The Practical Guide to Business Process Reengineering Using IDEF0

Clarence G. Feldmann

Foreword by John V. Tieso

The Practical Guide to Business Process Reengineering Using IDEFO

Also Available from DORSET HOUSE PUBLISHING CO.

Find Out More about These and Other DH Books:

The Practical Guide to Business Process Reengineering Using IDEFO

Clarence G. Feldmann

Foreword by **John V. Tieso**

Dorset House Publishing
353 West 12th Street
New York, New York 10014

Library of Congress Cataloging-in-Publication Data

Feldmann, Clarence G.
The practical guide to business process reengineering using IDEF0
/ Clarence G. Feldmann ; foreword by John V. Tieso.
p. cm.
Includes index.
ISBN 0-932633-37-4 (softcover)
1. Reengineering (Management) 2. Electronic data processing--Structured techniques. 3. Business--Graphic methods. I. Title.
HD58.87.F45 1998
658.4'063--dc21 98-20236
CIP

Cover Design: Jeff Faville, Faville Design
Cover Author Photograph: Joe Demb, Belmont, Massachusetts

Distributed in the English language in Singapore, the Philippines, and Southeast Asia by Toppan Co., Ltd., Singapore; in the English language in India, Bangladesh, Sri Lanka, Nepal, and Mauritius by Prism Books Pvt., Ltd., Bangalore, India; and in the English language in Japan by Toppan Co., Ltd., Tokyo, Japan.

Printed in the United States of America

Library of Congress Catalog Number: 98-20236

ISBN: 0-932633-37-4 12 11 10 9 8 7 6 5 4 3 2 1

Digital release by Pearson Education, Inc., June, 2013

Dedication

I dedicate this book to Douglas T. Ross, the father of SADT and IDEF0, with whom I worked for more than thirty years. The concepts presented here are his. I have tried to present his ideas as I have come to understand and apply them to the real world. I apologize to him for any misleading statements herein, or for dabbing on his canvas, as he used to put it.

This page intentionally left blank

GRAPHICS CREDITS

Many different graphics software packages were used to produce the figures in this book. Thus, the reader will see several variations in style when reading the IDEF0 diagrams presented here. The use of a particular package to produce a specific form of IDEF0 diagram is not intended to imply endorsement by the author. In most cases, the package used to develop an IDEF0 model was determined by the product used by the client for whom the model was developed.

The following graphics packages were used to produce the diagrams in this book, listed in alphabetical order of the package name:

Authormate, by Eclectic Solutions, Inc.
AutoCad, by Autodesk, Inc.
BPwin, by Logic Works, Inc.
Design/IDEF, by Meta Software, Inc.
SA/BPR, by Popkin Software and Systems, Inc.

This page intentionally left blank

CONTENTS

FIGURES & TABLES

FOREWORD

Over the past several years as I have traveled just about everywhere to discuss business reengineering, it has amazed me that most people think reengineering is a new phenomenon created by Mike Hammer. Some have been dumbfounded when I say how long it has been around and what it has accomplished during that time. I certainly give credit to Dr. Hammer's book for the publicity reengineering has received, but it is the unsung heroes that have made reengineering not just well known but successful.

One such hero is Clare Feldmann. He and I have been talking about and through his book for at least five years. Clare is one of those guys who prefers the background to the limelight. His years at SofTech with Doug Ross were the tough years as reengineering emerged into a modern method. More importantly, Clare was one of the pioneers in the belief that understanding processes and activities is the key to the application of technology to problems. This book shares with you many of his thoughts on how to effectively reengineer.

I will share one personal insight for you to bear in mind as you read the chapters that follow. Over the years, Clare and I have shared a number of opportunities, committees, boards, and so forth. At each, he takes very careful notes. In fact, more often than not, he converts them to IDEF0 activity diagrams and data diagrams. He has always claimed that he understands subjects better through models. By this time, he must have thousands of them. While I have no secret knowledge here, I will bet that a model exists for how to write a book. Ask him when you meet him.

Clare's love of models led to a deep understanding of how processes work. A lot of what we now say and do in reengineering efforts comes from that approach: Understand what you do now and look at what you can do to improve the process. When you model the reengineered process, it quickly becomes obvious how technology can help to further optimize the efficiency of the new process. All of this is quite logical—yet completely impossible to achieve without models. Clare's favorite technique is shown here—IDEF0—the integrated definition language that he helped create in the 1970's. That modeling technique captures information not obtainable otherwise. Understanding the controls on actions as well as the mechanisms that facilitate action is critical to creating business rules that clarify action.

Another benefit of Clare's effort is the timing. For those of us in Government, rapid change is upon us. The Information Technology Management Reform Act of 1996 now mandates that managers redesign processes before recommending information technology investments. In fact, this is a first step in bringing accountability to the hundreds of billions of dollars expended each and every year on technology in Government. With IDEF0, Clare shows managers a proven way to assure that those investments make sense.

Any time someone writes a book on reengineering, I cheer. Particularly so when the person writing the book has gotten his hands dirty with real work, in helping managers improve. We have too many volumes out there that discuss others' efforts, written by people who have never tackled a project themselves. Here you have a master who is describing his art—and the only charge for his services is the cost of the book. What a bargain!

If you only have time to read one book on the subject, this is it. Enjoy.

John V. Tieso
Former Deputy Director, Office of Functional Process Improvement
Office of the Assistant Secretary of Defense (OASD) for Command and Control, C3I
Arlington, Virginia

PREFACE

For many years, there has been a call for a comprehensive book on the IDEF0 method and its use in BPR. This book, written in answer to that call, is designed to introduce the method, its usage, and the benefits that can be achieved. It is also written to serve as a "do's and don'ts" document to identify the ways the method has been misused in the past and to specify ways to correct these misuses.

This book presents essential information about the IDEF0 method—its definition, basic rules of usage (including the standard language syntax and semantics as contained in the Federal Standard)—and lessons learned from many years of application in the real world. The examples depict actual commercial client and government agency models, from which names and proprietary information have been removed.

Who Should Read This Book

The material presented in this book is suitable for three categories of IDEF0 users:

- people who wish to explore the concepts and the application potential of IDEF0 for enterprise reengineering
- people who wish to determine how IDEF0 might be helpful to them in their systems analysis efforts

- people who have been introduced to IDEF0 concepts, and who wish to understand the theory behind the concepts so that they can use the method more effectively

The Department of Defense (DoD) Corporate Information Management (CIM) Office reports that more than 600,000 people worldwide fall into the third category, and one goal of this book is to provide meaningful assistance to such people during their IDEF0 analysis efforts.

After reading the book, a person should be familiar enough with IDEF0 to read and understand any IDEF0 model and should be able to understand how the method might be applied to enterprise or systems analysis and what goals and benefits are reasonable to expect from its application. However, to be considered an "author" of IDEF0 models, he must complete additional formal training and a brief apprenticeship period with an experienced IDEF0 author.[1]

Training courses in IDEF0 modeling are available from commercial vendors and at various universities, primarily in the United States;[2] SADT training courses are available worldwide.[3] In addition, there are more than a dozen good computer tools that support the IDEF0 method.

My hope is that this book will guide a new generation of modelers in producing the maximum benefit from IDEF0.

Winchester, Massachusetts C.G.F.
April 1998

[1] Throughout this book, the male pronoun is used generically to include both males and females. Also, please note that in IDEF0 terminology, a *reader* is someone who knows the rules of the IDEF0 graphics and can therefore understand the diagrams. An *author* is someone who is a reader but who also is trained in creating IDEF0 models.

[2] For a list of registered vendors of courses and computer support tools for the IDEF family, write to Society for Enterprise Engineering, ATTN: Secretariat, c/o The Koop Foundation, 15825 Shady Grove Road, Suite 22, Rockville, MD 20850.

[3] SADT is the precursor to IDEF0. For more on SADT and the history of IDEF0, see Chapter 1 and Appendix C.

The Practical Guide to Business Process Reengineering Using IDEFO

This page intentionally left blank

CHAPTER 3

Features and Benefits of the IDEF0 Activity Modeling Method

This chapter presents the basic principles of IDEF0—the assumptions behind its graphics and the rules for applying its analytical process. Expanding on Chapter 1, the discussion examines the scope of topics that can be handled by IDEF0, the benefits of its use, and a comparison to other popular graphical diagramming methods. Chapter 4 gets even more specific as it presents the detailed syntax and semantics of the IDEF0 graphical language, as officially defined in the Federal Standard.

The Seven Basic Principles

There are seven principles—or rules of conduct—that must be followed in order to apply IDEF0 techniques successfully:

1. ***The method must accurately represent the problem area:*** A graphical model of a system should be developed so that the system elements and their interactions can be defined, documented, communicated, discussed, and analyzed effectively.
2. ***The model must have a top-down, modular, hierarchical structure*:** The model should depict the system top-down by defining modular system elements that interact to form a hierarchy.
3. ***The model must separate function from design:*** By definition, *what* the system does (function) must be kept separate from *how* it does it

(design). That is, more than one design can be developed for a single function; this distinction is important. Because of this, design improvements may be developed without disturbing the basic function. If a more drastic (functional) change is called for, then basic functions can be changed and a new design created to meet the new functional requirements.

4. ***The model must reflect both objects and actions—things and happenings***: The modeling method must be able to depict all forms of things and happenings. Restriction of the modeling scope to raw data would leave out the people, resources, raw materials, human factors, and other important influences on the operation of the enterprise, all of which are critical in managing change.
5. ***The model form must be graphical:*** The form on which the model is recorded must be graphical, not mathematical or textual. The graphical form must communicate concisely and rigorously to the people who must validate that the model reflects the real functions and processes of the enterprise.
6. ***The model must be the product of disciplined, coordinated teamwork:*** To build a model and to achieve consensus among the enterprise staff requires disciplined, coordinated teamwork. The IDEF0 method must therefore contain the working rules and procedures for developing and validating the model in an organized way (reader/author cycle forms and procedures, librarian configuration management procedures, and the like).
7. ***The model must present all information in writing:*** By having a standardized form on which all information (interview notes, conclusions with their rationale, definitions of terms, and so on) is retained, the method assures that information is not lost during the early stages of the planning cycle. With its forms and procedures, IDEF0 provides a convenient means of retaining information.

Scope of Subject Matter That Can Be Handled by IDEF0

As we have stated, IDEF0 is a method for analyzing enterprises. An enterprise may be a company, a division of a company, a group of companies working together in a joint venture, a field office of a parent company, a support function within an office (for example, publications, accounting, or payroll), a government agency (for example, the Department of Defense or the Commerce Department), an individual running a small business, or a project within a company. All of these are enterprises, since they are composed of one or more people working to accomplish some form of product development or to provide some service.

The scope of the analysis encompasses manual as well as computerized aspects of the enterprise. IDEF0 is particularly useful for rigorously defining the juncture at which manual actions interface with the computer (the so-called man-machine interface). For example, using the *viewpoint* concept of IDEF0, the author may develop a model from the user's viewpoint as well as from the software developer's viewpoint, thus providing a requirements definition from both important aspects of a software system to be developed. The model bridges the communication gap between user and software cultures.

IDEF0 analysis does not stop at the man-machine interface, but may be used to model the operation of the software system itself. In fact, the method has been successfully used for many years to model the requirements, top-level design, and detailed design of software systems.

Benefits Resulting from the Use of IDEF0

The primary goal of IDEF0 is to facilitate change management, a goal that was introduced in Chapter 2, under the topic "Use of IDEF0 for Military Downsizing and Reorganization." In this chapter, we discuss specific benefits that can be expected from the IDEF0 modeling effort.

Stated simply, the goal of the modeling effort must not stop at developing a model. Such a goal leads to models that sit on a shelf and do not achieve real operational benefits. Instead, the purpose of an IDEF0 model should be "to provide a baseline on which to plan and manage change." The model serves as a communication vehicle for streamlining existing processes, for introducing new technology, for evaluating the impact of a change on related processes, and for achieving consensus among the staff involved in the change.

The initial benefits are achieved by the analyst as he creates (authors) the model. In developing the model, the analyst cannot help but identify problems and potential improvements in the operation of the enterprise. When the analysis is complete, the paper model is available, thereby providing documentation on-the-fly and not requiring a separate, after-the-fact documentation phase at the end of the analysis. Specific benefits of IDEF0 modeling are as follows:

> *Identifies Needs:* A good IDEF0 model will look very simple and easy to understand. Needs and opportunities for improvement revealed during the modeling effort may seem obvious at first, but they would have gone unnoticed otherwise.

Builds Consensus: The diversity of a staff's background, training, skills, and knowledge can impair communication. The reader/author model critique method can establish a well-defined common basis for understanding by the entire staff. Real consensus, not just imagined consensus, can be reached.

Enhances Vision: The condensed, graphical picture provided by the model presents a thought-provoking baseline on which to consider improvements. An analyst or an enterprise planner who considers potential use of a new technology or new method in light of this baseline may identify specific application opportunities.

Provides a Basis for an Open Architecture: The model structure may be used to define the interfaces between system elements, and to identify precise interfaces for the definition of an open systems architecture. The model bounds the scope precisely by showing where the modeled system fits into the bigger picture.

Broadens Automation and Commonality Potential: Almost any model of any size will reveal activities that are similar to those already modeled in other parts of the structure. These similarities can be used to define improvements, such as a subsystem to support these common activities.

Supports Management Control Through Metrics: The model may serve as a baseline for cost, time, flow capacity, peak loads, and other metrics relevant to the design of a new system. If the model detail is not sufficient to attach specific metrics, the lack of detail should be used to identify where further decomposition is needed. Finer detail in the modeling will facilitate more precise measurement.

Defines Variants for Broader Support System Use: Support systems that are applied to several activities in the model may benefit from a careful analysis of the actual functionality needed at each point of usage. Variants and versions of the support system could save considerable cost if the support system can be fine-tuned for each specific usage point.

Features of IDEF0 Analysis

The IDEF0 method analyzes a subject by investigating how it operates, by modeling the operational details uncovered, and by applying evaluation techniques (such as "value added" criteria) to identify potential improvements. The author of the model uses two-dimensional, simple graphics as a means of capturing and communicating the results of the analysis.

Applying IDEF0 to the analysis of an enterprise thus results in a paper, two-dimensional graphical model of the enterprise. The model is hierarchical in nature, consisting of a set of diagrams in which each diagram represents three to six activities of the enterprise. At the top of the IDEF0 model, the enterprise is decomposed into three to six very large activities or subsystems that represent the activities of the enterprise (see Figure 3-1). These activities are shown as rectangular boxes.

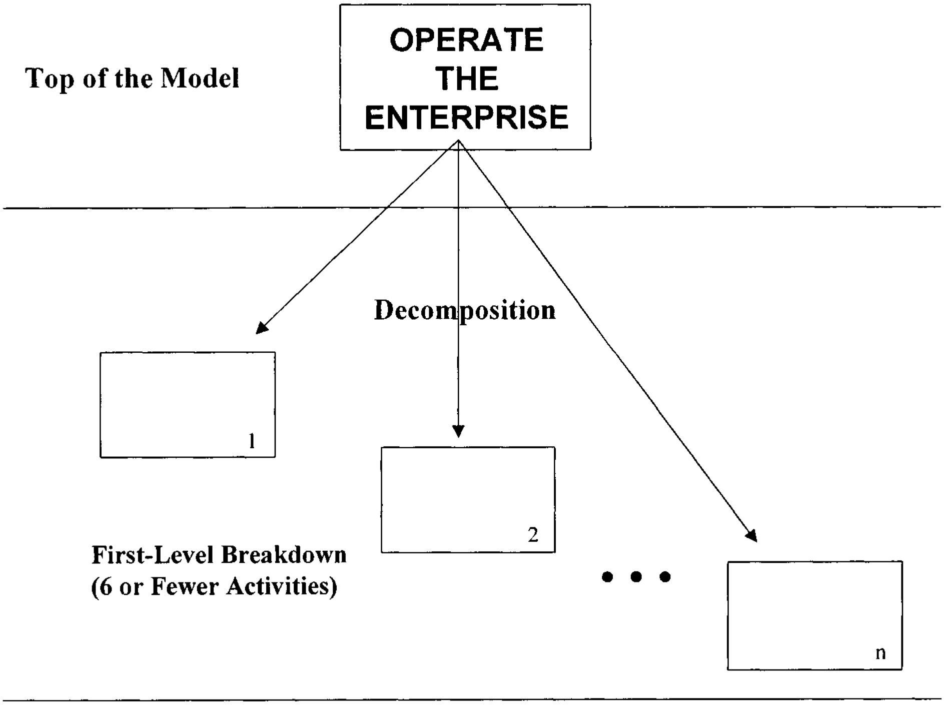


Figure 3-1: Breakdown into Six or Fewer Activities.

Once the top-level activity breakdown has reached consensus among the enterprise staff, each of the boxes is further broken down into the next level of detail on a new three-to-six activity diagram. Thus, the entire IDEF0 model provides a tree structure of diagrams. Anyone wishing to inspect a particular aspect of the enterprise may scan the tree structure, identify the activity and level of detail that is useful for his purposes, and then retrieve the corresponding IDEF0 diagram to study the complete description of the selected activity at the selected level of detail. Figure 3-2 shows a graphical representation of an IDEF0 model that breaks down an enterprise into three levels of detail.

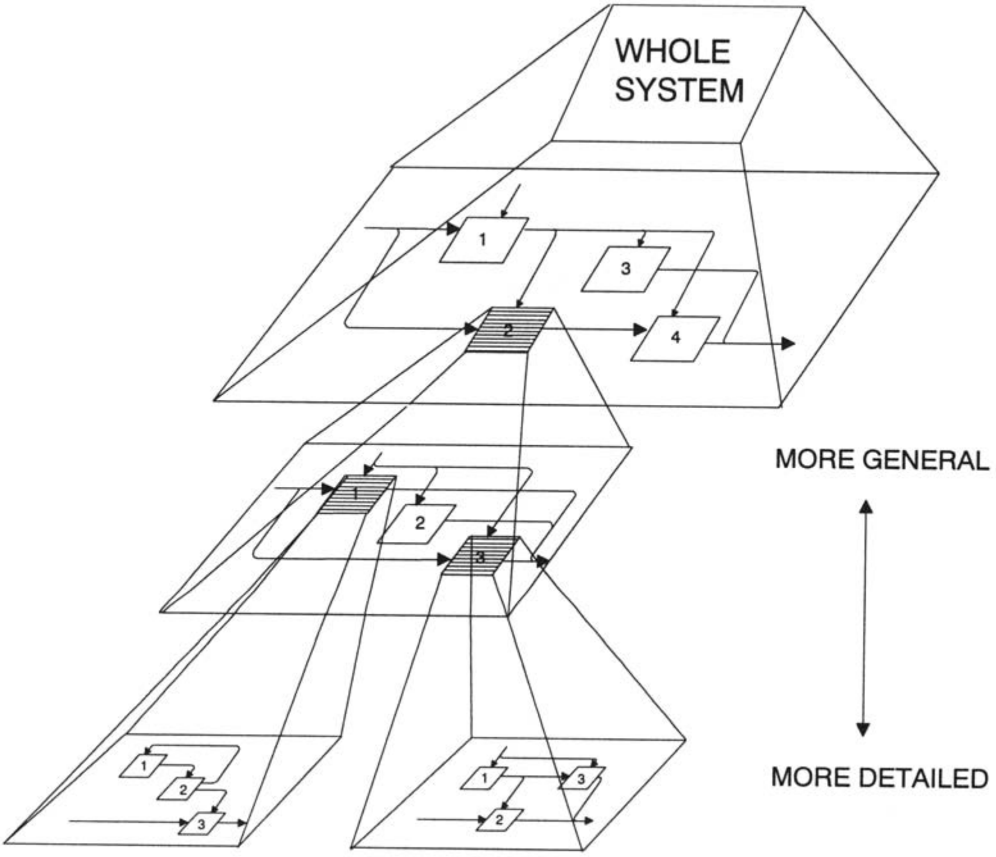


Figure 3-2: Top-Down IDEF0 Model Structure.

At the top level, the entire system being modeled (labeled Whole System in the figure) is shown as broken down into four parts, labeled 1, 2, 3, and 4. Part 2 is further broken down into a three-part diagram, shown below the initial diagram as emanating from a shaded box 2. Finally, boxes 1 and 3 of the detailed part 2 box are broken down into yet finer detail in the two diagrams displayed at the bottom of the figure. This illustrates an important feature of the IDEF0 model: Any aspect may be broken down to be as detailed as necessary, while other aspects may be less completely decomposed. This permits the analyst to zero in on the most critical aspects to be analyzed, while putting off the other aspects until needed.

Comparison to Data Flow Diagramming

Most popular in past decades, the data flow diagram approach to modeling a computer software system is still used today for software design. It provides a picture of the data that are input to and output from the software. Figure 3-3 shows that Location of Assembly and Quality Measurements are input to a Scheduling Database of information about the status of a part. When a process has been completed, the assembly machine sends a "Process Complete" Status signal to the database, as an additional input. Thus, the database has a running record of where the part is located and which processes have been performed. As output, the software produces a report on the status of the part.

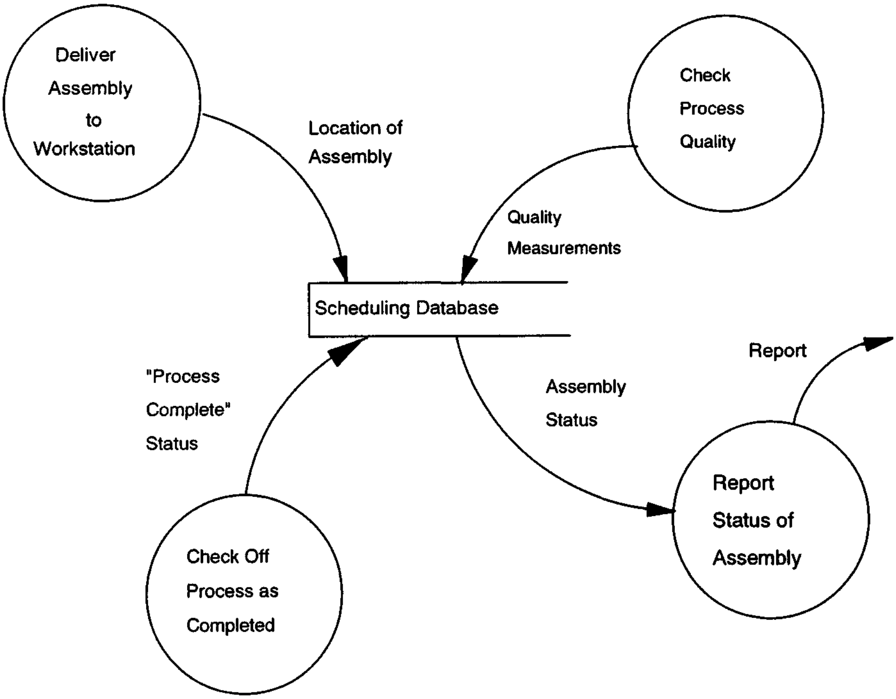
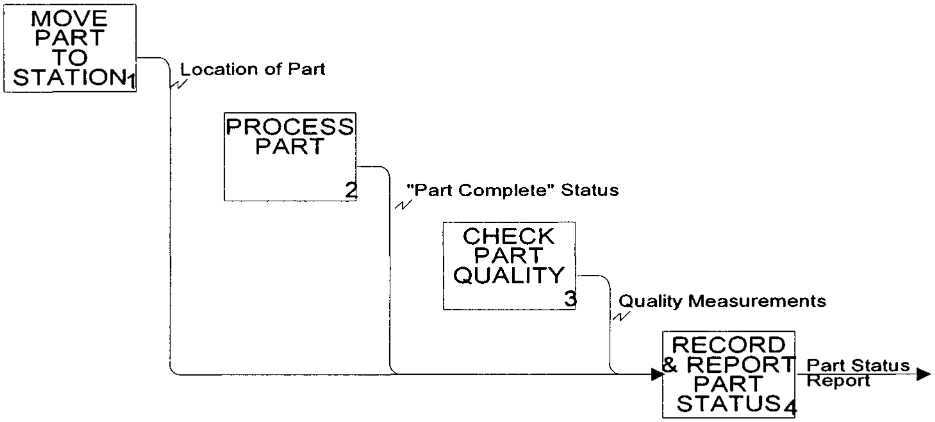

Deliver Assembly to Workstation
Location of Assembly
Check Process Quality
Quality Measurements
Scheduling Database
"Process Complete" Status
Report
Assembly Status
Report Status of Assembly
Check Off Process as Completed

Figure 3-3: A Data Flow Diagram to Track Assembly Status.

Figure 3-4 shows the same information as Figure 3-3, but this time as an IDEF0 diagram.



Equivalent Information to the Data Flow Diagram

Figure 3-4: Same Track Assembly Status Data Flow Using IDEF0 Format.

Note that the interfaces between the activity boxes in the IDEF0 diagram are represented by arrows, depicting one activity supplying input to or controlling one or more other activities (as we shall see in Figures 3-5 through 3-7). Unlike the simpler data flow diagram, the interactivity relationship can include data flow, raw material flow, management and other forms of control, resources (such as people, machines, and computers), and whatever else must be included to model the complete enterprise's operational details.

Figures 3-5 through 3-7 detail information that is not shown on the data flow diagram. Whereas a data flow diagram is adequate to depict information needed by a computer software program, it is not adequate for reengineering purposes, since there are other factors influencing an enterprise's operations. These other factors must be included if the reengineering process is to be understood and designed, and the changeover process controlled. The IDEF0 model shows what controls each activity and who performs it, as well as the resources needed by each activity.

Figure 3-5 shows the addition of the physical part proceeding through the stages of manufacturing, from Unassembled Part entering box 1 to Completed Part leaving box 3.

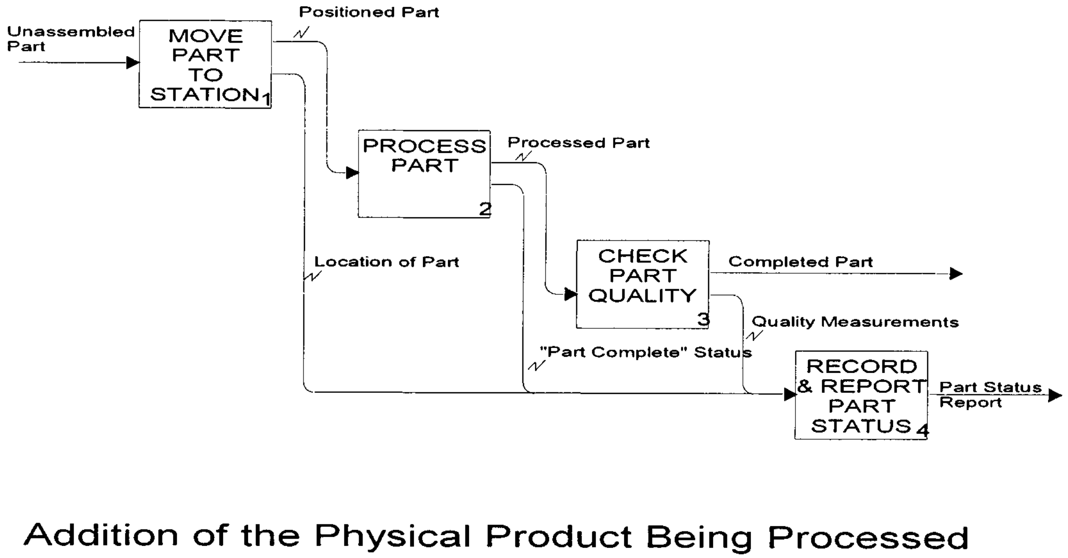
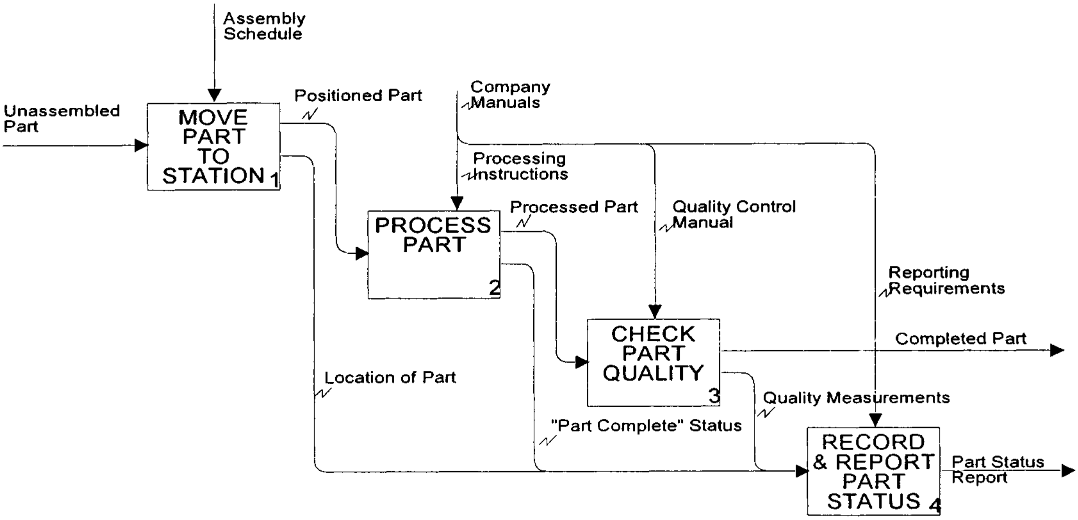


Figure 3-5: Assembly Status Tracking IDEF0 Diagram, with Physical Parts Assembly Added.

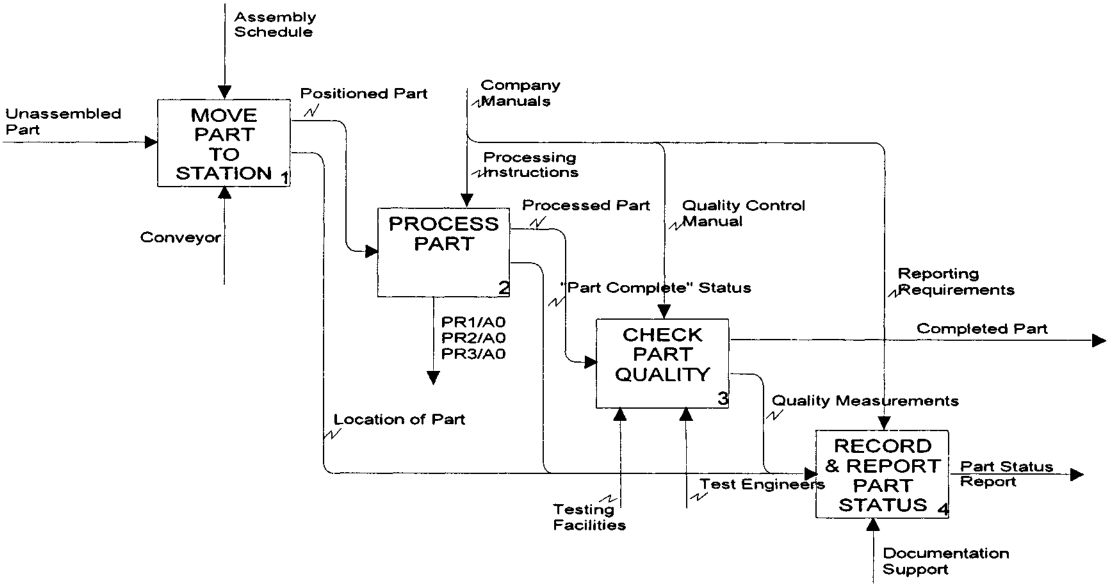
Figure 3-6 adds controls to each IDEF0 activity box. For example, we see that there is an Assembly Schedule that controls when the part is prepared for processing. Company Manuals are used to control the remaining three activities. These controls must be carefully considered when making changes, and therefore are important to include in the reengineering effort.

Finally, Figure 3-7 shows what mechanism performs each process. This includes people (for example, Test Engineers for box 3) as well as support tools and systems (for example, Testing Facilities for box 3). A common practice is to document the elements of the enterprise's organization chart as mechanisms in the model in order to show the assignment of responsibility to specific organizational elements. In the TO-BE picture, the revised organizational responsibility is likewise shown.



Addition of Control Information -- What Controls Each Activity

Figure 3-6: Previous Diagram, with Controls Added.



Addition of Mechanisms -- What Performs Each Activity

Figure 3-7: Complete IDEF0 Diagram, with Mechanisms Added.

An important difference between data flow diagrams and IDEF0 diagrams is the scope of the topic covered: DFDs were intended to depict fine detail needed by a software design specification, whereas IDEF0 diagrams were designed to capture very complex topics such as the operation of the aerospace industry, a government agency, or a business.

IDEF0 syntax may be used to capture fine detail, as we have just shown in the series of diagrams comparing the information content of DFDs to IDEF0 diagrams. However, many of the features of IDEF0 have the objective of handling great complexity, not depicting fine detail. The purpose and viewpoint statements of an IDEF0 model are examples of features that enable handling complexity.

While the purpose of a set of data flow diagrams is to depict the operation of computer software, the purpose of an IDEF0 model may be one of many, since the modeling method is applicable to any "system" comprised of "things" and "happenings." Focusing the analysis effort by purpose shortens the list of enterprise elements that must be modeled during a reengineering effort. In other words, the analyst must narrow the topic to be modeled and decrease complexity.

An IDEF0 model must also select a specific *viewpoint.* If all viewpoints of an enterprise were to be included in a single model, the information would be too complex to understand.

Understanding a Top-Level IDEF0 Diagram of an Enterprise

Figure 3-8 presents a typical top-level diagram in an IDEF0 model (it originally appeared as Figure 2-9). This diagram might represent the most general level of detail of a model entitled "Run the Enterprise."

The figure shows the running of an enterprise, broken down into five major functions: 1) Manage the Enterprise; 2) Manage Day-to-Day Operations; 3) Manufacture Products; 4) Provide Management & Technical Information; and 5) Provide Resources. Four of these major functions are shown as boxes positioned in a stairstep fashion that makes the control and feedback loops between the functions simpler to draw. The fifth box (Provide Resources) is shown below the stairstep, since it is a support function that provides resources (people and machines) to be used by the other functions.

By reading the names in the boxes, anyone can get a quick idea of the breakdown of the subject Run the Enterprise, as well as the topics to be modeled in subsequent lower-level diagrams. For example, if the reader wishes to examine further detail regarding the provisioning of resources, he would isolate the diagram for box 5 and study that at a finer level of detail.

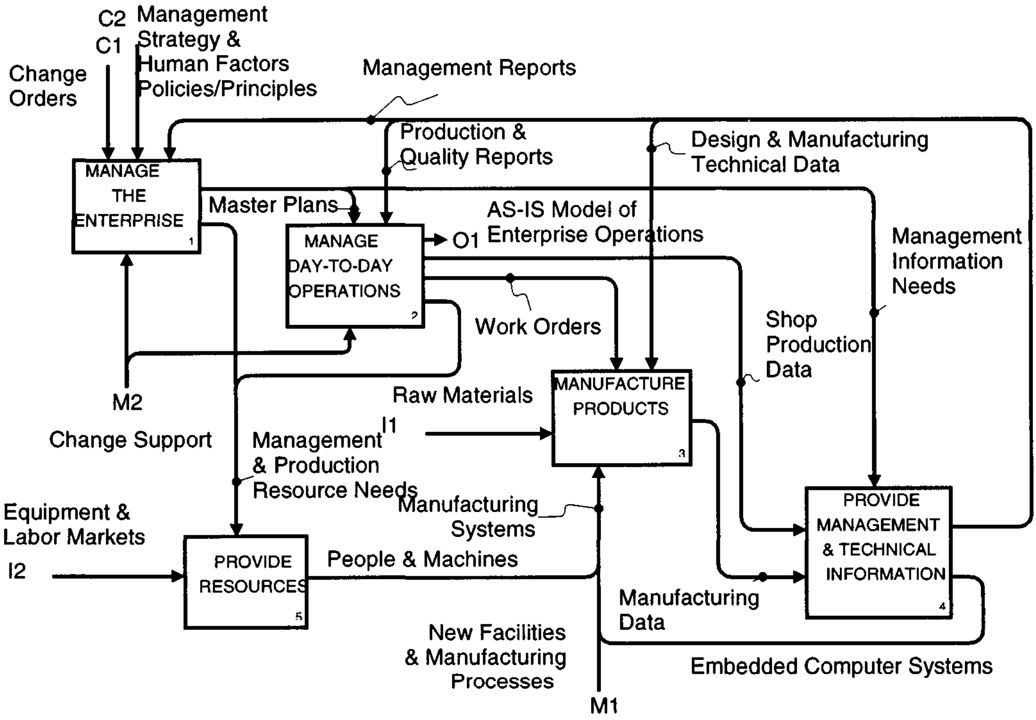

C2 Management Strategy &
C1 Human Factors Policies/Principles
Change Orders
Management Reports
Production & Quality Reports
Design & Manufacturing Technical Data
MANAGE THE ENTERPRISE 1
Master Plans
AS-IS Model of Enterprise Operations
O1
MANAGE DAY-TO-DAY OPERATIONS 2
Management Information Needs
Work Orders
Shop Production Data
M2
Change Support
Raw Materials
I1
Management & Production Resource Needs
MANUFACTURE PRODUCTS 3
Equipment & Labor Markets
I2
Manufacturing Systems
PROVIDE RESOURCES 5
People & Machines
PROVIDE MANAGEMENT & TECHNICAL INFORMATION 4
Manufacturing Data
New Facilities & Manufacturing Processes
Embedded Computer Systems
M1

Figure 3-8: A Typical IDEF0 Diagram.

IDEF0 graphics are undeniably straightforward and that quality is, in my view, one key to their success. Such elaborations as additional box shapes, dotted arrows, and formalisms for arrow branching structures have been experimented with by modelers since the inception of IDEF0. It seems only natural to conclude that such elaborations would provide additional expressive capabilities to the analyst.

However, I have learned to resist the temptation to elaborate the syntax, since this makes the graphics too complex to fulfill the essential role of the model to communicate. At the early stages of project planning, communication is the key element, and the simple graphics that are quickly learned and employed must remain a basic element of IDEF0.

Levels of Abstraction in IDEF0 Models

As stated, the breakdown approach used in IDEF0 modeling decomposes selected activities into finer and finer levels of detail. The resultant model set includes sufficient detail to plan and control the implementation of changes to the enterprise.

An IDEF0 model may also contain multiple levels of abstraction. To understand what is meant by "level of abstraction," let's consider a typical enterprise. Each level of management has its natural level of abstraction. Top management must think in broad terms, taking into account many viewpoints as well as abstractions of the details of what is going on in the enterprise. An upper-level manager cannot possibly understand every detail of all of the activities in the entire enterprise, but he must abstract business processes and characteristics that are important at his level of abstraction. He must consider such things as overall profit, hiring policies, or the quality of the enterprise's product line, rather than the specific factors that go into these high-level abstractions.

Conversely, a manager at a lower level needs fine-level detail about a specific topic from a single viewpoint, and he may get uneasy thinking in general terms. His level of abstraction is concerned with such things as the cost of raw materials, the capabilities of specific staff members, or the quality of the service for which he is responsible.

The model's level of abstraction is independent of its level of detail. That is, after selecting a level of abstraction to model, the author may decompose the activities to whatever level of detail he requires to satisfy the purpose of the model. However, if the purpose of the model requires information at a different level of abstraction, no amount of further detailing at the present level of abstraction will provide the needed information. Modeling at all required levels of abstraction must be completed to satisfy the enterprise analysis requirements.

The IDEF0 concept of a mechanism is different from the concept of level of abstraction. In IDEF0, a mechanism represents who performs the activity and what tools (software packages and equipment) are required to perform the activity. In other words, the mechanism identifies the resources needed to perform the function. A mechanism is therefore at a lower level of abstraction than the activity box it implements.

The fact that the mechanism depicts a lower level of abstraction does not mean that it cannot be modeled in IDEF0. It just means that the mechanism is shown as an arrow entering the bottom of the box, and that a separate model must be examined to understand how the mechanism works (see Figure 3-9).

The IDEF0 diagram in Figure 3-9 shows Activity A23 (Edit Document) using the word processor support mechanism. The IDEF0 model for the word processor is shown at the lower-right corner of Figure 3-9. What this tells us is that any word processor mechanism may be modeled and plugged into the diagram to show the precise activity performed when editing a document with any variety of word processors. This shows *how* the editing is done using the mechanism, and therefore indicates that there is a drop in level of mechanization from *what* is done to the document by the EDIT activity.

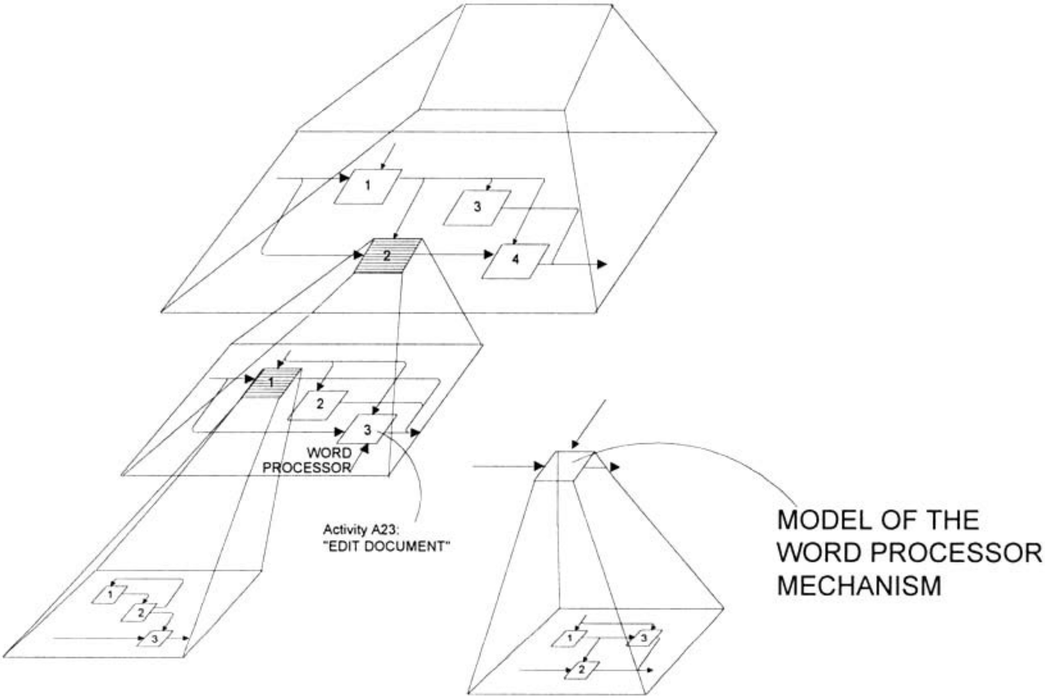


Figure 3-9: Mechanism at a Lower Level of Abstraction.

To accomplish the reengineering of an enterprise, an analyst may need to show multiple levels of abstraction and several levels of mechanism, as well as how those are integrated to perform the processes of the enterprise. Showing all this provides a clear, accurate *big picture* of the overall operation of the enterprise.

In addition to an IDEF0 process-oriented analysis, various additional analysis methods may be applied, using the IDEF0 model as a baseline. For example, the analyst may need to know where costs and labor are being expended, in order to estimate the return on investment of potential changes or to understand the impact of introducing new technology. Various analyses, such as activity-based costing (ABC) and work flow simulation, are typically applied using the IDEF0 model as a baseline.

The Role of Data Analysis Compared to IDEF0 Function Analysis

Detail needed regarding the processes requires not only IDEF0 process models, but also precise information about the arrow content. This requires a careful analysis of data.

An analyst may be confronted with a tidal wave of raw data when he first gathers information about an enterprise. He typically starts by finding out the precise meaning of the terms that are key to the system's operation; then he documents these definitions. Once he has done that, he has a solid basis for initiating discussion with the staff about the detailed operation of the system. But even before analyzing the system in detail, he must consider the system functions and their interactions to understand the need, the purpose, and the objective of the relevant data.

Not all analysts take this view. In fact, some prominent leaders in the database community have stated that there is no need to look at the functions and activities at all. One database proponent wrote that the development of a function model actually gets in the way of the programmer and should be avoided if possible, further proclaiming his First Rule of CASE: "Only one diagramming method is needed—the data model—and code can be generated automatically from a data model."[1]

This statement is true if the automated system to be built performs simple, well-known processes such as basic banking transactions (for example, put money in checking account, withdraw money from checking account, transfer money to savings account, and so on). These activities are so familiar that an IDEF0 model would not be helpful, and design of the data structure of the system can be started almost immediately. The statement is not true, however, if the software to be built has a complex processing aspect, such as that for an avionics system or a computer's operating system.

The field of automated programming would also lead one to believe that analysis, design, and programming are outmoded—as if the system will do all this for you and produce a running computer program. However, the type of system that is implied here is one that simply queries a database and displays answers on a screen. Automatic programming has a long way to go before it can build an avionics system, a computer operating system, or a compiler. And haven't we slipped into the solution domain? Automated programming tools are of no help if you are trying to reengineer a business operation and have not yet understood what the system will look like.

[1] Clive Finkelstein, *An Introduction to Information Engineering: From Strategic Planning to Information Systems* (Reading, Mass.: Addison-Wesley, 1989), p. 376.

Regardless of the system to be analyzed and the comparative importance of the data versus the activity analysis, at some point the data must be analyzed. To perform this portion of the analysis, the analyst must understand several important characteristics of the data.

Although there are many data characteristics that could be analyzed, two are critical: data dynamics and data structure. Data dynamics can be depicted by SADT data models, using a portion of SADT that was not adopted by the Air Force for IDEF0. Data structure may be modeled using IDEF1X, where the term "structure" signifies all non-dynamic data information, such as its attributes and its relationship to other data elements.

A type of function model, the SADT data model shows the breakdown of the kinds of data in the system, with arrows indicating which activities produce which data elements and which activities use the data for what purpose. This gives a dynamic picture of the data being created, used, and merged with other data to form new data entities. The SADT data model is a natural companion for an IDEF0 or SADT activity model, and it is useful for analyzing how the system manipulates data and what is affected when changes are implemented. It also provides a validation of the activity model. In fact, most data models developed using SADT result in corrections to the activity model!

The IDEF1X model provides all of the structural information about the data. Ultimately, relating the need for and the usage of specific attributes of data by the activity boxes of an IDEF0 model is the key to integrating the two models.

This page intentionally left blank

INDEX