

3.3.4.5 Reference Notation

A standard notation is used in writing text and notes to refer to diagrams and specific parts of diagrams. References are based on box numbers, node numbers, ICOM codes, and note numbers. The following table provides examples of reference notations.

REFERENCE NOTATION	MEANING
2I1	Box 2 Input 1
O2	The boundary arrow with ICOM code O2
2O2 to 3C1 or 2o2 to 3c1	The arrow from 2O2 to 3C1 (The I, O, C or M may be upper case or lower case.)
I2 to 2I3 to 2O2 to (3C1 and 4C2)	From the boundary arrow with ICOM code I2 to Box 2 Input 3, through the activation of Box 2 that yields Output 2, to the availability (via a forking branch) of that output as Control 1 on Box 3 and Control 2 on Box 4.
A21.3C2	On diagram A21 in this model, see Box 3 Control 2. An embedded period means "look specifically at".
A42. 3	On diagram A42, see model note 3.
A42. 3	Same as above, using optional notation (vertical pipes surrounding model note instead of boxed note).
A42.3	On diagram A42 in this model, see Box 3.

MFG/A42.1

On diagram A42 of the model abbreviated MFG, see Box 1.

3.4 Models

3.4.1 IDEF0 Model Description

One of the most important features of IDEF0 as a modeling concept is that it gradually introduces greater and greater levels of detail through the diagram structure comprising the model. In this way, communication is enhanced by providing the reader with a well-bounded topic with a manageable amount of detail to learn from each diagram.

An IDEF0 model starts by presenting the whole subject as a single unit – a box with external-arrow boundary conditions connecting it to functions and resources outside the subject. The single box is called the "top box" of the model. (This top box has node number A0.) Since the single top box of an IDEF0 model represents the subject as a whole, the descriptive name in the box is general. The same is true of the external arrows of the model, since they represent the complete set of external boundary conditions of the subject as a whole, including access to mechanism support that supplies additional means of performance.

3.4.2 Context Diagrams

The diagram in which the A0 top box appears represents the context of the model and is called a context diagram. The minimum context for a model is the special context diagram with the node number A-0. The A-0 context diagram has only the single named A0 top box, with its labeled external arrows, and also textual definitions of the Viewpoint and Purpose of the model. (The A-0 diagram has no ICOM codes or tunneling at unconnected arrow ends.)

Sometimes, however, in order to provide a more complete exposition of the environmental context of the model, an optional A-1 context diagram (having the appearance of an ordinary, non-context, diagram) is also presented. In the A-1 context diagram, the A0 box takes the place of one of the three-to-six numbered boxes (the other boxes retaining their expected box number), so the effect is to provide a complete parent diagram (with three to six boxes) for the model's top – the A1 through A6 nodes still being the first-generation children. In the case where an A-1 context diagram is used, an A-0 context diagram is still presented. This A-0 diagram still has only the single named A0 top box, with its labeled external arrows and also textual definitions of the Viewpoint and Purpose of the model.

Context diagrams are diagrams that have node numbers of the form "A-n" (with a minus sign included), where n is greater than or equal to zero. Ordinary, non-context diagrams lack the minus sign in their node numbers. The box number of the top box of the model (representing the whole of the modeled subject) always is 0. Box number 0 shall appear on the required A-0 context diagram of the model, and shall also appear on the optional A-1 context diagram (if any) where it takes the place of one of the boxes (1 to at most 6) of that A-1 (model-wide) parent diagram. Thus A0 always is the (shared) node number of the parent box and child diagram for the whole model and always is detailed by boxes with node numbers A1, A2, A3, to at most A6.

With only one box, A-0 is a proper context diagram but is not a (proper) parent diagram. Proper diagrams have three to six boxes. The parental context is that which provides or names the context for a diagram in the place of a proper parent diagram. The parental context of the A0 diagram is the required A-0, if there is no A-1 context diagram. If there is an A-1 context diagram, A-1 is the proper parent of the A0 diagram. The parental context of the A-0 context diagram always is "TOP".

3.4.3 High-Level Context Diagrams

High-level context diagrams have node numbers of the form A-n, for n greater than one. Thus A-1 is a context diagram, is a proper parent (of A0), but is not high level. For a given model presentation, the highest-level context diagram (largest n) has parental context "NONE", unless the highest-level context diagram is A-0.

Each high-level context diagram, A-n, syntactically is an ordinary detail diagram except that one of its three-to-six boxes has its box number replaced by "minus sign n-1", so that, for A-1, that box is the A0 top box of the model, and the model as a whole (that A0 box itself, the parent of the children) appears to have parent A-1, grandparent A-2, etc.

By providing a more complete description of the model's environmental context (not, however, intended to be definitive in all respects, but only "typical"), context modeling (characterized by negative node numbering) provides more-constraining specifications on the boundary conditions of the A0 diagram of the model.

Context modeling proceeds just as ordinary detail modeling, the only difference being the negative numbering (and the non-definitive, but normative interpretation) that preserves A0 as the "origin" of the node-number-based coordinate system for all model references.

All the negative-node-number modeling merely provides more and more details about the sources and uses of the external boundary conditions. That detail may not precisely be matched by any particular specific environment, completely. These context diagrams describe the "typical" context.

Figure 21 provides an illustration in node-tree form to show how rich high-level context might appear.

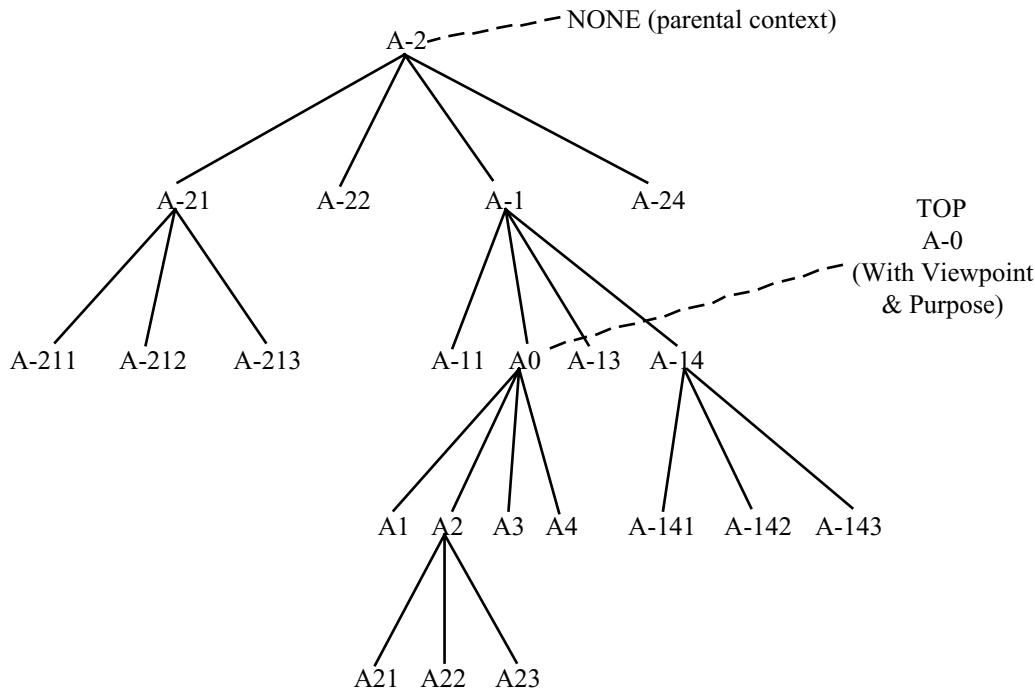


Figure 21. Negative Node-Numbered Context

3.4.4 FEOs, Text and Glossary

The node-numbering scheme provides the basis for coordinating FEOs, text, and glossary terms. During development it is important that each new element of information be associated with the node that brought it into consideration.

For each form (FEOs, text, and glossary), the node-numbering extension notation consists of a single letter appended to the associated node number. For example, node numbers for FEOs shall contain an "F" for FEO (e.g., A312F).

Some IDEF0 users record glossary definitions on IDEF0 diagram forms, though the use of this form for glossaries is not required. In this case, a glossary page shall define the key words, phrases and acronyms used with a particular, associated IDEF0 node. The node numbers for such glossary pages shall contain a "G" for glossary (e.g., A312G).

Likewise, some IDEF0 users record their textual comments on IDEF0 diagram forms, though the use of this form for text is not required. In this case, a text page shall provide the text comments for a particular, associated IDEF0 node. The node numbers for such text pages shall contain a "T" for text (e.g., A312T).

If there is more than one FEO, glossary or text page associated with a given IDEF0 node, the pages should be designated with an additional number to uniquely identify each (e.g., A312F1, A312F2,...A312G1, A312G2,..., A312T1, A312T2, ...).

3.4.5 Model Name

Each model has a unique, descriptive name that distinguishes it from other models with which it may be associated. This model name is normally abbreviated (uniquely) for use in node references. For example, a model named Manufacturing Operations may be abbreviated MFG. See Section 3.3.4.3 for a discussion of node references.

3.4.6 Presentation Rules

1. When there is text, it shall accompany the associated graphic diagram.
2. In non-publication models, the glossary associated with a specific graphic diagram shall accompany the diagram and shall define only the key words, phrases and acronyms used with the particular node.
3. In publication models, a glossary section shall define the key words, phrases and acronyms in alphabetical order for the entire model.
4. When a table of contents is provided for a model, it shall be presented as a node tree or node index, and shall contain node numbers, diagram titles and box names.

ANNEX A - IDEF0 CONCEPTS

A.1 Background

The desire of the U.S. Air Force to reduce costs and lead times by assisting the aerospace industry in its modernization efforts is evidenced in its many “Tech Mod” (Technology Modernization) programs. A similar goal, but using an industry-wide target rather than individual companies, was established under the ICAM (Integrated Computer Aided Manufacturing) Program. In ICAM, the goal was to develop “generic subsystems” which could be used by a large number of companies to provide a significant upgrade to the industry as a whole. These “subsystems” provide support for common functions such as management of information, shop floor scheduling and materials handling.

This ambitious goal needed a common “baseline” communication vehicle around which to plan, develop and implement the subsystems in the individual aerospace companies. The baseline was called the “Architecture of Manufacturing”, since it was to provide an industry-wide “architecture” showing how industry works today and around which generic subsystems could be planned, developed and implemented.

To develop the architecture, a “language” was needed in which to express and document current aerospace industry operations. At the outset of ICAM, the Air Force issued a Request for Proposal to build the architecture. An activity modeling technique was specified as the expressive language (where an activity was defined as a manufacturing cell or operational unit). To be successful, the language had to satisfy the following criteria:

- Since the architecture was to depict manufacturing, it had to be able to express manufacturing operations in a natural and straightforward way.
- Since the subject was so vast and complex, it had to be concise and provide a straightforward means of locating details of interest easily and quickly.
- Since it was to be used by a wide audience, it had to be able to communicate to a wide variety of aerospace industry personnel as well as to Air Force ICAM Program Office personnel.
- Since it was to serve as a baseline for generic subsystem planning, development and implementation, it had to permit sufficient rigor and precision to insure orderly and correct results.
- Since the baseline was to be developed through the cooperative effort of a large segment of the aerospace industry, it had to include a methodology (rules and procedures) for its use that would permit many diverse groups to develop

architecture pieces and that would permit wide-spread review, critique and approval.

- Since the baseline was to represent the entire aerospace industry rather than any one company or industry segment, the technique had to include a means of separating “organization” from “function”; that is, a common agreement could not be achieved unless the individual companies’ organizational differences were separated out and only the common functional thread was captured.

The SADT^a (Structured Analysis and Design Technique^a) originally developed in 1972 by Douglas T. Ross, of SofTech, was selected as the “The Architecture Method” for use in the Air Force Computer Aided Manufacturing (AFCAM) Project. The activity modeling technique was further developed and used in the follow-on ICAM Part I Program. The major subset of this technique used by the ICAM Part II Program Office was later re-named and documented as “IDEF0”.

A.2 IDEF0 Concepts

The original IDEF0 methodology incorporated basic concepts which address each of the needs listed above. These basic IDEF0 concepts are:

1. Activity Modeling Graphic Representation. The “box and arrow” graphics of an IDEF0 diagram show the manufacturing operation as the box, and the interfaces to/from the operation as the arrows entering/leaving the box. In order to be able to express real-life manufacturing operations, boxes may be interpreted as operating with other boxes, with the interface arrows providing “constraints” as to when and how operations are triggered and controlled.
2. Conciseness. The documentation of a manufacturing architecture must be concise to permit encompassing the subject matter. The linear, verbose characteristic of ordinary language text is clearly insufficient. The two-dimensional form provided by a blueprint-like language has the desired conciseness without losing the ability to express relationships such as interfaces, feedback and error paths.
3. Communication. There are several IDEF0 concepts which are designed to enhance communications:
 - Diagrams based upon very simple box and arrow graphics.
 - English text to specify box (function) and arrow (data or objects) meanings.
 - Gradual exposition of detail, featuring a hierarchy with major functions at the top and successive levels of sub-functions revealing well-bounded detail breakout.
 - A node index for locating details within the hierarchic structure of diagrams.
 - Limitation of detail on each successive diagram to not more than six sub-functions for ease of reader comprehension.

- Diagrams supported with text and glossary to increase the preciseness of the graphic representation.
4. Rigor and Precision. The rules of IDEF0 enforce sufficient rigor and precision to satisfy ICAM architecture needs without overly constraining the analyst. IDEF0 rules include:
 - Detail exposition control at each level (3-6 box rule).
 - Bounded context (no omissions or additional out-of-scope detail).
 - Syntax rules for graphics (boxes and arrows).
 - Uniqueness of names and labels on a diagram.
 - Diagram connectivity (Detail Reference Expressions [DRE]).
 - Data/object connectivity (ICOM codes and tunneled arrows).
 - Input vs. control separation (rule for determining role of data or objects).
 - Minimum control of function (all functions require at least one control).
 - Arrow branch (fork or join) constraint (labels for arrow segments).
 - Arrow label requirements (minimum labeling rules).
 - Purpose and viewpoint (all models shall have a purpose and viewpoint statement).
 5. Methodology. Step-by-step procedures are provided for modeling, review and interview tasks.
 6. Organization vs. Function. The separation of organization from function is included in the purpose of the model and carried out by the selection of functions and arrow labels during model development. Continual review during model development ensures that organizational viewpoints are avoided.

A.3 Discussion of Individual IDEF0 Concepts

In the remaining sub-sections descriptions of some of the basic concepts are elaborated to clarify them and show their utility.

A.3.1 Activity Modeling Graphics

The IDEF0 methodology may be used to model a wide variety of automated and non-automated “systems” or subject areas, including any combination of hardware, software, machines, processes or people. For new systems IDEF0 may be used first to define the requirements and specify the functions, and then to design an implementation that meets the requirements and performs the functions. For existing systems, IDEF0 can be used to analyze the functions the system performs and to record the mechanisms (means) by which these are done.

The result of applying IDEF0 is a model. A model consists of diagrams, text and glossary, cross-referenced to each other. Diagrams are the major components of a model. All functions and interfaces are represented as boxes (functions) and arrows (data or object interfaces) on diagrams.

The position at which the arrow attaches to a box conveys the specific role of the interface. The controls enter the top of the box. The inputs, the data or objects acted upon by the operation, enter the box from the left. The outputs of the operation leave the right-hand side of the box. Mechanism arrows that provide supporting means for performing the function join (point up to) the bottom of the box. Call arrows, a type of mechanism arrow which enables the sharing of detail between models or between portions of the same model, connect to the bottom of the box and point downward. These arrow positions are illustrated in Figure A1.

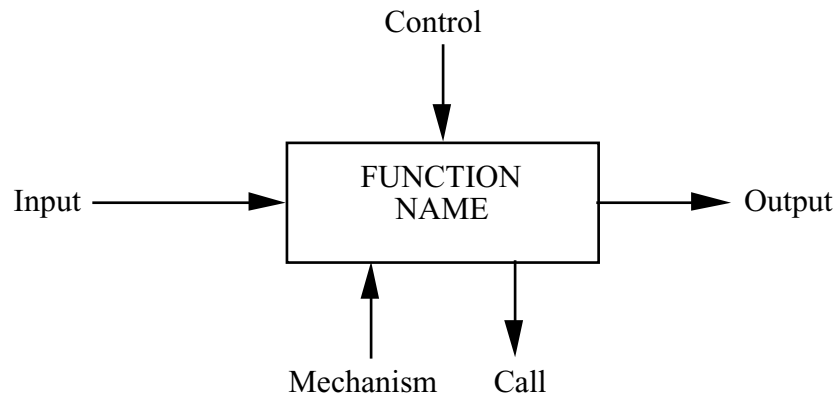


Figure A1. Function Box and Data/Objects Arrows

These box and arrow meanings are used to relate several sub-functions on a diagram comprising a more general function. This diagram is a “constraint diagram” which shows the specific interfaces which constrain each sub-function, as well as the sources and targets of the interface constraints (Figure A2).

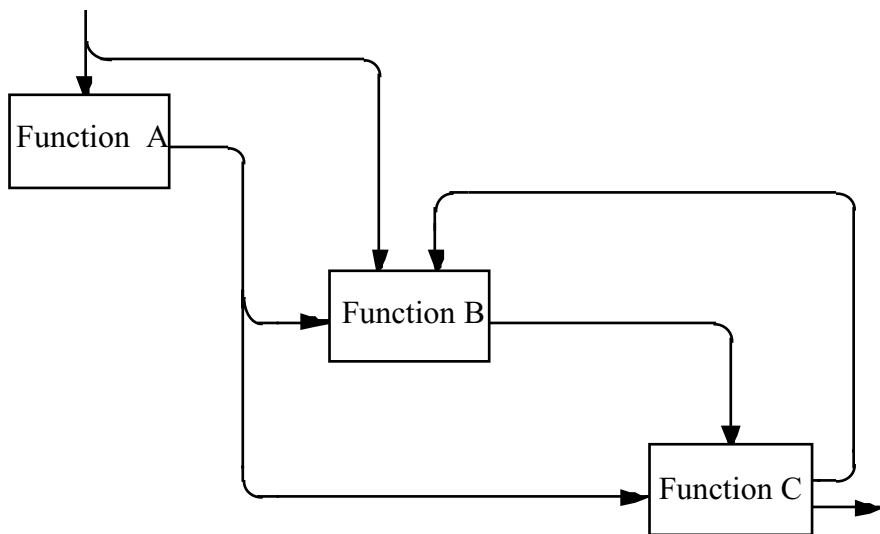


Figure A2. Constraint Diagrams

In Figure A2, Function B is constrained by one input and two controls, and produces a single output, which constrains Function C.

Here, the term “constrains” means that a function uses the data or objects shown entering the box, and therefore, is constrained from operating by the interface; the function cannot act until the contents of the interface arrow are provided, and the way in which the function operates depends upon the details (value, number, etc.) of the contents of the interface arrow.

A.3.2 Communication by Gradual Exposition of Detail

One of the most important features of IDEF0 is that it gradually introduces greater and greater levels of detail through the diagram structure comprising the model. In this way, communication is enhanced by providing the reader with a well-bounded topic with a manageable amount of new information to learn from each diagram.

The structure of an IDEF0 model is shown in Figure A3. Here, a series of four diagrams is shown with each diagram's relation to the others.

An IDEF0 model starts by representing the whole system as a single unit - a box with arrow interfaces to functions outside the system. Since the single box represents the system or subject area as a whole, the descriptive name written in the box is general. The same is true of the interface arrows since they also represent the complete set of external interfaces to the system as a whole. The diagram with the single box is called the “context diagram,” and shall define in text the viewpoint and purpose of the model.

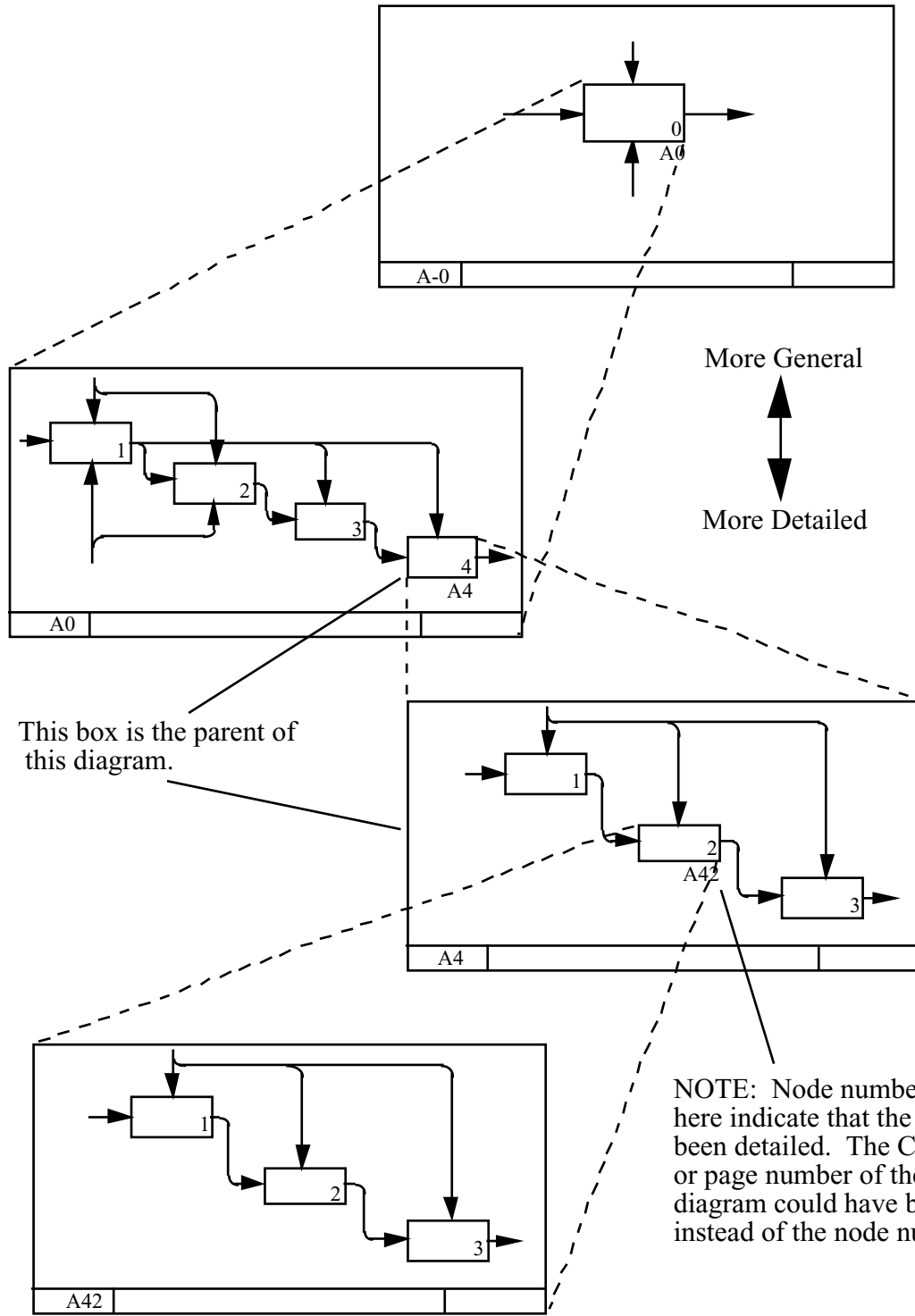


Figure A3. IDEF0 Model Structure

The box that represents the system as a single module is then detailed on another diagram with boxes connected by interface arrows. These boxes represent major sub-functions of the single parent function. This decomposition reveals a complete set of sub-functions, each represented as a box whose boundaries are defined by the interface arrows. Each of these sub-functions may be similarly decomposed to expose even more detail. In IDEF0, we use the following terminology: functions are “decomposed”; the boxes that represent functions are “detailed”.

A box, if detailed, is always detailed on a child diagram into no fewer than three boxes, but no more than six boxes. The upper limit of six forces the use of a hierarchy to describe complex subjects. The lower limit of three insures that enough detail is introduced to make the decomposition (detailing) of interest.

Each diagram in a model is shown in precise relationship to other diagrams by means of interconnecting arrows. When a function is decomposed into sub-functions, the interfaces between the sub-functions are shown as arrows. The name of each sub-function box plus its labeled interfaces define a bounded context for that sub-function.

In all cases, every sub-function is restricted to contain only those elements that lie within the scope of its parent function. Sub-functions are discrete and do not overlap. Further, the collection of sub-functions cannot omit any elements. Thus, as already indicated, the parent box and its interfaces provide a context for its child diagram. Except for tunneled arrows, nothing may be added or removed from this precise boundary.

A.3.3 Disciplined Teamwork

The IDEF0 methodology includes procedures for developing and critiquing models by a large group of people, as well as integrating support subsystems into an IDEF0 Architecture. Additional supporting procedures, such as librarian rules and procedures, and review cycle (see Annex C) procedures are also included in the IDEF0 methodology. (It should be noted that some of these rules and procedures, such as the Kit Cycle or Reader-Author Cycle critique procedures, are also used with other IDEF techniques.)

The creation of an IDEF0 model is the most basic of these “disciplined teamwork” procedures. The creation of a model is a dynamic process which usually requires the participation of more than one person. Throughout a project, authors create initial diagrams which are distributed to project members for review and comment. The discipline requires that each person expected to make comments about a diagram shall make them in writing and submit them to the author of the diagram. The author replies, also in writing. This cycle continues until the diagrams, and eventually the entire model, are officially accepted.

IDEF includes procedures for retaining written records of all decisions and alternate approaches as they unfold during the project. Copies of the diagrams created by an author are critiqued by knowledgeable readers who document suggestions directly onto the copies. Authors respond to each comment in writing on the same copy. Suggestions are accepted or rejected in writing

along with the reasoning used. As changes and corrections are made, outdated versions of diagrams are retained in the project files.

The diagrams are changed to reflect corrections and comments. More detail is added to the model by the creation of more diagrams which also are reviewed and changed. The final model represents an agreement on a representation of the system or subject area from a given viewpoint and for a given purpose. This representation can be easily read by others outside the initial project, used for presenting the system definition in short stand-up briefings or in walk-throughs, and for organizing new projects to work on system changes.

ANNEX B - CREATING AND INTERPRETING IDEF0 DIAGRAMS

B.1 Reading IDEF0 Diagrams

A model is made up of a collection of diagrams and associated materials arranged in a hierarchic manner. A node index (or table of contents) shall be provided. Placing the diagrams in hierarchical order gives an overall view of the model and allows access to any portion.

Reading is done top-down, considering each diagram as a context bounded by its parent box. After the top level diagrams are read, first level diagrams are read, then second level diagrams are read, etc. If specific details about a model are needed, the node index is used to descend through the levels to the required diagram.

When published, a model is bound in “page-pair” format and “node index” order. “Page-pair” format means that each diagram and the entire text associated with it appear on a pair of facing pages. (Figure B1.)

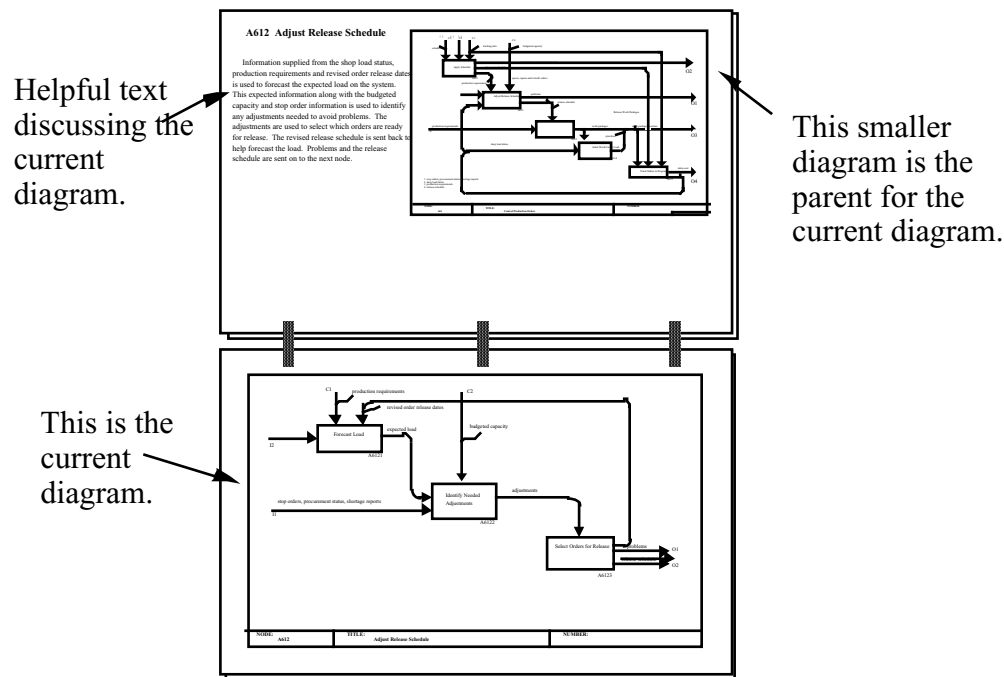


Figure B1. Page-Pair Format

“Node index” order means that all child diagrams relating to one box on a diagram are presented before the children of the next box. This places related diagrams together in the same order used in an ordinary table of contents. (Figure B2.)

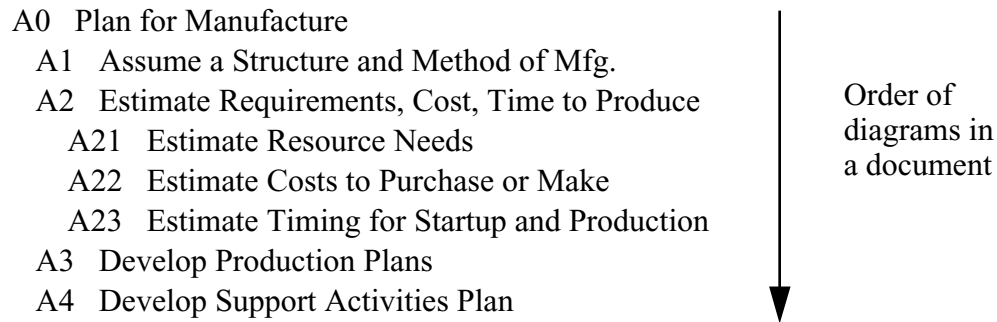


Figure B2. Node Index Showing Diagram Order

B.1.1 Approaching a Model

Models provide an overview of the whole system or subject area as well as details of a particular subject. To read a model for its overview, use the index to find all high-level diagrams. (Figure B3.)

A0 Manufacture Product

A1 Plan for Manufacture

- A11 Assume a Structure & Method of Mfg.
- A12 Estimate Requirements, Time, Cost to Produce
- A13 Develop Production Plans
- A14 Develop Support Activities Plan

A2 Make & Administer Schedules & Budgets

- A21 Develop Master Schedule
- A22 Develop Coordinating Schedule
- A23 Estimate Costs & Make Budget
- A24 Monitor Performance to Schedule & Budget

A3 Plan Production

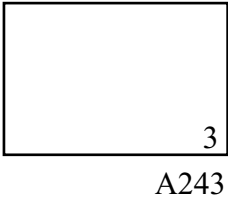
Figure B3. Node Index Showing Overview Diagrams

To read a model for detail, use the index to find all diagrams detailing the subject of interest. (Figure B4.)

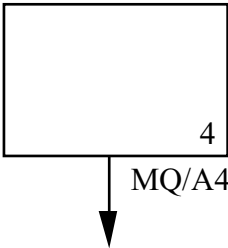
- A0 Manufacture Product
 - A1 Plan for Manufacture
 - A11 Assume a Structure & Method of Mfg.
 - A12 Estimate Requirements, Time, Cost to Produce
 - A13 Develop Production Plans
 - A14 Develop Support Activities Plan
 - A2 Make & Administer Schedules & Budgets**
 - A21 Develop Master Schedule**
 - A22 Develop Coordinating Schedule**
 - A23 Estimate Costs & Make Budget**
 - A24 Monitor Performance to Schedule & Budget**
 - A3 Plan Production

Figure B4. Node Index Showing Specific Detailed Diagram

Further detailing in a model may be traced by referring to the detail reference expression (DRE) just below the box number. This indicates the node number, C-number or page number of the child diagram that details the box. In the example below, details for box 3 on diagram A24 may be found on a diagram with node number A243. If no DRE appears, the box has not yet been detailed.



Details may be shared within a model or between different models. In both cases, a call arrow (downward pointing) indicates where the shared detailing appears via a reference expression that may include a unique, abbreviated model name. In the example below, box 4 is detailed by diagram A4 in model MQ. In this example, the reference expression is a diagram node reference.



B.1.2 Diagram Reading Steps

The precise information about a system is in the diagrams themselves. The following reading sequence is recommended:

1. Scan the boxes of the diagram to gain an impression of what is being described.
2. Refer back to the parent diagram and note the arrow connections to the parent box. Try to identify a “most important” input, control and output.
3. Consider the arrows of the current diagram. Try to determine if there is a main path linking the “most important” input or control and the "most important" output.
4. Mentally walk through the diagram, from upper left to lower right, using the main path as a guide. Note how other arrows interact with each box. Determine if there are secondary paths. Check the story being told by the diagram by considering how familiar situations are handled.
5. Check to see if a related FEO diagram exists.
6. Finally, read the text and glossary, if provided.

This sequence ensures that the major features of each diagram receive attention. The text will call attention to anything that the author wishes to emphasize. The glossary will define the author's interpretation of the terminology used.

Each diagram has a central theme, running from the most important incoming boundary arrow to the most important outgoing boundary arrow. This main path through the boxes and arrows outlines the primary function of the diagram. (Figure B5.) Other parts of the diagram represent qualifying or alternative conditions which are secondary to the main path.

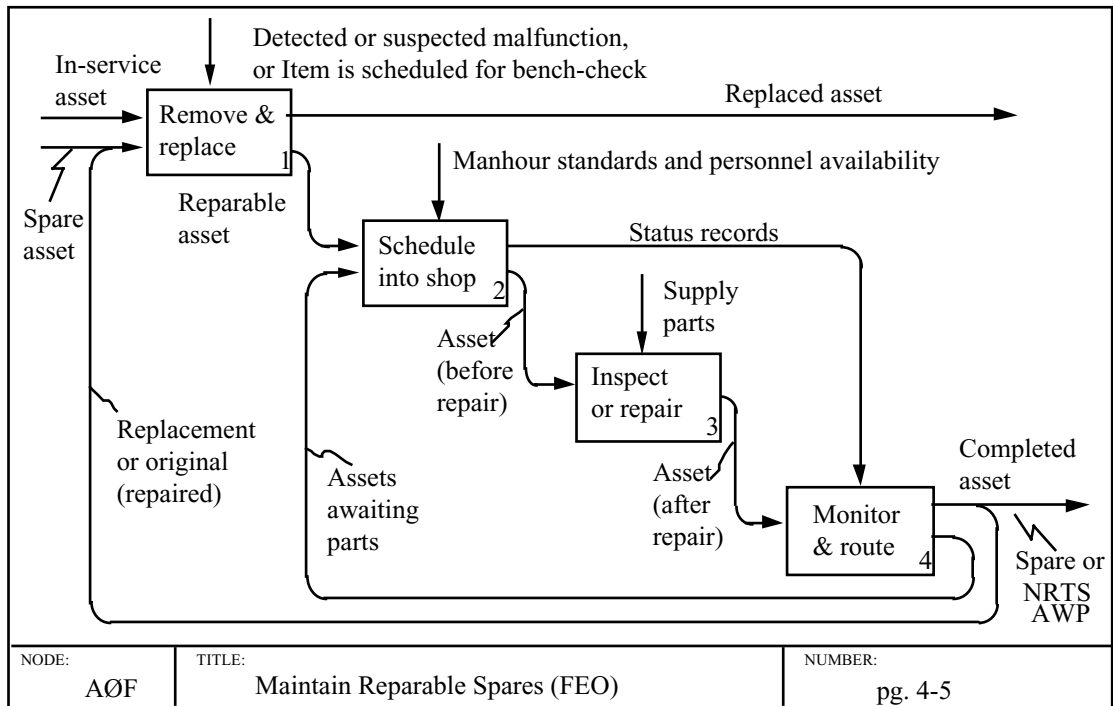


Figure B5. Example of Main Path

The system's operation can be mentally envisioned by pursuing the main path. Specific kinds of data inputs, the handling of errors and possible alternative outputs lend detail to the story. This walk-through enhances understanding of the diagrams.

B.1.3 Semantics of Boxes and Arrows

The fundamental notion which must guide the interpretation of any diagram or set of diagrams is: Only that which is explicitly stated is necessarily implied. This derives from the very nature of constraint diagrams. Unspecified constraints must not be assumed; necessary constraints must be explicit. The corollary is that: Any further detailing not explicitly prohibited is implicitly allowed.

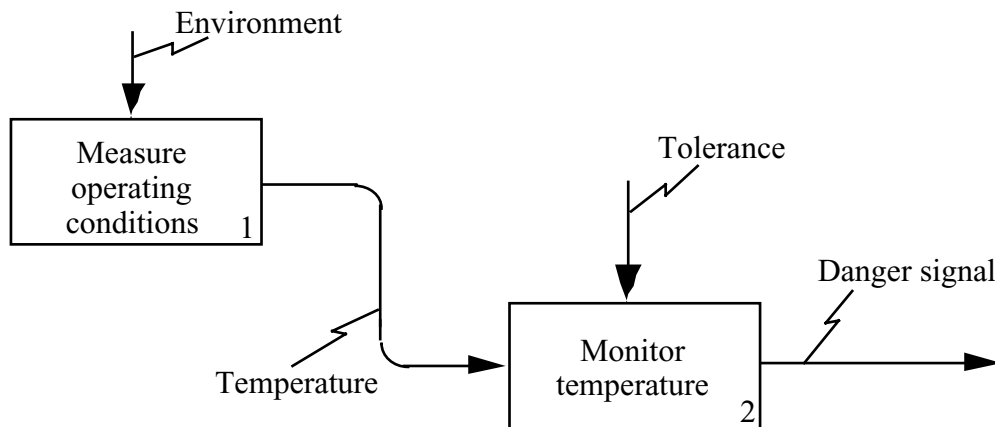


Figure B6. Example of Constraint

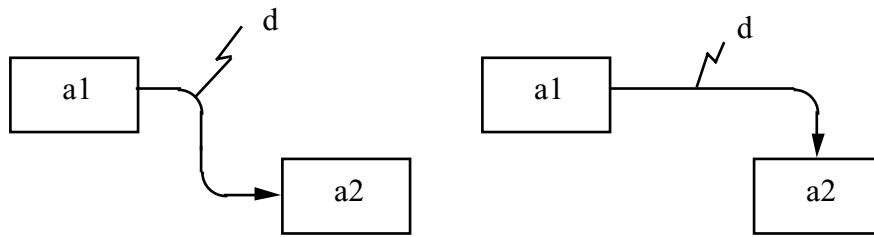
An assumption can be made using Figure B6 that the temperature is measured “often enough” and the tolerances are changed “when appropriate” and the temperature is monitored against the tolerances “often enough” that the danger signal will be produced “soon enough”. None of these intuitive understandings would conflict with subsequent detailing which showed that:

- a. the temperature was measured by periodic sampling, or
- b. current tolerances were requested only when the temperature increased by some fixed amount, or
- c. a series of temperature values produced by box 1 was stored by box 2 which examined the pattern of change to determine if the pattern was within the tolerances, etc.

The graphic notations of a diagram are, by themselves, abstract. However, they do make important fundamental distinctions. Their abstract nature should not detract from the intended breadth of possible interpretations that are permitted.

B.1.3.1 Constraints Omit How and When

Either of the two representations:



says that: the activity a2 is dependent on “d” which is created or modified by the activity a1.

Each representation defines a constraint relationship between the two boxes. All that is explicitly stated by the intermediate arrow for either representation is expressed as follows: some activation of box 2 requires something called “d” that is produced by some activation of box 1.

Frequently, diagrams imply strongly that two or more boxes may need the contents of an arrow. The meaning of the boxes and arrows shown in Figure B7 is that something produced by box 1 is needed by box 2 and by box 3. It may be that an activation of the arrow’s “source” (box 1) must precede every activation of its “destination” (box 2 or box 3). It may be that one activation of the source is sufficient for every activation of any destination. Without additional information, the boxes and arrows alone permit either interpretation.

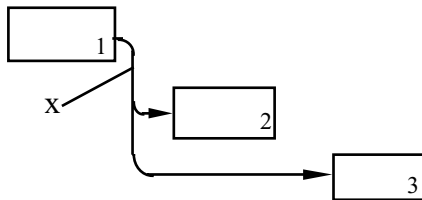


Figure B7. Two boxes using the contents of the same arrow

B.1.3.2 Multiple Inputs, Controls and Outputs

The basic interpretation of the box shown below (Figure B8) is: In order to produce any subset of the outputs [O1, O2, O3], any subset of the entries [I1, I2, I3, C1, C2, C3, C4, M1, M2, M3] may be required. In the absence of further detailing it cannot be assumed that:

- a. any output can be produced without all entries present, or
- b. any output requires all entries for its production.

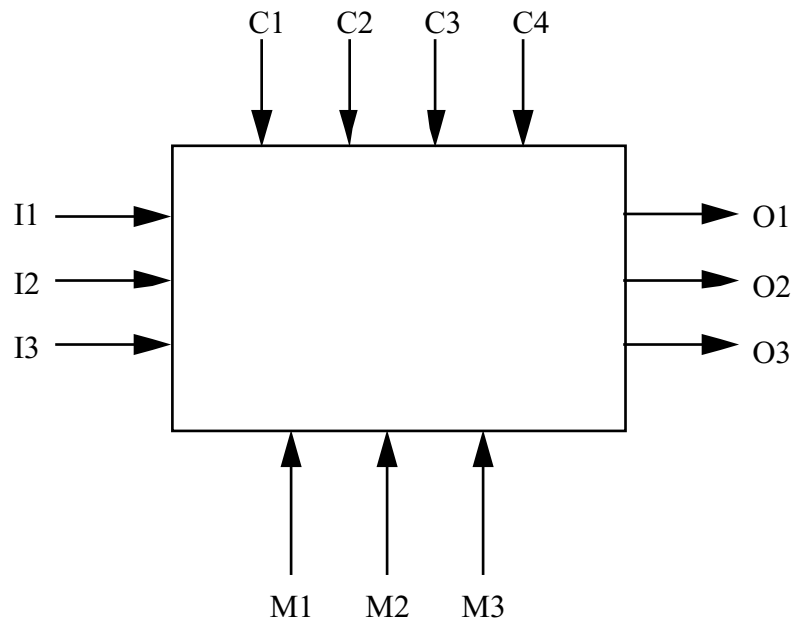


Figure B8. Illustration of ICOM coding

The partial detailing of the previous box (shown in Figure B9, as it might appear in a FEO diagram) indicates that I3, C2, C3 and C4 are not required for producing O1. Figure B9 illustrates the points that:

- a. some form of further detailing will specify the exact relationship of inputs and controls to outputs;
- b. until that detailing is provided, limiting assumptions about relationships “inside” each box should not be made;
- c. reading of a diagram should concentrate on the arrows, which are explicit, rather than on box names, which are only implicit.

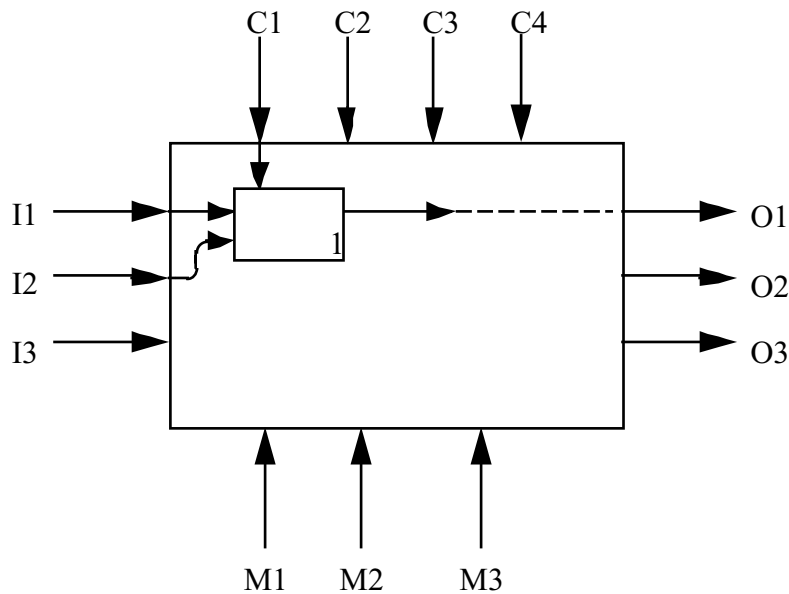


Figure B9. FEO representing detailing of multiple ICOMs

B.2 Author's Guide to Creating IDEF0 Diagrams

When creating any IDEF0 diagram, the requirements to be satisfied are that:

- a. its purpose and viewpoint shall match the stated purpose and viewpoint of the overall model;
- b. its boundary arrows shall correspond to the arrows that connect to its parent box;
- c. its content shall be exactly everything in its parent box.

B.2.1 Basic Steps of Authoring

The step-by-step discipline of authoring makes it possible to create diagrams that form useful and coherent models. The discipline to follow is:

- a. Bound the subject matter more precisely than the name of the function box suggests. This is done with a list of data and objects acted on or processed by the function.
- b. Study the bounded set of subject matter and form possible sub-functions of the total function.
- c. Look for natural patterns of connection of those sub-functions.

- d. Split and cluster sub-functions to make suitable boxes.
- e. Draw a final version of the diagram with careful attention to layout and clarity.

B.2.1.1 Selecting a Context, Viewpoint and Purpose

Before beginning any model, it is important to determine the model's orientation. This includes the context, viewpoint and purpose. These concepts guide and constrain the creation of a model. While they may be refined as authoring proceeds, they must be consistent throughout a model if its orientation is to remain clear and undistorted.

The context establishes the subject of the model as part of a larger whole. It creates a boundary with the environment by describing external interfaces. The context diagram establishes the context for the model.

The viewpoint determines what can be "seen" within the context, and from what "slant" or perspective. Depending on the purpose, different viewpoints may be adopted that emphasize different aspects of the subject. There is only one viewpoint per model.

The purpose establishes the intent of the model or the goal of communication that it serves. Purpose embodies the reason why the model is created (functional specification, implementation design, customer operations, etc.).

The starting point for every analysis is to bound the context. Decide what the focus is before the top-most box is created. Beware of drifting out of this carefully-selected starting domain. Every step should be checked against the starting purpose. Things that don't fit may be noted for later modeling of the relevant views. Clarity is derived from the rigors of detailing. Knowing how far to go, when to stop, when to change gears and how the pieces fit together will always depend on the purpose for which a model is created.

B.2.1.2 Creating the Context Diagram

To start a model, create the A-0 diagram. Draw a single box containing the name of the function which encompasses the entire scope of the system being described. Use input, control and output arrows entering and leaving the box to represent the data and object interfaces of the system to its environment. This single-box diagram bounds the context for the entire model and forms the basis for further decomposition efforts. State the purpose and viewpoint on A-0 context diagram.

Some authors find it easier to sketch the A0 and then draw the single box and interface arrows shown at level A-0. It may be necessary to switch diagramming efforts back and forth between A-0 and A0 several times to obtain a good start for the decomposition.

If the A-0 diagram has begun at too low a level of detail, make the A-0 box the basis of a new level A0 diagram. Move up one level to a new A-0 and revisit the Viewpoint and Purpose statements. Repeat this process until an A-0 is reached which has sufficient scope to cover all aspects of the system. (Sometimes such a higher level will broaden rather than clarify the chosen

viewpoint. If so, make an A-1 multi-box context diagram and keep the A0 diagram to the original intent.)

B.2.1.3 Creating the Top-Most Diagram

All system functions lie within the single box shown on the A-0 diagram. The diagram bounds the context of the system. The A0 diagram decomposes the single function on the A-0 diagram into its three to six major sub-functions.

The real “top” of the model is the A0 diagram. Its structure clearly shows what the A-0 diagram tried to say. The terms and structure of A0 also bound every subsequent level because it is a complete description of the chosen subject. Lower levels delineate the A0 functions (boxes). If the purpose of the model is to be achieved, this chain of detailing must be carefully followed at each step. Beginning at the top is the challenge of authoring. It forces the author to maintain a level of abstraction, keep an even model depth and relegate details to a lower level.

B.2.1.4 Creating Child Diagrams

To form the structure of diagrams, detail each box on the A0 diagram into its three to six major parts. Form a new diagram for each box, which covers the same topic as its parent box but in more detail.

To detail each box by 3 to 6 child boxes, obtain the needed additional facts. Create a first-draft diagram by first listing all data and objects related to the function being decomposed. Take care that the list covers the entire topic of the parent box so that no portion is lost in the decomposition. Then draw boxes which associate candidate sub-function names with appropriate data and objects from the list, and draw arrows between the boxes.

To derive the clearest possible diagram, modify or re-draw the diagram several times until satisfied. Split (break up a box into two or more parts) and cluster (combine two or more parts into a single box) until satisfied. Split and cluster until you express the parent function in three to six boxes.

Generate portions of more detailed-level diagrams to explore points which need clarification. Create several (3 or 4) diagrams as a set, rather than one diagram at a time.

B.2.1.5 Creating Supporting Material

Eventually, each diagram will be accompanied by a page of narrative text, glossary and, perhaps, FEOs. The text associated with the A-0 diagram should complete the model’s orientation and is written when the A-0 diagram is created. The text complements the context (expressed in A-0 itself) by expanding upon the stated viewpoint and purpose of the model.

Text for every other diagram (including A0) is quite different. It tells a brief, concise story. It does not duplicate what the diagram already says by merely describing each box function in words, but rather weaves through its patterns. At every level, this captures the viewpoint in a way that furthers the purpose. A graphic diagram may or may not have an associated text diagram.

The glossary explains the definitions the author gives to functions and data/objects in a diagram. These definitions are important because the terminology used in the model may have a completely different meaning in one company from the meaning in another company. Terminology often differs among units or disciplines within the same company.

FEOs are diagrams that highlight a particularly interesting or subtle aspect of a diagram. They are not bound by IDEF box and arrow syntax and may contain partial arrow structures and notes to emphasize their point.