arial" color="white">Object ID</font></tr>
<tr><font size="4" face="arial" color="white">Object Version</font></tr>
<% end if %>
```
<font size="3" face="arial" color="#543000"><%=rs2("version")%>
</td>
</tr>
<%}
rs2.MoveNext
wend
end if
%
</table>
</center>
</BODY>
</HTML>

A. 49
'**********************************************************
' // login_check.asp //
'**********************************************************

<% Language=VBScript %>
<% Response.Buffer=true %>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>
<%}

usr_name=Request.Form("usr_name")
usr_pwd=Request.Form("usr_pwd")

'session("usr_name")=Request.Form("usr_name")
'session("usr_pwd")=Request.Form("usr_pwd")

'set conn=server.CreateObject("adodb.connection")
conn="dsn=dpdm_db;uid=dpdm_user;pwd="
' conn.Open "dpdm_db"

sql="select * from dpdm_user where user_name="%usr_name&" and user_password="%usr_pwd&"

''set rs=conn.Execute(sql)
set rs=server.CreateObject("adodb.recordset")
rs.open(sql),conn

myUserName=rs("user_name")
myPassword=rs("user_password")
myUserClass=rs("user_class")
myRealName=rs("real_name")
session("myUserName")=myUserName
session("myPassword")=myPassword
session("myUserClass")=myUserClass
session("myRealName")=myRealName

if rs.eof then
  ' Response.Write("You are not a registered user.")
%

<table border="1" align="center" height="100" width="500">
<tr>
  <td align="center" border="1">
    <table align="center" border="1">
      <tr>
        <td><font size="5" color="red">YOU ARE NOT A REGISTERED USER!</font></td>
      </tr>
    </table>
  </td>
<tr align="middle">
  <br>
  <td>
    <form method="post"

action="dpdm_login.asp" name="retry">
  <input type="submit" name="try

value="retry">
  </form>
</td>
</tr>
</table>

A. 50
else
    Response.Write("Welcome")
    Response.Redirect "dpdm_main_menu.asp"
%
</tr>
</table>
</td>
</tr>
</table>
</body>
</html>
```vbnet
'// login_check.asp //

<% Language=VBScript %>
<% Response.Buffer=true %>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>
<%

usr_name=Request.Form("usr_name")
usr_pwd=Request.Form("usr_pwd")

'session("usr_name")=Request.Form("usr_name")
'session("usr_pwd")=Request.Form("usr_pwd")

'set conn=server.CreateObject("adodb.connection")
conn=dsn=dpdm_db;uid=dpdm_user;pwd=
'  conn.Open "dpdm db"
sql="select * from dpdm_user where user_name='"&usr_name&"' and
user_password='"&usr_pwd&"'"

'set rs=conn.Execute(sql)
set rs=server.CreateObject("adodb.recordset")
rs.open(sql),conn

myUserName=rs("user_name")
myPassword=rs("user_password")
myUserClass=rs("user_class")
myRealName=rs("real_name")
session("myUserName")=myUserName
session("myPassword")=myPassword
session("myUserClass")=myUserClass
session("myRealName")=myRealName

if rs.eof then
  Response.Write("You are not a registered user.")
<%

<table border="1" align="center" height="100" width="500">
<tr>
  <td align="center" border="1" face="arial">
    <table align="center" border="1">
      <tr>
        <td width="500" color="red">
          YOU ARE NOT A REGISTERED USER!
        </td>
      </tr>
      <tr align="middle">
        <br>
        <td>
          <form method="post" action="dpdm_login.asp" name="retry">
            <input type="submit" name="try value="retry">
          </form>
        </td>
      </tr>
    </table>
  </td>
</tr>
</table>
<%

else

```
Response.Write("Welcome")
Response.Redirect "dpdm_main_menu.asp"

<!-table border="1" align="center">
<tr>
<td width="200">
<table border="0" bgcolor="#ffdb9" height="100"
face="arial">
<tr align="middle">
<td>
<font size="5" color="red">
Welcome!
</font>
</td>
</tr>
</table>
<tr align="middle">
<form method="post" action="dpdm_main_menu.asp" name="hello">
<input type="submit" name="submit" value="Enter DPDM System" style="BACKGROUND-COLOR: orangered;"/>
</form>
</td>
</tr>
</table-->

<% end if
%>
</FORM>
</BODY>
</HTML>
<%@ Language=VBScript %>

<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgcolor="black">

<%
obj_id=Request("obj_id")
conn="dsn=dpdm_db;uid=dpdm_user;pwd=
sql="select * from dpdm_data where obj_id=''&obj_id&''",
sql2="select * from dpdm_data where obj_id=''&obj_id&''",
set rs=server.CreateObject("adodb.recordset")
rs.Open(sql),conn
rs2.Open(sql2),conn

%>
<table bgcolor="#c9c9ff" width="98%" border="5" cellspacing="1">
<tr>
<td align="center" height="50">
<font size="5" face="arial">Object Detail Information</font>
</td>
</tr>
</table>

<table width="98%" cellspacing="1" cellpadding="5" border="5"
background="img/background.gif">
<tr bgcolor="#6ec2ff">
<td>
<font size="3" face="arial">Object ID</font>
</td>
<td>
<font size="3" face="arial">Object Version</font>
</td>
<td>
<font size="3" face="arial">Object State</font>
</td>
<td>
<font size="3" face="arial">Created Date</font>
</td>
<td>
<font size="3" face="arial">Created By</font>
</td>
<td>
<font size="3" face="arial">Modified By</font>
</td>
<td>
<font size="3" face="arial">Last Modified Date</font>
</td>
<td>
<font size="3" face="arial">Site</font>
</td>
</tr>
<tr bgcolor="#fff6d8">
<td>
<font size="3" face="arial">&rs2("obj_id")&</font>
</td>
<td>
<font size="3" face="arial">&rs2("version")&</font>
</td>
<td>
<font size="3" face="arial">&rs2("state")&</font>
</td>
</tr>
</table>

A. 54
<table border="1">
  <tr>
    <td><font size="3" face="arial">$rs2("create_date")$</font></td>
  </tr>
  <tr>
    <td><font size="3" face="arial">$rs2("created_by")$</font></td>
  </tr>
  <tr>
    <td><font size="3" face="arial">$rs2("modified_by")$</font></td>
  </tr>
  <tr>
    <td><font size="3" face="arial">$rs2("last_modified_date")$</font></td>
  </tr>
  <tr>
    <td><font size="3" face="arial">$rs2("site_id")$</font></td>
  </tr>
</table>

```xml
<% if not rs.eod then %>
do while not rs.eof
<%
  rs.MoveNext
  loop
<% %>
```

A.55
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<!--BODY background="img/background.gif" bgColor=Gold-->
<BODY background="img/background.gif">
<center>

<% user_class=session("myuserClass")
    Response.Write("user class = ")
    Response.Write(user_class)
%>

<table cellspacing="1" border="5" width="95%" cellpadding="0" bgcolor="#cac9ff">
    <tr>
        <td height="80" valign="middle" align="center">
            <font size="5" face="arial" color=Navy>DOPM System Distribution</font>
        </td>
    </tr>
    <tr>
        <form method="post" action="SelectDistributeObj.asp" id=form1 name=form1>
            <td height="35">
                <font size="3" face="arial" color="blue">Project code</font>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
                <input type="text" name="projectCode" size="12"&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
                     style="width=120">
            </td>
        </form>
    </tr>
</table>

<% 'myFind_input=Request("myFind_input")
    'Response.Write("myFind_input = ")
    'Response.Write(myFind_input)
    find=Request("find")
    projectCode=Request("projectCode")
    'Response.Write("find_input = ")
    'Response.Write(find_input)

    class_1="Document"
    class_2="Drawing"
    class_3="Solid Part"
    conn=dsn=dpdm_db;uid=dpdm_user;pwd="

    sql_c1="select * from dpdm_data where project_id='"&projectCode&"' and data_class='"&class_1&"' and state='released'"
    sql_c2="select * from dpdm_data where project_id='"&projectCode&"' and data_class='"&class_2&"' and state='released'"
    sql_c3="select * from dpdm_data where project_id='"&projectCode&"' and data_class='"&class_3&"' and state='released'"

A. 56
set rs_c1=server.CreateObject("adodb.recordset")
rs_c1.open(sql_c1),conn
set rs_c2=server.CreateObject("adodb.recordset")
rs_c2.open(sql_c2),conn
set rs_c3=server.CreateObject("adodb.recordset")
rs_c3.open(sql_c3),conn

if projectCode<>"" then
%
</form method="post" action="distributeObj.asp" id=form1 name=form1>

<table cellspacing="1" border="5" width="95%" cellspacing="0">
<tr>
<td align="center" height="50">
<font size="3" face="arial" color="blue">Remote Site Code
</font>
<input type="text" name="remoteSite" size="10" maxlength="15">
</td>
</tr>
</table>

<table border="5" cellspacing="1" cellpadding="1" width="95%" background="img/background.gif" border="1">
<tr>
<td bgcolor="NavajoWhite" width="120">
<font size="3" face="varial" >Object Class</font>
</td>
</tr>
<tr>
<td bgcolor="NavajoWhite">
<font size="3" face="varial" >Object ID</font>
</td>
</tr>
<tr>
<td bgcolor="NavajoWhite" width="15%">
<font size="3" face="varial" >Object version</font>
</td>
</tr>
<tr>
<td bgcolor="NavajoWhite">
<font size="3" face="varial" >Object Description</font>
</td>
</tr>
</table>

<% if not rs_c1.eof then%

<tr>
<td bgcolor="DeepSkyBlue">
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&n
<table>
<thead>
<tr>
<th><strong>class_2</strong></th>
<th><strong>obj_id</strong></th>
<th><strong>version</strong></th>
<th><strong>description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A.58
```html
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY bgcolor="#cac9ff">
<p>&nbsp;&lt;/p&gt;
<center>
<table border="5" cellpadding="1" cellspacing="1" width="500" align="center">
  <tr height="60" align="center">
    <td height="60" align="center">
      <font size="5" face="arial" color="blue"> Register a new data object</font>
    </td>
  </tr>
</table>
<form method="post" action="registerOp.asp"
name="register Frm"&gt;
  &lt;tr&gt;
    &lt;td width="130"&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;font size="3" face="arial"&gt;&nbsp;Object ID:&lt;/font&gt;
    &lt;/td&gt;
    &lt;td width="130"&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;input type="text" name="objID" size="12" maxlength="15"&gt;
    &lt;/td&gt;
  &lt;/tr&gt;
  &lt;tr&gt;
    &lt;td&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;font size="3" face="arial"&gt;Project ID:&lt;/font&gt;
    &lt;/td&gt;
    &lt;td width="130"&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;input type="text" name="projectID" size="12" maxlength="20"&gt;
    &lt;/td&gt;
  &lt;/tr&gt;
  &lt;tr&gt;
    &lt;td&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;font size="3" face="arial"&gt;&nbsp;Created by:&lt;/font&gt;
    &lt;/td&gt;
    &lt;td width="130"&gt;
      &lt;input type="text" name="createdBy" size="12" maxlength="15"&gt;
    &lt;/td&gt;
  &lt;/tr&gt;
  &lt;tr&gt;
    &lt;td&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;font size="3" face="arial"&gt;Create Date:&lt;/font&gt;
    &lt;/td&gt;
    &lt;td&gt;&lt;p&gt;&nbsp;&lt;/p&gt;
      &lt;input type="text" name="create Date" size="12" maxlength="20"&gt;
    &lt;/td&gt;
  &lt;/tr&gt;
&lt;/form&gt;
</center>
</body>
</html>
```
| Classification: | <input type="text" name="objClass" size="12" maxlength="15"> |
| Content: | <input type="text" name="objContent" size="12" maxlength="20"> |
| Description: | <textarea name="objDescription" cols="55" rows="5"></textarea> |

```html
<form method="post" action="dpdmBlank.asp"
name="cancel Frm" value="Cancel" onclick="cancel Frm.submit()">
<input type="reset" name="reset" value="Reset"/>
<input type="button" name="cancel" value="Cancel" onclick="back Frm.submit()"></form>
<form method="post" action="javascript:history.go(-1);"
name="back Frm"></form>
</form>
</table>
</center>
</BODY>
</HTML>
```
```vbscript
\*
*/

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta name="GENERATOR" content="Microsoft Visual Studio 6.0">
</head>
<body>
<br>
objID = Request("objID")
projectID = Request("projectID")
createdBy = Request("createdBy")
createDate = Request("createDate")
objClass = Request("objClass")
objContent = Request("objContent")
userClass = Request("userClass")
objDescription = Request("objDescription")

'Response.Write("<br> objID="")
'Response.Write(objID)
'Response.Write("<br> projectID="")
'Response.Write(projectID)
'Response.Write("<br> createdBy="")
'Response.Write(createdBy)
'Response.Write("<br> createDate="")
'Response.Write(createDate)
'Response.Write("<br> objClass="")
'Response.Write(objClass)
'Response.Write("<br> objContent="")
'Response.Write(objContent)
'Response.Write("<br> objDescription="")
'Response.Write(objDescription)
'Response.Write("<br> userClass="")
'Response.Write(userClass)

conn = newODBC ("dsn=dpdm db;uid=dpdm user;pwd="
sql = "select * from dpdm_data where obj_id = "&objID&""
set rs_sql_1 = server.CreateObject("adoDB.recordset")
rs_sql_1.Open(sql_1), conn
if not rs_sql_1.EOF then
%
<tr>
<td>
/font size="3" color="red" face="arial">The object ID was already existed. Try another object ID!
<br>
<a href="javascript:history.go(-1)">Try again!</a>
</td>
</tr>
%
```

A. 62
<% end if %>

</BODY>
</HTML>
<table cellspacing="1" border="5" width="95%" cellspacing="0">
  <tr align="center" bgcolor="#c9c9ff">
    <form method="post" action="find_project.asp" id="form1" name="form1">
      <td height="35">
        <font color="navy" size="3" face="arial">Find project by project code</font>
        <input type="text" name="projectId" size="12"><br>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
        <input type="submit" name="go" value="Go">
      </td>
    </form>
  </tr>
</table>

<form name="form1">
  <input type="hidden" value=""></form>

\%
Response.Write("\% productId=\"")
Response.Write(productId)
class_1="Document"
class_2="Drawing"
class_3="Solid Part"
conn="dsn=dpdm_db;uid=dpdm_user;pwd="
sql_c1="select * from dpdm_data where project_id='\&projectId\' and data_class='\&class_1\' and state='released'"
sql_c2="select * from dpdm_data where project_id='\&projectId\' and data_class='\&class_2\' and state='released'"
sql_c3="select * from dpdm_data where project_id='\&projectId\' and data_class='\&class_3\' and state='released'"
set rs_c1=server.CreateObject("adodb.recordset")
rs_c1.Open(sql_c1),conn
set rs_c2=server.CreateObject("adodb.recordset")
rs_c2.Open(sql_c2),conn
set rs_c3=server.CreateObject("adodb.recordset")
rs_c3.Open(sql_c3),conn
%
</table border="5" cellspacing="1" cellpadding="1" background="img/background.gif" width="550">
  <tr>
    <td height="50" align="center" bgcolor="#c9c9ff">
      <font face="arial" size="5" color="navy">Product Data Distribution</b></font>
    </td>
  </tr>
</table>
<tr>
</tr>
</table border="1" cellpadding="0" cellspacing="0" width="550">
<tr>
</tr>
</table>
<table>
<thead>
<tr>
<th>Name</th>
<th>&lt;input type=&quot;checkbox&quot; name=&quot;objectId&quot; value=&quot;rs(obj_id)&quot;/&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>&lt;font face=&quot;arial&quot; size=&quot;3&quot;&gt;Object ID&lt;/font&gt;</td>
</tr>
<tr>
<td>Code</td>
<td>&lt;font face=&quot;arial&quot; size=&quot;3&quot;&gt;Project&lt;/font&gt;</td>
</tr>
</tbody>
</table>
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY background="img/background.gif" bgColor=Gold-->
<BODY background="img/background.gif">
<center>
<table cellspacing="1" border="5" width="95%" cellpadding="0"bgcolor="#cac9ff">
<tr><td height="80" valign="middle" align="center">
<font size="5" face="arial" color="Navy"><b>DPDM System Distribution</b><br><br>
Operation</font></td></tr>
</table>
</center>

<form method="post" action="SelectDistributeObj.asp" id=form1 name=form1>
<table cellspacing="3" border="5" width="95%" cellpadding="0"bgcolor="#cac9ff">
<tr><td align="center"><font size="5" face="arial" color="blue">Project code</font><br>
<input type="text" name="projectCode" size="12"><br>
<input type="submit" name="go" value="Retrieve Data"></td></tr>
</table>
</form>

' myFind_input=Request("myFind_input")
' Response.Write("myFind_input = ")
' Response.Write(myFind_input)
find=Request("find")
projectCode=Request("projectCode")
' Response.Write("find_input = ")
' Response.Write(find_input)

conn="dsn=dpdm_db;uid=dpdm_user;pwd="
sql_c1="select * from dpdm_data where project_id='"&projectCodes"' and data_class='"&class_144"' and state='released'
sql_c2="select * from dpdm_data where project_id='"&projectCodes"' and data_class='"&class_24"' and state='released'
sql_c3="select * from dpdm_data where project_id='"&projectCodes"' and data_class='"&class_36"' and state='released'"
set rs_c1=server.CreateObject("adodb.recordset")
rs_c1.open(sql_c1),conn
set rs_c2=server.CreateObject("adodb.recordset")
rs_c2.open(sql_c2),conn
set rs_c3=server.CreateObject("adodb.recordset")
rs_c3.open(sql_c3),conn

if projectCode<>"" then
%

<form method="post" action="distributeObj.asp" id=form1 name=form1>
<table cellspacing="1" border="5" width="95%" cellpadding="0">
<tr>
<td align="center" height="50">
<br />
<input type="text" name="remoteSite" size="10" maxlength="1"
</td>
</tr>
</table>
<table border="5" cellspacing="1" cellpadding="1" width="95%"
background="img/background.gif" border="1">
<tr>
<td bgcolor=NavajoWhite width="120">
<font size="3" face="varial">Object Class</font></td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Object ID</font></td>
</tr>
<tr>
<td bgcolor=NavajoWhite width="154">
<font size="3" face="varial">Object version</font></td>
</tr>
<tr>
<td bgcolor=NavajoWhite>
<font size="3" face="varial">Object Description</font></td>
</tr>
</table>
%
if not rs_c1.eof then
%

<tr>
<td bgcolor=DeepSkyBlue>
<br />
<b class="s1"><b>&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;Object ID</b></b>
</td>
<td colspan="3">
<br />
</td>
</tr>
%
while not rs_c1.EOF
%

<tr>
<td align="right">
<br />
</td>
<td>
<input type="checkbox" name="objectId"
value="&lt;%=rs_c1("obj_id")%&gt;">
<font size="3" face="varial"&gt;&lt;%=rs_c1("obj_id")%&gt;&lt;/font>
</td>
</tr>
</form>

A.67
rs_c1.MoveNext
wend
endif

if not rs_c2.eof then
<tr bgcolor=DeepSkyBlue>
 &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbs
<table>
<thead>
<tr>
<th>&lt;td&gt;</th>
<th>&lt;font size=&quot;3&quot; face=&quot;varial&quot; &gt;$% rs_c3(&quot;version&quot;)%$&lt;/font&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&lt;font size=&quot;3&quot; face=&quot;varial&quot; &gt;$% rs_c3(&quot;description&quot;)%$&lt;/font&gt;</td>
</tr>
</tbody>
</table>

```$%
rs_c3.MoveNext
wend
end if
$%```

```<table><tr><td><input type="submit" name="distributeObj" value="Distribute"></td><td><input type="reset" name="reset" value="Re-selected"></td><td><input type="button" name="cancel" value="Cancel" onclick="javascript:history.go(-1)"></td></tr></table></form><%end if%><center></BODY></HTML>```