

Spring 2000

## An Integrated Engineering-Computation Framework for Collaborative Engineering: An Application in Project Management

Hisham Mohamed El-Sayed AbdelSalam  
*Old Dominion University*

Follow this and additional works at: [https://digitalcommons.odu.edu/mae\\_etds](https://digitalcommons.odu.edu/mae_etds)



Part of the [Industrial Engineering Commons](#), [Mechanical Engineering Commons](#), and the [Operational Research Commons](#)

---

### Recommended Citation

AbdelSalam, Hisham M.. "An Integrated Engineering-Computation Framework for Collaborative Engineering: An Application in Project Management" (2000). Master of Science (MS), Thesis, Mechanical & Aerospace Engineering, Old Dominion University, DOI: 10.25777/y52z-1513  
[https://digitalcommons.odu.edu/mae\\_etds/106](https://digitalcommons.odu.edu/mae_etds/106)

This Thesis is brought to you for free and open access by the Mechanical & Aerospace Engineering at ODU Digital Commons. It has been accepted for inclusion in Mechanical & Aerospace Engineering Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact [digitalcommons@odu.edu](mailto:digitalcommons@odu.edu).

**AN INTEGRATED ENGINEERING-COMPUTATION FRAMEWORK FOR  
COLLABORATIVE ENGINEERING: AN APPLICATION IN PROJECT  
MANAGEMENT**

by

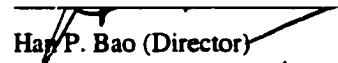
**Hisham Mohamed El-Sayed AbdelSalam**  
B.Sc. July 1996, Cairo University

A Thesis Submitted to the Faculty of  
Old Dominion University in Partial Fulfillment of the  
Requirement for the Degree of

**MASTER OF SCIENCE  
MECHANICAL ENGINEERING**

**OLD DOMINION UNIVERSITY**  
May 2000

Approved by:

  
Hao P. Bao (Director)

\_\_\_\_\_  
K. Williamson (Member)

\_\_\_\_\_  
Resit Unal (Member)

## **ABSTRACT**

# **AN INTEGRATED ENGINEERING-COMPUTATION FRAMEWORK FOR COLLABORATIVE ENGINEERING: AN APPLICATION IN PROJECT MANAGEMENT**

Hisham Mohamed El-Sayed AbdelSalam

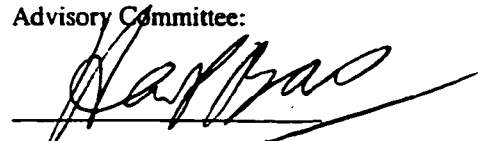
Old Dominion University, 2000

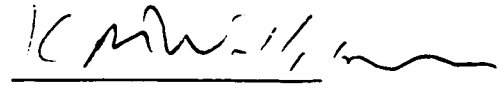
Director: Dr. Han P. Bao

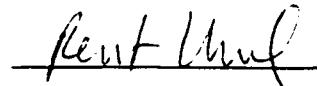
Today's engineering applications suffer from a severe integration problem. Engineering, the entire process, consists of a myriad of individual, often complex, tasks. Most computer tools support particular tasks in engineering, but the output of one tool is different from the others'. Thus, the users must re-enter the relevant information in the format required by another tool. Moreover, usually in the development process of a new product / process, several teams of engineers with different backgrounds / responsibilities are involved, for example mechanical engineers, cost estimators, manufacturing engineers, quality engineers, and project manager. Engineers need a tool (s) to share technical and managerial information and to be able to instantly access the latest changes made by one member, or more, in the teams to determine right away the impacts of these changes in all disciplines (cost, time, resources, etc.). In other words, engineers need to participate in a truly collaborative environment for the achievement of a common objective, which is the completion of the product / process design project in a timely, cost effective, and optimal manner.

In this thesis, a new framework that integrates the capabilities of four commercial software, Microsoft Excel<sup>™</sup> (spreadsheet), Microsoft Project<sup>™</sup> (project management), What's Best! (an optimization add-in), and Visual Basic<sup>™</sup> (programming language), with a state-of-the-art object-oriented database (knowledge medium), InnerCircle2000<sup>™</sup> is being presented and applied to handle the Cost-Time Trade-Off problem in project networks. The result was a vastly superior solution over the conventional solution from the viewpoint of data handling, completeness of solution space, and in the context of a collaborative engineering- computation environment.

Advisory Committee:

  
Han P. Bao (Director)

  
K. Williamson (Member)

  
Resit Unal (Member)

Dear Mom and Dad,

I am gratefully dedicating this work to you.

Just trying to say thank you.

Thank you for being you.

For everything you did for me, for everything you are doing for me, and for everything you wouldn't do, that sometimes I didn't understand until years later, thank you.

## **ACKNOWLEDGMENTS**

I would like to express my grateful thanks to Dr. Han Bao. This thesis was prepared under his advisorship. His consistent advice, encouragement, and confidence were essential to the completion of this thesis and are highly appreciated. I also wish to extend my thanks to my committee members for their support and encouragement during the preparation of the thesis.

## TABLE OF CONTENTS

LIST OF TABLES .....	x
LIST OF FIGURES .....	xi
CHAPTER .....	PAGE
I. INTRODUCTION.....	1
1.1 RESEARCH MOTIVATION .....	1
1.2 RESEARCH OBJECTIVES .....	2
1.3 PROBLEM STATEMENT .....	2
1.4 READER'S GUIDE .....	3
II. O2DBMSs: TOWARD A COLLABORATIVE ENGINEERING ENVIRONMENT .....	4
2.1 INTRODUCTION.....	4
2.2 DATA AND DATABASE MANAGEMENT SYSTEMS CONCEPTS .....	4
2.2.1. <i>Historical Background</i> .....	4
2.2.2 <i>Traditional File Management Systems</i> .....	5
2.2.3 <i>Definitions</i> .....	6
2.2.4 <i>Data Models</i> .....	9
2.2.5 <i>Database Approach Characteristics</i> .....	9
2.2.6 <i>Classification of DBMSs</i> .....	10
2.2.8 <i>Data Management Control Topics</i> .....	11
2.3 WHY DATABASES –OR WHY NOT .....	12
2.3.1 <i>Advantages of DBMSs</i> .....	12
2.3.2 <i>Disadvantages of DBMSs</i> .....	13
2.4 CONVENTIONAL DATA MODELS AND SYSTEMS.....	13
2.4.1 <i>The Hierarchical model</i> .....	13
2.4.2 <i>The Network Model</i> .....	14
2.4.3 <i>The Relational Model</i> .....	15
2.4.4 <i>The Pseudo-Relational Model</i> .....	17
2.5 SHORTCOMINGS OF THE CONVENTIONAL DATABASE MODELS .....	17
2.6 OBJECT-ORIENTED DATABASE MANAGEMENT SYSTEMS (O2DBMSs) .....	18
2.6.1 <i>Why O2DB</i> .....	19
2.6.2 <i>What Is An O2DB?</i> .....	19
2.6.3 <i>Approaches To Object-Oriented Databases</i> .....	22
2.6.4 <i>Object-Oriented Database in Engineering</i> .....	23
2.6.5 <i>Why O2DBMSs in Engineering Design and Manufacturing?</i> .....	25
2.6.6 <i>O2DBMS as an Integration Environment for Engineering Applications</i> .....	26
2.6.7 <i>O2DBMSs in Engineering, a Literature Review.</i> .....	29
2.6.8 <i>Shortcomings of O2DBMSs</i> .....	33
2.7 COMPARISON OF DIFFERENT DATA MODELS AND SYSTEMS .....	34
2.7.1 <i>Comparison of Concepts</i> .....	34
2.7.2 <i>Performance Characteristics</i> .....	35
2.8 THE FUTURE.....	36

III. THE INTEGRATION FRAMEWORK (AN O2DBMS APPLICATION FOR INTEGRATED ENGINEERING -COMPUTATION ENVIRONMENT) .....	37
3.1 INTRODUCTION.....	37
3.2 THE PROPOSED FRAMEWORK STRUCTURE .....	38
3.3 INNERCIRCLE2000™ .....	39
3.3.1 <i>Survey of InnerCircle2000™</i> .....	41
3.4 ACTIVE X AND OLE: APPLICATION INTEGRATION.....	45
3.4.1 <i>OLE</i> .....	45
3.4.2 <i>Linking Vs. Embedding</i> .....	46
3.4.3 <i>COM</i> .....	47
3.4.4 <i>Active X</i> .....	47
3.4.5 <i>ActiveX Controls</i> .....	48
3.4.6 <i>OLE for Design and Modeling</i> .....	48
3.5 N-TIER THIN-CLIENT SERVER ARCHITECTURE .....	49
3.6 UNIFIED MODELING LANGUAGE .....	50
3.6.1 <i>Development Project Artifacts</i> .....	50
3.6.2 <i>Goals of the UML</i> .....	51
3.7 OPTIMIZATION AND DECISION MAKING .....	53
3.7.1 <i>Mathematical Programming (Optimization)</i> .....	54
3.7.2 <i>Linear Programming</i> .....	56
3.8 SPREADSHEETS.....	57
3.8.1 <i>Spreadsheets in Engineering</i> .....	57
3.8.2 <i>Microsoft Excel™</i> .....	58
3.8.3 <i>"What's Best! 4"</i> .....	58
3.9 PROJECT MANAGEMENT.....	59
3.9.1 <i>Microsoft Project™</i> .....	62
3.9.2 <i>Project Management in Manufacturing</i> .....	62
3.10 PROJECT CRASHING AND TIME-COST TRADE-OFF .....	65
3.10.1 <i>Project Cost Models</i> .....	65
3.10.2 <i>Enumerative Method for Project Crashing</i> .....	66
3.10.3 <i>Project Crashing with Linear Programming</i> .....	67
3.10.4 <i>Time-Cost Trade-Off Problem in Literature</i> .....	67
IV. APPLICATION (CASE STUDY / DEMONSTRATION). ....	71
4.1 INTRODUCTION.....	71
4.2 PROBLEM STATEMENT .....	71
4.3 CASE STUDY I .....	72
4.3.1 <i>Project Network</i> .....	72
4.3.2 <i>Cost Model Assumptions</i> .....	72
4.3.3 <i>Problem Formulation</i> .....	73
4.3.4 <i>Application</i> .....	74
4.4 CASE STUDY II.....	75
4.4.1 <i>Project Network</i> .....	75
4.4.2 <i>Cost Model Assumptions</i> .....	76
4.4.3 <i>Problem Formulation</i> .....	76
4.4.4 <i>Application</i> .....	77
4.5 DATA MANIPULATION.....	83
4.6 SUMMARY .....	84
REFERENCES .....	86



APPENDIX .....	105
VITA .....	109

## LIST OF TABLES

TABLE	PAGE
1. COMPARATIVE TERMINOLOGY OF DIFFERENT DATA MODELS.....	35
2. OPTIMIZATION PROBLEMS CLASSIFICATION .....	56

.

## LIST OF FIGURES

FIGURE	PAGE
1. USE OF DATABASE MANAGEMENT SYSTEM IN MANUFACTURING .....	1
2. THE EVOLUTION OF DATABASES.....	5
3. MACHINE FILE .....	6
4. RELATIONSHIPS.....	7
5. A SIMPLIFIED DATABASE SYSTEM ENVIRONMENT .....	8
6. CENTRALIZED FILE MANAGEMENT IN A DATABASE MANAGEMENT SYSTEM .....	10
7. MACHINE SHOP HIERARCHY .....	14
8. OBJECT-ORIENTED DATABASE.....	20
9. NO INTEGRATION .....	26
10. INTEGRATION LEVEL 0: ISOLATION.....	27
11. INTEGRATION LEVEL 1. CONVERTERS.....	28
12. INTEGRATION LEVEL 2. NEUTRAL FILE FORMAT .....	28
13. INTEGRATION LEVEL 3. A CENTRALIZED DATABASE .....	29
14. INTEGRATION LEVEL 4. INTEGRATION OF STAND ALONE COMPONENTS.....	29
15. THE PRESENTED FRAMEWORK .....	38
16. EVOLUTION OF ENTERPRISE DECISION MAKING .....	40
17. INNERCIRCLE2000™, N-TIER THIN CLIENT ARCHITECTURE.....	42
18. TEAM INTEGRATION .....	43
19. FUNCTIONAL INTEGRATION (A) .....	43
20. FUNCTIONAL INTEGRATION (B) .....	44
21. APPLICATION INTEGRATION .....	44
22. LINKING VS. EMBEDDING .....	46
23. TYPES OF MINIMUM FOR UNCONSTRAINED OPTIMIZATION PROBLEMS (SCHEMATIC) .....	55
24. TYPES OF MINIMUM FOR CONSTRAINED OPTIMIZATION PROBLEMS (SCHEMATIC).....	55
25. PROJECT MANAGEMENT AS A MEANS TO INTEGRATE VARIOUS MANUFACTURING ACTIVITIES.....	63
26. DIFFERENT COST MODELS .....	66
27. PROJECT NETWORK (CASE I).....	72
28. COST MODEL (CASE I).....	73
30. TIME-COST TRADE-OFF WORKSHEET .....	74
31. TIME-COST TRADE-OFF CURVE. ....	75
32. PROJECT NETWORK (CASE II).....	75
33. COST MODEL (CASE II) .....	76
34. PROJECT DESCRIPTION WORKSHEET.....	78
35. PROJECT DATA ENTRY WORKSHEET.....	78

FIGURE	PAGE
36. MODEL WORKSHEET .....	79
37. TIME-COST TRADE-OFF CURVE .....	80
38. CRASHED PROJECT DATA WORKSHEET .....	81
39. WHAT'S BEST! 4 REPORT .....	82
40. DATA MANIPULATION THROUGH THE FRAMEWORK .....	83

# CHAPTER I

## INTRODUCTION

### 1.1 Research Motivation

Computers already have a pervasive influence on modern manufacturing engineering. This ranges from supporting the design process through production planning and control, and on the toe control of manufacturing equipment and distribution system. As computing becomes more widely available, each discipline will come up with its unique set of applications. The difficult goal of computer integrated manufacturing requires the effective management of information. This difficulty arises from the fact that manufacturing, the whole process, combines several disciplines: design, production planning, project management, optimization, etc. An integrated approach to manage manufacturing information requires compatible representation and manipulation of information. A major emphasis in research today is being directed towards developing object-oriented databases particularly suited to manufacturing applications. Figure 1 shows the variety of data and applications in the manufacturing environment, and how they may be supported once data is captured.

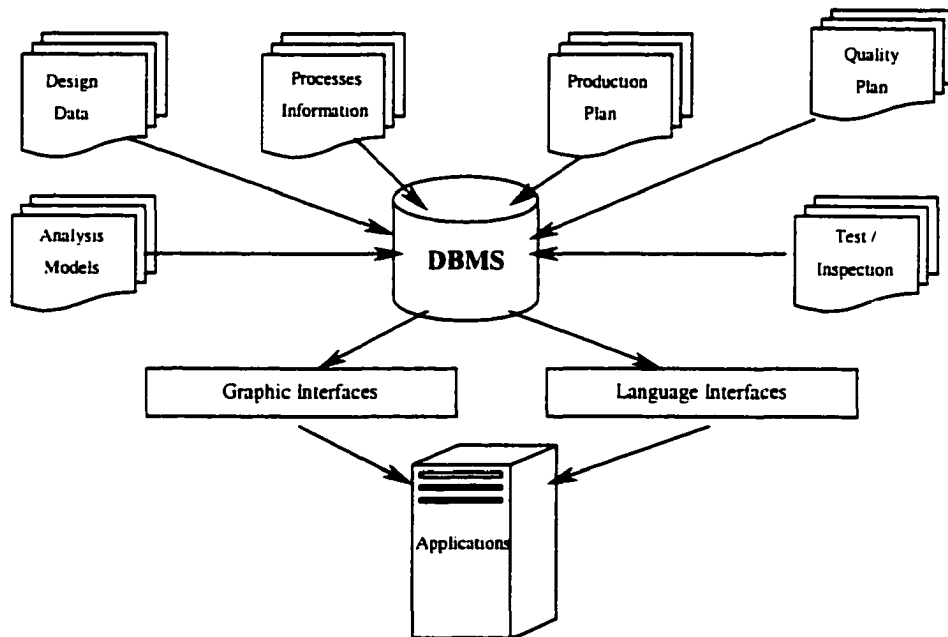


Figure 1. Use of Database Management System in Manufacturing

---

The format of this thesis is based on "The American Society of Mechanical Engineers Journal."

## 1.2 Research Objectives

The specific objectives of this thesis are as follows:

- To investigate the expected future role of object-oriented databases management systems as an applications / activities integration tool in manufacturing;
- To investigate the role of project management in manufacturing;
- To introduce an integrated engineering-computation framework that combines the capabilities of several commercial software with a state-of-the-art knowledge medium (based on an object-oriented database); and
- To demonstrate how the “Time-Cost Trade-Off Problem” in project networks could be manipulated interactively through the framework.

The fulfillment of these objectives will provide some guidance for the role that such an integration framework can play in manufacturing development, in the sense that several disciplines (project management, modeling, and optimization) can work together to adopt a successful manufacturing practice.

## 1.3 Problem Statement

The disciplines basic to making progress in manufacturing belong not only to mechanical engineering, but also to industrial engineering, mathematics, management science, and computer science. These separate disciplines are individually supported by their research, methods, and software. There is a lack of focused attention on how to integrate knowledge from many disciplines into knowledge that furthers manufacturing goals. Moreover, at the same time that this lack of strategy is apparent, all dimensions of manufacturing (e.g. products, processes, markets) are becoming more complex and diverse. Complex new products based on information content and their accompanying information-dominated design and manufacturing methods already require us to deal with an entirely new scale of complexity. The integration among various elements of manufacturing on one side, and among manufacturing and other disciplines on the other side, can be achieved through this framework. Providing ways to facilitate and manage the complexity of these information and computation intensive activities plays an important role in supporting and even enabling the complex practice of manufacturing.

## 1.4 Reader's Guide

This thesis is organized as follows:

- Chapter (II) provides background on object-oriented database management systems by reviewing the history and current state of database management systems in manufacturing, and develops the main research goal by examining expected and desired development direction for the future of object-oriented databases as an integration tool of manufacturing applications / activities.
- Chapter (III) presents a framework for integrating mathematical modeling, optimization, and project management software packages with a state-of-the-art knowledge medium (based on an object-oriented database).
- Chapter (IV) illustrates how to handle the Time-Cost Trade-Off problem in project networks interactively through the presented framework and provides a summary of the thesis.

## **CHAPTER II**

### **O2DBMSS: TOWARD A COLLABORATIVE ENGINEERING ENVIRONMENT**

#### **2.1 Introduction**

Our world, that is, our conception of the world, is continuously becoming more complex. In any type of organization, considerable resources and activities are dedicated to the gathering, filing, processing, and exchange of data based on well-established procedures in order to achieve specific goals.

In recent years, due to the marked changes in computer technology, databases are having a major impact on all managerial and administrative areas. Furthermore, as a result of hardware innovations, new data intensive applications have emerged. For example, engineering applications such as CAD/CAM, CASE, and CIM. All require extensive databases and highly efficient means of retrieving the relevant information.

In this chapter, a background on object-oriented databases is being provided. An emphasis is being given to the role of O2DBMSs as an integration tool in engineering (manufacturing).

#### **2.2 Data and Database Management Systems Concepts**

##### **2.2.1. Historical Background**

Object-oriented databases constitute an important step in the evolution of database technologies. During the 1960's, Hierarchical and Network database management systems appeared. Both data models were primarily navigational and suffered from the lack of a strong theoretical foundation. Moreover, they did not support the notion of physical and logical data independence. Trying to overcome these shortcomings, the Relational Data Model was introduced in the early 1970's, which became increasingly popular in the 1980's and 1990's. An alternative database modeling technique the Semantic Data Model, was proposed after its forerunner, the Entity-Relationship Model. In the mid-1980's, object-oriented databases started to appear. These databases integrate both object-oriented concepts and conventional database capabilities. Figure 2 traces the evolution of databases.



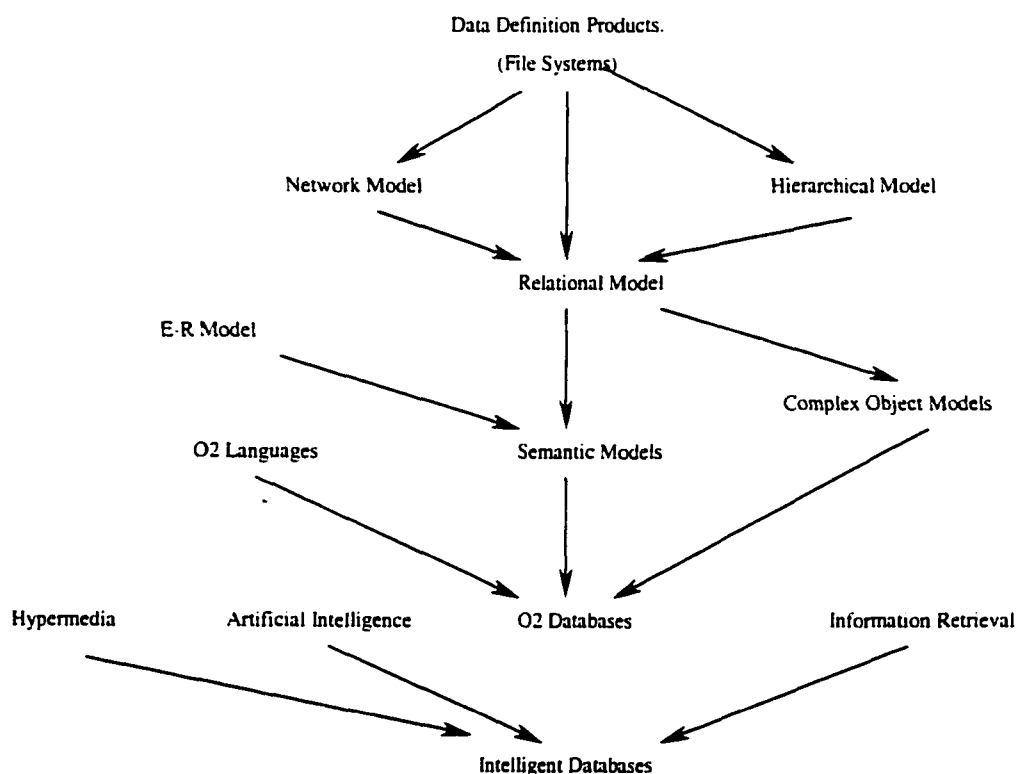


Figure 2. The Evolution of Databases

### 2.2.2 Traditional File Management Systems

The forerunners of DBMSs were the file management systems. When designing such a system, one can choose any of several approaches. These approaches vary in the way data records are accessed. The simplest approach is using a “sequential file structure” where a file is to be read / written one record a time, and each record follows its predecessor physically on the storage device. On the other hand, we can find a more complicated approach, that is, “indexed file structure”, where the system here maintains a key for each record in an index component of the file, which is linked to the corresponding data component. Generally, in all traditional file management systems, we may notice that they are: 1) easy to create and simple to use, and 2) require minimal overhead to access and use. But at the same time: 1) any change in the structure or the content requires simultaneous changes to all programs, 2) relationships across files must

be imbedded within application logic, and 3) these systems encourage the proliferation of redundant data; that is, the same fields exist at the same time or offset within each record.

### 2.2.3 Definitions

#### Data versus Information

A single piece of data represents a single fact about something in which we are interested. Usually we have many facts describing something of our interest, and it certainly seems reasonable to collect all of these facts and hold them together. On the other hand, "Information is what you get when you distill data. A collection of row data facts does not help anyone to make a decision until it is reduced to high-level abstraction." (Celko, 1994) So, when we are concerned about databases, we have to understand that having data in the database is not the same as knowing what to do with it.

#### Linear Files

Figure 3 represents the data of a machine shop. The collection of facts about a particular machine (one row of the table) is called a record. Each kind of fact (each column) is called a field. And the collection of operation facts for all of the machines the entire table, is called a file. The file in figure (3) is usually called a simple, or linear, file. In general, we may say that each thing we are interested in keeping track of is called an entity. The center lathe is an entity. A collection of entities of the same type is called an entity set. An attribute is a property or characteristic of an entity. Relating these terms back to files, we can see that a record describes a particular entity (at least in terms of one or several applications). A file describes an entire entity set. A given attribute is represented as a field value.

Machine #	Machine Name	Standard Rate	.....
1001	Center Lathe	100 /hr	
1002	Center Lathe	120 /hr	
1003	Shaper	50 /hr	

Figure 3. Machine File

## Relationships

In addition to maintaining facts about a particular entity, another very important kind of information arises, that is, the collection of ways that different entities relate to each other. Such interactions are described as associations and relationships. Figure 4 demonstrates different relationships.

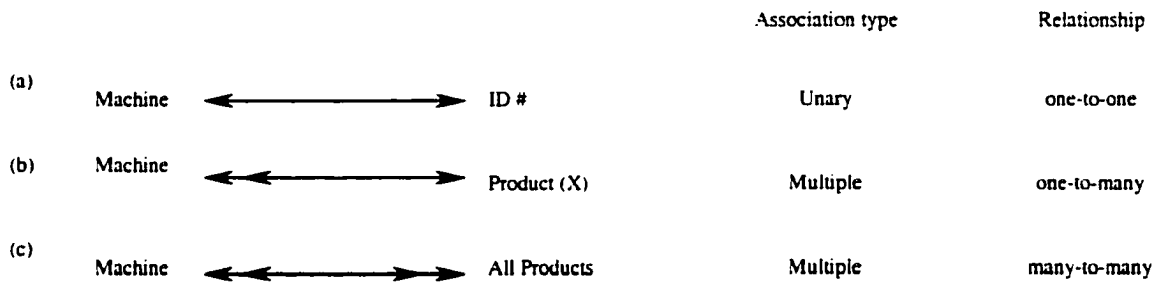


Figure 4. Relationships

## Database

The term “database” denotes a collection of related data stored as a set of permanent or so-called standing files in a data processing system. These files either remain unchanged over time, or they are updated by transactions that arrive in batches or one by one into the system.

The preceding definition of database is quite general. In fact, the common use of database is usually more restricted. In this context, we may say that, “A database is a well organized collection of data. One should be able to process, update, and make additions to the contents of a database in a simple and flexible way. It should also be easy to make different kinds of unplanned as well as planned retrievals of data from the database” (Sundgrer, 1985).

Although this definition cannot exhaust the database concept, it contains some important key words: 1) well organized, 2) simple and flexible, 3) process, update, add, and retrieve, and 4) unplanned as well as planned

From a functional point of view, there are two main distinguished purposes of a database: (1) To satisfy the information needs of different database users; information consumption oriented database, or user database, (2) To facilitate the management of data within a data processing system; an information production oriented database, or a working database. Sometimes one and the same database has the double function of being a user database and a working database, either at the same time, or during different phases.

## Database Management Systems and Database Languages

According to the database definition a database must be well organized, flexible, and easy to use. In order to fulfill these requirements, a computerized database must be managed by a piece of software- a system of programs that makes it possible to add, retrieve, delete, and update data in a simple and efficient way. Such a program system is called a database management system (DBMS). Thus, a DBMS, as shown in Fig. 5 serves as the interface between data files and the people who seek the data in these files (users).

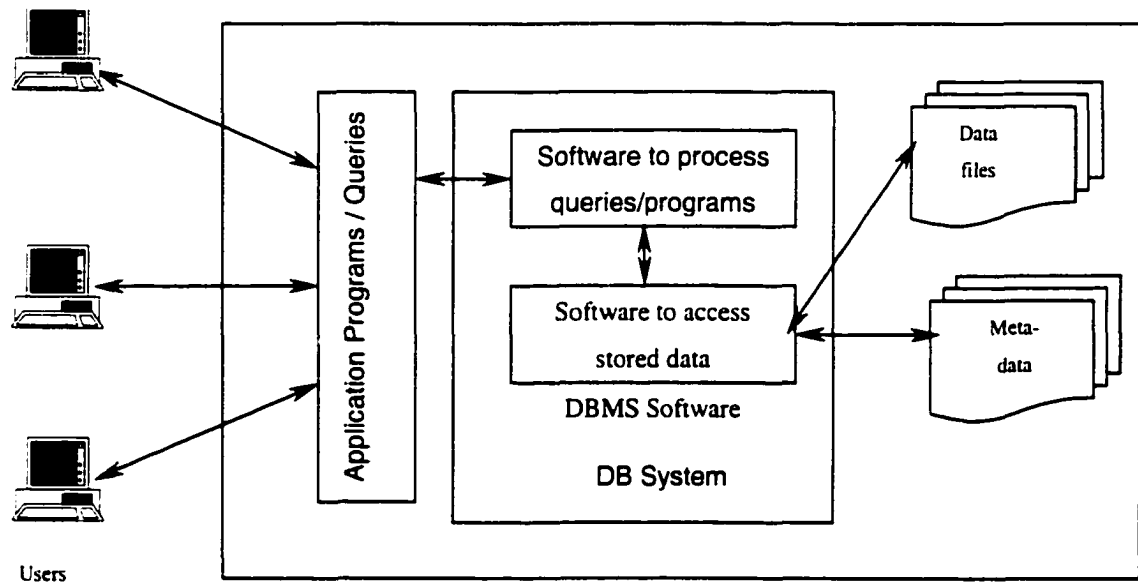


Figure 5. A Simplified Database System Environment

### Metadata

“Metadata is “data about data”. Thus metadata represents:

1. information about the information that is represented by certain stored data, and
2. information about how the data are physically stored” (Sundgrer, 1985).

In a database system the metadata base is an integrated part of the database, that is, “a database within the database”, and it contains the metadata needed by the database end-users, the database administrator, and the database management system.

### 2.2.4 Data Models

Underlying the structure of a DB is a data model, a collection of conceptual tools for describing the real world entities to be modeled in the DB and the relationships among these entities (Silberschatz et. al., 1996). Or, as Sabin stated, a model is "something that can be interpreted as an analog for some entity of interest" (Sabin, 1988). Data models can be classified into four types:

1. Physical data models, used to describe data at the lowest level.
2. Object-based logical models, which make use of the concepts of entities (objects) and relationships between them. The two most widely-used representatives of these models are the entity-relationship model (E-R model) and the object-oriented model (O2 model).
3. Record-based logical models, where, the DB is structured in fixed format records of several types. The relational database is a direct application of such data model.
4. Object-relational data models, which are hybrids of the O2 and relational data models.

On the other hand, several conceptual data models exist, such as EER (Extended-Entity Relationship), and SOM (Semantic Object Model).

### 2.2.5 Database Approach Characteristics

Using a database approach in a data processing system involves several characteristics that distinguish this approach from the traditional approaches. These characteristics are:

- A. **Centralized File Management.** In a database system the file management is concentrated on one subsystem, the database management (sub)system, as shown in Fig. 6. The operations of this subsystem are controlled by programs that are part of the database management software. All other subsystems are communicating with the database management subsystem via a database language.
- B. **Self-Describing Nature of the Database System.** A fundamental characteristic of the database approach is that the system contains not only the database itself but also a complete definition or description of the database stored in the system catalog, which contains information such as the structure of each file, the type and storage format of each data item, etc.. This information stored in the catalog is what we call the meta-data.
- C. **Data Independence.** In DBMSs, the application programs in a system and the data that are processed by the application programs are in a certain sense independent of each other. It should be possible to change the data in various ways without changing the application programs. This property is called the program-data independence
- D. **System Integration.** In a database system, the data required by all application programs is stored in the same database allowing different applications to use the same part of the data, which in turn reduces data redundancy.

- E. **Support of Multiple Views of the Data.** A database typically has many users, each of whom may require a different perspective or view of the database. A multi-user DBMS whose users have a variety of applications must provide facilities for defining multiple views.
- F. **Concurrent Sharing of Data.** A multi-user DBMS must allow multiple users to access the database at the same time within a concurrency-controlled framework.

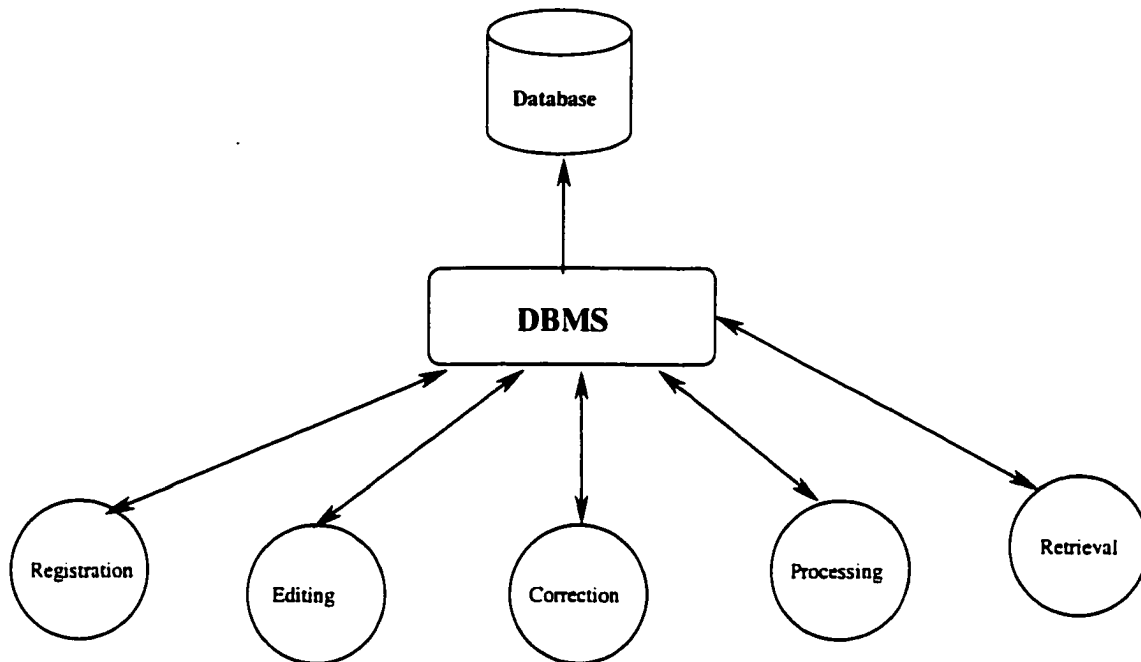


Figure 6. Centralized File Management in a Database Management System

### 2.2.6 Classification of DBMSs

Several criteria may be used to classify DBMSs. The main one is the data model on which the DBMS is based. In this context, DBMSs are categorized as relational, hierarchical, network, object-oriented, and others. Another criterion is the number of users supported by the system- single-user or multi-user system. A third criterion is the number of sites over which the database is distributed- centralized or

distributed database. Finally, a DBMS can be general-purpose, that is, used in several database systems and in combination with different databases, or special-purpose, tailor-made for certain application systems.

### 2.2.8 Data Management Control Topics

There are several concerns about the way we manage data, regardless the file structures used, that must be considered in any data processing environment. These control topics are inherent in storing, managing, and accessing the data. The areas are the following.

- A. **Security.** There are several aspects of security which are specifically data related, however, and thus are of interest here. The two key points involved are preventing people from seeing data that they are not entitled to see and preventing them from changing data that they are not entitled to change. It is certainly possible to design and build systems that can protect the data in independent linear files. The problem is that unless a unified system was agreed upon, every programmer is left to his or her own devices, either omitting such security considerations or reinventing the wheel for each application.
- B. **Backup and Recovery.** Clearly, it is always advisable to have more than one copy of every data file. Furthermore, the copies must be kept in a way that prevent a catastrophe from destroying all copies of the data. The process of copying a file, which must be done on a periodic basis to reflect changes in the data, is known as backup, or backing-up the file. The process of using the file copy, plus other data, to correct a data damage is known as recovery. Here again, as with the security case, it is certainly feasible to have backup and recovery systems for independent linear file based data processing. But, again, this means that everyone in the environment must use the same system, which is often contrary to the independent nature of the application in such environments.
- C. **Concurrency.** In a multi-user application, a particular problem can surface involving changing field values. When two users try to update the same record simultaneously (or as simultaneously as a multi-programmed system allows), they have a rather nasty way of interfering with each other so that one of those two updates may wind up being ignored. A common solution to this problem involves a technique called "lockout". Unfortunately, as often happens, the use of this technique itself causes other problems that did not previously exist and that have to be handled. Once again, there is an important concern: who and what software system should be responsible for implementing these protective devices?
- D. **Auditability.** As more and more of an enterprise's data is stored and processed in automated data processing systems, it becomes an increasingly important focus of the auditing function. There is an endless list of reasons for management, accountants, and others to ask who had access to or made changes to what data, and when-not to mention what the new and old values of the data were. The primary tool of the auditors is the audit trail: a permanent record of the nature of the changes made to the data. But here, as in the other three situations discussed who or what should be responsible for the audit function?

## 2.3 Why Databases –or Why Not

When one argues for a database system of a system or group of systems, it is customary to point out the favorable influence of database technique on such things as: 1) The availability, up-to-date-ness, and consistency of the information. 2) The flexibility and viability of the systems, and 3) The costs for collection, storage, and updating of data. On the other hand the very efficiency of database technique implies risks within areas such as: 1) Confidentiality and privacy, 2) Data quality, and 3) Data integrity and security. These risks can be eliminated by adequate protection measures, which will of course imply certain costs. One should also calculate with a certain increase of costs for certain tailor-made outputs, and certain common resources, such as the database administration function.

The efficiency of the database is to a certain degree based on coordination and standardization. Measures of this nature will not automatically be applauded by everybody involved. Hence, if one wants to introduce database, one should be prepared to explain, why a database system may be very efficient from an overall point of view, although certain applications may become a little more expensive to run than they would be in a conventional system that has been (sub)optimized for its specific and more narrowly defined purposes.

### 2.3.1 Advantages of DBMSs

- 1- Data independence. This means that the application program doesn't know (or care) about any of the physical attributes of the data sets involved, or how the individual data records are arranged within any given data set. That's the job of the DBMS.
- 2- Support complex data relationships. This complexity greatly enhances the ability of a designer to put a piece of data in one place, "where it belongs," and provide a path to that data whenever needed.
- 3- Sophisticated data security features.
- 4- Data base recovery. Backup/recovery capabilities often distinguish between a true DBMS and a software package that only claims that name. A true DBMS has a logging or recording mechanism that captures information on changes to data records within a database under its control. If, at a later time, a database is lost as a result of a hardware failure, the DBMS would have a utility capable of rebuilding it by using a backup copy of the data and the log of changes as input.
- 5- Advanced capabilities such as sophisticated on-line communications systems and ad hoc reporting capabilities.



### 2.3.2 Disadvantages of DBMSs

- 1- Additional overhead is required to access data. To take advantage of a DBMS's capabilities, you have to plan. More research, knowledge, and time are required.
- 2- Application programmers require more training to code efficient programs that will run under a DBMS.
- 3- Rushed data analysis can result in an incomplete or incorrect data structure. A later change in structure can be costly in terms of conversion and testing of existing programs.
- 4- Data must be considered a corporate resource. Companies are recognizing that there will be an increased need to share data across applications.
- 5- Data structures need to be fully and consistently defined in data dictionary system.

## 2.4 Conventional Data Models And Systems

During the past three decades, database technology has undergone five generations of evolution. We may link these transitions from one generation to the next, mainly, with the complexity increasing of database applications. At the beginning, it was the "file systems", such as ISAM, and VSAM. The second generation was the Hierarchical databases, such as IMS. The third generation was the network databases (CODASYL), such as IDS, and IDMS. The lack of data independence and the tedious navigational access to the database gave rise to the fourth generation, relational databases. And now comes the recent generation, object-oriented databases, which is characterized by a richer set of database facilities necessary to satisfy the requirements of the emerging complicated applications.

In this section we will present, in brief, four fundamental database structure models, namely; the Hierarchical model, the Network model, the Relational model, and the Pseudo-Relational model.

### 2.4.1 The Hierarchical Model

The first approach to DBMS that we are going to consider is the hierarchical approach. IBM's Information Management System (IMS), one of the oldest (became commercially available in 1969) and most widely used DBMSs, is based upon that approach. Another popular hierarchical DBMS product is System2000 from SAS Institute, which was originally marketed by MRI and then later by Intel Corporation. The hierarchical data model represents the structure of the persistent database as a collection of trees.

Now, let us briefly describe how data can be stored in the hierarchical form of the IMS. Figure 7 represents a machine shop hierarchy. The figure shows that each node of this tree represents a different

record type, called a segment in IMS. Each segment consists of a number of fields that describes the segment. Every branch in the tree represents a one-to-many relationship. In fact, the hierarchical structures has some limitations, both in the degree of data integration that it permits and in the fact that it cannot directly handle many-to-many relationships. In order to overcome these shortages, the concepts of "logical relationships" and "bi-directional logical relationships" were introduced to the facility in IMS. See (Gillenson, 1990) for details.

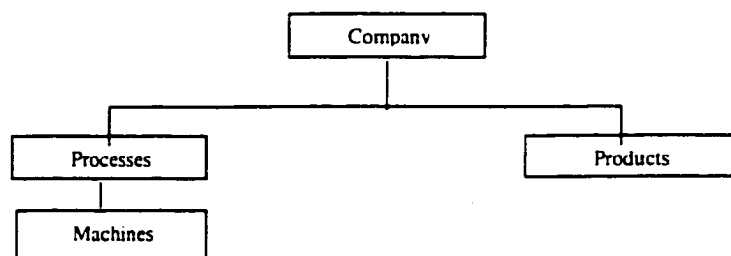


Figure 7. Machine Shop Hierarchy

#### 2.4.2 The Network Model

Another approach to DBMSs is based on structures called networks. Over the last two decades many DBMSs have been developed that are implementation, to a greater or lesser extent, of the network-oriented DBMSs specifications developed by the Conference on Data Systems Languages (CODASYL) and its data Base Task Group (DBTG) and Data Description Language Committee (DDLC). The CODASYL work on uniform specifications for a DBMS began in the late 1960s. Reports or set of CODASYL specifications have been produced from time to time. The different commercially available "CODASYL DBMSs" are based on some percentage of the specifications in one or more of the CODASYL reports.

Understanding the differences between trees and more general networks is very important. There are two ways of distinguishing a tree from a more general network: 1) a tree is a network that has no cycles. And 2) in a tree, a node may not have more than one parent. The network model is more general than the hierarchical model. Although the only type of relationship supported by the network model is also one-to-many, it is possible for the same record type to be a child, or a member, with multiple parent, or owner, record type.

Both the hierarchical and network data models were primarily navigational: A user would start from a parent or owner record and navigate through the members of a relationship through get next, get first, or get last constructs. The owner-member relationship (for the network model) or parent-child relationship (for the hierarchical model) were explicitly stored in the database records. More specifically, the network and hierarchical database implementations did not have physical data independence. This meant that the user's view of the navigational and hierarchical databases reflected the way the data was organized, stored, and accessed from the underlying physical storage media. In some cases, the user or the database management system administrator (DBA) needed to specify details of record placement, storage areas, record ordering, record locations, and so on. Besides the specification hassle, this approach severely limited the extensibility, maintainability, reusability, and portability of the database management system applications that were developed from these models.

In order to provide more flexibility in organizing large databases and to alleviate some of the problems of the earlier models, Dr. E. E Codd (1970) in the early 1970s introduced the relational data model.

#### 2.4.3 The Relational Model

"In 1970, Dr. Edgar F. Codd of IBM published a paper entitled "A Relational Model of Data for Large Shared Data Banks." This paper marked the beginning of the field of relational database". (Gillenson, 1990) The relational approach to database received a great deal of publicity through the years, but it was not until the early 1980's when it has commercially viable relational DBMSs been available. Today, the applications of RDBMSs cover most of the financial and engineering systems.

To begin with, let's consider the data structure used in relational database. The data appears to be stored in what we have been referring to as simple linear files. Those simple linear files are called relations or tables. Thus, "a relation is a two-dimensional array or table of data containing descriptive information about an entity." (Hogan, 1990) We will assume that no two rows, records, or tuples of a relation are identical. Technically, the columns, fields, or attributes of a relation can be arranged in any order without affecting the meaning of the data. The set of all possible values that a particular attribute may have is called "Domain". A relation always has a unique key, a field or a group of fields whose values are unique throughout all of the tuples of the relation. The number of fields involved in the key is always the minimum number of fields that provide the uniqueness quality. If a relation has more than one unique key, then they are each called candidate keys, and the one that is chosen as "the key of the relation" is called the primary key. If in a collection of relations that make up a relational database a field or a group of fields serves as the primary key of one relation and also appears in another relation, then it is called a foreign key in that other relation. Thus, We may summarize the properties of relations as: 1) no duplicate rows exist, 2) the order of rows or columns is insignificant, and 3) columns are all elemental.

A relational DB is simply a collection of relations. In term of the way we think about the data, the relations are quite independent, physically, as opposed to the tightly pointer-connected structures in the hierarchical- and network-based approaches.

### Extracting Data from a Relation

An inherent feature of a RDBMS is the capability to accept high-level data retrieval commands aimed at relations and to process them, even to the extent of matching related records in different files (integrating data). Since a relation fundamentally a rectangular arrangement of data values, it would seem to make sense to want to approach data retrieval horizontally (retrieving one or more records), vertically (retrieving one or more attributes for all of its rows), or in a combination of the two. The commands controlling these operations are extracted from a database formalism called "relational algebra". Two operators called "select" and "project" are capable of the both kinds of horizontal and vertical manipulations. The result of any relational operation will always be a relation. Another important operator called "join", used in parallel with "select" and "project", helps solving the problem of "multiple relation searches".

Representing a many-to-many relationships in a RDBMS requires the creation of an additional relation. That relation will consist of fields which serve one or the other of two purposes. First, it will have as its key the combined unique identifiers of the two entities in the many-to-many relationship. In addition, it will have fields that constitute intersection data, that is, attributes or fields that describe not one entity or the other alone, but the relationship between the two entities.

### Operating Characteristics

In abstract, it is relatively easy to express and describe database operations against a relational data model, since the user/programmer does not have to think about which access paths are favored by the database management system. Instead, the database management system automatically select an efficient access path for every database operation that is expressed by the user in the way that the most "natural" or "logical" to him.

### Why Relational DBMSs Have Only Recently Become Commercially Viable

Recall the "join" operator, the thought of comparing every tuple of one relation included in a RDBMS with every tuple of the other to accomplish the join process, data retrieval, might take a very long time considering large DBs.

Modern, sophisticated RDBMSs rely either on special hardware or on clever software techniques to speed up the join process. They also take advantage of the increase in speed of computer processors that has developed continually over the years.

#### 2.4.4 The Pseudo-Relational Model

Many of those who work in the field of DB consider the hierarchical, network, and relational models of DB system design to be the only ones in existence. We will demonstrate a fourth approach to DB, which has been labeled in the literature as the “pseudo-relational”, “flat-file integrated”, “semi-relational”, or “relational-like” approach. The pseudo-relational approach is really a hybrid, composed of several of the features of the DB approaches that we have already discussed. The fundamental data structure is that of simple linear files, a feature it shares with the relational approach. But as a general rule, pseudo-relational DB systems are not geared towards performing execution-time joins the way relational systems are. Like the hierarchical and network approaches, they require that some of the physical linkage between related records of different files be constructed and stored with the DB in advance of any on-line query. A new concept to be introduced is the “link file”, that indicates which records of a certain two files are related to each other. So in the pseudo-relational approach the data remains in independent, simple linear files, but additional link files are created – normally well in advance of any queries being posed to the data – that indicate how the records of the different files relate to each other in the sense of the join concept.

The advantages and disadvantages of the pseudo-relational approach parallel those aspects of the other DB approaches from which it is derived. The pseudo-relational approach has what many would consider to be the relative simplicity of simple linear files. It also has the performance boost of joins being done in advance, with the results of those joins being stored for future use by queries. On the other hand, it has the disadvantage of not having the true relational on-the-fly join capability. Also, at the time that data values are changed in a file, the system must take the time to update any affected indexes. Another drawback is that, as with the hierarchical and network approaches, decisions must be made at the time the DB is designed as to which files must be prejoined, and the results stored in link files.

### 2.5 Shortcomings of the Conventional Database Models

As discussed by (Kim, 1990; Elmasri and Navathe, 1994), several shortcomings of the conventional databases were the reasons for O2DB to be developed. They are:

- 1- **Lack of Data Abstraction.** Conventional data models, especially the relational model, is too simple for modeling complex nested entities found in many current advanced applications.

- 2- **Only Built in Data Types.** Conventional database systems support only a fixed collection of atomic data types, such as integer, string, etc., that are “hard-wired” into the system. They don’t support application-specific data types and they don’t even allow the storage and retrieval of long structured data (long data) as in multimedia application.
- 3- **Segmentation.** “One of the most severe drawbacks of the relational model is the need to decompose logically coherent application objects over several base relations.” (Kemper and Moerkotte, 1994)
- 4- The performance of conventional Database systems, especially relational, is not compatible with various types of compute-intensive application, such as simulation programs in CAD.
- 5- An impedance-mismatch problem arises when some database languages are embedded in a programming language due to large differences in both data model and data structure.
- 6- Transaction models supported by conventional databases are inappropriate for interactive, cooperative design environments.
- 7- **Identifier Attributes.** Numerous problems arise, in relational databases, with using attribute that model the state of an object as a key to identify tuples.

## 2.6 Object-Oriented Database Management Systems (O2DBMSs)

The field of computer science has been characterized by two new trends during the past few years. One is the high complexity of application environments, and this includes the great diversity in the way information is accessed, manipulated, and processed. The other one is the emergence of the downsized client/server architecture. Large mainframe systems are being replaced by inter networked LANs (Local Area Networks) at a much lower cost- yielding much greater efficiency.

Object-oriented databases combine and integrate these two trends to satisfy the computational needs of not only advanced database applications, but also general corporate computing as well. In fact Object orientation provides an enabling technology that makes it easy to construct and maintain complex systems, and database servers act as the repository of all concurrently shared information on the network.

Moreover, advanced applications, including Computer Aided Engineering (CAD) and Computer Aided Manufacturing(CAM), and integrated environments need powerful databases. In this respect, the databases management system technology of the 80’s – relational DBMS- fell short of providing the necessary abstraction to act as the repository for such integrated, advanced, and emerging applications. Meanwhile, O2DBMSs, the emerging trend of the 1990’s, integrates the O2 features with the DB capabilities. On one hand, O2 features (abstract data typing, inheritance, and object identity) satisfy the requirements of a new generation of DB applications. And on the other hand, applications of O2 systems are requesting DB capabilities (concurrency, recovery, etc.). These capabilities make O2DBMSs more powerful of handling the mentioned advanced applications.

### 2.6.1 Why O2DB

The main target of Object-oriented databases has been engineering-computer-aided design, computer aided manufacturing, computer aided software engineering, and intelligent offices (office automation). These applications differ from the traditional business or accounting applications in: 1) much larger amount of information stored in the database, 2) more complex connections among these information, and 3) the heterogeneous and complex structures of databases in these applications; there are many object types corresponding to each individual component and composite object, and 4) in typical design applications each design object will undergo refinements and have multiple versions or alternatives. Furthermore, we may notice that most of the manufacturing processes share the same properties (name, machining time, setup time, labor rate...etc.) but with different values. These requirements make O2DB the most suitable DB for such applications.

"Object-oriented databases remove the so-called semantic gap between an application domain and its representation in persistence storage. Since the real world is modeled as closely as possible, the links and relationships among entities in it are represented and manipulated directly. Object-oriented databases achieve their modeling capability through the object-oriented concepts of abstract data types, inheritance, and object-identity." (Khoshafian, 1993).

Object-oriented databases are more complete in the sense that they typically provide the necessary expressive power to perform the computations of an application through the data manipulation language of the object-oriented database management system.

### 2.6.2. What Is An O2DB?

The object-oriented technology provides users with powerful modeling facilities that are based on object-oriented concepts. Before defining the object-oriented concepts, we must first understand what an object is. An object is "a combination of data, of any type, and a program, or a collection of programs, that represents some real world entity." (Kim, 1995) The object nature, characterized by encapsulating data and programs, enables application developers to build large programs from lots of small, prefabricated ones.

Object-oriented databases, as shown in Fig. 8, integrate the object-oriented technology (object orientation) with database functionality. "Through object-oriented constructs users can hide the details of implementation to their modules, share objects referentially, and extend their systems by specializing existing modules. Database functionality is needed to ensure persistence and concurrent sharing of information in applications. Therefore, the framework characterizing object-oriented database may be defined as follows:

Object-Oriented Databases = Object Orientation + Database Capabilities.” (Khoshafian, 1993)

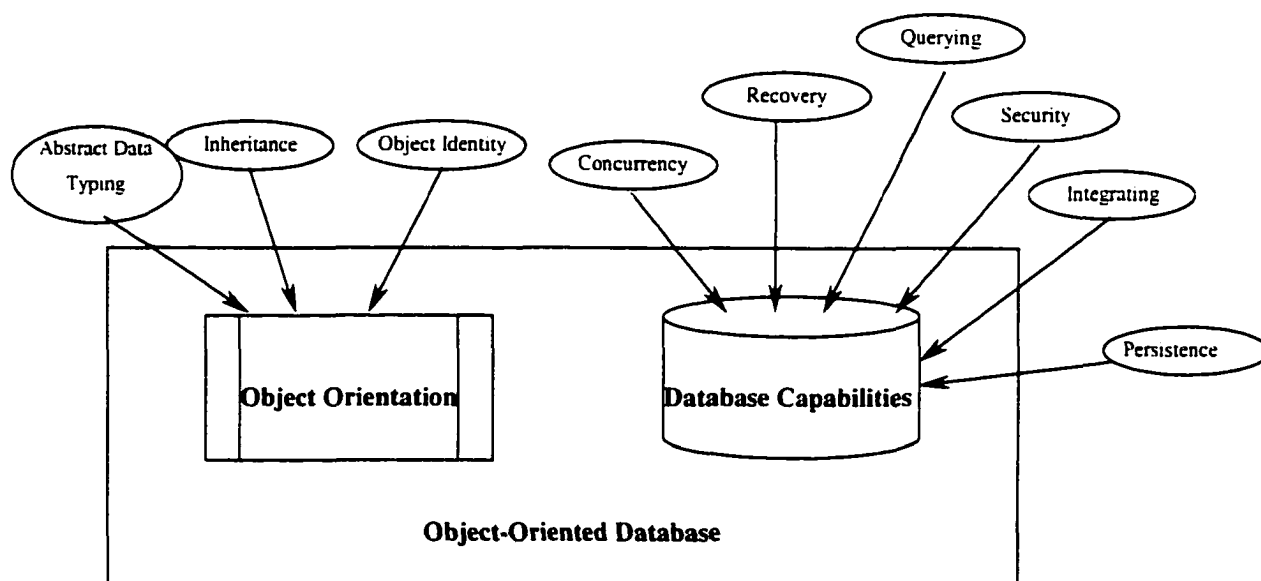


Figure 8. Object-Oriented Database

There have been many definitions of object orientation and object-oriented databases (Atkinson et al., 1992; Brown, 1991; Cattell, 1991; Hughes, 1991; Khoshafian and Abnous, 1990; Kim, 1990; Stonebraker et. al., 1990).

#### Object Orientation

Object orientation concepts are in use today in many areas in computing, including; languages, interfaces, artificial intelligence, operating systems, and databases. In fact, object-oriented concepts are being implemented within the first four areas. But, when it comes to applying object-oriented concepts to databases, things are not as clear. Part of the confusion stems from the fact that there is no single agreed-upon definition of object orientation, proposals and descriptions of object-oriented database management systems have been made to clarify the confusion and define the important features and characteristics of object-oriented databases. (Atkinson et al., 1992; Brown, 1991; Cattell, 1991; Hughes, 1991; Khoshafian and Abnous, 1990; Kim, 1990; Stonebraker et. al., 1990).

Object orientation can be loosely defined as “the software modeling disciplines that make it easy to construct complex systems out of individual components” (Khoshafian, 1993).



The intuitive appeal of object orientation is that it provides concepts and tools with which to model and represent the real world. The advantages of object orientation in programming and data modeling are many. As pointed out by (Ledbetter and Cox, 1985),

.....object (oriented) programming allows a more direct representation of the real-world model in the code. The result is that the normal radical transformation from system requirements (defined in user's terms) to system specification (defined in computer terms) is greatly reduced.

The object-oriented technology promises (and has delivered) significant benefits to each of the following groups of computer users:

1. End Users. Object orientation provides direct representation of the "physical" objects manipulated by end users and a more natural "object-message" paradigm for interacting with objects. In addition, it helps integrate multimedia data types into the computing environment.
2. Application Developers. Object orientation provides tools that are easier to use and help create the most natural abstraction of the user's object space.
3. System Programmers. Object orientation enhances the engineering and configuration management tools, that enables those expert programmers to build complex systems more quickly.

"Object-oriented methods and systems adhere to normal forms and improves integrity between databases and application" (Blaha et. al., 1988).

(Bancilhon, 1996) covers the main concepts of ODBs, object data model and classical database features, and other subjects related to database performance. (Kim, 1990) discusses: 1) The major reason for the concerns and questions about the field of O2DBMSs that have persisted, 2) The issues of standardization and performance, and 3) Several areas of research which not been well explored thus far. (Pancake, 1995) discusses in what areas does object technology offer the most promise, and as OT expands into these areas, what new challenges lie ahead? (Ullman, 1987) sketches the development of the various branches of DB theory, emphasizing on relational databases.

## Overview of Object-Oriented Concepts

The four most fundamental aspects of the object-oriented paradigm are abstract data types, object identity, encapsulation, and inheritance. Each of these concepts contributes to the software engineering and modeling properties of object-oriented systems. Usually, each specific object-oriented language, system, or database emphasizes one or two of these concepts without supporting the others directly.

### i. Abstract Data Types

Data types are used to describe a set of objects with the same representation. Several operations are associated with each data type. Abstract data types extend the notion of a data type by "hiding" the implementation of the user-defined operations associated with the data type. Languages that support

abstract data types provide constructs to define data structures and the operations used to manipulate occurrences ("instances") of the data structures directly. In addition, the instances of the data type are allowed to be manipulated only through a prescribed collection of operations associated with the type. In other words, the internal structure of the object is hidden, and the object is only accessible through a number of predefined operations.

#### ii. Object Identity

"In an object model, every object instance has a unique, unchanging identity called an Object Identity." (Barry, 1996). Thus, an identity is that property of an object that distinguishes each object instance from all others. Object identity organizes the objects of the object instances manipulated by an object-oriented program. "Several important characteristics are associated with these OIDs. First, OIDs are independent of data contained in the object. The internal data values are not used to generate identification. Second, OIDs are generated by the object system. Users or programmers have no control over identification. And finally, OIDs last the lifetime of the object." (Barry, 1996). Object identity is independent of object contents.

#### iii. Encapsulation

This concept refers to including processing or behavior with the object instances defined by the class. In other words, code and data are allowed to be packaged together.

#### iv. Inheritance

Inheritance is a means of defining one class in term of another. Through inheritance new classes can be built on top of an existing hierarchy of classes. Allowing code sharing (and hence reusability) among, these classes. "With inheritance, a class called a subclass can be defined on the basis of the definition of another class called a superclass. The subclass inherits the attributes, methods, and messages of its superclass. In addition, a subclass can have its own specific attributes, methods and messages which are not inherited. (Bertino and Martino, 1993)

### 2.6.3. Approaches To Object-Oriented Databases

Several approaches for the incorporation of object-oriented capabilities into databases are currently available;

1. Novel database data model/data language approach. To develop an entirely new database language and database management system with object-oriented capabilities.
2. Extending an existing database language with object-oriented capabilities. Since SQL is a standard and is by far the most popular database language, the most reasonable solution is to extend this language

with object-oriented constructs, reflecting the object-oriented capabilities of the underlying database management system.

3. Extending an existing object-oriented programming language with database capabilities.
4. Providing extendable object-oriented database management system libraries.
5. Embedding object-oriented database language constructs in a host (conventional) language.
6. Application-specific products with and underlying object-oriented database management system. The development of application/domain-specific tools and environments that either use object-oriented database technologies or provide an object-oriented database view for the application domain. The intention in application/domain-specific solutions is not to provide a general-purpose object-oriented environment. Rather, only useful or application-specific constructs, possibly with some object-oriented features, are made visible to the user.

(Barsalou et. al., 1991) discusses how by combining the relational database concept of view and the programming language concept of objects, the resulting view-object model supports simultaneously abstract complex units information and sharing of these units. O2DB showed potential to overcome several know weaknesses of the RDB- based project control. However, the O2D concept is still maturing. Many technological difficulties are awaiting to be resolved. (Kim and Ibbs, 1992). (Chung and Chang, 1995) explores several approaches how to adapt relational database system for objects. (Premeriani et. al., 1990) describes a technique for constructing an O2DBMS from an existing technology relational database management system (RDBMS) and a small amount of human written object-oriented programming language (O2PL) code. The described technique takes an existing RDBMS and hide it beneath an O2PL combining the best features of both of them. RDBMS have a sound theoretical base and work well for different applications. The O2PL allows complex algorithms to be written that would be hard to express in and RDBMS. It also substitutes the lack of some important features in RDBMS needed for advanced applications such as abstract data typing and versioning. The technique has been applied successfully to support an editor for electric circuit design (the editor stores its data in a DB for access by other programs such as mathematical simulators). (Zorn and Chaudhri, 1995) issues in integrating relational databases with object oriented systems were discussed in this workshop.

#### 2.6.4 Object-Oriented Database in Engineering

Designing complex systems is one of the most difficult tasks engineers undertake. As pointed out by (Cattell, 1994), several reasons lie behind this difficulty:

- i. The design itself may be very large.
- ii. Many Independent subsystem designs exist.
- iii. The design evolves with time.
- iv. A huge number of engineers of different disciplines are involved.

Engineers have always used tools to help them design large systems. In this section, we will, briefly, show some of these tools and how could an O2DBMS integrate them.

### Computer-Aided Design

As its name suggests, computer-aided design denotes the use of computerized systems and tools for designing products. Any product development (computerized or otherwise) has a design life cycle. The steps include requirement specification, analysis and design, implementation, testing, manufacturing, and so on. With CAD, computerized tools are used specifically in the design phase of product development. There are at least two variations of CAD: electronic CAD (ECAD), for the design and implementation of VLSI circuits, and Mechanical CAD, for the design and implementation of physical machines and their parts. The support of versioning when designs are iterated upon, and the control of concurrent accesses and updates in projects, object-oriented databases provide ideal persistent repositories for CAD objects.

### Computer-Aided Manufacturing

CAM refers to a software system that offers assistance in the manufacturing or production of components or machines. With CAM, computer systems are involved in monitoring and controlling the production cycle. This means that computer networks operate the manufacturing floor. The status of various machines and monitors is continually processed and communicated by the system. This might involve the monitoring of temperature and pressure. The role of the underlying object-oriented database system is the storage of the objects, the object states, and the history of object states in the manufacturing process.

### Computer-Aided Software Engineering

As with most design engineering disciplines, there are well-known steps in the software development cycle. The requirement specification is a high-level description of the problem or the product. Requirement analysis is a detailed specification of the problem, identifying the entities or classes and the relationships between the entities and operations performed for the application or the product. The design specification is a detailed specification that, in addition to the entities, relationships, and operations identified in the analysis phase, specifies the algorithms and supporting classes for the implementation. The implementation deals with the coding. Testing deals with the quality assurance and testing of the software. Release and maintenance are other phases.

Object-oriented databases can be used to store and retrieve the code base of complex software engineering projects. The various libraries that are used to construct the system can be modeled as

complex objects. Also, the object-oriented database can be used to keep track of the various versions and "builds" of the system. It can also provide a check-out/check-in mechanism for controlling the concurrent updates of software modules.

### 2.5.5 Why O2DBMSs in Engineering Design and Manufacturing?

Engineering design and manufacturing data is difficult to capture and represent with conventional data models for the following reasons:

- This data contains a non-homogeneous collection of objects. Conventional data models deal with homogeneous collections.
- Conventional DBMSs are good for formatted data, short strings, and fixed-length records. Engineering application need long strings and have variable-length records or texture information.
- Temporal and spatial relationships are important in design for layout, placement, and assembly operations.
- This data is characterized by a large number of types, each with a small number of instances. Conventional databases have just the opposite situation.
- Schemas evolve constantly in manufacturing databases because designs go through a long period of evolution.
- Transaction in manufacturing databases are of long duration.
- It is necessary to keep old versions and to create new versions of the same object.
- Data must not be duplicated at lower levels.

Because of these demands, object-oriented models are favored because they possess the following characteristics:

- *Common models.* The designer's mini-world can be mapped into the database objects by a one-to-one mapping.
- *Uniform interface.* All objects are treated uniformly by accessing them through user-defined operations.
- *Support of complex objects.* O2 models allow creation of arbitrarily complex objects involving hierarchies and lattices.
- *Information hiding and support of abstraction.*
- *Versioning.*
- *Modularity, flexibility, extensibility, and tailor-ability.*

For a detailed discussions, see(Ahmed and Navathe, 1991; Batory and Buchmann, 1984; Kim, 1990b, Kim, et. al., 1989; Ketafchi and Berzins , 1986; Spooner, et. al., 1984).

### 2.6.6 O2DBMS as an Integration Environment for Engineering Applications

Today's engineering applications suffer from a severe integration problem. Engineering –the entire process starting with conceiving a product idea, to conceptual design, to planning, to manufacturing, to testing, and finally, distribution and after sales services- consists of a myriad of individual, often complex steps. These steps depend on very large volumes of data, and they yield volumes of new data. Many computerized tools are available to handle each of these different activities, such as CAD, CAM, CAPM, etc. Unfortunately, as shown in Fig. 9, each of these tools has its own interface and customized file structure, which are not accessible by other tools.

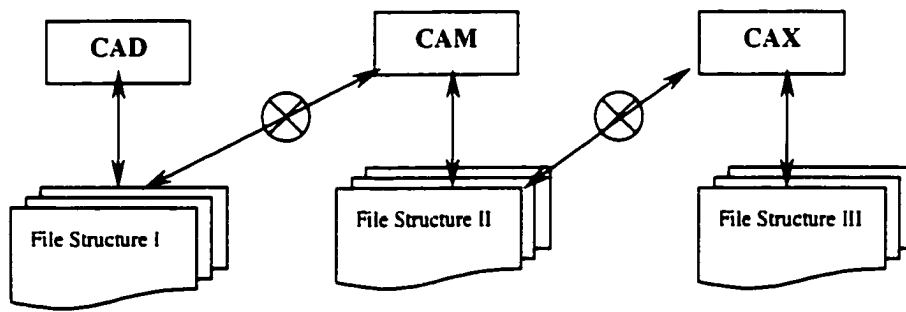


Figure 9. No Integration

#### Databases : An Integration Tool For Engineering applications

One of the main challenges in information management in engineering applications is the integration diversification, special-purpose application modules. As discussed in (Kemper and Moerkotte, 1994), different integration levels exist. These are:

- I. Integration Level 0. Characterized by a complete isolation, no integration whatsoever. Each of the diverse application modules manages its own information in highly customized database and file structure. See Fig. 10.

- II. Integration Level 1. Here, the interface of different modules is achieved via special, customized converters that transfer the information structure required by one application module to the needs of the other application module. Fig. 11.
- III. Integration Level 2. As shown in Fig. 12, the number of converters is reduced. A neutral database now exists to serve as the common link for all application modules.
- IV. Integration Level 3. Fig. 13. Here, the database system forms the central agency of integration. All application modules interface with a common database.
- V. Integration Level 4. Fig. 14. From a database point of view, the integration level 3 is the best we can achieve. However, the reality is that many customized CAX modules exist, and possess highly customized data formats, and, therefore, cannot easily be rewritten to interface with a database system. Therefore, means to integrate such key systems into an integrated database environment must be provided. Now the database system plays the role of the neutral data format.

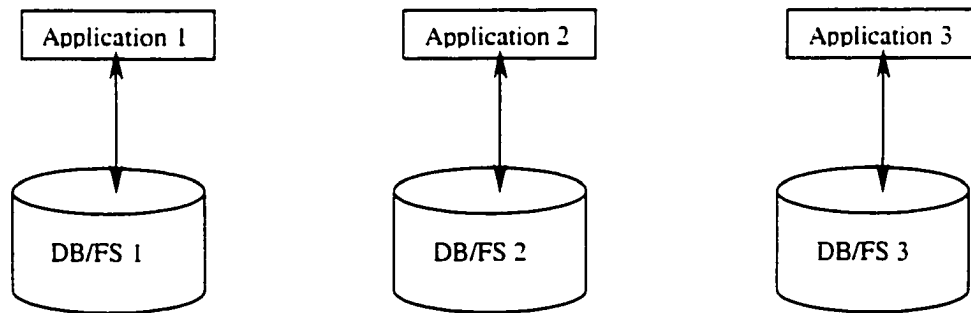


Figure 10. Integration Level 0: Isolation

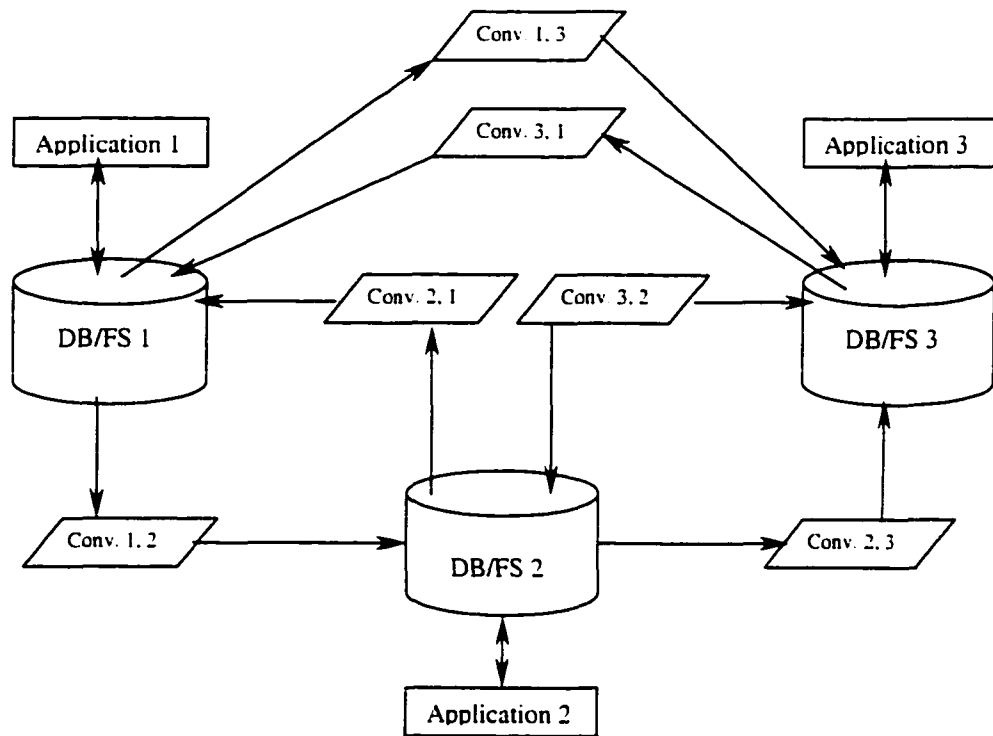


Figure 11. Integration Level 1. Converters

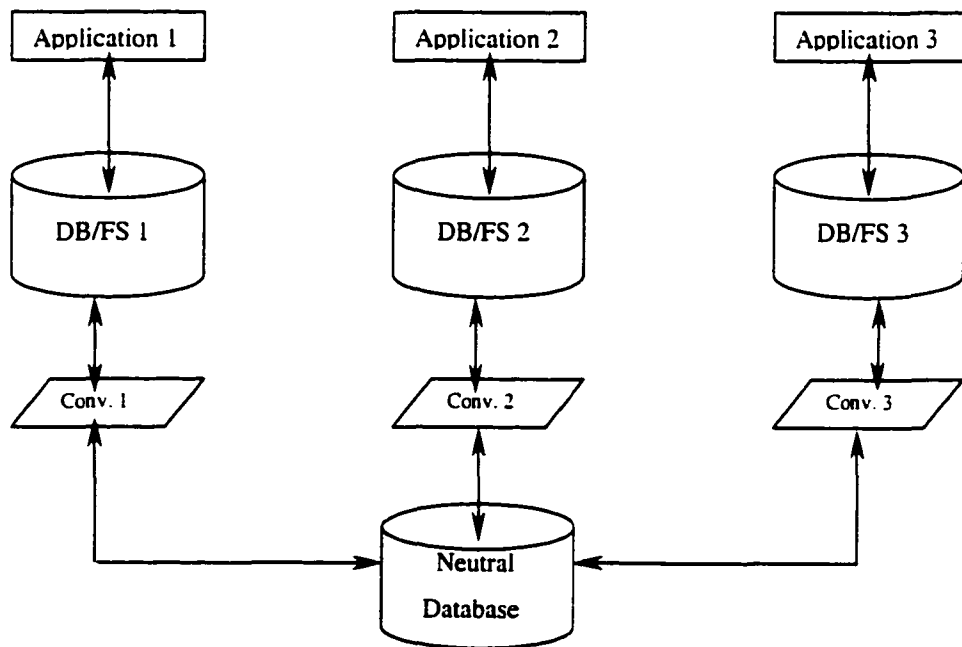


Figure 12. Integration Level 2. Neutral File Format



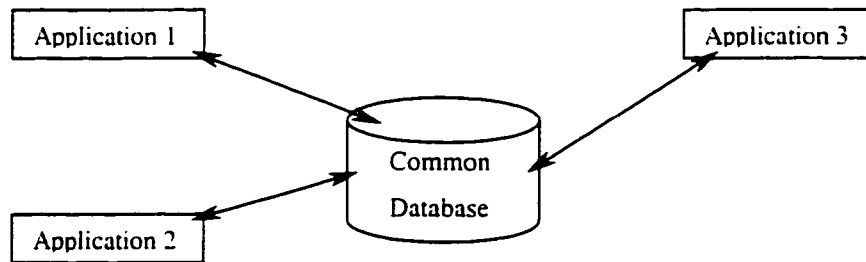


Figure 13. Integration Level 3. A Centralized Database

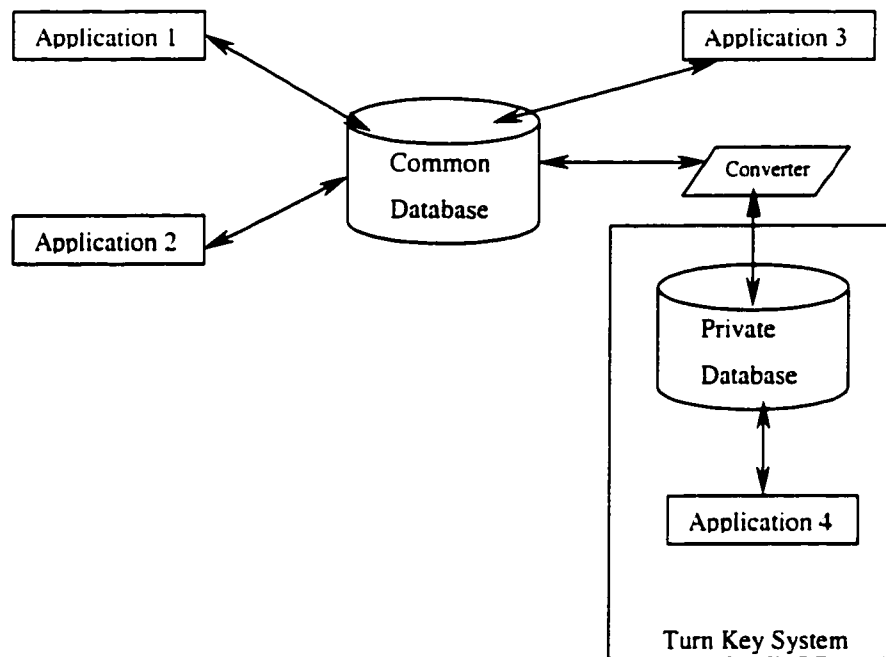


Figure 14. Integration Level 4. Integration of Stand Alone Components

#### 2.6.7 O2DMSs in Engineering, a Literature Review.

Why are database systems infrequent components of computer-aided design (CAD) systems? The answer is the lack of modeling power and performance. Requirements for advanced CAD tools and design environments are different from requirements of traditional database systems, and current models are not up to handling complex design structures for very large scale integration (VLSI) or mechanical (CAD).

(Bancilhon et. al., 1985) provides both an intuitive description and formal development of a new model of CAD transaction that allows a group of cooperating designers to: 1) arrive at a design without being forced to wait over a long time, and 2) collaborate on a design with another group by assigning subtasks. (Banerjee et. al., 1990) presents the data model for ORION, a prototype database system that adds persistence and sharability to objects created and manipulated in object-oriented applications. This data model consolidates and modifies a number of major concepts found in many object-oriented systems, such as objects, classes, class lattice, methods, and inheritance. These enhancements are strongly motivated by the data management requirements of the ORION applications from the domains of artificial intelligence, computer-aided design and manufacturing, and office information systems with multimedia documents. (Maier, 1989) argues why O2DBs could satisfy the needs of design application, especially speed. (Fishman et. al., 1990) presents a research prototype of the DBMS requirements at the beginning of the 1990s, Iris. The Iris DBMS consists of

- i. A query processor that implements the Iris object-oriented data model.
- ii. A relational storage subsystem (RSS) – like storage manager that provides access paths and concurrency control, backup, and recovery, and
- iii. A collection of programmatic and interactive interfaces.

The data model supports high-level structural abstractions, such as classification, generalization, and aggregation, as well as behavioral abstraction.

In industrial environments, lack of consistency between interrelated data creates difficulties in interoperation between systems. (Bellcore and Karabatis, 1993) demonstrates a framework that allows precise and detailed specifications of complex interdependencies that lead to efficient strategies to enforce the consistency requirements among the corporate data managed in multiple databases. (Berchtold and Kriegel, 1997) presents the prototype of a DB system (S3) supporting the management and similarity retrieval of industrial CAD parts. Thus, reducing the cost for developing and producing new parts by maximizing the reuse of existing parts. The implementation of the S3 system is based on an O2 design. The advantage of this design is the good extensibility of an O2 system: similarity algorithms and index structures may be added to the S3 system without changing the other components. (Bjork and Penttila, 1991) illustrates the potential of an O2 semantic product data model approach for facilitating data exchange between heterogeneous computing applications based on RDB and CAD systems. As a step towards the achieving of an O2 programming environment for CAD application, through a tailored O2 DBMS, (Buchmann, 1985) presents an architecture for a process-plant CAD system. In (Kemper and Wallrath, 1987), the data modeling and computational requirements for (CAM) DBs are analyzed, and the most common representation schemas for modeling solid geometric objects in a computer are described.

The development of computer integrated design systems requires well-defined models. The main two types of models are:

- i. Product models, which describe the product of design; and
- ii. Process model, which describe the process of design.

O2 approaches provide a powerful means to express and implement both models. Product and process models are essential types toward computer integrated systems, and the O2 approach is attractive as an implementation tool and as a unifying concept between models. A general discussion of models for engineering design is presented in (Sause et. al., 1992), followed by more detailed discussion of both product and process models. To support collaboration among different project phases (design, planning, construction, and monitoring), an O2 model that integrates both product and process information was proposed in (Stumpf et. al., 1996)

Most of the current CAE tools for design do not separate standard representation and standard usage within their programs. (Abdalla and Yoon, 1992) describes a software development using an object-oriented approach to integrate finite element and graphics application programs, which are to become parts of an integrated civil (mechanical) engineering system. The short-term objective of the project, that is, developing a general-purpose data translation facility for finite element and graphics-based programs was achieved. Potential for its application to other engineering environment will be the findings of the long-term objective. The model (facility) took advantage of the effective O2 features (encapsulation, inheritance...) but still it shows run-time inefficiency. In (Garrett and Haki, 1992), an O2 model for representing a design standard was described. An O2 approach provides organizational structure and representational flexibility, both of which are lacking in the logic-based approach currently used. Furthermore, the O2 approach makes feasible the development and usage of a collection of editors specialized to each class of data item, then enhancing the ease of use of this O2 model. (Golendziner et. al., 1997) presents an approach to extend O2 data model. The approach was used to model an engineering application fulfilling its requirements.

Computational objects –objects with functional interpretations- are a valuable addition to structural objects in O2 data models for engineering CAD/ CAM applications. (Zhu and Maier, 1989) presents an approach to add computational semantics to structural semantics in O2 data models.

Many literatures have recognized the need of composite objects for CAD/CAM applications, and tried to develop some frameworks for modeling composite objects in these application areas. (Batory and Buchmann, 1984; Batory and Kim, 1985; Emond and Marechal, 1983; Kim et. al., 1987; Kim et. al., 1989; Lorie and Plouffe, 1983; Navathe and Cornelio, 1990; Stonebraker et. al., 1983).

Workgroup computing systems are emerging to support a group of users engaging in common tasks such as group decision making, engineering design, or group scheduling (a group of people performing collaborative tasks in a shared environment). (Huh, 1998) proposes an object-oriented model for a change management framework supporting workgroup systems to facilitate managing dependency relationships between shared objects and dependent user views and to coordinate change and propagation activities between the two in client-server computing environments.

The VLSI design environment is characterized by a large volume of data, with diverse modalities and complex data descriptions. Both data and descriptions of the data are dynamic, as is the underlying collection of design techniques and procedures. (Afsarmanesh et. al., 1990) describes an approach to the

specification and modeling of information associated with the design and evolution of VLSI (very large scale integrated circuits) components. The approach is characterized by combined structural and behavioral descriptions of a component. An extensible object-oriented information management framework, the 3DIS (3 dimensional information space), is presented. The framework has been adapted to capture the underlying semantics of the application environment by the addition of new abstraction primitives. The 3DIS (Afsarmanesh and Mcleod, 1984; Afsarmanesh, 1985) is a simple but extensible object-oriented information management framework. The 3DIS is mainly intended for applications that have dynamic and complex structures and whose designers, manipulators, and evolvers are non-database experts. (Singhal et. al., 1993) presents an O2 data model for VLSI/CAD data. Then uses this model to implement a design data manager. A design data manger is emerging as an important component of an integrated VLSI/CAD system. Two earlier papers (Mehmood et. al., 1987; Singhal et. al, 1989) presented data models for design versions, design configurations, and circuit description data.

The problem of concurrency control in O2DBs cannot be handled using methods addressed to classical databases, due to differences in structure.

(Cellary and Wieczrzycki, 1993) presents an efficient new method for concurrency control in O2DBs. Two former attempts to adopt the classical hierarchical concurrency control to the requirements of objects are discussed in (Cart and Ferrie, 1990; Garza and Kim, 1988).

The ability of O2 language to create new objects is one of the important features distinguishing them from relational languages. In (Denninghoff and Vianu, 1993), the expressive power of various O2 languages is investigated with respect to their ability to create new objects. The results clarify the impact of the various constructs on object creation.

There are essentially two characteristics of design objects, which impact upon the iterative and exploratory nature of the design process. First, they are hierarchically formed assemblies of component objects. Second, they go through phases of design giving rise to multiple versions. The iterative and exploratory nature of the design process lacks to two aspects of design objects which must be dealt with. First, they are usually complex; that is, they are assemblies of components that themselves may be constructed hierarchically from constituent objects. Second, there can be several alternative descriptions, called versions, of a design object. Traditional DB systems, designed to deal with only regular and structured data with no built in concepts of versioning, cannot efficiently manage design data.

(Ahmed and Navathe, 1991) proposes a concrete approach for defining a unifying framework for the effective management of versions of composite objects. The approach takes advantage of O2 concepts to deal with the modeling of composite objects and their version management.

One of the important notions of the O2 paradigm is the object subclass hierarchy (OSH). An object is said to be a subclass of another object class if every member of the former (object class) is also a member of the latter (object class). OSH provides a useful way for specifying property and behavior inheritance (Blakeley et. al., 1986; Jacky and Kalet, 1987). (Kung, 1990) presents a method for implementing OSH in SQL to provide an O2 view of data at the expense of affecting retrieval performance.

The part-whole semantic relationship (the part relationship, for short) is an important modeling primitive in many advanced modeling domains such as manufacturing and design. (Halper et. al., 1994) demonstrates how to seamlessly integrate a powerful part relationship model into an existing O2DB system. A comprehensive part relationship model for O2DBs was introduced in (Halper, 1993; Halper et. al., 1992a; Halper et. al., 1992b; Halper et. al., 1993). Utilization of part relationship in some O2DBSs was demonstrated in (Kim et. al., 1989; MacKellar and Peckham, 1992; Nguyen and Rieu, 1991). (Bertino and Guerrini, 1996) shows how an O2 view mechanism can be exploited to allow different applications to see the same DB according to different viewpoints.

(Katz, 1990) describes the general requirements for version management systems, and proposes a terminology unification for version management in engineering databases.

### 2.6.8 Shortcomings of O2DBMSs

During the past fifteen years, there was a great rush to develop O2DBs in both commercial and research fields, but until today, they have not had significant impact in the current database market. Behind this weak market position lie several distinct aspects to the shortcomings of the current O2DBs technology.

- i. Absence of a standard.
- ii. Lack of a formal foundation for a database language.
- iii. Lack of query facilities, or nonconforming of their query languages with ANSI SQL.
- iv. Lack of authorization support- that is, to allow users to grant and revoke privileges to read or change object and/or relations they created to other users.
- v. The limited capability of parameterized performance data.
- vi. Non-automated locks set and release.
- vii. They don't allow users to dynamically change the database schema.

The revolution of O2DBs has not been accomplished by an equivalent evolution of theoretical models, leaving the foundations for O2DBs ill defined. Design of an O2DBMS has been hampered by the lack of design techniques/tools which correspond to the theoretical model. To date there is not a common theoretical model- or core data model - for the components of O2DBMS, the O2DB system itself, or their inheritance strategies (Cattell, 1994b; Hurson and Pakzad, 1993; Kim, 1990a; Kim, 1990b; Kotz-Dittrich and Dittrich, 1995; Fong et. al., 1991). This deficiency hinders both research and adoption of O2DBMS by industry. (Kotz-Dittrich and Dittrich, 1995) A core model, based in a mathematical foundation, is necessary to support development and implementation of O2DBs and to provide a foundation for theoretical investigation. (Andrews, 1991; Beerli, 1990). (Hines, 1998) defines a core conceptual O2DB model providing a foundation and framework for theoretical research.

(Hughes and Winslett, 1995b) presents a performance evaluation framework (PEDCAD) for applications which use an O2DBMS. (Ching et. al., 1996) describes a new model of object database application for use in performance modeling. The model can be used to get rough performance estimates or more precise estimates if more information is provided and more computation is expended. Both kinds of estimates are useful in different stages of application design, which typically involves several cycles of identifying and improving performance bottlenecks.

## 2.7 Comparison of Different Data Models and Systems

“Prior studies can be classified into three categories: (1) comparison of classical data models, (2) comparison of conceptual data model with classical data model, and (3) comparison of conceptual data models.” (Lee and Choi, 1998)

When comparing the various data models presented in the chapter, several aspects may be taken into consideration. In this section, we will, briefly, compare different models’ data representation concepts, and then compare some of their performance characteristics. The aim of the section is to show the general differences among different data models, not to find the best one.

This comparison belongs to the first category mentioned above, while (Lee and Choi, 1998) conducts an empirically comparison among four conceptual data modeling techniques: EER (Extended-entity Relationship), SOM (Semantic Object Model), ORM (Object Role Modeling), and OMT (Object Modeling Technique). these techniques were compared with respect to three dimensions: (1) model correctness, (2) modeling time, and (3) perceived ease-of-use.

Experimental results indicate some significant differences among the four techniques. Although the study does not answer why one model is better than another, its findings may help data modelers better understand different modeling technique. The importance of proper selection of the used conceptual data model arises from the fact that using a conceptual data model makes it easier to understand and interpret a reality and then describing it properly which results in an effective database design

### 2.7.1 Comparison of Concepts

The terminology used in the ER, Relational, Network, Hierarchical, and O2 models are compared in Table 1.

Table 1. Comparative Terminology of Different Data Models

E-R Model	Hierarchical Model (IMS)	Network Model	Relational Model	O2 Model
Entity/Object Type	Segment Type	Record Type	Relation Table	Collection of Objects
Object Instance	Segment instance (occurrence)	Record Occurrence	Table Row	Object
Variable / Attribute	Field/Data Item	Field/Data Item	Table Column	Attribute
Value Set	Data Type	Data Type	Data Type	Atomic Data Type
Key	No Equivalent Concept or Term	No Equivalent Concept or Term	Candidate Key	No Equivalent Concept or Term
No Equivalent Concept or Term	Sequence Key or Sequence Field	Key or Unique Field	Primary Key	Object Identifier
Relationship	Parent-Child Relationship (PCR) Type	Record Type	Established by using Foreign Keys	Established by using References

References (Sundgrer, 1985; Elmasri and Navathe, 1994)

### 2.7.2 Performance Characteristics

- 1) Regardless of what the underlying structure is in the DBMS being used for a particular application, one fundamental concept of data integration remains the same: records describing different, but related, entities in the application environment must in some way be capable of being associated each with another. In hierarchical and network databases, joins are performed implicitly at the time that the hierarchies and networks are designed, and explicitly with pointers when the data is loaded according to those structures. The power of data integration is primarily a function of the data structure. In relational database joins are performed "on the fly" at execution time by the DBMS according to commands given to the system at that time. The data structure, that is, the relations, must have join fields present in the right places, but the power of data integration is fundamentally an execution-time function of the DBMS.
- 2) There are several prominent methods for designing a database. Each method has proponents who feel that the one that they favor is easier to use. One result of the design process of a relational database is the proper distribution of join fields in the relations, based on the way the application environment operates. This assures that any

semantically correct query (that is, one that makes sense in terms of the application) that requires a join will find the join field in the right places when it needs them. But join operations, particularly brute force joins that require full file scans of one or both files, can be painfully slow for large files. When designing a hierarchical or network database, decisions must be made as to which of the integrating associations are actually designed into the structure. In the hierarchical and network environments, the performance of join-oriented accesses based on relationships that have been designed into the structure (and are ultimately implemented with pointers) can be very fast, although accesses based on relationships that have not been designed into the structure can be extremely slow.

- 3) Later modifications to the database, depending on their nature, can often be made more easily in a relational database, too. We're not talking about adding additional record or tuple occurrence to a database, but rather changing or expanding the number of record types. Adding a new record types to hierarchical or network databases may require fitting them into existing structures, reloading the data into the database, and resetting pointers. It may also affect queries in existing application programs.
- 4) In relational database, data integration is accomplished at the time the data is loaded and is maintained by all of the physical pointers that interconnect the records. Hierarchies and networks may appear to be complex structures and may present problems in future database structure modifications, but their physical pointers provides speedy retrieval.

(Hughes and Winslett, 1995a) discusses how index choice for an ODBMS application differs from RDBMS index optimization. The index suggestion problem for an ODBMS is to automatically identify access paths which might be useful to the application. The paper defines the index suggestion problem for ODBMS applications and describes an intuitive technique for solving this problem.

## 2.8 The Future

The steady increase of the number of complex industry applications, which entails large-scale integration of computerized tools into the manufacturing process, means that O2DB technology will indeed become more and more important. (Kim, 1995) suggests three major conditions to be satisfied before the O2DB can deliver on its promises. First, new database systems that incorporate object-oriented data model must be full-fledged database systems that are compatible with RDBs.

Second, application development tools and database access tools must be provided for such database systems, just as they are critical for the use of RDBs.

Third, a migration path (a bridge) is needed to allow coexistence of such systems with currently installed RDBs, so that the installations may use RDBs and new systems for different purposes and also gradually migrate from their current products to the new products.



## CHAPTER III

### THE INTEGRATION FRAMEWORK (AN O2DBMS APPLICATION FOR INTEGRATED ENGINEERING -COMPUTATION ENVIRONMENT)

#### 3.1 Introduction

Future product development process faces many challenges. Accurate, timely, and efficient decisions must be made throughout the design process, particularly early on when such decisions have the most influence on cost, quality, and market demand. On the other hand, this development process is “characterized by decentralized design teams cooperating with a high degree of process parallelism.” (Roller and Eck, 1999) To make the best product development decisions, management needs to leverage the cumulative knowledge from every discipline across the enterprise and with careful regard to supply chain resources. Tools for cooperative work are required to provide mechanism for facilitating and controlling collaborative work as well as to support the individual work of single teams dealing with large amount of data.

Over the last few years, businesses have become increasingly dependent upon Packaged Applications (PAs) to serve their information management needs. In many companies, accounting, product data management, and manufacturing logistics are performed within PA software. As the information needs of companies have continued to grow, the demand for increased functionality has escalated. Hence, complexity of the PAs has also increased as the limits of their existing technologies are stretched beyond their original intent. This has caused PA software to become less flexible, more difficult to use, and very expensive to acquire and maintain. Furthermore, integrating PAs with other mission-critical systems is exceedingly complicated, and when finished, the links between them are unique to the site for which they are developed. This means that any time a company upgrades their PA software, the integration pieces must be re-developed, which again adds further cost and complexity. The information system also becomes less flexible and less able to support the “what if?” analyses that are so vital to continuous development of higher value and more profitable products.

In this chapter, a framework that integrates the capabilities of four commercial software, Microsoft Excel<sup>™</sup> (spreadsheet), Microsoft Project<sup>™</sup> (project management), What’s Best! (Optimization Add-In), and Visual Basic<sup>™</sup> (Programming Language), with a state-of-the-art object-oriented database (knowledge medium), InnerCircle2000<sup>™</sup> is being presented.

The framework shows how a knowledge medium (based on an active object-oriented database) would be the backbone of an integrated engineering-computation environment.

The components integrated in the framework are discussed in details below, with an emphasis on manufacturing applications.

### 3.2 The Proposed Framework Structure

The proposed framework, as shown in Fig. 15, consists of the following components:

1. An Adaptive Knowledge Network “AKN”(based on an object-oriented database), InnerCircle2000™,
2. Spreadsheet software, as a mathematical modeling tool, Microsoft Excel™,
3. An optimization tool, Excel add-in, What’s Best!.
4. A project management software, Microsoft Project™, and
5. A programming language, Visual Basic™.

The role and capabilities of each tool will be discussed below.

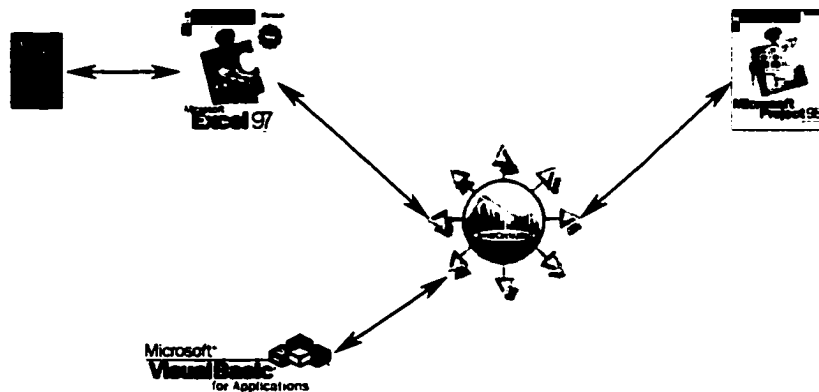


Figure 15. The Presented Framework

#### 3.2.1 What Can Such an Integration Framework Achieve?

The disciplines basic to making progress in manufacturing include mechanical engineering, industrial engineering, mathematics, management science, and computer science. These separate disciplines are individually supported by their research, methods, and software. There is a lack of focused attention on how to integrate knowledge from many disciplines into knowledge that furthers manufacturing goals. Moreover, at the same time that this lack of strategy is apparent, all dimensions of manufacturing (e.g.

products, processes, markets) are becoming more complex and diverse. Complex new products based on information content and their accompanying information-dominated design and manufacturing methods already require us to deal with entirely new scale of complexity. The integration among various elements of manufacturing on one side, and among manufacturing and other disciplines on, the other side, can be achieved through this framework. By providing ways to facilitate and manage the complexity of these information intensive activities, AKN can play an important role in supporting and even enabling the complex practice of manufacturing.

Furthermore, when project management is being applied in a manufacturing environment, large amounts of data are dealt with on a day-to-day basis. Linking data for project management purposes will give flexibility needed to make more effective decisions. A database has the capability to store, retrieve, and manage large amounts of data. Pieces of data can be quickly retrieved and processed as appropriate to generate information needed for decision-making. Moreover, with appropriate data input and instructions, a spreadsheet offers a powerful program that combines three major functions: spreadsheet(mathematical modeling), graphics(charts), and data analysis. Once data is entered on the spreadsheet, we can apply a variety of calculations can be applied. All these functions can be put together to be used for project management purposes.

### 3.3 InnerCircle2000™

InnerCircle2000™, TeamVision's Adaptive Knowledge Network (AKN) software, enables teams in all disciplines across an enterprise to collaborate on projects and enhance the design process by improving the way information is presented and analyzed. Moreover, InnerCircle2000 provides a flexible environment enabling decision-makers to explore the project before committing scarce resources to their maturity.

Adaptive Knowledge Networks (AKNs) represent the fourth era of product development collaboration, with the goal to enhance enterprise decision-making. The first era, computer-aided design (CAD), was aimed at improving engineering productivity. The second era, computer-aided engineering (CAE), focused on automating the engineering effort by incorporating other discrete analyses such as computational fluid dynamics and finite element modeling. The third era, Virtual Product Data Management (VPDM), combined the previous two and added web-enabled digital definition, reuse of design and process technology, and innovation synthesis. These initial stages were purely vertical in nature, thereby automating the design analysis process as the product definition matured through time.

AKNs are the next logical step, attempting to simultaneously integrate data cross-time and cross-function with a special emphasis on the financial relationship between design and manufacturing technologies. This includes the adoption of more robust and readily available design representation by integrating CAD/CAM

technologies, capacity assessment, factory simulation, cost estimating, and business case analysis to arrive at affordable marketing variants. Object-oriented technologies and tools for leveraging intellectual capital, similar to VPDM, also drive AKNs. Figure 16 illustrates how AKNs are the next logical step in the evolution of enterprise decision making.

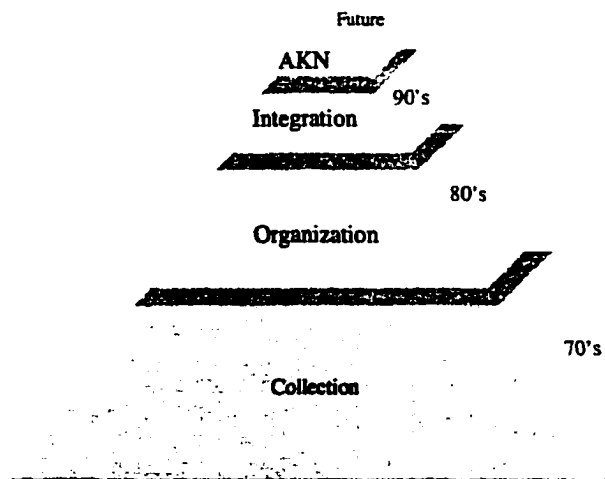


Figure 16. Evolution of Enterprise Decision Making

InnerCircle2000™ technology has the five key attributes that a corporate decision-analysis technology must possess for customers to realize significant Returns On Investment:

1. Provide rapid and accurate "what if" analysis throughout the design process
2. Holistically evaluate product and process technology interaction
3. Prevent design process overlap - eliminating redundant or needless effort
4. Eliminate errors and omissions that necessitate redesign and re-certification
5. Provide objective value improvement goals and reporting metrics.

Using InnerCircle2000™, it is now possible for subject matter experts across the enterprise and throughout the supply chain to collaborate effectively on a project. For example, as an engineer makes modifications to a design within a CAD system or an accountant updates figures in a spreadsheet, InnerCircle2000™ detects these changes and dynamically passes updates to the team members who depend upon the information to make solid business decisions. In this way, everything from factory simulations that predict resource consumption to financial spreadsheets that model production costs are synergistically included in the design process.

### 3.3.1 Survey of InnerCircle2000™

In this section, a survey of the InnerCircle2000™ architecture is presented to clarify how InnerCircle2000 fit into the heart of an integrated-computation engineering environment.

InnerCircle2000 is realized as a “knowledge base” because: 1)it’s goal is not just to store data, but also to represent its semantics explicitly, and 2)information is represented in a high abstraction level, a property gained from the use of an O2DBMS. As defined in (Stefik, 1986), a knowledge medium is “a computational environment in which explicitly represented knowledge serves as a communication medium among people and their programs.” (Mylopoulos et. al., 1996) discusses the difference between the terms “knowledge base” and “database”.

The development and implementation of knowledge bases are the subject of a number of research articles. In (Agrawal, 1995), an approach to support activities in collaborative environments is proposed. (Centeno and Stnadridge, 1991) describes an integrated simulation modeling environment which utilizes an information-based approach, based on a relational database, for the description of manufacturing systems. In (Chen, et. al., 1995), O2 techniques are employed to model a knowledge-based manufacturability assessment tool. (Gruber, et. al., 1992) proposes a framework, a knowledge medium, for supporting active information sharing and coordinated communications among members of a product development organization. (Huh, 1998) implements an O2DBMS for a change management framework supporting work group computing system, that is, a group of users engaging in common tasks such as group decision making or engineering design. (Kandt, 1994) presents a software system (SeeQFD), developed to support concurrent engineering, that applies an object management system.

Another approach to cooperative transactions is given in (Korth, et. al., 1990). (Maltez, 1987) presents a knowledge-base approach to project management. (Manola et. al., 1990) describes the development of the data model of PROBE, a knowledge-oriented DBMS being developed at CCA (Dayal, et. al., 1985; Dayal and Smith, 1986) The data model, called PDM, illustrates an integration of functional, relational, and object-oriented approaches. The extensions are primarily those required to handle the requirements of new database applications, such as engineering applications (especially design and manufacturing) and cartography, having special or temporal semantics. (Mylopoulos et. al., 1996) presents a shared knowledge base for advance applications such as CAD. In (Portougal and Janczewski, 1998), an information system for operation management database integration was presented. In (Wang, 1999), an O2 approach was used to facilitate communication in the course configuration management. COSMO, a communication scheme for cooperative knowledge-based systems, is presented in (Wong, 1993).

Furthermore, InnerCircle2000™ is an active database in the sense that it initiates actions automatically when certain conditions arise (the event-condition-action-rule).

Approaches to active database are discussed in (Dayal, et. al., 1995; McCarthy and Dayal, 1989; Mylopoulos, et. al., 1996; Stonebraker, 1992). An active knowledge base system (DEVICE) was presented in (Bassiliades and Vlahavas, 1997) is based on a high-level rule integration scheme in an O2DBMS.

To facilitate concurrent engineering, an active database system, Product Information and Knowledge Server (PIKS), was presented in (Domazet and San, 1997). PIKS integrates an expert system with a passive O2MS. (Gatzia, et. al., 1991) sketches some properties of the active O2 DB system, SAMOS, including its knowledge and its execution model. Another active database (Ode) was presented in (Gehani and Jagadish, 1991). (Montesi and Torlone, 1995) proposes a new formal semantics of active databases based on a transaction rewriting technique in the context of the relational model. (Roller and Eck, 199) describes a knowledge base system (an active, distributed, O2DB system) relevant in modern product development.

InnerCircle2000™ architecture is based on:

1. An object-oriented database.
2. N-Tier web based thin client architecture. See Fig. 17.
3. ActiveX and OLE links.
4. An advanced application server.

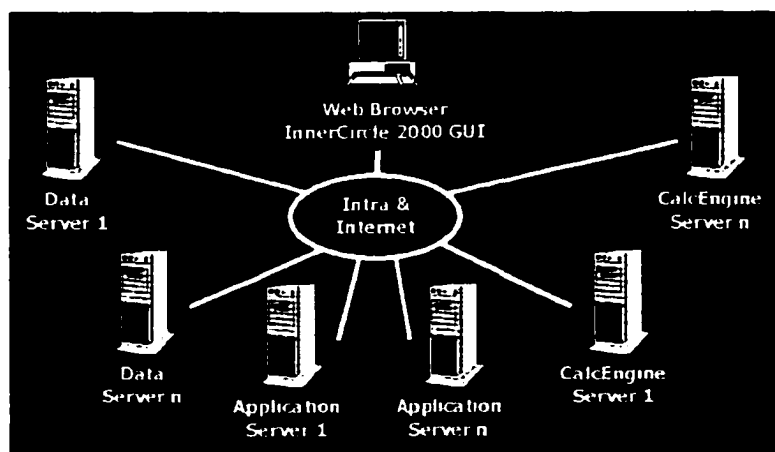


Figure 17. InnerCircle2000™, N-Tier Thin Client Architecture

The use of InnerCircle2000™ as the backbone of an integrated engineering-computation environment provides:

1. Team Integration. As shown in Fig. 18, several teams of different disciplines can work concurrently, and then the results of each team is being collected in a central database (InnerCircle2000™).
2. Functional Integration. Figure 19 shows how data resulted from different activities is being collected centrally. And Fig. 20 shows how is this data being sent to other teams.
3. Application Integration. Figure 21 shows how the capabilities of several commercial software packages can be integrated.

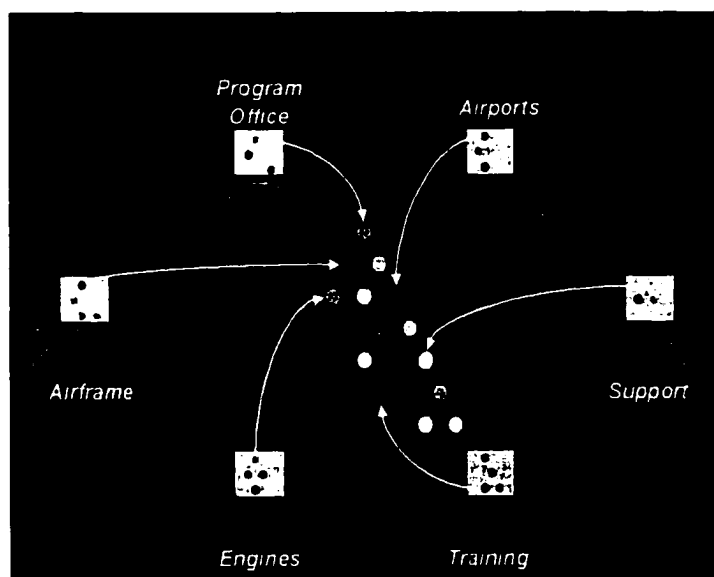


Figure 18. Team Integration (Ref., TeamVision Inc., 1998)

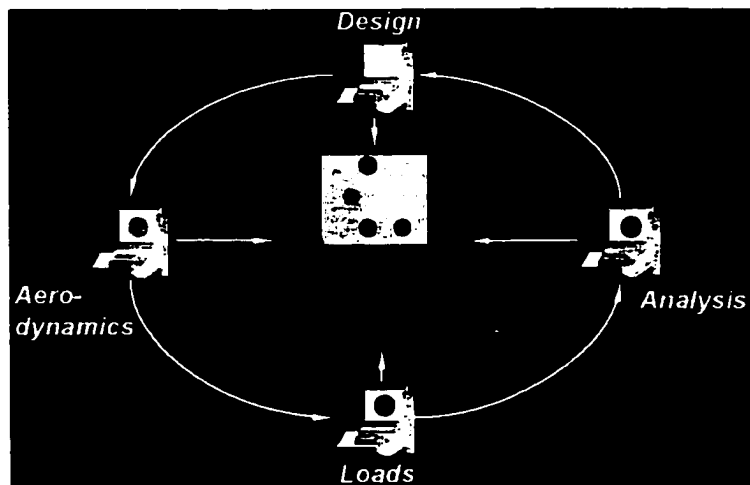


Figure 19. Functional Integration (a) (Ref., TeamVision Inc., 1998)

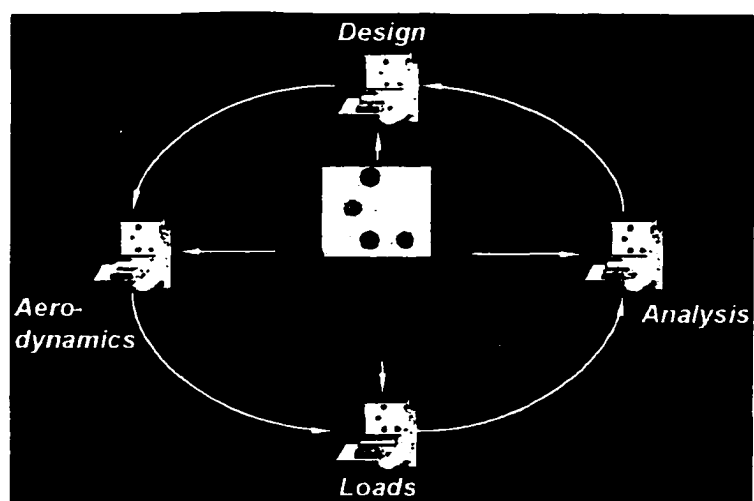


Figure 20. Functional Integration (b) (Ref., TeamVision Inc., 1998)

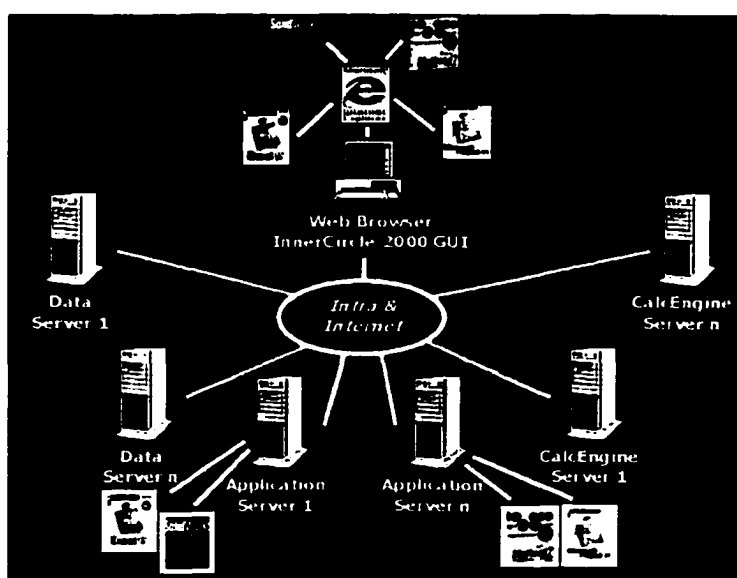


Figure 21. Application Integration (Ref., TeamVision Inc., 1998)



### 3.4 Active X and OLE: Application Integration

The current computing life suffers from the lack of application integration. Thus, end users who need to create compound documents suffer from difficulty when doing this or may be even unable to do it, while corporate developers are continuing to build custom software solutions instead of using the existing off-the-shelf packages. "Microsoft Active X and OLE are a step toward the creation of better software. "Better" here means software that's more reliable, certainly, and more effective as well." (Chappell, 1996)

#### 3.4.1 OLE

Microsoft's Object Linking and Embedding Technology (OLE) is the standard for inter-application communication that is characterized by its tight integration with the underlying windows architecture. OLE technology puts a new form of integration into the hands of end users and developers. This technology lets end users integrate off-the-shelf applications to create rich documents composed of data from a multitude of sources. OLE controls how Windows applications should work with objects, where objects are a combination of some kind of data--text, graphics, video, sound, and so forth--and the functionality needed to create and use that data. OLE makes it possible for software developers to take advantage of the O2 technology as they are able to pack their programs' data as objects that end users and corporate developers can access.

In traditional software applications, developers would typically bind in all the functionality needed by that application. OLE makes it possible to avoid such duplication by facilitating the creation and use of software components. A developer can use such components- sound or graphing features, for example-in multiple applications without having to build them into each one. For the end user, the benefits are obvious: no need to clutter a system with huge, partially redundant applications. And best of all, end users can, in their own data files, store or link to data that resides in other files.

When you insert an OLE object into a container's file, the file becomes an OLE compound document. The container's file doesn't have to know anything about the object itself, instead, OLE tells this file to perform some process to it. All a container has to know is where the data is stored and that the data corresponds to the on-screen image of the data. The container's file doesn't even have to know how to display the data.

There are several advantages to embedding items in a document. First, when you distribute the compound document, the embedded items travel with it, reducing the bill of materials required to distribute it. And the user who receives the compound document needs only the container application installed to display or print the document: the object server doesn't need to be present.

On the other hand, several disadvantages rise up. First, a compound document with several embedded objects can grow quickly, since the data from these embedded objects is physically stored in it. Second, the only way to get to an embedded object is by first launching the container and then invoking the object

server. And finally, any user who needs to edit an embedded item must have an appropriate object server installed.

### 3.4.2 Linking Vs. Embedding

The disadvantages of embedding an OLE object in a compound document can be avoided by choosing to link the item instead. With object linking, as shown in Fig. 22, the container doesn't embed the object's data in its compound document, but instead stores a path or other information that points to the file containing the data.

The main advantages of this approach are: 1) the compound documents grow by only a few hundred bytes per linked object. And 2) the object server can run as a standalone application to access the object data.

On the other hand, distribution can still be a problem. 1) If a linked object has been moved, the object location must be updated in the compound document. 2) If the compound document moves to a different system, the links will probably have to be established all over again. and 3) while other users can display and print a compound document with linked objects, to edit the objects they'll need both the linked objects themselves and an object server capable of editing those kinds of objects.

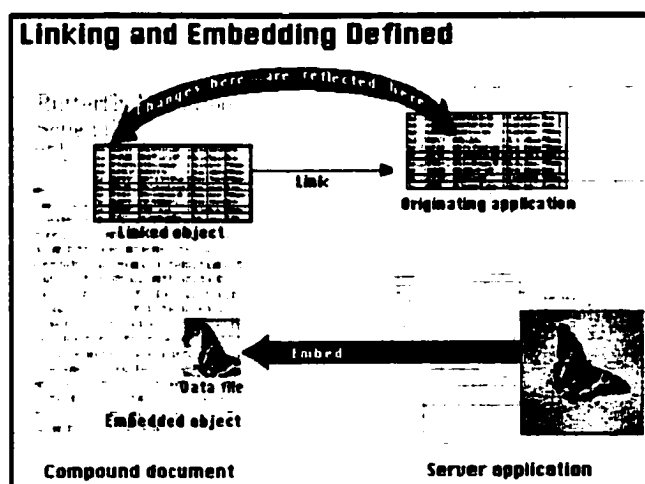


Figure 22. Linking Vs. Embedding

### 3.4.3 COM

Each of OLE's features is based on a set of interfaces, with OLE's Component Object Model (COM) at its core. COM prescribes how applications that use OLE will behave and interact, and provides a mechanism for one application to connect to the OLE interfaces supported by another. All OLE components assume that applications using them will do so through COM.

COM's mission is to ensure that disparate Windows applications can connect and communicate through OLE and that the details of the connection remain transparent to the participating applications. The connection between two applications will be made even if the applications are written by different vendors, and if necessary, COM will drop out altogether once the connection is established. Moreover, COM will hide any discrepancies that may exist between different releases of an application, such that a newer release of one application won't cause another application to break as they communicate through OLE.

### 3.4.4 Active X

The term ActiveX was officially coined by Microsoft at the PDC (professional Developers Conference) March 1996.

Fundamentally, the ActiveX platform is a re-adaptation of existing Microsoft technologies for the web. This standard rests on OLE and Com. Windows development standards. The language used to program Active X objects is VB Script, which a dialect of MS Basic. In fact, "The available tools used to build Web pages with ActiveX functionality owe an obvious debt to Microsoft Visual Basic. By and large, this re-adaptation is a success." (Mace and Rubenking, 1996)

"OLE lets windows programs communicate with each other automatically. The numbers in a spreadsheet can update the numbers in a document, which in turn generate a chart in a graphic package. Double-click those spreadsheet numbers in your document, and up pops the spreadsheet, ready for editing. ActiveX works in similar fashion but across a LAN or the internet (it works between your copy of Explorer and a distance Web site)." (Crowe, 1997) Theoretically, one could give a real-time presentation over the net, with his/her accounting staff in another city plugging in the numbers.

As discussed in (Shadish, 1996), two main reasons lie behind the change from OLE into ActiveX: 1) OLE had grown beyond the linking and embedding concepts of the late 80's; into something that was an all encompassing framework of the technologies and applications, and 2) Microsoft wanted to come up with something catchy to set against the Netscape plug-in browser and Java development combination. And that's how we ended up with ActiveX.

The ActiveX framework technologies can be lumped into three sections as follows:

- 1) Tools: ActiveX controls, Java, and VBScript.
- 2) Applications: Internet Explorer, and Microsoft Office.

3) **Server Framework:** ISAPI, IIS, Microsoft Merchant, Proxy Server, and Media Sever.

### 3.4.5 ActiveX Controls

An ActiveX control is a user interface element created using ActiveX technology, and Component Object Model (COM) technology.

ActiveX controls for Windows CE also support the following features that may be enabled to increase a control's efficiency and effectiveness in certain applications:

- **Just-in-time activation.** Although an ActiveX control can be visible, activation is deferred until the user interacts with the control. Thus, saving time and resources if the control is never activated.
- **Mouse interaction while inactive.** When a control is visible, but not activated, it can still process mouse messages delegated to it by the container. Once the control is activated, it processes messages in the usual manner.
- **Flicker-free activation.** This option prevents screen flicker when an Active X control makes the transition from the inactive to the active state.
- **Windowless activation.** A windowless ActiveX control does not have its own window. The container is responsible for routing user-input messages to the control.
- **Unclipped device context.** This option speeds up its drawing operations of an Active X control, if it never draws outside its client rectangle.
- **Optimized drawing.** When a container draws several controls into the same device context, each control selects its required graphics device interface (GDI) objects. If each control does not have to restore the previously selected objects, it can save time. The container can restore the original objects after all the controls have been drawn

### 3.4.6 OLE for Design and Modeling

With the purpose of further refining and enhancing OLE for Design and Modeling as an industry solution, the Design and Modeling Applications Council (DMAC) was formed at a meeting at Microsoft in Jan of 1995. OLE for Design and Modeling (OLE for D&M) is the logical extension of OLE technology from Microsoft Office applications to ED Cad, CAM, and CAE. It aims to deliver direct access to native CAD models without data translation, giving the same benefits to 3D users that Office users have enjoyed for years. OLE for D&M supports the concept of a container and a server where one application (the server) draws its own 3D data within the graphics window of another application (container).

Four key advantages of using OLE for D&M. 1) No data translation means no loss of high-level data. OLE for D&M enables data access rather than data exchange and the CAD model represented in the container is still using the data structures of the server application. 2) Associativity. The 3D model in the

downstream application is always kept current with design changes in the CAD model. Any changes made in the server can be immediately reflected in any container using the model. 3) Flexibility. A container may connect to any number of servers so that a collection of related geometry from different systems can be displayed in the container's window. 4) Best in class applications. OLE for D&M gives end users the freedom to integrate best in class software tools to meet their engineering needs.

### 3.5 N-Tier Thin-Client Server Architecture

The Internet introduced the concept of linking powerful, state-of-the-art servers (that can be located anywhere in the world) to almost any type of computer device over a basically uncontrolled network environment. Through this medium, users can access graphics, text, audio and video files in a timely and inexpensive manner.

One key to the Internet's success was the development of a "thin-client" architecture built around the Web browser (using HTML scripting language) and the HTTP protocol. Browser software itself is a simple application created to run on almost any computer, thus the term thin client. The browser simply displays the screen interpretation of the HTML code in the document, otherwise it does no computing.

The computing is actually done on a Web server located somewhere on the Internet. In turn, this server accesses database and application servers using a protocol such as the Common Gateway Interface (CGI). Because three computers are involved in the process (the client workstation, the Web server and the database server), the architecture is known as three-tiered architecture.

Such an architecture basically enables complex, memory-intensive applications to be created between the Web and database servers while the client software (the browser) is a small, universally available, standard program.

In retrospect, the concept of thin-client computing is similar to the terminal/host computing model. The architecture offers the flexibility to use a simple terminal device in lieu of a high-end PC loaded with complex processors, drives, memory and numerous software applications.

First generation systems were a 2-tiered architecture where a client presents a GUI interface to the user, and uses the user's data entry and actions to perform requests of a database server running on a different machine. In the latter, application logic is typically tied to the client application and a 'heavy' network process is required to mediate the client-server interaction. A new generation of client-server implementations takes this a step further and adds a middle tier to achieve a '3-tier' architecture. Generally, client-server can be implemented in an 'N-tier' architecture where application logic is partitioned. This leads to faster network communications, greater reliability, and greater overall performance.

Today there are basically four types of thin-client architectures:

1. X-Windows system — for many years used mainly in UNIX environments to remotely display graphical screen information from central servers.
2. Citrix Winframe/Microsoft system — uniquely designed to run Windows® applications from a modified Windows NT® server platform using the Intelligent Console Architecture client.
3. Java Virtual Machine — a relatively new architecture that launches small applets written in JavaScript from servers to be run on the local device (personal or network computer).
4. Web browsers — uses the Internet concept of linking an HTML Web server to a client browser.

### 3.6 Unified Modeling Language

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time-to-market. These techniques include component technology, visual programming, patterns and frameworks. Businesses also seek techniques to manage the complexity of systems as they increase in scope and scale. In particular, they recognize the need to solve recurring architectural problems, such as physical distribution, concurrency, replication, security, load balancing and fault tolerance. Additionally, the development for the World Wide Web, while making some things simpler, has exacerbated these architectural problems. The Unified Modeling Language (UML) was designed to respond to these needs.

“In a relatively short period of time, the Unified Modeling Language has emerged as the software industry’s dominant modeling language. UML is not only a de facto modeling language standard; it is fast becoming a de jure standard.” (Kobryn, 1999) “The Unified Modeling Language (UML) is a language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of the best engineering practices that have proven successful in the modeling of large and complex systems.” (OMG UML Specification, 1999)

#### 3.6.1 Development Project Artifacts

The choice of what models and diagrams one creates has a profound influence upon how a problem is attacked and how a corresponding solution is shaped. Abstraction, the focus on relevant details while ignoring others, is a key to learning and communicating. Because of this:

- Every complex system is best approached through a small set of nearly independent views of a model. No single view is sufficient.
- Every model may be expressed at different levels of fidelity.

- The best models are connected to reality.

In terms of the views of a model, the UML defines the following graphical diagrams:

- Use case diagram
- Class diagram
- Behavior diagrams
- Statechart diagram
- Activity diagram
- Interaction diagrams
- Sequence diagram
- Collaboration diagram
- Implementation diagrams
- Component diagram
- Deployment diagram

Although other names are sometimes given to these diagrams, this list constitutes the canonical diagram names. These diagrams provide multiple perspectives of the system under analysis or development.

The underlying model integrates these perspectives so that a self-consistent system can be analyzed and built. These diagrams, along with supporting documentation, are the primary artifacts that a modeler sees, although the UML and supporting tools will provide for a number of derivative views.

### 3.6.2 Goals of the UML

The primary design goals of the UML are as follows:

- Provide users with a ready-to-use, expressive visual modeling language to develop and exchange meaningful models.
- Furnish extensibility and specialization mechanisms to extend the core concepts.
- Support specifications that are independent of particular programming languages and development processes.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of the object tools market.
- Support higher-level development concepts such as components, collaborations, frameworks and patterns.
- Integrate best practices.

These goals are discussed in detail in (OMG UML Specification, 1999).

Because the UML authors were in effect designing a language (albeit a graphical one), they had to strike a proper balance between minimalism (everything is text and boxes) and over-engineering (having an icon for every conceivable modeling element). To that end, they were very careful about adding new things, because they didn't want to make the UML unnecessarily complex. Along the way, however, some things were found that were advantageous to add because they have proven useful in practice in other modeling. There are several new concepts that are included in UML, including

- Extensibility mechanisms (stereotypes, tagged values, and constraints),
- Threads and processes,
- Distribution and concurrency (e.g., for modeling ActiveX/DCOM and CORBA),
- Patterns/collaborations,
- Activity diagrams (for business process modeling),
- Refinement (to handle relationships between levels of abstraction),
- Interfaces and components, and
- A constraint language.

Many of these ideas were present in various individual methods and theories but UML brings them together into a coherent whole.

### Use Cases

The Use Cases package is a subpackage of the Behavioral Elements package. It specifies the concepts used for definition of the functionality of an entity like a system. The package uses constructs defined in the Foundation package of UML as well as in the Common Behavior package. The elements in the Use Cases package are primarily used to define the behavior of an entity, like a system or a subsystem, without specifying its internal structure. The key elements in this package are UseCase and Actor. Instances of use cases and instances of actors interact when the services of the entity are used. How a use case is realized in terms of cooperating objects, defined by classes inside the entity, can be specified with a Collaboration. A use case of an entity may be refined to a set of use cases of the elements contained in the entity. How these subordinate use cases interact can also be expressed in a Collaboration.

### Class

A class is the descriptor for a set of objects with similar structure, behavior, and relationships. The model is concerned with describing the intension of the class, that is, the rules that define it. The run-time execution provides its extension, that is, its instances. UML provides notation for declaring classes and specifying their properties, as well as using classes in various ways. Some modeling elements that are similar in form to classes (such as interfaces, signals, or utilities) are notated using keywords on class symbols: some of these are separate metamodel classes and some are stereotypes of Class. Classes are declared in class diagrams and used in most other diagrams. UML provides a graphical notation for



declaring and using classes, as well as a textual notation for referencing classes within the descriptions of other model elements.

### Class Diagram

As defined by (OMG UML Specifications, 1999), a class diagram is a graph of Classifier elements connected by their various static relationships. Note that a “class” diagram may also contain interfaces, packages, relationships, and even instances, such as objects and links. Perhaps a better name would be “static structural diagram” but “class diagram” is shorter and well established.

Although InnerCircle2000™ does not use class diagrams in the same way that they are drawn by the UML standard, many of the concepts involved with the diagrams are still valid. InnerCircle2000 uses a class tree which allows classes, instances, properties, and values to be laid out in a manner which is easy to view and manipulate.

InnerCircle2000™ uses a variety of different objects from which it creates models. There are classes and properties. Classes are the theoretical example or definition of an object that will be modeled. Classes have properties which are attributes, associations, other classes, or subtype that help describe characteristics of an object. Once a class and its properties have been defined, it is ready to be used as a part in an InnerCircle2000™ model. The class can then be modeled as an instance, a specific object with the properties of its class, and have its values defined. Values are equivalent to what properties are for a class. Values are specific instances of a property, and while representing the same characteristic, may have different actual values.

## 3.7 Optimization and Decision Making

Decision-making is a key activity in all organizations. A large portion of management time is spent in deciding what activities should be carried out, when and how they should be implemented, and how to improve the performance of the facility. Decision-making process usually involves four main phases: intelligence, design, choice, and review. In any decision or design process, one attempts to make the best decision within a specified set of possible ones. The notion “best” refers to the decision that either optimize a single criterion, or the decision that simultaneously optimize several criteria.

Although it is difficult to be applied in the real world, optimization would seem to be the ideal decision-making process. Optimization involves selecting the course of action (alternative) with the highest utility or level of satisfaction to the decision-maker. This requires estimating the comparative value of every viable alternative in terms of anticipated benefits and costs. The optimization process goes through four main phases:

1. Model building. Within which we identify

- Some objective (qualitative measure of performance).
  - The variables which affects this objective, and
  - The constraints that restrict the variables.
2. Solution. Once the model has been formulated, the appropriate optimization algorithm can be used to find the solution.
  3. Check the optimality of the solution.
  4. Improve the solution by applying techniques such as sensitivity analysis.

Many ways for finding the optimal solution exist, such as mathematical programming (mathematical optimization), simulation, heuristic algorithms, artificial intelligence, and finally meta-heuristic algorithms.

### 3.7.1 Mathematical Programming (Optimization)

Mathematical programming is the formal title given to the branch of computational science that seeks to answer the question 'what is best?' for problems in which the quality of any answer can be expressed quantitatively.

The goal of an optimization problem can be formulated as follows: to find the combination of some independent variables (decision variables) which optimize (either maximize or minimize) a given quantity (known as the objective function), possibly subject to some restrictions on the allowed variable ranges (constraints). Figure (23) represents an unconstrained optimization problem, while Fig. 24 represents the constrained type.

The general optimization problem may be stated mathematically as:

$$\begin{array}{lll}
 \text{Minimize} & f(X), & X = (x_1, x_2, \dots, x_n)^T \\
 \text{Subject to} & c_i(x) = 0, & i = 1, 2, 3, \dots, m' \\
 & c_i(x) \geq 0, & i = m'+1, \dots, m
 \end{array}$$

Where  $f(X)$  is the objective function,  $X$  is the column vector of the  $n$  independent variables, and  $c_i(x)$  is the set of constraint functions.

There are many optimization algorithms available. Many methods are appropriate only for certain types of problems. Within each class of problems there are different minimization methods, varying in computational requirements, convergence properties, and so on. Optimization problems are classified according to the mathematical characteristics of the objective function, the constraints, and the decision variables.

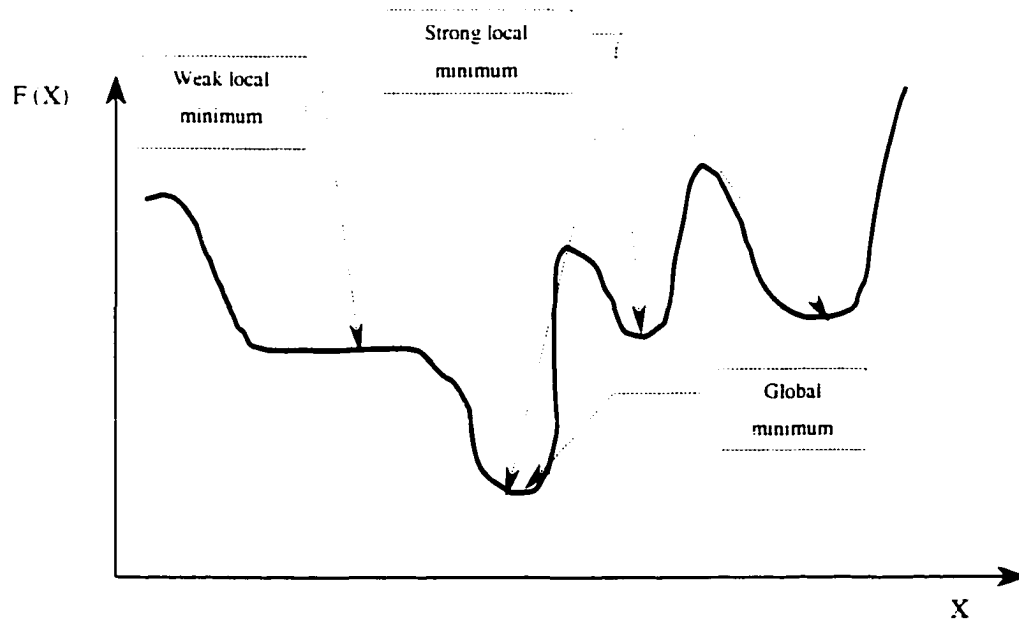


Figure 23. Types of minimum for unconstrained optimization problems (schematic)

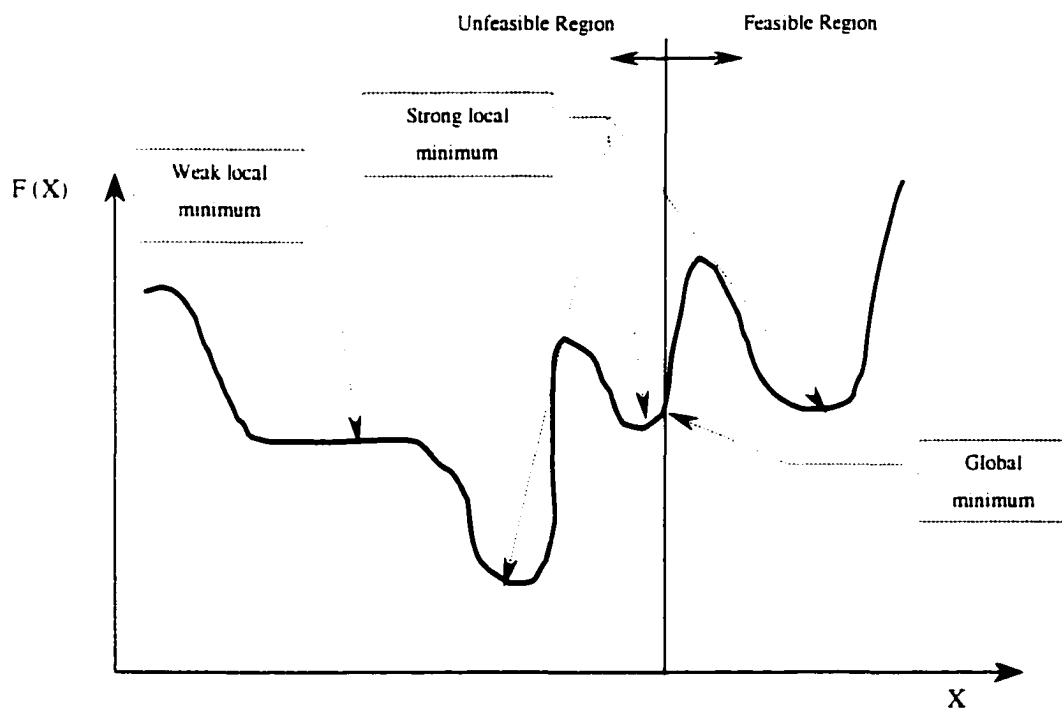


Figure 24. Types of minimum for constrained optimization problems (schematic)

There are a special class of problems, examples of which are particularly common in the fields of operations research and engineering design, in which the task is to find the optimum permutation of some decision variables. These are known as combinatorial optimization problems. The solutions to such problems are usually presented as ordered list of integers, and they are, of course, constrained, since not all integer lists present valid solutions. These and other classifications are summarized in Table 2.

Table 2. Optimization Problems Classification

Characteristic	Property	Classification
Number of Decision Variables	One	Univariate
	More than one	Multivariate
Type of Decision Variables	Continuous real numbers	Continuous
	Integers	Integer
	Both continuous real numbers and integers	Mixed integer
	Integers in permutations	Combinatorial
Problem Functions (Objective Function and Constraints)	Linear	Linear
	Quadratic	Quadratic
	Nonlinear	Nonlinear
Problem Formulation	Subject to constraints	Constrained
	Not subject to constraints	unconstrained

### 3.7.2 Linear Programming

A linear programming problem is a special case of mathematical programming problems, in which the following conditions are met:

1. The decision variables are non negative.
2. The objective function is linear in terms of the decision variables, and
3. Constraints imposed are either linear equations or linear inequalities or a combination of them.

“Linear programming is a quite young and yet very active branch of applied mathematics.” (Fang and Puthenpura, 1992) Linear programming techniques are widely used to solve a large number of economics, industrial, managerial, and military problems. “In a recent study of Fortune 500 companies, 85% of those responding said that they had used linear programming.” (Fang and Puthenpura, 1992)

Three primary reason lie behind this wide spread applications of linear programming;

1. The large variety of problems in diverse disciplines that can be represented (either directly or approximately) as linear programming problem.
2. Many efficient techniques for solving linear programming problems are available, and
3. The ease through which data variation (sensitivity analysis) can be handled in a linear programming model.

Dantzig's development of the simplex method in the late 1940's marked the start of the modern era in optimization. This method made it easy to handle large linear programming models in a systematical and efficient way.

A few years ago, the main disadvantage of linear programming was that its procedures are iterative in nature, which results in highly time and effort consuming. But with the recent breakthrough in computer technology, the solution of large linear programming problems has not only become feasible but inexpensive and very fast as well. "Today, linear programming and the simplex method continue to hold sway as the most widely used of all optimization tools." (Nocedal, and Wright, 1999)

In order to represent an optimization problem as a linear programming problem, implicitly the following assumptions are made:

1. Proportionality. For each decision variable, contribution to the objective function and to each constraint is directly proportional to its value.
2. Additivity. The contribution to the objective function or any constraint of any decision variable is independent of the values of other decision variables.
3. Divisibility. Non-integer values for the decision variables are permitted.
4. Certainty. No probabilistic or stochastic element is involved in a linear programming problem.

### 3.8 Spreadsheets

#### 3.8.1 Spreadsheets in Engineering

A spreadsheet is "an application program for which the user supplies data and instructions in the form of commands and formulas to make the desired computations." (Kral, 1992) The tabular organization of data with several implicit computational procedures applies to many disciplines, including engineering. Since the developer is constrained by displaying information in a tabular form, the problem must be formulated in such a way that both data structure and its computational methodology conform with spreadsheet characteristics.

Although initially applied to business applications, the uses of spreadsheets in engineering is expanding. Various applications of the program have been discussed in the literature.

(Banks, 1986a; Banks, 1986b; Coleman, 1987; Diaz and Lezman, 1988; Fargher, 1987; Fields, 1986; Hong and Maleyeff, 1987; Kennedy and Martinez, 1987; Kuo and Folkers, 1986; Lue, 1987; Martin, 1986; Masri and Moodie, 1985; Oden, 1986; Oden, 1988; Rickles and Elliot, 1985; Sitton, 1989; Sounderpandian, 1989; Trevino and McGinnis, 1986) discuss issues concerning spreadsheet applications in production planning and process improvement.

Cost and cost analysis issues were discussed in (Chong and Veress, 1988; Earnest, 1987; Graff, 1989; Hamilton and Rothe, 1985)

A relatively small number of spreadsheet implementations in optimization and decision making in (Evans, 1986; Godin, 1987; Parlar, 1986), in simulation (Godin and Rao, 1987), and in quality control (Liu, 1989; Zimmerman and Gibson, 1989)

The applications of spreadsheet in manufacturing were discussed in (De Lurgio and Zhao, 1989; Dix, 1985; Evans, 1986; Fields, 1986; Kleinfeld, 1984; Whitehouse, 1984)

### 3.8.2 Microsoft Excel

Although several spreadsheet programs are available, Microsoft Excel, or MS Excel, seems to have the greatest market share among these programs. In fact, many reasons lie behind this domination of MS Excel, but what is of our interest here is "why MS Excel for engineering applications?"

In fact, MS Excel has many features that make it the most suitable spreadsheet program for engineering environment:

- 1- It's great range of built-in mathematical and scientific functions.
- 2- It's capability for hierarchical modularity with provision for user-defined modules.
- 3- The availability of common data types to engineering computations.
- 4- it's powerful graphing facility.
- 5- The availability of logical statements and decisions, and
- 6- It's capability to communicate with other applications within MS Windows environment.

On top of all these features, many companies have developed add-ins that seamlessly add evermore functionality to MS Excel. In particular, we have an optimization add-in for MS Excel, that is, "What's Best! 4"

### 3.8.3 "What's Best! 4"

"What's Best!" is an Excel add-in developed by Lindo Systems Inc.. "What's Best!" is a powerful constrained-optimization feature that can calculate solution to what-if scenarios based on adjustable

decision variable cells, constraint cells, and, optionally, cells that must be optimized. The basic purpose of the "What's Best!" is to find a solution –that is, values for the decision variables or changing cell in the model- which satisfies the constraints and which maximizes or minimizes the objective or target cell value.

Professional versions of "What's Best!" are able to solve large LP models of up to 8,000 variables and 4,000 constraints in a matter of seconds by setting up a spreadsheet, while the largest version of "What's Best!" has no capacity limit.

The mathematical form of the relationship between the objective function and the decision variables has important implications for the difficulty of the problem and the speed of solution. "What's Best!" supports linear, non-linear, and quadratic functions. "What's Best!" applies a state-of-the-art Primal and Dual Simplex solvers for linear programming models. The linear solver algorithms have numerous enhancements for maximum speed and robustness. Preprocessing routines include scaling procedures that can improve speed and robustness on numerically difficult models. Using reduction techniques, "What's Best!" can often make models solve faster by analyzing the original formulation and mathematically condensing it into a smaller problem. For nonlinear programming models, the primary underlying technique used by "What's Best's!" optional nonlinear solver is based upon a Generalized Reduction Gradient (GRG) algorithm. The nonlinear solver takes advantage of sparsity for improved speed and more efficient memory usage. It also offers a variety of algorithmic options including a Crash procedure, a Steepest Edge/Steepest Decent option, and Sequential Linear Programming procedures. "What's Best!" automatically selects the solution approach that appears best suited to the specific model at hand. The solution approach is dynamically adjusted during the solution process based upon the model's behavior. For models with general and binary integer restrictions, "What's Best!" includes an integer solver that works in conjunction with the linear and nonlinear solvers. For linear models, the integer solver does extensive preprocessing and adds constraint "cuts" of several different varieties to greatly improve solution times on large classes of integer models.

Moreover, intelligent integration appears in "What's Best!," that it handles the details of the solution process allowing user to focus on modeling. When the Solve command is initiated, "What's Best!" analyzes the problem and, when possible, reduces the problem and even substitutes out variables. With "What's Best!," the user never have to specify whether to use the linear or nonlinear solver. Based upon the model's structure, "What's Best!" automatically selects the appropriate solver and intelligently adjusts internal parameters.

### 3.9 Project Management

A project can be defined as "a group of tasks performed in a definable time period in order to meet a specific set of objectives." (Badiru, and Whitehouse, 1989) Or as defined in (Gido & Clements, 1999) "an endeavor to accomplish a specific objective through a unique set of interrelated tasks and the effective

utilization of resources.” As seen from these definitions, a project exhibits most of the following conditions:

1. It is likely to be a unique, one-time program.
2. A project has a customer. The customer is the entity that provides the funds necessary to accomplish the project.
3. It has a well-defined objective stated in terms of scope, schedule, and cost.
4. It has a specific time frame, a life cycle or finite life span. It has a start time and a date by which the objective must be accomplished.
5. A project is carried out through a series of independent tasks- that is, a number of nonrepetitive tasks that need to be accomplished in a certain sequence in order to achieve the project objective.
6. A project utilizes various resources to carry out the tasks.
7. It has a budget.
8. A project involves a degree of uncertainty.

The management of a project is quite different from the management of a continuing operation. The generally accepted definition of management is “the planning, organizing, directing, and controlling of company resources to meet the company’s financial and non-financial objectives.” Project management, on the other hand, can be defined as “the application of knowledge, skills, tools, and techniques to project activities in order to meet or exceed stakeholder needs and expectations from a project.” (PMI Standards) Or as in (Rabun and Sommers, 1998) “the planning, organizing, directing, and controlling of resources for a specific time period to meet a specific set of one-time objectives.” Or in another way, “the process of managing, allocating, and timing resources in order to achieve a given objective in an expedient manner.” (Badiru and Whitehouse, 1989). Meeting the project objective(s) involves compromising competing demands on: scope, time, cost, and quality.

From these definitions, two major differences appear between the two kinds of management. The first difference is that; in project management the manager is not directly responsible for staffing and must use and direct resources from other components or companies. The second one is that project management concerns about “specific time period” to meet “one-time objectives”.

Project management involves a process of first establishing a plan and then implementing that plan to accomplish the project objective. Once the project starts, the project management process involves monitoring progress to ensure that everything is going according to plan. The ultimate benefit of implementing project management techniques is having a satisfied customer-whether the customer is the project owner or a business (contractor) being paid by a customer to perform the project. Completing the full scope of work of the project in a quality manner, on time, and within budget provides a great feeling of satisfaction.



The project management process means planning the work and then working the plan. This planning effort includes the following steps:

1. Clearly defining the project objective.
2. Dividing and subdividing the scope into major work packages. A work breakdown structure (WBS) is a hierarchical tree of work elements or items accomplished or produced by the project team during the project.
3. Defining the specific activities that need to be performed for each work package in order to accomplish the project objective.
4. Graphically portraying the activities in the form of a network diagram. This diagram shows the necessary sequence and interdependencies of activities to achieve the project objective.
5. Making a time estimate for how long will it take to complete each activity. It is also necessary to determine which types of resources and how many of each resource are needed for each activity to be completed within the estimated duration.
6. Making a cost estimate for each activity. The cost is base on the types and quantities of resources required for each activity.
7. Calculating a project schedule and budget to determine whether the project can be completed within the required time, with the allotted funds, and with the available resources.

#### Traditional project management models

The following project management models have influenced the development of project management software:

1. Critical Path Method (CPM). A mathematical model that calculates the total duration of a project based on individual task durations and dependencies, and identifies which tasks are critical. This model is the fundamental scheduling method used in project management softwre today including Microsoft Project.
2. Program Evaluation Review Technique (PERT). Uses statistical probabilities to calculate expected durations.
3. Gantt chart. A way to graphically represent activities across a time scale.

A large number of procedures and techniques have been developed to enhance the application of project management to a wide range of application in either industrial or administrative environments. The main reason that lies behind this rapid spreading of project management is the availability of computer software packages that make it easily to implement project management techniques efficiently in short time. Computers are used as a tool to facilitate the implementation of proved mathematical/heuristic techniques. The primary advantage of computers here is the speed at which it will perform the accurate quantitative analysis needed to develop schedules and generate a variety of outputs and reports.

### 3.9.1 Microsoft Project

Today's organizations are often not as rigidly hierarchical as they were in the past, and thus information, tasks, and responsibilities are increasingly dispersed within and among various workgroups. Project management has become vital to the functioning of organizations large and small. Microsoft Project 98 provides powerful tools to build customized planning solutions, and has the flexibility to appeal to a range of user types, from novice planners to expert project management professionals.

Microsoft Project 98 is the fifth Microsoft Windows®-based release of what has grown to be the world's most popular project management software. The Microsoft Project installed base, which is more than 2 million users worldwide, comprises a wide variety of people, ranging from novice planners to expert project management professionals. Microsoft Project 98 is the result of an enormous research and development effort focused on this entire spectrum of planners, and offers major feature additions and enhancements for anyone with a planning need.

MS project helps manage projects in a variety of ways, including:

- Ease of use
- Scheduling
- Tracking
- Reporting
- Workgroup
- Import/export
- Customization

### 3.9.2 Project Management in Manufacturing

“Manufacturing is the act of making something through deliberate processing from raw material to the desired object, usually with the use of machinery.” (Badiru, 1996) This act encompasses several functions that must be strategically planned, organized, scheduled, controlled, and terminated. A manufacturing cycle includes, but not limited to, such functions as forecasting, decision analysis, cost analysis, inventory control, process planning, machine scheduling, quality control, production planning, process control, work and time analysis, and a host of others. These are all functions that fall within the planning organizing, scheduling, and control functions of project management.

*Manufacturing System Definition.*

A system is defined as a collection of interrelated elements brought together to achieve a specified objective. In a managerial context, the purposes of a system are to develop and manage operational

procedures and to facilitate effective decision-making process. A system approach is particularly essential for manufacturing and high tech endeavors because of the various factors are expected to interact.

Manufacturing professionals have to serve as systems integrators. One of their primary responsibilities involves ensuring the proper flow of information to control manufacturing tasks. The classical approach to decision process follows rigid lines of organizational charts. By contrast, the systems approach consider all interactions necessary between the various elements of an organization

The various elements (or subsystems) of the system act concurrently in a separate but interrelated fashion to achieve a common goal. This synergism helps to expedite the decision process and to enhance the effectiveness of the decisions. The overall effectiveness of the system is expected to be greater than the sum of the disjoint efforts of the subsystems, since the supporting commitments from some subsystems serve to counterbalance the weaknesses of each other.

“The increasing complexity of organizations and projects makes the systems approach essential in today’s management environment. As the number of complex projects increases, there will be an increasing need for project management professionals who can function as systems integrators.”(Badiru, 1996) Figure (25) shows different manufacturing interface component with project management.

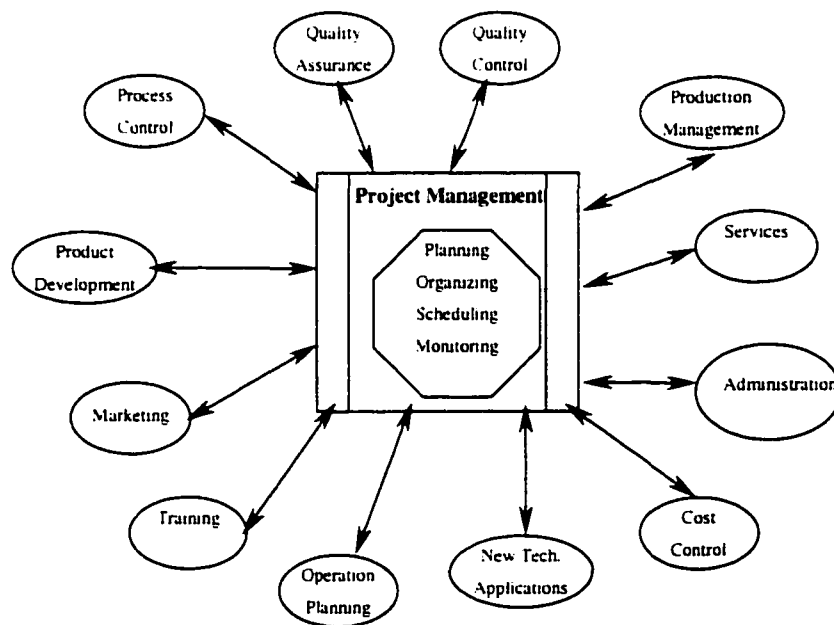


Figure 25. Project Management as a Means to Integrate Various Manufacturing Activities.

In (Ambrosy, et. al., 1996), the conception of an integrated product and process model is introduced which is particularly suitable for areas of project management, design and assembly planning. In order to assist *simultaneous engineering in product development*, the various computer tools employed have to gain access to a common database. All product specific data are stored in a model of the product. The representation of dynamic sequences requires a process model. In (Beghini and Romanin, 1991), an integration among the solutions of the problems concerning project planning, material purchasing and information exchange, in a firm working by orders, is studied. A new discipline which faces the project evolution, starting from the feasibility study up to the production delivery is 'concurrent engineering; the work reported was developed by following its principles. (Brown, 1984) applies Project management for the design and supply of power station mechanical and electrical plant.

New product development best practice models advocate the integration of teamwork, simultaneous engineering, tools and techniques, a front loaded process and project management. The design of the model is relatively straightforward compared with implementation which is significantly more difficult. An approach that has lead to some implementation success is the pilot project approach where a carefully selected product development project is used as a vehicle for good practice demonstration and learning. (Boznak, 1988) discusses the role that project management could play in employing a company strategies to reduce new product development time. (Brown, 1995) describes alternative implementation scenarios when using this approach. In (Churchill, 1988), the principles behind quality assurance as an effective strategy for management of large-scale capital projects are discussed. "Fitness-for-purpose" objectives are analyzed, and the need to minimize economic and other consequences of failure is shown to be essential as a means of obtaining customer confidence. It is of vital importance for the manufacturing industry to respond to the requirements of the market in a flexible, cost favourable and above all quick way. (Feldermann, 1993) describes the setup and the concept of an effective project planning and control in the manufacturing area. (Maio, et. al., 1994) proposes an interpretative model that explains firms' dynamic behavior in multi-project management of new product development. The model could be used as a unique and homogeneous framework supporting the processes of project selection, resource allocation, risk management, priority management and ongoing control. (Larson and Gobeli, 1988) assessed the relative effectiveness of five different project management structures by comparing the performance of 540 development projects in terms of cost, schedule, and technical performance. (Ryba and Baitinger, 1996) presents a novel approach supporting administrative tasks within the lifecycle of design projects. Based upon comprehensive models of design environments and design activities it combines techniques known from project management and mechanisms for design flow control.

Project management is characterized by qualified collaborators and by suitable planning and controlling methods. The strong point of the management concept for innovation projects lies in the formulation of the task and in the clear representation of the project situation. For coordination projects, the strong point for the project manager is the setup and care of the project information system. The management of an organization project must introduce methods and procedures for the planning and the

supervision of tasks, time and costs. (Heuer, 1976) discusses the applications of project management in mechanical engineering, planning and controlling of industrial intentions. The failure of some newly installed manufacturing systems to live up to their pre-installation expectations has been blamed on a number of factors. One overriding factor is poor project planning. (Osullivan, 1991) discusses one approach to project planning that uses the structured methodology IDEF/sup 0/ to model the development process and perform many project-planning activities that are normally left out when traditional planning techniques are used. IDEF/sup 0/ is a graphical modeling methodology that was developed for systems design and is now used widely in manufacturing for activity modeling.

There is a dramatic rise in the use of project management as organization shift to provide customer-driven results and systems solutions. (Englund and Graham, 1999) reviews actions that upper managers can take to create an environment for more successful projects in their organizations.

### 3.10 Project Crashing and Time-Cost Trade-Off

In addition to scheduling projects, the project managers are frequently confronted with the problem of having to reduce the scheduled completion time (indicated by the CPM or PERT network analysis) to meet a pre-specified deadline.

Project duration reduction can be achieved by assigning more resources (labor, material, equipment, etc.). however, additional resources cost money, and hence increase the overall project cost. Thus, the decision to reduce the project duration, and by how much, must be based on an analysis of the trade-off between time and extra cost added.

"project crashing is a method for shortening the project duration by reducing the time of one or more of the critical project activities to a time that is less than the normal activity time." (Taylor, 1996)

#### 3.10.1 Project Cost Models

Network analysis can be extended to incorporate cost explicitly, thus integrating the planning and control aspects of project management with the financial and budgeting activities. This is done via defining some cost model representing the activity time-cost relationship. The simplest cost models are extensions of the basic CPM calculations. Each activity has, on one hand, normal duration and an associated normal cost and, on the other hand, a crash duration and an associated crash cost.

As discussed in (Chapman et. al., 1987), the activity time-cost relationship may take one of several shapes, as shown in Fig. 26:

- (a) It may be concave,

- (b) The precise shape is difficult to specify, in this case we assume linear relationship, or
- (c) Piecewise linear approximation to more general functions.

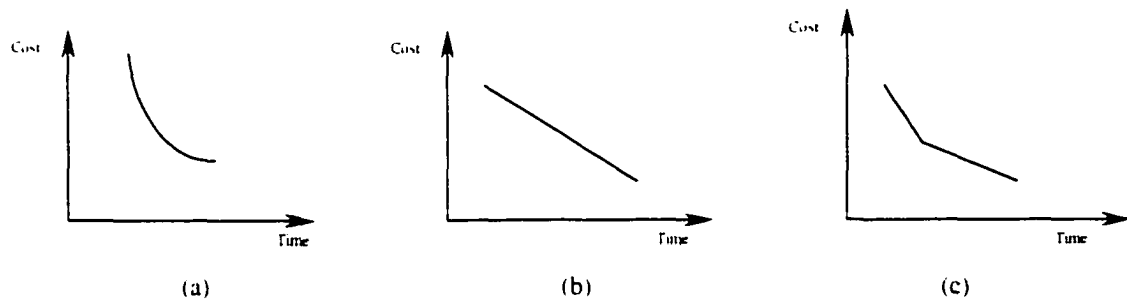


Figure 26. Different Cost Models

### 3.10.2 Enumerative Method for Project Crashing

For small projects, the optimal project schedule could be obtained by this method. The basic idea behind this method is that the project length can be reduced by reducing the duration of the critical path. Hence, the critical activities are crashed as long as we are still within the allowable crash time for the activity. The main difficulty with this method is that the critical path of the projects may change once we start crashing the critical activities. So, in each step, we have to crash a critical activity only by one unit time and check whether there is a new critical path or not before continuing to the required crash time. Furthermore, when applying this method for large projects, we are faced with many parallel critical paths, and the critical path may have a large number of activities. Examining all possible combinations of the activities on the parallel paths by the enumerative method will not only be inefficient but also expensive and time consuming.

The steps of this method could be summarized as follows:

1. Find the normal critical path and identify the critical activities.
2. Compute the crash cost per unit time for all critical activities.
3. Select the activity on the critical path with the smallest crash cost per unit time. Crash this activity by a unit time.
4. Check that the critical path is still the same. If it is, then re-crash the same activity by another unit time making sure that you don't exceed the allowable crash time for the activity.
5. If the critical path change, repeat from step (2).

### 3.10.3 Project Crashing with Linear Programming

The project management problem is to determine the amount by which the various jobs are to be crashed which will minimize the total cost of the project. This problem has been solved by several methods during the last two decades. In the beginning, people used an enumerative method, by which the project length can be reduced by reducing the duration of the critical activities. Although it is direct, several difficulties were faced when using such a method; 1) the critical path of the project changes once we start crashing the critical activities, 2) inapplicable to large projects, and 3) time consuming.

In order to handle these difficulties, people started using mathematical programming methods (Linear programming, Integer programming, and Nonlinear programming) which proved to be more efficient in determining the optimal project schedule. But still, in large projects, mathematical programming seemed to be time-consuming methods and may be inapplicable.

The last stage was using heuristic algorithms and artificial intelligence. These methods although faster, usually do not give an optimal solution (give a near optimal solution).

Nowadays, the computing world has witnessed a new generation of PCs with very high computational speed, beside the emerging of new, capable optimization software. Using these software enables us to handle time-cost trade-off problem in large projects as a mathematical programming problem and then solve it. So, we can combine both optimality and speed.

As discussed in (Render and Stair, 1991; Taylor, 1996; Phillips et. al., 1976; Williams, 1999), both CPM/PERT network and project crashing network can be formulated as a linear programming problem; "to minimize the cost of crashing given the limits on how much individual activities can be crashed."

### 3.10.4 Time-Cost Trade-Off Problem in Literature

The time-cost trade-off problem is the subject of a number of research articles. (Deckro and Hebert, 1989) developed models for two specific resource constrained project crashing cases; 1. A model for resource critical crashing case, and 2. A model for the activity duration crashing case. (Rosenblatt and Roll, 1985) analyzed optimal project duration for situations where project duration can be shortened by 'crashing' activities. The cost components considered are: regular direct costs, crashing costs and overhead costs. (Tufekci, 1982) introduces an iterative solution procedure for solving the time-cost trade-off problem that utilizes a labeling algorithm for locating a minimal cut in the flow network. In (Law and Hsing-Wei, 1987), two predictive models for estimating the computer execution time required by two network flow based algorithms to solve the time-cost trade-off problem are presented. (Nair et. al., 1993) considers a network in which each arc is associated with a time-cost trade-off function. This function is assumed to be non-increasing, piece-wise linear and convex and objected to enumerate all efficient chains in the context

of two criteria, the total time and the total cost required to traverse from source node to sink node. (Skutella, 1998) presents approximation algorithm for the discrete time-cost trade-off problem.

The importance of the time-cost trade-off problem arises from the wide range of its application involvement's. (Batson, 1987) discusses an implementation of a time-cost trade-off algorithm in aircraft technology development projects. (Reda and Carr, 1989) handled the problem among related activities. (De et. al., 1997) addresses the discrete version of the well-known time-cost trade-off problem for project networks, and discusses the complexities of various network structures and validate an old conjecture that certain structures are necessarily more difficult to solve. (de Reyck and Herroelen, 1996) Investigate the relation between the hardness of a problem instance and the topological structure of its underlying network, as measured by the complexity index. It also demonstrates that the complexity index plays an important role in predicting the computing effort needed to solve easy and hard instances of the multiple resource-constrained project scheduling problem and the discrete time/cost trade-off problem. (Haffner and Graves, 1988) uses the time-cost trade-off to maintain the planned market entry of a product. (Hajdu, 1996) deal with some special problems concerning least cost scheduling problem in precedence diagramming. And give an algorithm for the PDM/cost problem using the assumption that both splitting and non-splitting of activities are allowed. (Icmeli et. al. 1993) performed a survey of project scheduling problems since 1973 limited to work done specifically in the project scheduling area. The survey includes the work done on several fundamental problems such as the time/cost trade-off problem. (Gander, 1985) introduced . different forms of government involvement in the innovation process, both direct and indirect, into a standard innovation time-cost trade-off model. (Graves, 1987) presents a brief review of the key concept of a convex time-cost trade-off, which by assumption forms the basis for both static and dynamic models of research and development investment expenditure. (Levy and Tayl, 1989) considered a cost-minimization model to investigate scheduling strategies for multistage projects in a client-contractor environment. The model is designed primarily to address the interaction between earliest-, intermediate-, and latest-start options and project-crashing strategies for a broad range of penalty costs.

The time-cost trade-off problem has been attacked by several methods, we can classify them as follow:

#### 1) Mathematical Programming

(Chassiakos et. al., 1998) Presents an algorithm, that employs an integer programming formulation, for obtaining the optimal solution for the time-cost trade-off problem in large projects. (Demeulemeester et. al., 1996) Describes two algorithms, base on dynamic programming logic, for optimally solving the discrete time-cost trade-off problem in deterministic CPM networks. In (Demeulemeester et. al., 1998), an optimal solution for the discrete time-cost trade-off problem in deterministic networks is derived using a branch-and-bound procedure. (Erenguc et. al., 1993) determines the activity durations and a schedule of activity start times so that the net present value of cash flows is maximized in a project scheduling problem. The problem was formulated as a mixed-integer nonlinear problem. (Hamacher and Tufekci, 1983) Introduces a multiobjective project crashing model where the problem is formulated as a lexicographical optimization



model. An efficient lexicographical maximal flow algorithm is implemented to obtain the lexicographical minimal cuts at each step to determine the activities to be modified. In (Kanda and Rao, 1984), a procedure is developed to obtain the project-cost curve when there are linear penalty costs for delays of certain key events in a project in addition to crashing costs for activities. A linear programming formulation is given. (Liu et. al., 1995) presents an algorithm to assist construction planners in making time-cost trade-off decisions is presented. This approach, called the LP/IP hybrid method, takes advantage of linear programming and the convex hull method for efficiency, and integer programming to find the precise solutions. This hybrid method, along with a spreadsheet tool, provides the construction planner with an efficient means to obtain resource selections that optimize time and cost of a construction project. (Shouman et. al., 1991) presents a model that involves a mixed integer linear programming formulation to determine the optimum allocation of the project duration reduction. The main advantage of this model is its ability to determine the optimum allocation among activities for four different time/cost functions. The functions are straight line, multistage piecewise-linear, discrete points, and one point for dummies and incompressible activities.

### 2)Heuristic Algorithms

(Barber and Boardman, 1986)established the definition of an easy-to-use tool for project crashing problems with two key features: an algorithm to generate a range of increasingly pragmatic solutions by the inclusion of heuristics to portray real-world objectives; and an intelligent knowledge-based system to assist in the generation of strategies and to postulate the resultant time-cost trade-off function, for each activity considered. (Barber, 1989) Presents a prototype system which allows a project network to be portrayed graphically as a CPA network and then crashed using a heuristic algorithm with the aid of a knowledge based system.(Bowman, 1994) presents a heuristic using the gradient estimators to give close to locally optimal performance relatively quickly for PERT networks. In (Sunde and Lichtenberg, 1995), a new heuristic for cost-time trade-off which balances cost, time, and resources is presented. The new method is called net-present-value cost-time trade-off. In (Taeho and Erenguc, 1998), a combination of the time-cost trade-off problem and the resource constrained project scheduling problem is solved using a heuristic procedure, a multi-pass algorithm.

### 3)Simulation

In (Patrick and Topuz, 1995), a project-scheduling simulation model of the longwall move process was developed to analyze and assess the economic viability of innovative transfer methods and equipment. Longwall face-to-face equipment transfers or moves are the largest source of nonproductive time in a longwall-mining system. In (Ramani, 1986), a computer simulation project has been outlined to achieve optimal crashing of a PERT network, where a probabilistic PERT model is converted into a deterministic CPM model for the purpose of carrying out the time-cost trade-off analysis.

#### 4) Artificial Intelligence.

(Chishaki and Tatish, 1992) Demonstrates a new procedure for the time-cost trade-off problem. The procedure involves new assumptions and a fuzzy linear programming formulation. (Fan, et. al., 1997) presents an algorithm based on genetic algorithms principles for construction time-cost trade-off optimization, and a computer program that can execute the algorithm efficiently. (Li et. al., 1999) presents a computer system called Machine Learning and Genetic Algorithms based System (MLGAS). With MLGAS, quadratic time-cost curves are generated from historical data and used to formulate the objective function that can be solved by the genetic algorithm. The capacity of the GA was enhanced to prevent premature convergence. When compared with an experienced project manager, MLGAS generated better solutions to nonlinear time-cost trade-off problems.

## CHAPTER IV

### APPLICATION (CASE STUDY / DEMONSTRATION).

#### 4.1 Introduction

In addition to scheduling projects, project managers are frequently confronted with the problem of having to reduce the schedule completion time (indicated by the CPM network analysis) to meet a pre-specified deadline. This problem, project crashing or Time-Cost Trade-Off, has been solved by several methods, starting with enumerative methods and ending with artificial intelligence. In this chapter, mathematical programming formulations, to solve two different cases of this problem interactively through the presented framework, is being implemented. While the demonstrations are based on relatively small networks, the potential for application to huge projects will be the key finding of this chapter.

#### 4.2 Problem Statement

The Time-Cost Trade-Off problem aims at reducing the overall completion time of a project by 'crashing', i.e. reducing the time of a number of tasks in the project while holding the total cost of the project to a minimum. This problem has been solved by several methods over the last two decades. At the beginning, people used an enumerative method, by which the project length can be reduced by reducing the duration of the critical activities. Although it is direct, several difficulties were faced when using such a method: 1) the critical path of the project changes once we start crashing the critical activities, 2) inapplicable to large projects, and 3) time consuming.

In order to handle these difficulties, people started using mathematical programming methods (Linear programming, Integer programming, and Nonlinear programming) which proved to be more efficient in determining the optimal project schedule. But still, in large projects, mathematical programming remains time-consuming and may be inapplicable. The last attempt was to use heuristic algorithms and artificial intelligence. These methods although faster, usually do not give an optimal solution, but rather a near optimal solution only.

Nowadays, the computing world has witnessed a new generation of PCs with very high computational speed, beside the emerging of new, capable optimization software. Using these software enables us to handle Time-Cost Trade-Off problem in large projects as a mathematical programming problem and then solve it. So, we can combine both optimality and speed.

To illustrate the handling of this problem through the proposed integrated engineering-computation framework, two case studies of the mentioned problem are presented below that involve different set of assumptions.

### 4.3 Case Study I

This first case is the typical Time-Cost Trade-Off problem solved in the context of the proposed framework.

#### 4.3.1 Project Network

Figure 27 shows a network representing some project that involves several activities as indicated. Normal CPM project completion time = 36 time units, and the critical path is 1-2-3-4-6-7.

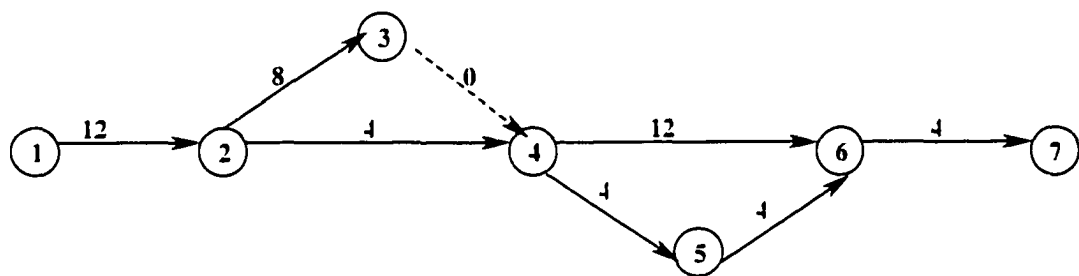


Figure 27. Project Network (case I)

#### 4.3.2 Cost Model Assumptions

The following assumptions apply for the used cost model:

1. Cost functions are linear, as shown in Fig. 28.
2. Activity duration's can be varied independently, i.e. no close co-ordination between tasks is required.
3. Cost functions are independent. The cost of crashing of one activity does not vary according to whether or not some other activity is also crashed.
4. The variances of costs and time are not important.
5. Variations in the overall requirements for resources do not affect the project cost.

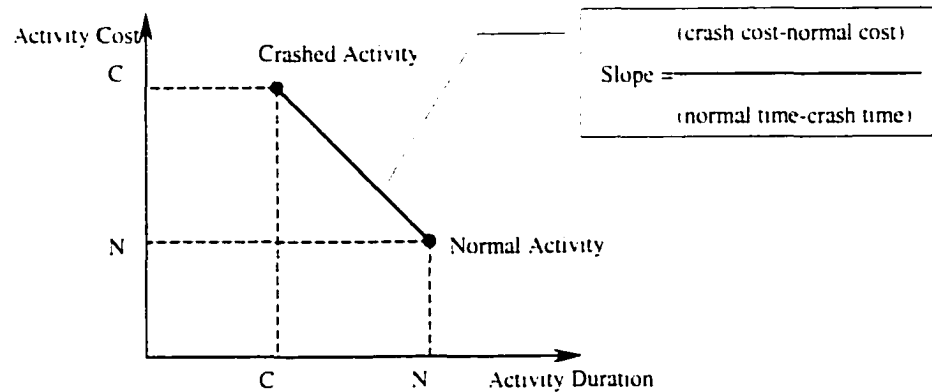


Figure 28. Cost Model (case I)

#### 4.3.3 Problem Formulation

A Linear Programming formulation is being used to solve the problem, as follows:

##### Decision Variables

- $x_i$  : earliest event time of node  $i$
- $x_j$  : earliest event time of node  $j$
- $y_{ij}$  : amount of time by which activity  $i \rightarrow j$  is crashed

##### Objective Function

To minimize the total crashing cost of the project,  $Z$

minimize  $Z$ , where  $Z = \sum c_{ij} y_{ij}$

$i = 0, 1, 2, \dots, m-1$  and  $j = 1, 2, \dots, m$

##### Constraints

- $y_{ij} \leq a_{ij}$  (crash time constraints)
- $x_j - x_i + y_{ij} \geq t_{ij}$  (constraints describing the network)
- $x_m \leq \text{DPTC}$  (project completion constraint)
- $x_i, x_j, y_{ij} \geq 0$  (non-negativity constraints)

where

- $m$  : number of nodes
- $c_{ij}$  : crashing cost per unit time of activity  $i \rightarrow j$  (slope)
- $a_{ij}$  : amount by which activity  $i \rightarrow j$  can be crashed
- $t_{ij}$  : normal duration for activity  $i \rightarrow j$
- $\text{DPTC}$  : desired project completion time

#### 4.3.4 Application

In order to solve the previous formulation, the data was transferred to Excel (as discussed later in section 4.5) and the model was solved using an Excel add-in, 'Solver'. Figure 29 shows the 'project data entry' worksheet, while Fig. 30 demonstrates the Time-Cost Trade-Off calculations performed in 'model' worksheet. And finally, Fig. 31 demonstrates the Time-Cost Trade-Off Curve as drawn by Excel.

Activity	Time		Cost		Total/Allowable Crash Time	Crash Cost per week
	Normal	Crash	Normal	Crash		
1→2	12	7	3000	5000	5	400
2→3	8	5	2000	3500	3	500
2→4	4	3	4000	7000	1	3000
3→4	0	0	0	0	0	0
4→5	4	1	500	1100	3	200
4→6	12	9	50000	71000	3	7000
5→6	4	1	500	1100	3	200
6→7	4	3	15000	22000	1	7000
Total Cost			75000			

(Normal Time - Crash Time)

Project Total Cost (Normal Durations)

Slope

Figure 29. Project Data Entry Worksheet

After Crashing					
			Required Duration	28	Required New PCT (entered)
Activity	Earliest Event Time of Node i	Earliest Event Time of Node j	Crashed By	Crashing Cost	Crashed Activity Time
1→2	0	7	5	2000	7
2→3	7	12	3	1500.00	5
2→4	7	12	0	0	4
3→4	12	12	0	0	0
4→5	12	20	0	0	4
4→6	12	24	0	0	12
5→6	20	24	0	0	4
6→7	24	28	0	0	4
Total Crashing Cost				3500.00	

Additional Cost due to crashed activities

Press Here to run a VBA Macro that runs the Solver on set of different PCTs, and then draws the Time-Cost Trade-off Curve

Press Here to Run the Solver

Project Crashing

Time-Cost Trade-off Curve

Figure 30. Time-Cost Trade-Off Worksheet

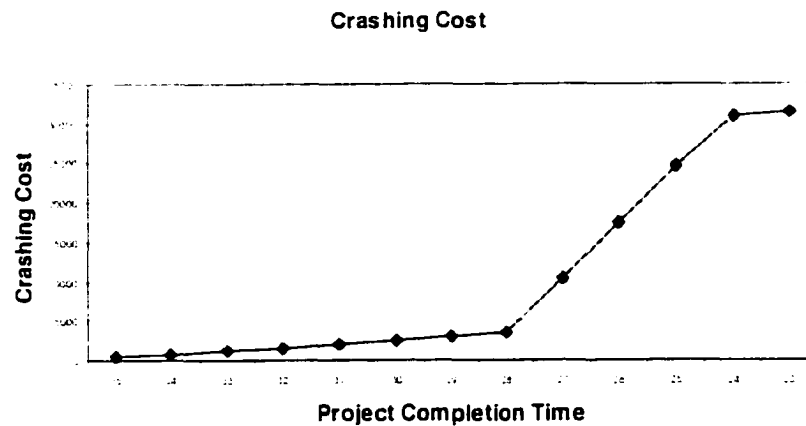


Figure 31. Time-Cost Trade-Off Curve.

#### 4.4 Case Study II

This second case study introduces multiple crash options per activity, thus raising the level of complexity higher than the one for the previous case study.

##### 4.4.1 Project Network

Figure 32 shows a network representing the project.

Normal CPM project completion time = 38 time units, and the critical path is 1-3-5-7-8-9.

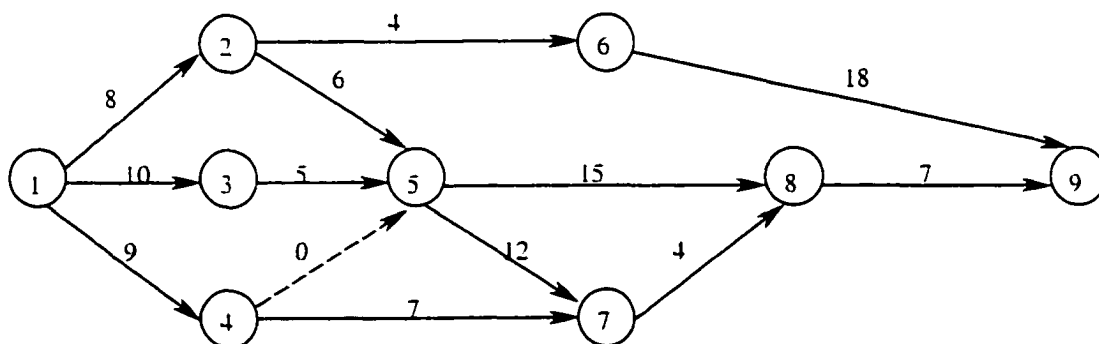


Figure 32. Project Network (case II)

#### 4.4.2 Cost Model Assumptions

In addition to the assumptions used in case study I, the new ones are:

1. Two crashing scenarios are available depending on the type of resource added, either over-time or out-source.
2. Only one crash scenario is allowed to be used per activity.
3. No conflicts among resources of different activities.

Figure 33 represents the cost model used.

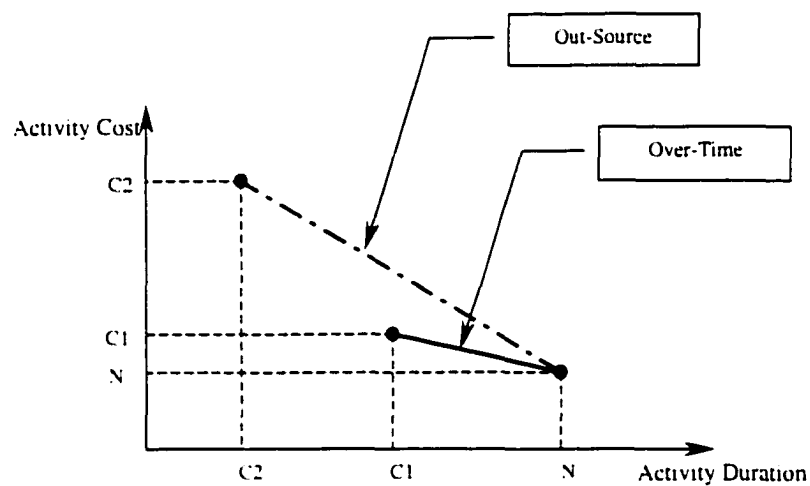


Figure 33. Cost Model (case II)

#### 4.4.3 Problem Formulation

A Mixed-Integer Linear Programming formulation is being used to solve the problem, as follows:

##### Decision Variables

- $x_i$  : earliest event time of node  $i$
- $x_j$  : earliest event time of node  $j$
- $y_{1ij}$  : amount of time by which activity  $i \rightarrow j$  is crashed if over-time is used
- $y_{2ij}$  : amount of time by which activity  $i \rightarrow j$  is crashed if out-source is used
- $\delta_{ij}$  : an indicator variable to distinguish between the state when over-time is used and the state where out-source is used in case that activity  $i \rightarrow j$  is to be crashed.



### Objective Function

To minimize the total crashing cost of the project, Z

minimize Z, where  $Z = \sum (c_{1ij} y_{1ij} + c_{2ij} y_{2ij} + \delta_{ij} a_{1ij} (c_{2ij} - c_{1ij}))$

,  $i = 0, 1, 2, \dots, m-1$  and  $j = 1, 2, \dots, m$

### Constraints

$y_{1ij} \leq a_{1ij}$  (crash time constraints)

$y_{1ij} + y_{2ij} \leq a_{1ij} + a_{2ij}$

$x_j - x_i + y_{1ij} + y_{2ij} \geq t_{ij}$  (constraints describing the network)

$x_m \leq \text{DPTC}$  (project completion constraint)

$y_{2ij} - (a_{2ij} - a_{1ij}) \delta_{ij} \leq 0$  (resource type constraints)

$\delta_{ij}$  is binary (0 or 1)

$x_i, x_j, y_{1ij}, y_{2ij} \geq 0$  (non-negativity constraints)

where

$m$  : number of nodes

$c_{1ij}$  : crashing cost per unit time of activity  $i \rightarrow j$  (slope) if over-time is used

$c_{2ij}$  : crashing cost per unit time of activity  $i \rightarrow j$  (slope) if out-source is used

$a_{1ij}$  : amount by which activity  $i \rightarrow j$  can be crashed if over-time is used

$a_{2ij}$  : amount by which activity  $i \rightarrow j$  can be crashed if out-source is used in addition to  $a_{1ij}$

$t_{ij}$  : normal duration for activity  $i \rightarrow j$

DPTC : desired project completion time

### 4.4.4 Application

In order to solve the previous formulation, the data was transferred to Excel (as discussed later in section 4.5) and the model was solved using an Excel add-in, 'What's Best! 4'. Figure 34 shows the 'project description' worksheet, while Fig. 35 shows 'project data entry' worksheet where data of each activity is being transferred from InnerCircle2000. Figure 36 demonstrates the 'model' worksheet', and Fig.37 demonstrates the Time-Cost Trade-Off Curve as drawn by Excel. Figure 38 represents the 'crash project data' worksheet, where data is being sent to InnerCircle2000. Finally, Fig. 39 shows the output report created by 'What's Best!' indicating that a 'globally optimum solution' achieved.

Activity	Name	Description
1 → 2	A	
1 → 3	B	
1 → 4	C	
2 → 5	D	
2 → 6	E	
3 → 5	F	
4 → 5	G	
4 → 7	H	
5 → 7	I	
5 → 8	J	
6 → 9	K	
7 → 8	L	
8 → 9	M	

Figure 34. Project Description Worksheet

Project Data Entry Table													
From InnerCircle2000		Name	Normal		Over-Time		Out-Source						
			Time	Rate	Time	Rate	Time	Rate					
		C	9	110	7	190	4	520					

Activity	Name	Normal			Crashing						Allowable Crash Time		Crash Cost per Day	
		Time	Rate	Cost	Over Time			Out-Source			Over Time	Out-Source	Over Time	Out-Source
1 → 2	A	8	100	800	6	200	1200	3	700	2100	2	5	200	260
1 → 3	B	10	105	1050	9	160	1440	5	800	4000	1	5	390	590
1 → 4	C	9	110	990	7	190	1330	4	520	2080	2	5	170	218
2 → 5	D	6	120	720	4	200	800	2	500	1000	2	4	40	70
2 → 6	E	4	125	500	3	220	660	1	1400	1400	1	3	160	300
3 → 5	F	5	100	500	4	200	800	2	850	1700	1	3	300	400
4 → 5	G	0	0	0	0	0	0	0	0	0	0	0	0	0
4 → 7	H	7	100	700	5	200	1000	2	950	1900	2	5	150	240
5 → 7	I	12	140	1680	10	230	2300	6	650	3900	2	6	310	370
5 → 8	J	15	90	1350	12	150	1800	8	370	2960	3	7	150	230
6 → 9	K	18	80	1440	14	220	3080	9	650	5850	4	9	410	490
7 → 8	L	4	150	600	3	280	840	1	1500	1500	1	3	240	300
8 → 9	M	7	105	735	6	185	1110	3	845	2535	1	4	375	450

Figure 35. Project Data Entry Worksheet

## Objective function

Activity	Y1	Slope1	Y2	Slope2	Delta	Diff.	Crashing Cost
A	2	200	3	260	1	120	1300
B	1	390	4	590	1	200	2950
C	2	170	0	218	0	96	340
D	2	40	0	70	0	60	80
E	1	160	2	300	1	140	900
F	1	300	2	400	1	100	1200
G	0	0	0	0	0	0	0
H	0	150	0	240	0	180	0
I	2	310	3	370	1	120	1850
J	3	150	4	230	1	240	1610
K	4	410	0	490	0	320	1640
L	1	240	2	300	1	60	900
M	1	375	3	450	1	75	1800
Total							14570

## Constraint Set # 1

Activity		Xi	Xj	Y1	Y2	L.H.S.		N.T.
A	1 → 2	0	3	2	3	8	=>=	8
B	1 → 3	0	5	1	4	10	=>=	10
C	1 → 4	0	7	2	0	9	=>=	9
D	2 → 5	3	7	2	0	6	=>=	6
E	2 → 6	3	4	1	2	4	=>=	4
F	3 → 5	5	7	1	2	5	=>=	5
G	4 → 5	7	7	0	0	0	=>=	0
H	4 → 7	7	14	0	0	7	=>=	7
I	5 → 7	7	14	2	3	12	=>=	12
J	5 → 8	7	15	3	4	15	=>=	15
K	6 → 9	4	18	4	0	18	=>=	18
L	7 → 8	14	15	1	2	4	=>=	4
M	8 → 9	15	18	1	3	7	=>=	7

## Constraint Set # 2

Activity	Y1		limit	Y2	Y1+Y2		limit
A	2	=<=	2	3	5	=<=	5
B	1	=<=	1	4	5	=<=	5
C	2	=<=	2	0	2	=<=	5
D	2	=<=	2	0	2	=<=	4
E	1	=<=	1	2	3	=<=	3
F	1	=<=	1	2	3	=<=	3
G	0	=<=	0	0	0	=<=	0
H	0	=<=	2	0	0	=<=	5
I	2	=<=	2	3	5	=<=	6
J	3	=<=	3	4	7	=<=	7
K	4	=<=	4	0	4	=<=	9
L	1	=<=	1	2	3	=<=	3
M	1	=<=	1	3	4	=<=	4

Figure 36. Model Worksheet

Constraint Set # 3					
Activity	Y2	Y2 limit	Delta	L.H.S.	R.H.S.
A	3	3	1	0	0
B	4	4	1	0	0
C	0	3	0	0	0
D	0	2	0	0	0
E	2	2	1	0	0
F	2	2	1	0	0
G	0	0	0	0	0
H	0	3	0	0	0
I	3	4	1	-1	0
J	4	4	1	0	0
K	0	5	0	0	0
L	2	2	1	0	0
M	3	3	1	0	0

#### Constraint Set # 4

PCT		
18	=<=	18

Total Crashing Cost **14570**

**Solve**

**Time-Cost Trade-Off Curve**

Figure 36 Continued

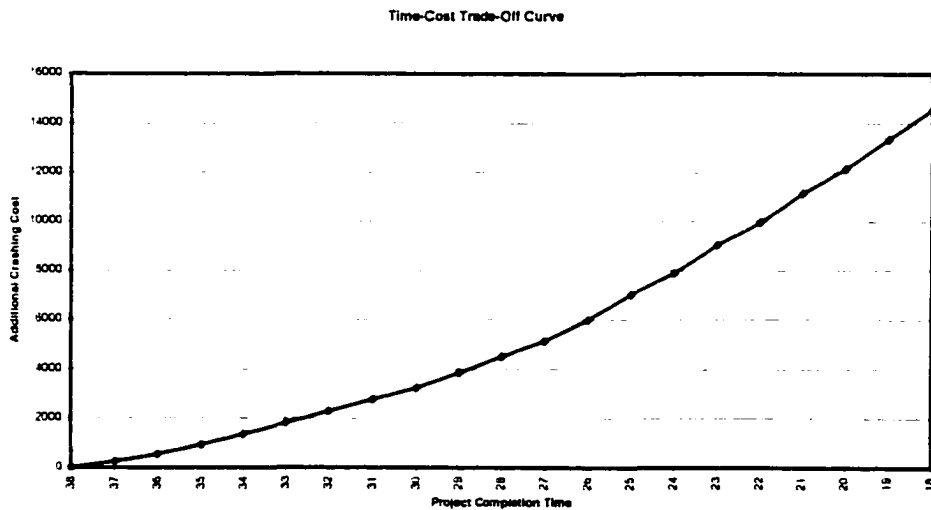


Figure 37. Time-Cost Trade-Off Curve

Crashed Project Data					
Activity	Name	Duration	Additional Resource Used	Rate	Total Cost
1 → 2	A	3	Out-Source	700	2100
1 → 3	B	5	Out-Source	800	4000
1 → 4	C	7	OverTime	110	1330
2 → 5	D	4	OverTime	120	800
2 → 6	E	1	Out-Source	1400	1400
3 → 5	F	2	Out-Source	850	1700
4 → 5	G	0	None	0	0
4 → 7	H	7	None	100	700
5 → 7	I	7	Out-Source	650	3530
5 → 8	J	8	Out-Source	370	2960
6 → 9	K	14	OverTime	220	3080
7 → 8	L	1	Out-Source	1500	1500
8 → 9	M	3	Out-Source	845	2535

Project Completion Time	18
-------------------------	----

To	Activity		Resource	
	Name	Duration	Name	Rate
InnerCircle2000	E	1	Out-Source	1400

Figure 38. Crashed Project Data Worksheet

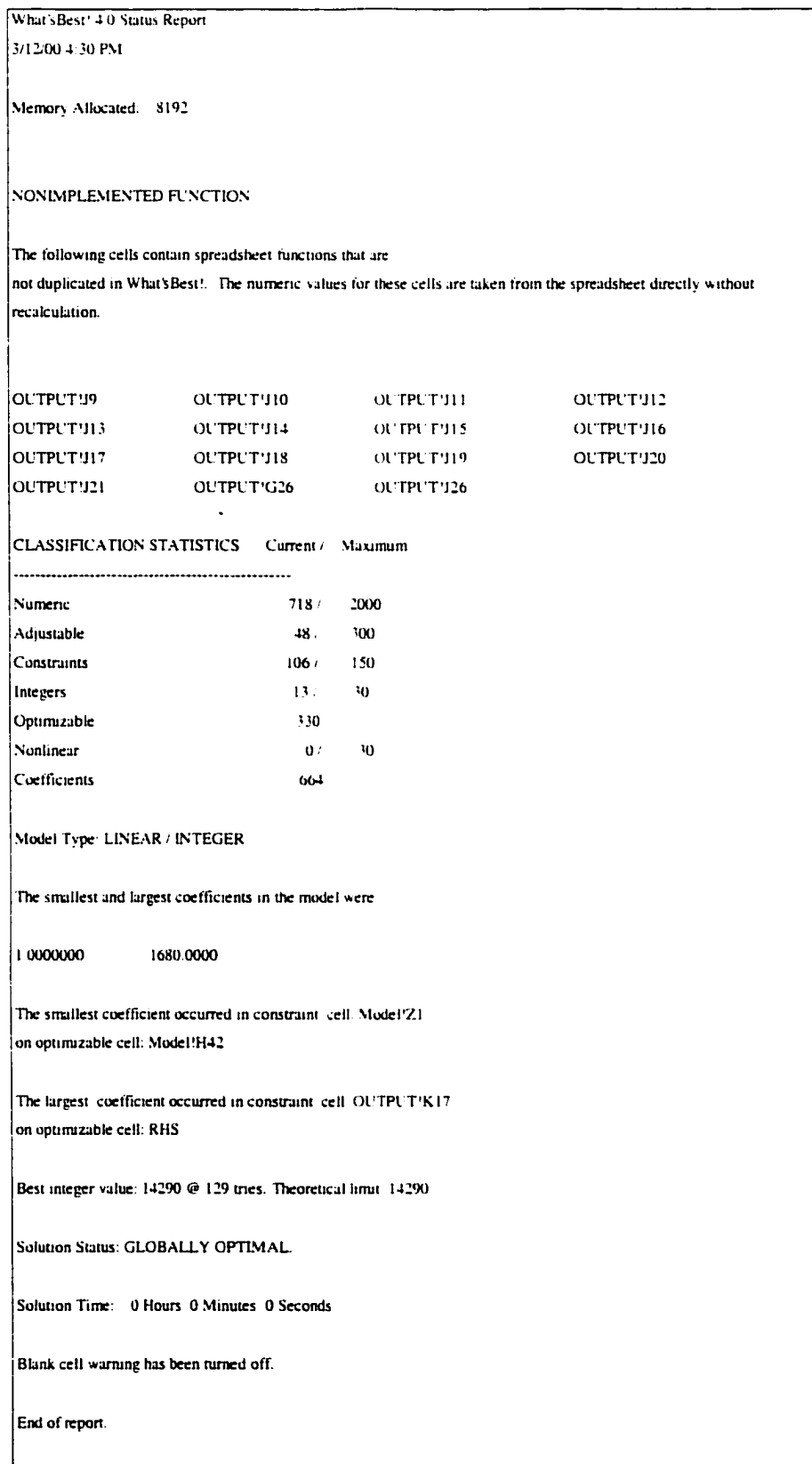


Figure 39. What's Best! 4 Report

## 4.5 Data Manipulation

As Shown in Fig. 40, the framework is used as follows: an object-oriented database model representing the project work break-down structure is being created. Through InnerCircle2000 linkages: First, the activity duration, normal resource rate, different crash times, and corresponding resource rate are entered into the database. Second, the project time chart will be drawn and resource analysis is to be performed. Third, project network crashing and Time-Cost Trade-Off analysis are to be done allowing the decision-maker to choose the best time span to execute the project. And fourth and finally, the crashed project time chart will be drawn in a separate file allowing new resource analysis to take place. Typically resource analysis in Microsoft Project means 'resource usage graph' and 'cash flow report.'

Screen shots of class definition and instance diagram, taken from InnerCircle2000, are displayed in the appendix.

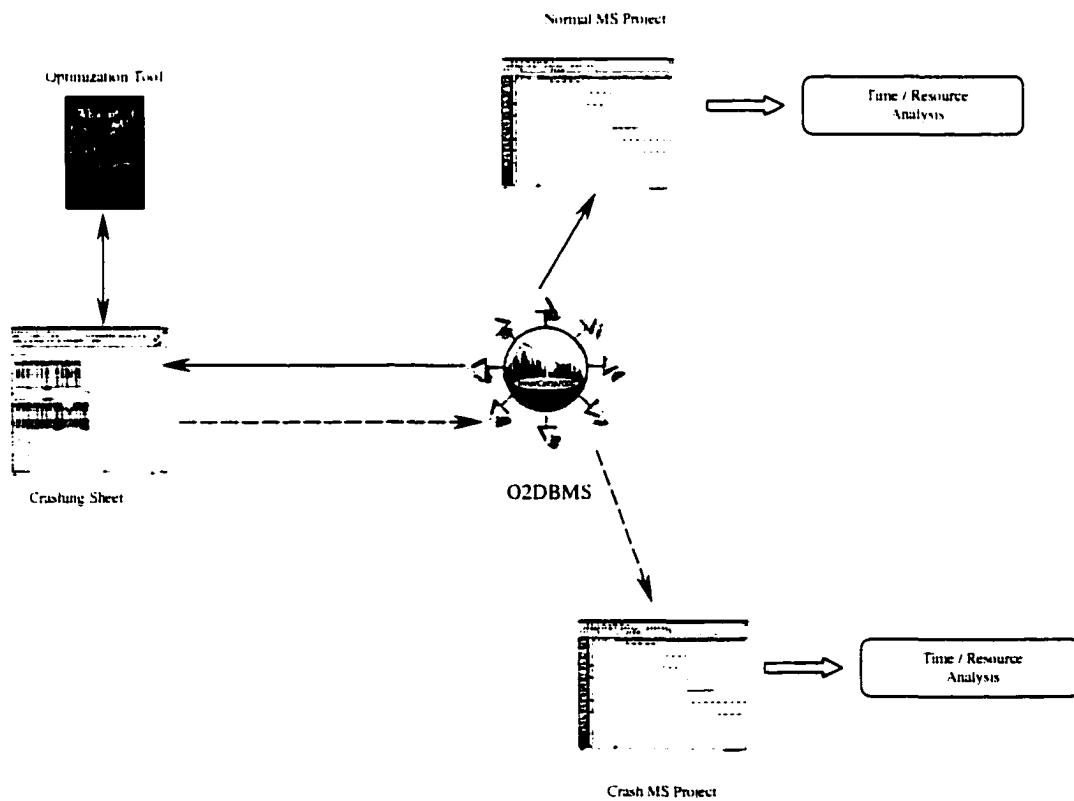


Figure 40. Data Manipulation through the Framework

## 4.6 Summary

Engineering, the whole process, is a knowledge- and communication-intensive process. For a collaborative work environment, team members need to have access to the knowledge underlying the decisions by each member, and notifying the affected parties in time. While computers are used extensively during each phase, existing tools do a little to facilitate information sharing and decision coordination, since information is isolated at tool boundary.

In an integrated manufacturing environment, database plays a central role. Different conventional database models, hierarchical, network, and relational databases, in addition to object-oriented data models were presented in chapter II. Conventional models have proved quite successful in developing database application required for traditional business environments. However, they have certain shortcomings with regard to emerging complex applications, such as engineering design and manufacturing. object-oriented database was proposed to meet the needs of these applications, since it offers the flexibility needed to handle structures of complex objects and the operations applied to these objects.

A look at research activities concerning object-oriented database application in engineering would result in the following:

1. Research concerning object-oriented databases was active during the period from 1981 to 1988, and then declined till the mid-90's.
2. Research concerning object-oriented database applications in engineering concentrates on CAD applications, and especially in Civil Engineering, while Mechanical Engineering application do not seem to have an appropriate share!
3. Almost no research concerning Mechanical Engineering Management Applications.
4. Some few articles concerned the use object-oriented databases as an integration tool in engineering.

The presented framework, in chapter III, integrates the capabilities of two commercial software tools without the need to adopt new links among them each time we change the application area. Furthermore, the use of an object-oriented database, as the backbone of the framework, reduces data redundancy to a minimum, since the data needed by all application is stored in one medium (server).

Two different cases of Time-Cost Trade-Off problem in project network were solved and manipulated through the framework. It can be seen that using the framework to handle such a problem would result in the following:

1. Team Integration (Planning Office, Project Management Dept., and Operation Research Office).
2. Functional Integration (Planning, Project Management, and Optimization).
3. Application Integration (Spreadsheets, Optimization Tools, Database, Project Management Software).



To solve each of the two cases, a mathematical model was formulated and then solved using an Excel add-in for optimization.

In case one, the traditional Excel Solver was used successfully. But in case two, where a more complicated mathematical formulation was to be applied, the Solver felt short of its objective (the solution didn't converge), and so another tool was used, that is, 'What's Best!'.

A quick look at both tools would show that 'What's Best!' has:

1. A better interface that enables user to handle the problem formulation easily.
2. A more powerful optimization engine (faster and more efficient).

## REFERENCES

- Abdalla, J. A., and Yoon, C. J., 1992, "Object-Oriented Finite Element and Graphics Data-Translation Facility," Journal of Computing in Civil Engineering, Vol. 6, No. 3, pp. 302-321.
- Achuthan, N. R., and Lotulelei, T., 1992, "Project crashing with earliness and tardiness penalties: a goal programming model," Proceedings, the International conference on Optimization: techniques and applications, pp. 375-384.
- Afsarmanesh, H., 1985, " The 3 Dimensional Information Space (3DIS): An Extensible Object-Oriented Framework for Information Management," PhD thesis, Dept. of Computer Science, Univ. of Southern California.
- Afsarmanesh, H., and Mcleod, D., 1984, "A framework for semantic database models," Proceedings, NYU Symposium on New Directions for Database Systems.
- Afsarmanesh, H., Knapp, D., Mcleod, D., and Parker, A., 1990, "An Extensible Object-Oriented Approach to Databases for VLSI/CAD," Readings in Object-Oriented Database Systems, ed. Zdonik, S. B., and Maier, D., Morgan Kaufman Publishers, Inc., California, USA, pp.
- Agrawal, D., Bruno, J. L., El Abbadi, A., and Krishnaswamy, V., 1995, "Managing concurrent activities in collaborative environments," Proceedings, The 3<sup>rd</sup> International Conference on Cooperative Information Systems, pp. 112-124.
- Ahmed, R., and Navathe, S. B., 1991, "Version Management of Composite Objects in CAD Databases," Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data, pp. 218 – 227.
- Ambrosy, S., Assmann, G., Bindbeutel, K., Cuiper, R., Feldmann, C., and Schmalzl, B., 1996, "Integrated product and process model for planning and design," Zwf Zeitschrift Fuer Wirtschaftlichen Fabrikbetrieb, vol.91, no.12, pp.607-11.
- Andrews, R., 1991, "The vbase object database environment," Research Foundations in Object-Oriented and Semantic Databases, A Cardenas, and D. Mcleod (eds.), Prentice Hall, Englewood Cliffs, NJ.
- Atkinson, M., Bancilhon, F., DeWitt, D., Dittrich, K., Maier, D., and Zdonik, S., 1992, "The Object-Oriented Database Systems Manifesto," In Bancilhon et. Al. (eds.), Building an Object-Oriented Database System: The Story of O2, Morgan Kaufman.

Babu, A. J. G., Stillman, W. P., and Suresh, N., 1994. "Project Crashing: Modeling and Optimization." Proceedings, 23<sup>rd</sup> Annual meeting of the Western Decision Sciences Institute, pp. 751-755.

Badiru, A. B., 1996. "Project Management in Manufacturing and High Technology Operations." 2<sup>nd</sup> ed., John Wiley & Sons, INC., New York.

Badiru, A. B., and Whitehouse, G. E., 1989. "Computer Tools, Models and Techniques for Project Management," TAB Professional and Reference Books, PA.

Bancilhon, F., 1996. "Object Databases." ACM Computing Surveys, Vol. 28, No. 1, pp. 137-140.

Bancilhon, F., Kim, W., and Korth, H., 1985. "A Model of CAD Transactions." Proceedings, the 11<sup>th</sup> International Conference on Very Large Data Bases.

Banerjee, J., Chou, H., Garza, J. F., Kim, W., and Kim, H., 1990. "Data Model Issues for Object-Oriented Applications." Readings in Object-Oriented Database Systems, ed. Zdonik, S. B., and Maier, D., Morgan Kaufman Publishers, Inc., California, USA. pp.

Bankes, W.F., 1986a, "Automated system for scheduling pipeline time for small batch production using a Symphony spreadsheet," Computers & Industrial Engineering, Vol. 11, No. 1-4, Proceedings, 8<sup>th</sup> Annual Conference Computers & Industrial Engineering, pp. 303-307.

Bankes, W.F., 1986b, "Automated system for matching tool capacity and utilization," Proceedings, 8<sup>th</sup> Annual Conference Computers & Industrial Engineering, pp. 308-311.

Barber, T.J., 1989, "Heuristics for project expedition," Proceedings, IEE Colloquium on Advances in Optimisation, pp.7/1-6.

Barber, T.J., and Boardman, J.T., 1986, "A pragmatic approach to the optimisation of project control using heuristic techniques," Proceedings, IEE Colloquium on The Application of Optimisation Techniques to Real Engineering Processes, pp.5/1-4.

Barry, D. K., 1996. "The Object Database Handbook: How to Select, Implement, and Use Object-Oriented Databases," John Wiley & Sons, Inc. New York.

- Barsalou, T., Siambela, N., Keller, A. M., and Wiederhold, G., 1991, "Updating Relational Databases through Object-Based Views," Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data, pp. 248 - 257.
- Bassiliades, N., and Vlahavas, I., 1997, "Processing production rules in DEVICE, an active knowledge base system," Data & Knowledge Engineering, No. 24, pp. 117-155.
- Batory, D. S., and Buchmann, A. P., 1984, "Molecular Objects, Abstract Data Types, and Data Models- A Framework," Proceedings, VLDB Conference.
- Batory, D. S., and Kim, W., 1985, "Modeling Concepts for VLSI CAD Objects," ACM TODS, Vol. 10, No. 3, pp. 322-346.
- Batson, R. G., 1987, "Critical path acceleration and simulation in aircraft technology planning," IEEE Transactions on engineering Management, Vol. EM-34, No. 4, pp. 244-251.
- Beeri, C., 1990, "A formal approach to object-oriented databases," Data and Knowledge Engineering 5.
- Bellcore, A.P.S., and Karabatis, G., 1993, "Multidatabase Independencies in Industry," Proceedings, The 1993 ACM SIGMOD International Conference of Management on Data, pp. 483-486.
- Beghini, G., and Romanin J. G., 1991, "Project management and production planning: study for an integrated solution," Proceedings, Engineering Systems with Intelligence: Concepts, Tools and Applications, Kluwer Academic Publishers, pp.549-54.
- Berchtold, S., and Kriegel, H., 1997, "S3: Similarity Search in CAD Systems," Proceedings, The ACM SIGMOD International Conference on Management of Data, pp. 564-567.
- Bertino, E., and Guerrini, G., 1996, "View Points in Object Database Systems," Proceedings, 2<sup>nd</sup> International Software Architecture Workshop (ISAW-2) and International Multiple Perspectives in Software Development (viewpoint'96) on SIGSOFT'96 Workshop, pp. 289-293.
- Bertino, E., and Martino, L. 1993, "Object-Oriented Database Systems: Concepts, and Architecture," Addison-Wesley Publishing Co., New York.
- Bjork, B., and Penttila, H, 1991, "Building Product Modeling Using Relational Databases, Hypermedia Software and CAD Systems," Microcomputers in Civil Engineering, Vo. 6, pp. 267-279.

- Blaha, M.R., Permerlani, W. J., and Rumbaugh, J.E., 1988, "Relational database design using an object-oriented methodology," Communications ACM, Vol. 31, No. 4, pp. 414-427.
- Blakeley, J. A., Larson, P., and Tompa, F. W., 1986, "Efficiently updating materialized views," Proceedings, the International Conference on Management of Data (SIGMOD.86).
- Bowman, R.A., 1994, "Stochastic gradient-based time-cost tradeoffs in PERT networks using simulation," Annals of Operations Research, vol.53, Nov. 1994, pp.533-51.
- Boznak, R., 1988, "Achieving a Competitive Manufacturing Advantage Through Effective Multi-Project Management," Proceedings, International Industrial Engineering Conference, pp. 285-290.
- Brown, A.M., 1984, "Project management for the design and supply of power station mechanical and electrical plant," IEE Proceedings-C Generation Transmission & Distribution, vol.131, no.6, pp.236-9.
- Brown, A. W., 1991, "Object-Oriented Databases. Applications in Software Engineering." McGraw Hill, New York.
- Brown, T.T., 1995, "Integrated design and manufacture: a pilot project approach to implementation," Colloquium on 'Concurrent Engineering - Getting it Right First Time' (Digest No.1995/127), pp.10/1-5.
- Buchmann, A. P., 1995, "An Architecture and Data Model for CAD Database," Proceedings, the 11<sup>th</sup> International Conference on Very Large Data Bases.
- Burns, S., Liu, L., Feng, C. W., 1995, "From Algorithm to Practical Tool - The Development of a Construction Time-Cost Trade-Off Optimization System," Proceedings, Research Transformed Into Practice: Implementation of NSF Research, pp. 269-280.
- Cart, M., and Ferrie, J., 1990, "Integrating concurrency Control into Object-Oriented Database system," EDBT Proceedings, Venice, Italy.
- Cattell, R. (ed), 1994b, " The Object Database Standard: ODMG-93." Morgan Kaufmann, San Francisco, CA.
- Cattell, R. G. G., 1991, "Object Data Management, Object-Oriented and Extended Relational Database Systems, Reading," Addison Wesley, MA.

- Cattell, R. G. G., 1994, "Object Data Management: Object-Oriented and Extended Relational DB Systems," Addison-Wesley Publishing Co., New York.
- Celko, J., 1999, "Joe Celko's Data and Database: Concepts in Practice," Morgan Kaufman Publishers, California, USA.
- Cellary, W., and Wiczerzycki, W., 1993, "Locking Objects and Classes in Multiversion Object-Oriented Databases," Proceedings, 2<sup>nd</sup> International Conference on Information and Knowledge Management, pp. 586 – 595.
- Centeno, M. A., and Stnadridge, C., 1991, "Modeling Manufacturing systems: An Information-Based Approach," Proceedings, 24<sup>th</sup> Annual Simulation Symposium, pp. 230 – 239.
- Chapman, C. B., Cooper, D. F., and Page, M. J., 1987, "Management for Engineers," John Wiley & Sons, New York.
- Chappell, D., 1996, "Understanding Active X & OLE," Microsoft Press, Washington.
- Chassiakos, A.P., Samaras C.I., and Theodorakopoulos D.D., 1998, "An algorithm for determining the optimal duration of large projects," Proceedings, Advances in Civil and Structural Engineering Computing for Practice, pp.449-54.
- Chen, Y., Miller, R. A., and Sevenler, K., 1995, "Knowledge-based Manufacturability Assessment: an Object-Oriented Approach," Journal of Intelligent Manufacturing, Vol. 6, pp. 321-337.
- Ching, N., Hughes, E., and Winslett, M., 1996, "A Model of Object Database Applications and Its Use in Cost Estimation," Proceedings of the Fifth International Conference on Information and Knowledge Management, pp. 125-133.
- Chishaki, T., and Tatish, M., 1992, "New model for project planning by time-cost trade-off procedure using fuzzy durations for project activities,"
- Chong, A. K., and Veress, S. A., 1988, "Cost estimates of photogrammetric related services using electronic spreadsheets," Photogrammetric Engineering & Remote Sens., pp. 47-50.
- Chung, J., Lin, Y., and Chang, D. T., 1995, "Objects and Relational Databases," ACM SIGPLAN OOPS Messenger, Vol. 6, No. 4, pp. 164-169.

Churchill, G.F., 1988, "Quality assurance - an effective project management technique," International Journal of Project Management, vol. 6, no. 4, pp. 241-244.

Coleman, H. W., 1987, "A PC spreadsheet application for safety stock calculations," Production & Inventory Management Journal, 2<sup>nd</sup> qtr, pp. 110-116.

Corwe, E., 1997, "Internet Basics," ActiveX: Microsoft Inside, July 22, 1997.

Dayal, U., and Smith, J. M., 1986, "PROBE: A knowledge-Oriented Database Management System," On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies, M.L. Brodie and J Mylopoulos (eds.), Springer-Verlag.

Dayal, U., et. al., 1985, "PROBE- A Research Project in Knowledge-Oriented Database Systems: Preliminary Analysis," Technical Report CCA-85-03, Computer Corporation of America, July 1985..

Dayal, U., Hanson, E., and Widom, J., 1995, "Active database systems," Modern Database Systems, W. Kim (ed.), ACM Press, pp. 434-456.

De Lurgio, S. A., and Zhao, J-G, 1989, "Teaching integrated production and information control system principles using a spreadsheet simulator," Production & Inventory Management Journal, 1<sup>st</sup> qtr, pp. 29-35.

De Reyck, B., and Herroelen, W., 1996, "On the use of the complexity index as a measure of complexity in activity networks," European Journal of Operational Research, Vol.91, No.2, pp.347-366.

De, P., Dune, E.J., Ghosh, J.B., and Wells, C.E., 1997, "Complexity of the discrete time-cost tradeoff problem for project networks," Operations Research, Vol.45, No.2, pp.302-306.

Deckro, R.F., and Hebert, J.E., 1989, "Resource constrained project crashing," Omega, vol.17, no.1, pp.69-79.

Demeulemeester, E.L., De Reyck, B., Foubert, B., Herroelen, W., and Vanhoucke, M., 1998, "New computational results on the discrete time/cost trade-off problem in project Networks," Journal of the Operational Research Society, Vol.49, No.11, pp.1153-63.

Demeulemeester, E.L., Herroelen, W.S., and Elmaghraby S.E., 1996, "Optimal procedures for the discrete time/cost trade-off problem in project networks," European Journal of Operational Research, Vol.88, No.1, pp.50-68.

Denninghoff, K., and Vianu, V., 1993. "Database Methods Schemas and Object Creation." Proceedings, 12th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 265 – 275.

Diaz, I., and Lezman, S., 1988, "Material handling simulation: Minimizing bottlenecks and improving product flow using Lotus 1-2-3." Industrial Engineering, June 1988, pp. 40-46.

Dix, R. C., 1985, "Electronic spreadsheets in manufacturing," Production Engineering, April 1985, pp. 78-81.

Domazet, D. S., and San, L. S., 1997, "Active database servers for concurrent engineering environments." Proceedings, the 5<sup>th</sup> International Conference on Database Systems for Advanced Applications.

Earnest, D. L., 1987, "Capital equipment justification: A spreadsheet application template." Proceedings, 9<sup>th</sup> Annual Conference Computers in Ind. Engineering, pp. 36-46.

Elmasri, R., and Navathe, S.B., 1994, "Fundamentals of Database Systems," 2<sup>nd</sup> edition, Addison-Wesley Publishing Co., New York.

Emond, J., C., and Marechal, G., 1983, "A Computer Aided-Design System Based On A Relational DBMS." ACM SIGMOD/IEEE Engineering Design Applications.

Englund, R. L., and Graham, R. J., 1999, "From Experience: Linking Projects to Strategy," J PROD INNOV MANAG, No. 16, pp. 52-64.

Erenguc, S.S., Tufekci, S., and Zappe, C.J., 1993, "Solving time/cost trade-off problems with discounted cash flows using generalized Benders decomposition," Naval Research Logistics, Vol.40, No.1, Feb. 1993, pp.25-50.

Evans, J. R., 1986, " Spreadsheets and optimization: Complementray tools for decision making," Production & Inventory Management. Journal, 1<sup>st</sup> qtr, pp. 36-46.

Fang, S.-C., and Puthenpura, S., 1992, "Linear Optimization and Extensions: Theory and Algorithms," Prentice Hall, New Jersey.

Fargher, J. S. W., Jr., 1987, "Industrial engineering spreadsheet applications from a manufacturing resource planning (MRP-II) system," Proceedings, 9<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 100-106.



Feldermann, J., 1993. "Project management in production scheduling experience of a large scale company in starting series type production of new products." Vdi-Z. vol.135, no.8,pp.40-2.

Feng, Ch-W, Liu, L., and Burns, S.A., 1997. "Using genetic algorithms to solve construction time-cost trade-off problems." Journal of Computing in Civil Engineering, Vol. 11, No. 3, pp. 184-189.

Fields, F. E., 1986, "Industrial use of a spreadsheet," Proceedings, 8<sup>th</sup> Annual Conference Computers & Industrial Engineering, pp. 312-315.

Fishman, D. H., Beech, D., Cate, H. P., Chow, E. C., Connors, T., Davis, I. W., Derrett, N., Hoch, C. G., Kent, W., Lyngbaek, P., Mahbod, B., Neimat, M. A., Ryan, T. A., and Shan, M. C., 1990, "Iris: An Object-Oriented Database Management System," Readings in Object-Oriented Database Systems, ed. Zdonik, S. B., and Maier, D., Morgan Kaufman Publishers, Inc., California, USA, pp.

Fong, E., Kent, W., Moore, K., and Thompson, C. (eds.), 1991, "X3/SPAR/DBSSG/OOBTG Final Report," NIST Technical Report, National Institute of Standards and Technology.

Gander, J.P., 1985, "Cooperative research, government involvement, and timing of innovations," Technological Forecasting & Social Change, Vol.28, No.2, pp.159-72.

Garrett, J. H., and Hakim, M. M., 1992, " Object-Oriented Model of Engineering Design Standards," Journal of Computing in Civil Engineering, Vol. 6, No. 3, pp. 323-347.

Gatzju, S., Geppert, A., and Dittrich, K. R., 1991, "Integrating Active Concepts into an Object-Oriented Database System," Proceedings, Database Programming Languages: Bulk Types & Persistent Data, the 3<sup>rd</sup> International Workshop, pp. 399-415.

Gehani, N. H., and Jagadish, H. V., 1991, "Ode as an active database: Constraints and triggers," Proceedings, the 17<sup>th</sup> International Conference on Very Large Data Bases, pp. 327-336.

Gido, J., and Clements, J. P., 1999, "Successful Project Management: A Practical Guide for Managers," South-Western College Publishing, Canada.

Gillenson, M. L., 1990, "Database Step-By-Step," 2<sup>nd</sup> edition, John Wiley and Sons.

Godin, V. B., 1987. "Solving decision tree analysis using IFPS or LOTUS," Industrial Engineering, April 1987, pp. 20-21, 24-25, 27.

Godin, V. B., and Rao, A., 1988, "LOTUS 1-2-3 can produce dynamic graphic simulation without stops in the program," Industrial Engineering, March 1988, pp. 42-43, 45-46.

Golendziner, L.G. Santos, C. S., and Wagner, F. R., 1997, "Modeling an Engineering Design Application Using Extended Object Oriented Concepts," Proceedings, 5<sup>th</sup> International Conference on Database Systems for Advanced Applications, pp. 343-352.

Graff, L., 1989, "Spreadsheet applications in benefit/cost analysis," Proceedings, 11<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 293-297.

Graves, S.B., 1987, "Optimal R and D expenditure streams: An empirical view," IEEE Transactions on Engineering Management, Vol.EM-34, No.1, pp.42-48.

Graza, J., and Kim, W., 1988, "Transaction Management in an Object-Oriented Database system," Proceedings, ACM SIGMOD Conference.

Gruber, T. R., Tenenbaum, J. M., and Weber, J. C., 1992, "Toward a Knowledge Medium for Collaborative Product Development," Artificial Intelligence in Design '92, Kluwer Academic Publishers, pp. 413-432.

Haffner, E.W., Graves, R.J., 1988, "Managing new product time to market using time-cost trade-off methods," Omega, Vol.16, No.2, 1988, pp.117-124.

Hajdu, M., 1996, "PDM time cost trade-off: activities are splittable or non-splittable," Mathematische Operationsforschung und Statistik, Series Optimization, Vol.38, No.2, pp.155-71.

Halper, M., 1993, "A comprehensive Part Model and Graphical Schema Representation for object-Oriented Databases," PhD thesis, NJIT.

Halper, M., Geller, J., and Perl, Y., 1992, "'Part' relationships for object-oriented databases," Proceedings, 11<sup>th</sup> International Conference on Entity-Relationship Approach, pp. 406-422.

Halper, M., Geller, J., and Perl, Y., 1992, "An OODB 'part' relationship model," Proceedings, ISMM 1<sup>st</sup> International Conference on Information and Knowledge Management, pp. 602-611.

Halper, M., Geller, J., and Perl, Y., 1993, "Value propagation in OODB part hierarchies," Proceedings, ISMM/ACM 2<sup>nd</sup> International Conference on Information and Knowledge Management, pp. 606-614.

Halper, M., Geller, J., Perl, Y., and Klas, W., 1994, "Integrating a Part Relationship into an Open OODB System using Metaclasses," Proceedings of the third international conference on Information and knowledge management, pp. 10-17

Hamacher, H., and Tufekci, S., 1983, "A lexicographical time-cost tradeoff problem," Proceedings, Operations Research Verfahren, No.45, pp.257-68.

Hamilton, W., R., and Rothe, K.E., 1985, "Computerized cost estimation spreadsheet and cost data for fusion devices, Proceedings, 6<sup>th</sup> Topical Meet. On Tech. Of Fusion Energy, pp. 356-361.

Heuer, G., 1976, "Project management in mechanical engineering. Planning and controlling of industrial intentions," Vdi-Z., vol.118, no.7, pp.304-8.

Hines, M. L., 1998, "Conceptual Object-Oriented Database: a Theoretical Model," Journal of Information Sciences, Vol. 105, pp. 31-68.

Hogan, R., 1990, "A Practical Guide to Database Design," Prentice-Hall Inc., New York.

Hong, S., and Maleyeff, J., 1987, "Production planning and master scheduling with spreadsheets," Production & Inventory Mngement Journal, 1<sup>st</sup> qtr, pp. 46-54.

Hughes, E., and Winslett, M., 1995a, "The Index Suggestion Problem for Object Database Applications," Proceedings of the 1995 International Conference on Information and Knowledge Management, pp. 50 – 57.

Hughes, E., and Winslett, M., 1995b, "PEDCAD: A Framework for Performance Evaluation of Object Database Applications," Proceedings, The 1995 ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems, pp. 309 – 310.

Hughes, J. G., 1991, "Object-Oriented Databases," Prentice Hall, NY.

Huh, S., 1998, "An Object-Oriented Database Model for a Change Management Framework in Workgroup Computing systems," Information and Software Technology, Vol. 40, No. 2, pp. 79-92.

Hurson, A., and Pakzad, S., 1993, "object-oriented database management systems: evolution and performance issues," IEEE Computer, Vol. 26, No. 2.

- Icmeli, O., Erenguc, S.S., and Zappe, C.J., 1993, "Project scheduling problems: a survey," International Journal of Operations & Production Management, Vol.13, No.11, pp.80-91.
- Jacky, J. P., and Kalet, I. J., 1987, "An object-oriented programming discipline for standard Pascal," Communications ACM, Vol. 30, No. 9, pp. 772-776.
- Kanda, A., and Rao, U.R.K., 1984, "A network flow procedure for project crashing with penalty nodes," European Journal of Operational Research, Vol.16, No.2, pp.174-82.
- Kandt, K., 1994, "Object Management in SeeQFD," Distributed Object Management, M.T. Ozsü, U. Dayal and P. Valduriez (eds), Morgan Kaufmann Publishers, San Mateo, Calif, pp. 132-146.
- Katz, R. H., 1990, "Toward a Unified Framework for Version Modeling in Engineering Databases," ACM Computing Surveys, Vol. 22, No. 4, pp. 375-409.
- Kemper, A., and Moerkotte, G., 1994, "Object Oriented Database Management: Applications in Engineering and Computer Science," Prentice Hall, New Jersey.
- Kemper, A., and Wallrath, M., 1987, "An Analysis of Geometric Modeling in Database Systems," ACM Computing Surveys, Vol. 19, No. 1, pp. 47-91.
- Kennedy, S., and Martinez, J. R., 1987, "Spreadsheet application to labor determination," Proceedings, 9<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 317-318.
- Ketafchi, M., and Berzins, V., 1986, "Component Aggregation: A Mechanism for organizing Efficient Engineering Database," IDCE.
- Khoshafian, S., 1993, "Object-Oriented Databases," John Wiley & Sons , Inc., New York.
- Khoshafian, S., and Abnous, R., 1990, "Object Orientation: Concepts, Languages, Databases, User Interfaces," John Wiley & Sons , Inc. New York.
- Kim, J., and Ibbs, C. W., 1992, "Comparing Object-Oriented and Relational Data Models for Project Control," Journal of Computing in Civil Engineering, Vol. 6, No. 3, pp. 348-369.
- Kim, W., 1990a, "Introduction to Object-Oriented Databases," The MIT Press, Massachusetts.

- Kim, W., 1990b, "Research Directions in Object-Oriented Database Systems," Proceedings, 9<sup>th</sup> ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 1-15.
- Kim, W., 1995, "Object-Oriented Database Systems: Promises, Reality, and Future," Modern Database Systems: The Object Model, Interoperability, and Beyond, Kim, W. (ed.), ACM Press, New York, pp. 225-280.
- Kim, W., Bertino, E., and Garza, J. F., 1989, "Composite objects revisited," Proceedings, 1989 ACM SIGMOD International Conference on the Management of Data, pp. 337-347.
- Kim, W., Chou, H-T., and Banerjee, J., 1987, "Operations and Implementation of Complex Objects," Proceedings, Data Engineering Conference.
- Kleinfeld, I. H., 1984, "General purpose software adapted for IE applications," Industrial Engineering, February 1984, pp. 18-20.
- Kobryn, C., 1999, "UML 2001: A Standardization Odyssey," Communications of the ACM, Vol. 42, No. 10, pp. 29-37.
- Korth, H. F., Kim, W., and Bancilhon, F., 1990, "On long-duration CAD transactions," Readings in Object-Oriented Database System, S. B. Zdonik and D. Maier (eds), Morgan Kaufmann Publishers, San Mateo, Calif, pp. 408-432.
- Kotz-Dittrich, A., and Dittrich, K., 1995, "Where object-oriented DBMSs should do better: A critique based on early experience," Modern Database Systems: The Object Model, Interoperability, and Beyond, W. Kim (ed.).
- Kral, I. H., 1992, "The Excel Spreadsheet for Engineers and Scientists", Prentice Hall, New Jersey.
- Kung, C., 1990, "Object Subclass Hierarchy in SQL: a Simple Approach," Communications of the ACM, Vol. 33, No. 7, pp. 117-127.
- Kuo, W., and Folkers, R., 1986, "Mini-micro computers: Spreadsheet programs solve systems dynamics problems," Industrial Engineering, November 1986, pp. 24-26, 28-29, 31.
- Larson, E., W., and Gobeli, D., 1988, "Organizing for Product Development Projects," J PROD INNOV MANAG, No. 5, 180-190.

Law, J.S., Hsing-Wei, C., 1987, "Models to predict efficiency of two network flow based algorithms on the time-cost trade-off problem," Computers & Industrial Engineering, Vol.12, No.2, pp.91-97.

Ledbetter, L., and Cox, B., 1985, "Software-IC.s", Byte, June.

Lee, H., and Choi, B. G., 1998, "A Comparative Study of Conceptual Data Modeling Techniques," Journal of Database Management, Vol. 9, No. 2, pp. 26-35.

Levine, H. A., 1986, "Project Management using Microcomputers," Osborne McGraw-Hill, California.

Levy, J.B., and Tayi, G.K., 1989, "Analysis of project scheduling strategies in a client-contractor environment," Naval Research Logistics, Vol.36, No.1, pp.69-87.

Li, H., Cao, J.-N., Love, P. E. D., 1999, "Using machine learning and GA to solve time-cost trade-off problems," Journal of Construction Engineering and Management, Vol. 125, No.5, pp. 347-353.

Liu, L.; Burns, S. A., and Feng, C.-W. , 1995, "Construction time-cost trade-off analysis using LP/IP hybrid method. ," Journal of Construction Engineering and Management, Vol. 121, Dec. '95, pp. 446-454.

Liu, M. C., 1989, "Using spreadsheet to teach on-line statistical process control," Proceedings, 11<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 191-195.

Lorie, R. A., and Plouffe, W., 1983, "Complex Objects and Their use in Design Transactions," ACM SIGMOD/IEEE Engineering Design Applications.

Lue, T. W., 1987, "Using integrated spreadsheets for production and facilities planning," Proceedings, 9<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 88-91.

Mace, T., and Rubenking, N. J., 1996, "Inside the ActiveX platform," PC Magazine, September 24, 1996.

MacKellar, B. K., and Peckham, J., 1992, " Representing Design Objects in SORAC: A data model with semantic objects, relationships and constraints," Proceedings, 2<sup>nd</sup> International Conference on Artificial Intelligence in Design.

Maier, 1989, "Making Database Systems fast enough for CAD applications," Object-Oriented Concepts, Databases, and Applications, W. Kim and Lochovsky (eds.), ACM Press, New York, pp. 573-582.

Maio, A. D., Verganti, R., and Corso, M., 1994, "A multi-project management framework for new product development," European Journal of Operational Research, No. 78, pp. 178-191.

Maletz, M. C., 1987, "Knowledge-Based Project Management for Manufacturing Projects," Proceeding Autofact'87 Conference, pp. 31-53.

Manola, F., and Dayal, U., 1990, "PDM: An Object-Oriented Data Model," Readings in Object-Oriented Database Systems, ed. Zdonik, S. B., and Maier, D., Morgan Kaufmann Publishers, Inc., California, USA.

Martin, J., 1986, "Multi-machine assignment workload calculation sheet. A Lotus 1-2-3 program for calculation of randomly serviced multi-machine assignments for measured daywork systems," 8<sup>th</sup> Annual Conference Computers & Industrial Engineering, pp. 346-350.

Masri, S., and Moodie, C., 1985, "Using an electronic spreadsheet to analyze manufacturing flow systems," Computers & Industrial Engineering, Vol. 9, No. 2, pp. 183-193.

McCarthy, D. R., and Dayal, U., 1989, "The architecture of an active database management system," Proceedings, ACM SIGMOD International Conference on Management of Data, pp. 215-224.

Mehmood Z., Singhal A., et al., 1987, "A design data management system for CAD," Proceedings, ICCAD Conference, pp. 220-223.

Microsoft Corporation, 1999, "How to Write and Use ActiveX Controls for Microsoft Windows CE". Microsoft Corporation, June 1999.

Microsoft Corporation, 1995, "User's Guide for Microsoft Project for Windows 95 and Windows 3.1."

Montesi, D., and Torlone, R., 1995, "A Rewriting Technique for the Analysis and the Optimization of Active Databases," Proceedings, Database Theory – ICDT'95: 5<sup>th</sup> International Conference, pp. 238-251.

Murali, K. M., and Rao, T. A., 1998, "Epitomising the Structure of a Project Network for Crashing by Identifying Non-Requisite Activities," Proceedings, 9<sup>th</sup> Symposium: Information Control in Manufacturing, pp. 857-864.

Mylopoulos, J., Chaudhri, V., Plexousakis, D., Shrufi, A., and Topaloglou, T., 1996, "Building knowledge base management systems," The VLDB Journal, No. 5, pp. 238-263.

- Nair, K.P.K., Prasad, V.R., and Aneja, Y.P., 1993, "Efficient chains in a network with time-cost trade-off function on each arc," European Journal of Operational Research, Vol.66, No.3, pp.392-402.
- Navathe, S. B., and Cornelio, A., 1990, "Modeling Engineering Data By Complex Structural Objects and Complex functional Objects," Proceedings, the International Conference on Extending Database Technology.
- Nguyen, G. T., and Rieu, D., 1991, "Representing Design Objects," AI in Design, J. Gero (ed.), Butterworth-Heinemann Ltd.
- Nocedal, J., and Wright, S. J., 1999, "Numerical Optimization," Springer, New York.
- Object Management Group, INC., "OMG Unified Modeling Language Specification," Version 1.3, June 1999.
- Oden, H. W., 1986, "An interactive productivity measurement model using spreadsheet software," Proceedings, 8<sup>th</sup> Annual Conference Computers & Industrial Engineering, pp. 161-170.
- Oden, H. W., 1988, "Developing flexible productivity measurement models using spreadsheet software," Computers & Industrial Engineering, Vol. 14, No. 2, pp. 161-170.
- Osullivan, D., 1991, "Project management in manufacturing using IDEF/sup 0/," International Journal of Project Management, vol.9, no.3, pp.162-8.
- Pancake, C. M., 1995, "The Promise and the Cost of Object Technology: a Five Year Forecast," Communications of the ACM, Vol. 38, No. 10, pp. 32-49.
- Parlar, M., 1986, "Dynamic programming on an electronic spreadsheet," Computers & Industrial Engineering, Vol. 10, No. 3, pp. 203-213.
- Patrick, C., and Topuz, E., 1995, "Time-cost, trade-off analyses of longwall face transfers," Mining Engineering, Vol. 47, Apr. '95, pp. 366-370.
- Phillips, D. T., Ravindran, A., and Solberg, J. J., 1976, "Operations Research: Principles and Practice," John Wiley & Sons, Inc., New York.



PMI Standards Committee. " A guide to the Project Management Body of Knowledge", 1996. Project Management Institute, PA, USA.

Portugal, V., and Janctewski, L. J., 1998. "An Information system for operations management: Co-ordination and integration." Industrial Management and Data Systems, No. 98/8, pp. 356-361.

Premierani, W. J., Blaha, M. R., Rumbaugh, J. E., and Varwing, T. A., 1990, "An Object-Oriented Relational Database." Communications of the ACM, Vol. 33, No. 11, pp. 99 – 109.

Ramani, S., 1986. "A simulation approach to time-cost trade-off in project network." Proceedings, Modeling and Simulation on Microcomputers (SCS), pp.115-20.

Rabun, A., and Sommers, J., 1998, " Microsoft Project 98 Support Course," Microsoft Corporation, Redmond, WA.

Reda, R., and Carr, R. I., 1989. "Time-cost trade-off among related activities." Journal of Construction Engineering and Management, Vol. 115, Sept. '89, pp. 475-486.

Render, B., and Stair, R. M., 1991, " Introduction to Management Science," Allyn & Bacon, Massachusetts.

Rickles, H. V., and Elliot, K. A., 1985, "Spreadsheet programs enable quick custom analyses of material handling problems." Industrial Engineering, February 1985, pp. 80-85.

Roller, D., and Eck, O., 1999, "Knowledge-based Techniques for Product Database," International Journal of Vehicle Design, Vol. 21, No. 2, pp 243-265.

Rosenblatt, M.J., and Roll Y., 1985, "A future value approach to determining project duration," IIE Transactions, Vol.17, No.2, pp.164-167.

Ryba, M., and Baitinger, U. G., 1996, " An Integrated Concept for Design Project Planning and Design Flow Control," Proceedings of the Conference with EURO-VHDL'96 and Exhibition on European Design Automation, pp. 98 – 103.

Sabin, M., 1980, "Geometric Modelling," Geometric Modelling and Computer Graphics, R. A. Earnshaw (ed.), The Tech. Press, Aldershot, Hants, England, pp. 50-74.

Sause, R., Martini, K., and Powell, G. H., 1992, "Object-Oriented Approaches for Integrated Engineering Design Systems," Journal of Computing in Civil Engineering, Vol. 6, No. 3, pp. 248-265.

Shadish, B., 1996, "ActiveX---what is this stuff?" VBTech Journal, August 1996.

Shaw, R. H., 1995a, "Understanding OLE: Application Integration," PC Magazine, 11/07/1995.

Shaw, R. H., 1995b, "OLE's Component Object Model," PC Magazine, 11/21/1995.

Shouman, M.A., El-nour, A., Ebrahim, S., and El-mahalawy M., 1991, "A mixed integer linear programming model for a time cost trade-off," Advances in Modelling & Simulation, Vol.22, No.4, pp.53-63.

Silberschatz, A., Korth, H. F., and Sudarshan, S., 1996, "Data Models," ACM Computing Surveys, Vol. 28, No. 1, pp. 105-108.

Singhal, A., Parikh, N., Dutt, D., and Lo, C. Y., 1989, "A data model and architecture for VLSI/CAD databases" Proceedings, ICCAD Conference, pp 276-279.

Singhal, A., Arlein, R. M., and Lo, C., 1993, "DDB: An Object Oriented Design Data Manager for VLSI CAD," Proceedings, The 1993 ACM SIGMOD International Conference on Management of Data, pp. 467 - 470.

Sitton, R. W., 1989, "How microcomputers and electronic spreadsheets can be used to educate industrial engineering students," Proceedings, 11<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 175-179.

Skutella, M., 1998, "Approximation of algorithms for the discrete time-cost trade off problem," Mathematics of Operations Research, Vol. 23, No., 4, pp. 909-929.

Sounderpandian, J., 1989, "MRP on spreadsheets: A do-it-yourself alternative for small firms," Production and Inventory Management Journal, 2<sup>nd</sup> qtr, pp. 6-11.

Spooner D., Michael, A., and Donald, B., 1986, "Modeling CAD Data Abstraction and Object Oriented Techniques," ICDE.

Stefik, M., 1986, "The next knowledge medium," The AI magazine, Vol. 7, No., 1, pp. 34-46.

- Stonebraker, M., 1992, "The integration of rule systems and database systems," IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 5, pp. 415-423.
- Stonebraker, M., Rowe, L., Lindsay, B., Gray, J., and Carey, M., 1990, "Third-Generation Data Base System Manifesto," Memorandum No. UCB/ELR. M90/23, April. The committee for Advanced DBMS Function, Univ. of California, Berkeley, CA.
- Stonebraker, M., Rubenstein, B., and Guttman, A., 1983, "Application of Abstract Data Types and Abstract Indices to CAD databases," ACM SIGMOD/IEEE Engineering Design Applications.
- Stumpf, A. L., Ganeshan, R., Chin, S., and Liu, L. Y., 1996, "Object-Oriented Model for Integrating Construction Product-and Process Information," Journal of Computing in Civil Engineering, Vol. 10, No. 3, pp. 204-212.
- Sunde, L., and Lichtenberg, S., 1995, "Net-present-value cost/time tradeoff," International Journal of Project Management, Vol.13, No.1, pp.45-49.
- Sundgrer, B., 1985, "Data Bases and Data Models," Studentlitteratur AB Chartwell-Bratt Ltd.,
- Taeho, A., and Erenguc, S.S., 1998, "The resource constrained project scheduling problem with multiple crashable modes: a heuristic procedure" European Journal of Operational Research, Vol.107, No.2, pp.250-259.
- Taylor III, B. W., 1996, "Introduction to Management Science," 5<sup>th</sup> ed., Prentice Hall, New Jersey.
- Trevino, J., and McGinnis, L.F., 1986, "Electronic spreadsheet implementation of a lot sizing procedure for a single-card kanban system," Proceedings, 8<sup>th</sup> Annual Conference Computers & Industrial Engineering, pp. 340-345.
- Tseng, H., Chishaki, T., Tatsumi, H., and Shih, W., 1996, "An Optimal Time-Cost Trade-Off Procedure of Project Scheduling under Fuzzy Activities Duration and Fuzzy Cost Estimation," Proceedings, International Conference: Urban engineering in Asian cities in the 21st century, pp. 259-264.
- Tufekci, S., 1982, "A flow-preserving algorithm for the time-cost trade-off problem," IEE Transactions, Vol.14, No.2, pp.109-113.

Ullman, J. D., 1987, "Database Theory: Past and Future," Proceedings, 6<sup>th</sup> ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp. 1-10.

Wang, S., 1999, "An Object-Oriented approach to plant configuration management information systems analysis," Industrial Management and Data Systems, No. 99/4, pp. 159-164.

Whitehouse, G. F., 1984, "Electronic spreadsheets and their applications," Proceedings, Annual International Industrial Engineering Conference & Societies, pp. 99-104.

Williams, H. P., 1999, "Model Building in Mathematical Programming," 4<sup>th</sup> ed., John Wiley & Sons, LTD, New York.

Wong, S. T. C., 1993, "COSMO: A communication scheme for cooperative knowledge-based systems," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23, No. 3, pp. 809-824.

Zhu, J., and Maier, D., 1989, "Computational Objects in O2 data models," Proceedings, the 2<sup>nd</sup> International Workshop on Database Programming Languages.

Zimmerman, S. M., and Gibson, D. R., 1989, "A proposed method to use electronic spreadsheets to develop quality control charts," Proceedings, 11<sup>th</sup> Annual Conference Computers in Industrial Engineering, pp. 384-389.

Zorn, B. G., and Chaudhri, A. B., 1995, "Object Database Behavior, Benchmarks, and Performance Workshop Addendum," Addendum to the Proceedings of the 10<sup>th</sup> Annual Conference on Object-Oriented Programming Systems, Languages, and Applications (Addendum), pp. 159 – 163.

## APPENDIX

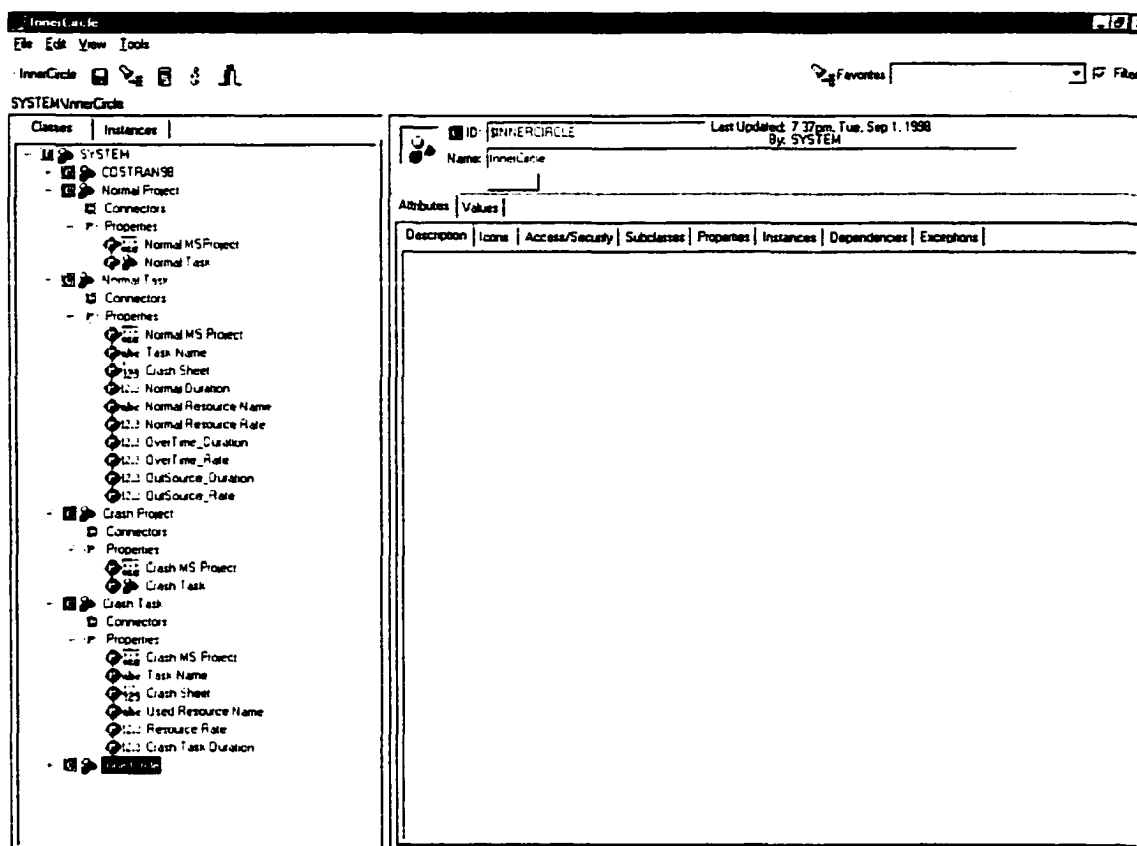


Figure a. 'Normal Project' and 'Crash Project' Class Tree

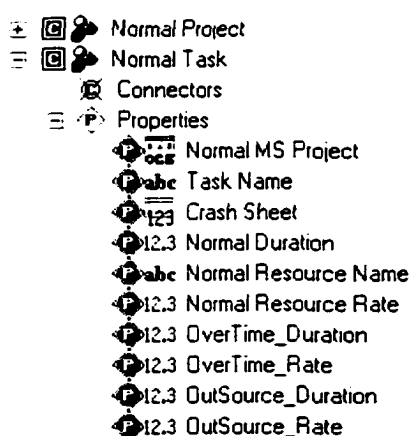


Figure b. 'Normal Task' Class Structure

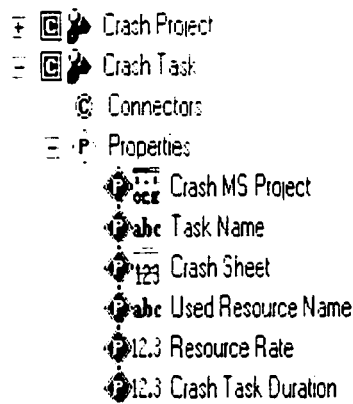


Figure b. 'Crash Task' Class Structure

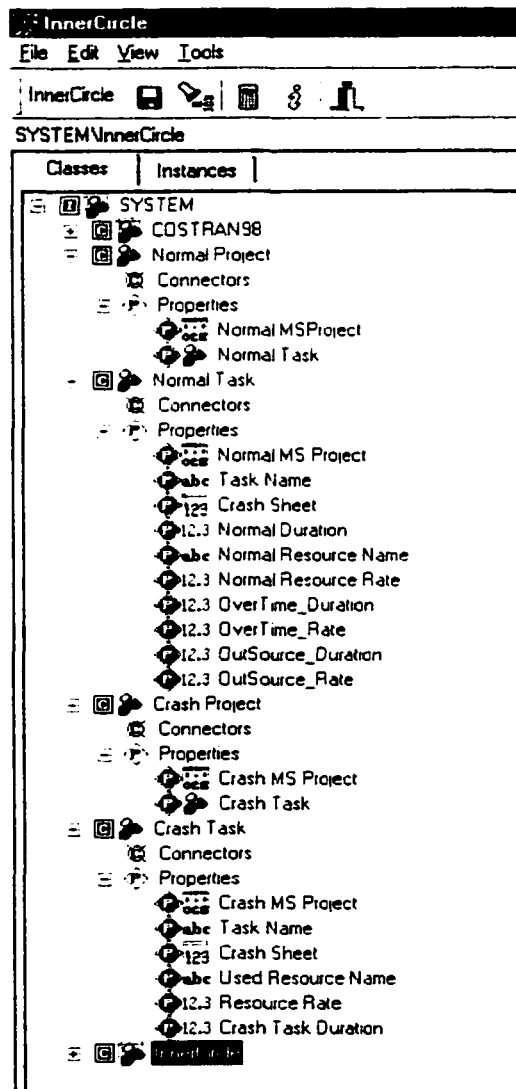


Figure d. 'Normal Project' and 'Crash Project' Class Tree. A Detailed View

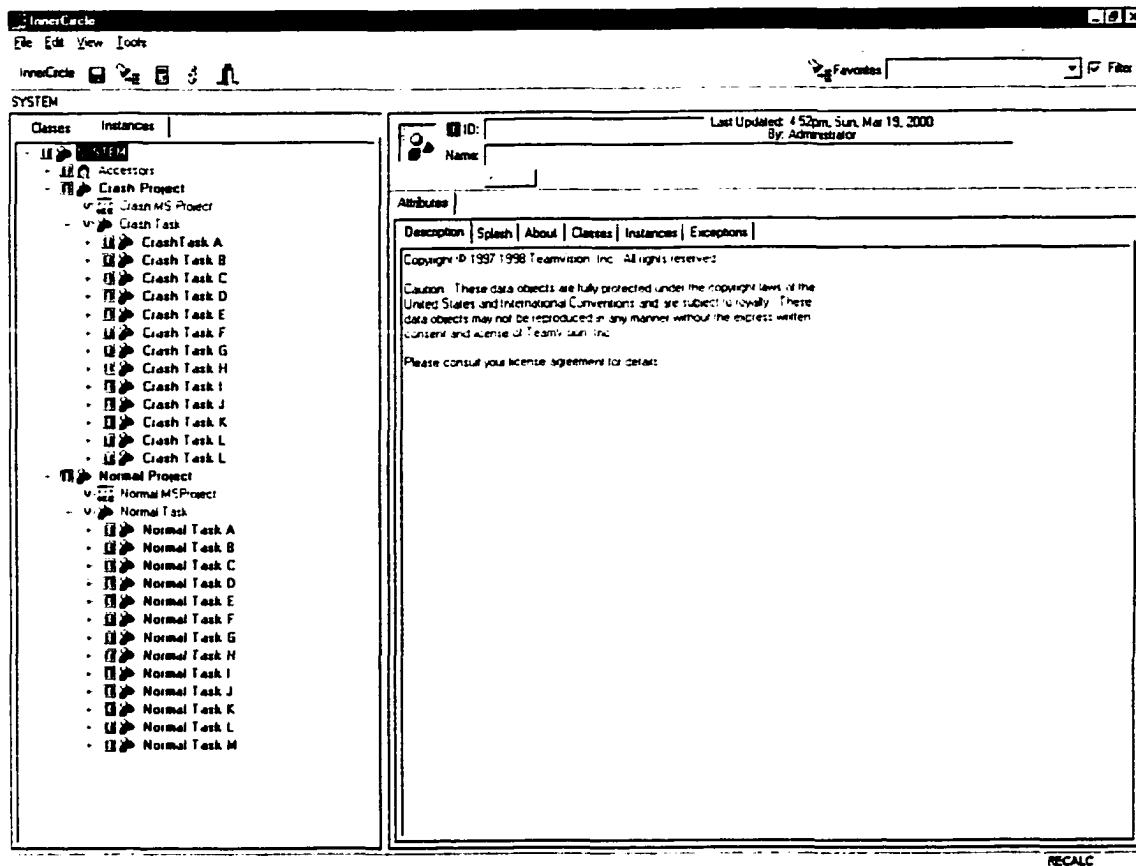


Figure e. Instance Diagram

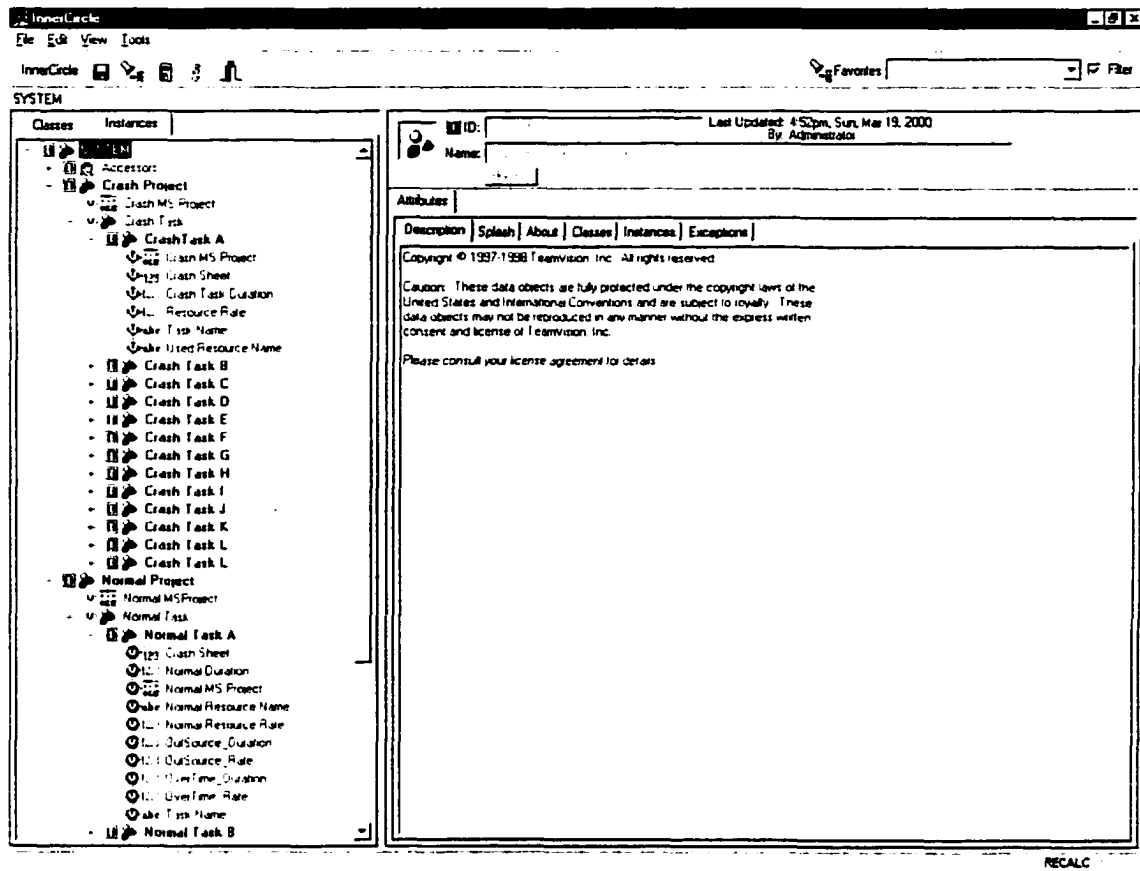


Figure f. Instance Values Assignment



## VITA

Hisham M. E. AbdelSalam was born on November 1<sup>st</sup>, 1973 in Giza, Egypt. He graduated from the Faculty of Engineering, Cairo university with a Bachelor of Science degree in Mechanical Design and Production Engineering in July 1996 by a very good with honor degree. After his graduation he worked as a teaching assistant at The Higher Technological Institute. At November 1997 he started his work as a teaching assistant at Operations Research and Decision Support Department, Faculty of Computers and Information, Cairo University. Besides working as a teaching assistant, he participated in quality improvement and pollution prevention consultations for several Egyptian companies. In May 2000, he received his Masters of Science degree in Mechanical Engineering from the College of Engineering and Technology, Old Dominion University. He has published two conference papers during his academic studies, and he is currently working on three other papers. He is currently working as a research/teaching assistant at the Department of Mechanical Engineering, Old Dominion University.