

IDEF0 1982 年 Ver

Function Modeling Manual (IDEF0) - Defense Technical

www.dtic.mil/cgi-bin/GetTRDoc?AD=adb062457

第 1 章 はじめに

統合 CAM (ICAM) という米空軍計画は、コンピューター技術をシステムティックに適用する事を通して、直接的に生産性を向上させることを狙ったものである。ICAM 計画では、生産にコンピューター技術を適用する構造化された手法を開発し、製造生産性の改良に何が最適であるかを理解する為にこれを利用して行く。

ICAM 計画では、生産性の改善に関係する人達にとって、情報伝達と製造の分析とをより強化する必要があると認識している。この必要性を満足させるために、製造の特性を詳細に表現する手法として、IDEF (ICAM 定義) を開発した。IDEF は製造を図表で表現する、以下の 3 つのモデリング手法によって構成されている。(訳注: 1982 年の時点では、IDEF0~2 の 3 つのみ。現在はかなり増殖している)

・ IDEF0

製造システムもしくは製造環境の機能と、その機能に関連する情報とオブジェクトの、構造化表現であるところの、機能モデルの作成に使用される。

・ IDEF1

製造システムもしくは製造環境の機能を支援する為に必要となる情報構造を表現する、情報モデルの作成に使用される。

・ IDEF2

製造システムもしくは製造環境の、機能、情報、リソースが、時間と共に移り変わる挙動を表現する、変遷 (dynamics) モデルの作成に使用される。

これら 3 つのモデルは、それぞれが独立して、もしくは幾つかのモデルの組み合わせにより、モデルされたシステム環境が、製造環境もしくは製造システムの最終的な目標である生産を達成する為に共同して機能すべきである、構成システム、組織、技術によって構成することで、一つの「基本設計概念 (? ARCHITECTURE)」を形作ることが可能である。「基本設計概念 (? ARCHITECTURE)」としてモデルが参照されているという事の意義は、

それらが基礎的關係を図形によって定義された、機能的インターフェース、共通性の識別、共有情報と個別情報、そしてリソースの動的な相互作用といった、設計図もしくはフレームワーク（?構造化された作業フロー?）であるということである。製造環境や製造システムといった主題をより良く理解し、分析する為だけでなく、それを構成しているシステム、組織、技術をどのように強調させるかということにも使用する事で、**IDEF** モデルは基本設計概念（?ARCHITECTURE）と成り得るという事を認識する事が重要である。

IDEF は手法（method）であり、基本設計概念は手段（means）であり、製造生産性の改善は結果（end）であるが、これらは米国航空産業において **ICAM** 計画が追い求めてきたものである。

以下の内容は、**IDEF0** を機能モデルの作成に使用する事を考慮した、基本コンセプトと技術、手続きについて議論したものである。

第2章 IDEF のコンセプト

IDEF0 のコンセプト

2. 1 背景

航空産業の近代化努力を補助する事で、コスト削減とリードタイム短縮を行おうという米空軍の願望は、進行中の幾つもの技術革新（Tech Mod、Technology Modernization）計画において形成されつつある。そして同様な目標が、こちらは個々の会社というよりも産業全体を対象としているが、ICAM 計画の下で設定されている。ICAM の目的は、多くの会社により利用されることで、業界全体が大きく進歩することになる「一般サブシステム（generic subsystem）」を開発する事である。こうしたサブシステムは、情報の運用管理や、工場の定盤日程作成、素材の取扱といった共通の製造機能を提供してくれるのである。

この野心的な目的には、個々の航空機関連会社においてサブシステムを計画、開発、適用に使用する、共通の基準情報媒体（baseline communication vehicle）が必要である。この基準（baseline）は、工業が今日どのように働いているかと、そして一般サブシステムが計画、開発、適用され得る場所とを示すことで、産業界一般での「基本構造」を提供している事から、「製造の基本構造」と呼ばれている。

製造の基本構造を開発する為には、現在の航空機産業の運営を表現し記述する、ある種の「言語」が必要である。ICAM を始めるに当たり、米空軍は基本設計概念の構築の提案要求（request of proposal）を行った。「セルモデリング技法（cell modeling technique、モジュール的表現技法のこと?）」。「セルモデリング技法」は表出言語（expressive language）として明記されている（ここでの「セル」は、製造セルもしくは運用ユニットとして定義されている）。成功したものとするには、この言語は以下の基準を満たさなければならない：

- ・ 製造を表現する為に用いる事から、基本構造は製造作業（manufacturing operation）を自然で、直接的な方法で表現可能でなければならない。
- ・ 主体となる物（subject）が余りに巨大で複雑であることから、基本構造は簡潔でありつつも、興味対象となる部分を簡単に素早く詳細化する直接的な手法を持たせる。
- ・ 広い層に使用されるべきものであることから、空軍 ICAM 計画の事務局スタッフ（Office

staff) だけでなく、航空機産業の製造担当者に至るまでの幅広い種類の人々との間で、情報交換が可能でなければならない。

- ・一般サブシステムの計画、開発、適用の為の基準となるべきものであるため、整理された正確な結果を保証する、十分な厳密性と精度とを持っていないといけない。

- ・航空機産業の多くの部門での共同作業を通して基準が作成されなければならないため、多くの異なったグループが基本構造の部品を作成することが可能となり、また広く批評、評論、認可が可能となるように、方法論 (methodology、基準にはルールと手法) が含まれていないといけない。

- ・基準は、個別の会社や工業部門というよりも航空機産業全体を表現しなければならない為、「機能 (function)」から「組織 (organization)」を分離する手段 (means) が、手法 (method) に含まれていないといけない。なぜなら、それぞれの会社の組織的相違を分離し、共通な機能的流れのみを抽出しなければ、共通の同意を得る事ができないからである。

空軍によって選択されたセルモデリング技法 (cell-modeling technique) は、1970 年代初頭に開発された SADT (Structured Analysis and Design Technique、構造化された分析と設計技法) を基にしている。ICAM プログラムの事務局 (Office) によって使用されているこの技法の主要な一部分が、後に名前を変えて「IDEF0」となっている。

訳注 : SADT については Wiki 等の説明が多い。

2. 2 IDEF0 のコンセプト

IDEF0 の方法論には、前節に挙げられた需要それぞれに対応する為に、以下のような基本コンセプトが含まれている。

(1) セルモデリングによる図形的表現 (Cell Modeling Graphic Representation)

IDEF0 ダイアグラムの「箱 (box) と矢印 (arrow)」の図は、製造作業を箱として、また作業に対するインターフェースを箱への/箱からの矢印として、それぞれ表現している。実際の製造作業を表現する為に、インターフェース矢印に、いつ・どのように作業 (箱) が開始・管理されるかという形の「拘束」を付与してやることで、箱 (作業) は他の箱 (作業) と同時に実行することが可能となっている。

(2) 簡潔 (conciseness)

製造作業の基本設計概念 (architecture) の記述は、主体となる事象 (subject matter) を包含するように、簡潔でなければならない。直線的で (linear) で冗長な (verbose) 通常の文字言語だけでは、明らかに不十分である。青図 (訳注：設計図面のこと?) のような言語によって作図された 2 次元図は、インターフェースやフィードバック、エラーパス等の関係性を表現する機能を失うことなく、要求された簡潔性を満たしている。

(3) 情報伝達 (Communication)

IDEF0 には、情報流通をより向上させる為に、以下のようなコンセプトが含まれている：

- ・非常に単純な箱と矢印で構成されたダイアグラム
- ・英数字のラベルにより、箱と矢印の意味を記述すると共に、用語集 (glossary) と説明文 (text) により、ダイアグラム要素の意味をより正確なものにする。
- ・主要 (major) な機能を上層に、従属 (sub) 機能を下層にした階層構造で、段階的に詳細を開示してゆくことで、詳細情報が爆発してしまわないように制限している (訳注：理解能力を越した詳細情報は害にしかならない。抽象化としても後で説明されている)。

- ・「ノード図 (node chart)」により、ダイアグラムの階層構造に配置された詳細情報を、素早く索引付 (index) することができる。

- ・読解者の理解力を容易にするため、各継承 (successive) ダイアグラム内の従属機能 (sub-function) が 6 個以上にならないように、詳細情報を制限している。

(4) 厳密性と精度 (rigor and precision)

IDEFO のルールでは、分析者を過剰に拘束しない範囲で、ICAM 構成の需要を満たすようなだけの、厳密性と精度とを要求している。

- ・各階層毎の詳細開示管理

 - 階層毎に 3~6 の箱というルール

- ・境界による環境の切り取り (Bounded Context)

 - 省略の禁止、もしくは範囲外の詳細の追加の禁止

- ・ダイアグラムインターフェース接続性 (Diagram Interface Connectivity)

 - ノード番号、箱番号、C 番号 (C-number)、

 - 詳細参照表現 (DRE、Detail Reference Expression)

- ・データ構造接続性 (Data Structure Connectivity)

 - ICOM コードと、丸括弧 (parentheses) の使用

- ・ラベル (label) とタイトル (title) の唯一性 (uniqueness)

 - 同一の物に複数の名前を与えない

- ・図の表記ルール (Syntax Rule for Graphics)

 - 箱と矢印

- ・データ矢印の分岐制限 (Data Arrow Branch Constraint)

 - 分岐でのデータフローを制限する為のラベル (?)

 - Labels for constraining data flow on branches

- ・入力と管理の分離 (Input vs. Control Separation)

データの役割を決定する為のルール

- データ矢印のラベルへの要求 (Data Arrow Label Requirements)
ラベル付を最小化するルール
- 機能の制御の最小化 (Minimum Control of Function)
全ての機能には、少なくとも 1 つの制御 (control) が必要
- 目的と視点 (Purpose and Viewpoint)
全てのモデルには、目的と始点を提示 (statement)

(5) 方法論 (Methodology)

モデリング、チェック (review)、統合の各作業 (task) の為に、段階的な手続き (step by step procedure) を提供している。こうした手続きについては、航空機産業関係者の訓練の為に、方法論の教育を行う為の公式コースを利用可能である。

(6) 組織と機能 (Organization vs. Function)

機能からの組織の分離は、モデルの目的に含まれており、モデル開発の中で、機能とインターフェース名の選択によって実行される。このコンセプトは IDEF0 の教育コース内で教えられ、またモデル開発の間も常にチェックし、組織的観点 (organizational viewpoints) が紛れ込まないようにしている。

2. 3 個々の IDEF0 コンセプトについての議論

ここからは、幾つかの基本コンセプトを詳細に説明して行くことで、そのコンセプトをより明確にし、ICAM での使い方を示してゆく。

2. 3. 1 セルモデリング図表 (cell modeling graphic)

IDEF0 の方法論は、ハードウェア、ソフトウェア、そして人のいかなる組み合わせをも含んだ、幅広い多様な「システム」をモデリングする為に用いられる。新しいシステムに対して使用する際には、IDEF0 は、まずは要求と（それを満たす）機能とを明確にするために使用され、そしてその後に、その要求を満たし、かつ機能を発揮するような実装 (implementatoin) をデザインする為に使用される。既存のシステムに対して使用する際には、IDEF0 はシステムの機能の分析と、そのメカニズムの記録とに使用する事が可能である。

IDEF0 を適用することにより、モデルを得ることができる。モデルはダイアグラム、説明文 (text)、用語集 (glossary) とによって構成され、その中の要素は相互参照している。これらの内、モデルの主要な構成物はダイアグラムである。全ての製造機能とインターフェースは、ダイアグラム上で箱（機能）と矢印（インターフェース）とによって表現されている。

箱に入る矢印の位置により、インターフェースの意味が異なっている。箱の上に入る矢印は製造の制御 (control) であり、製造作業に使用される資材や情報は箱の左から入り、その結果としての出力物は箱の右へと出て行く。そして作業者もしくは自動化システムといった作業を実行する機構 (mechanism) は、箱の下から入る (図 2-1)。

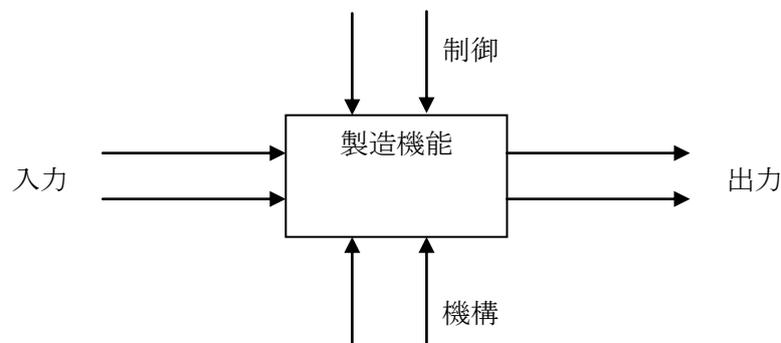


図 2-1 機能箱とインターフェース矢印

こうした箱と矢印は、より一般的な機能を構成するダイアグラム上にある従属機能 (sub-function) を関係付ける為に用いられる。このダイアグラムは「拘束ダイアグラム (constraint diagram)」であり、それぞれの従属機能を拘束する特定のインターフェースだけでなく、インターフェース拘束のソースと目標 (source and target) も示している (図 2-2)。

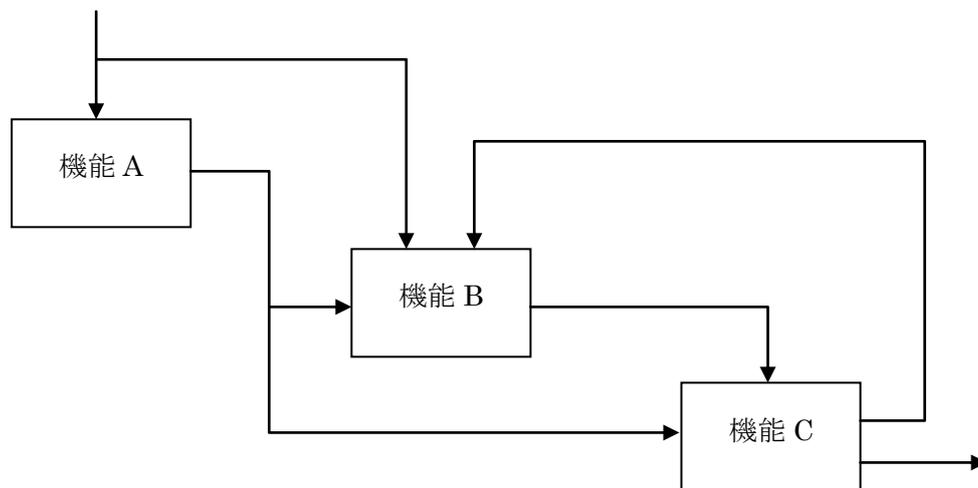


図 2-2 拘束ダイアグラム

機能 B は 1 つの入力と 2 つの運用管理により拘束されており、それにより生成される 1 つの出力は、機能 C を拘束している

ここにおける「拘束 (constrains)」という言葉であるが、機能は、箱に入るように図示された資材や情報を使用するが、これはつまりはインターフェースによって実行が拘束されているということであり、インターフェース矢印の中身が供給されるまでは、その機能を発揮できないという意味での拘束を示している。そして同様に、この機能の実行は、インターフェース矢印の中身の詳細 (量、数、等) にも依存し、拘束されている。

2. 3. 2 詳細の段階的開示による情報伝達

(Communication by Gradual Exposition of Detail)

IDEF0 の最も重要な特徴の一つが、モデルを構成するダイアグラム構造を通して、段階的に詳細が階層を増して実装されているということである。この方法により、読解者はそれぞれのダイアグラムから、新しい情報を取扱い可能な量で十分に囲い込まれた話題 (topic) として情報を提供されるため、情報伝達能力が優れている。

IDEF0 モデルの構成を図 3-3 に示す。ここでは一連の 4 つのダイアグラムが、それぞれのダイアグラムに関する形をとっている。

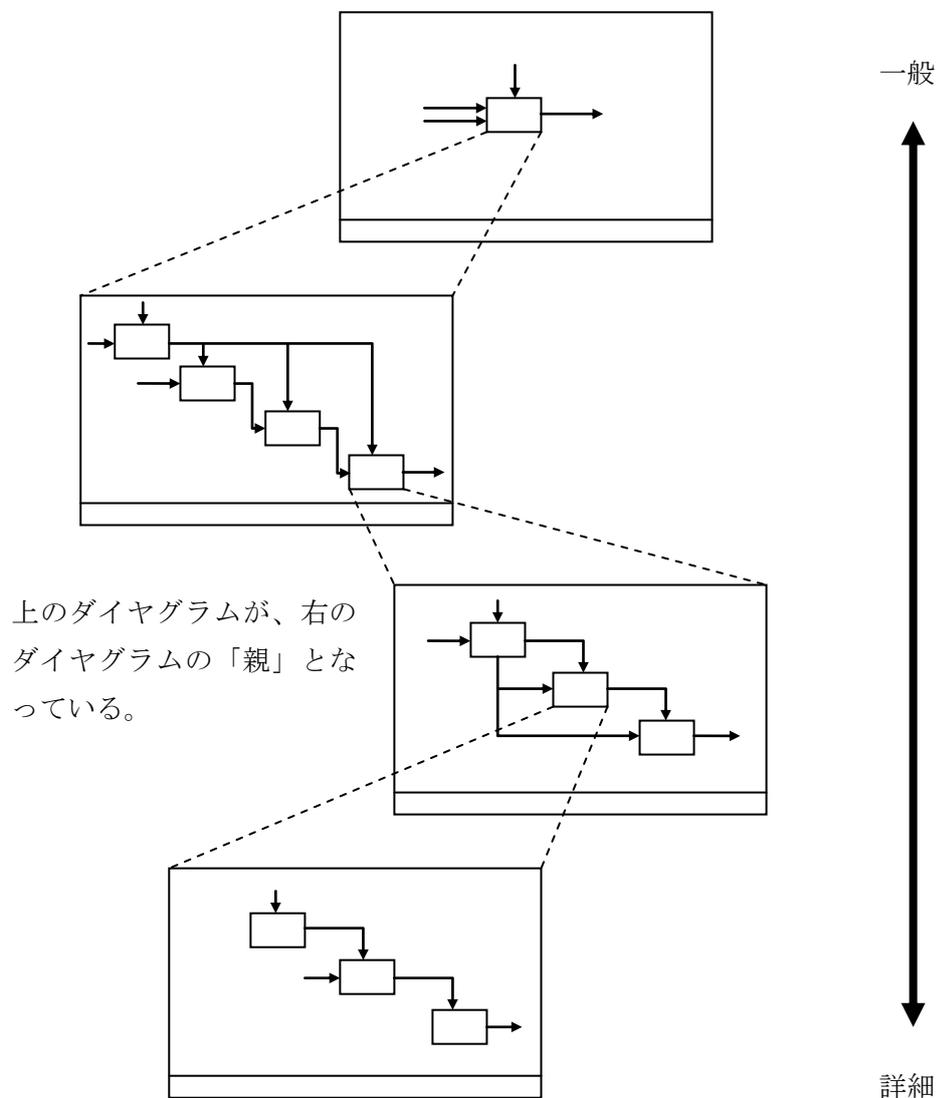


図 2-3 IDEF0 モデル構造

全ての内容物が、他のダイアグラム内で分解することができる。
各ダイアグラムが親ダイアグラムの箱の「中身」を示している

IDEFO のモデルは、システム全体を、箱（機能）1つと、システム外部からその箱（機能）へと繋がるインターフェース矢印という、単純なユニットとして表現する事から始められる。システム全体を1つの箱として表現していることから、この箱に説明的な名前をつけるのが一般的である。インターフェースもまた、外部からシステム全体へと入るインターフェースセットであることから、同様に名前を記述する。

システムを1つのモジュールとして表現している箱は、次の段階で、インターフェース矢印によって接続された複数の箱によって表現された別のダイアグラムとして表現される。この複数の箱は、1つの親モジュールにある主要な従属機能（サブモジュール）を表現している。この分解（*decomposition*）によって完全なサブモジュールのセットが生成される。この時、各サブモジュールは、インターフェース矢印によって境界が定義された箱として表現される。このサブモジュールの箱は、それぞれがまた同様に、より詳細な物へと分解されて行く。

IDEFO では、この分解による詳細化において、規則を設けている。1つのモジュールは常に、3個以上6個以下のモジュールへと分割するようになっている。上限が6個になっているのは、複雑な主体物を記述する際に階層構造の使用を促す為であり、下限が3個になっているのは、対象を分解する際に、十分な詳細化を行わせる為である。

モデル内の各ダイアグラムは、別のダイアグラムと相互接続矢印によって結び付けられることで、関係性を正しく表現している。1つのモジュールがサブモジュールへと分解されるとき、サブモジュール間のインターフェースは矢印として表される。サブモジュールの名前と、そのサブモジュールが持つラベル付きのインターフェースとは、そのサブモジュールの切り取られた背景（*bounded context*）を定義している。

全ての場合において、各サブモジュールは、その親モジュールの範囲内にある要素のみしか持つことが出来ないように制限されている。更に、モジュールはいかなる要素を省略する事も許されない。従って、これまでも指摘したように、親箱とそのインターフェースによって背景（*context*）が表されるのである。そしてこの厳密な境界線に対して、何も足されず、何も引かれないのである。

（訳注：サブモジュールがモジュールの境界を跨がず、全てそのモジュールの境界内にある。またサブモジュールによって表現されない部分が残ってはならず、全てのサブモジュールの合計がぴったりモジュールとならなければならない）

2. 3. 3 統制されたチームワーク (disciplined teamwork)

IDEF0 の方法論 (methodology) には、複数のサブシステムを 1 つの IDEF0 構造へと統合する支援をするだけでなく、多数の集団によってモデルの開発と評論とを行う為の手続きが含まれている。司書 (librarian) ルールと司書手続のような追加の支援手続もまた、IDEF0 方法論に含められている。(こうしたルールと手続きの幾つかは、キットサイクル評論手続 (? Kit Cycle critique procedure) のように、他の IDEF 技法でも使用されている)

IDEF0 モデルの作成は、こうした「統制されたチームワーク」の最も基本的なものである。モデルの作成は、通常 2 人以上が参加が必要となる動的なプロセスである。プロジェクトを通して、著者 (author) によって作成された初期 (initial) ダイアグラムは、プロジェクトメンバーへと配布され、評価とコメントが行われる。ダイアグラムへのコメントを要求されている全てのメンバーは、コメントの記述と同意をダイアグラムの著者へと行うように決められている。この作業サイクルは、ダイアグラムが最終的なモデルとなり、公的に承認されるまで繰り返される。

IDEF には、全ての決定と、その代替アプローチを記録し、プロジェクト期間中に開示し続けるための手続きも含まれている。著者によって作成されたダイアグラムのコピーは、見識のあるコメンター (knowledgeable commenter) によって評論され、コピーの上に直接に提案が書き込まれる。提案には受諾、もしくは拒否と記述されると共に、その理由も書き込まれる。変更と修正が行われると、古くなったバージョンのダイアグラムは、プロジェクトファイルに保存される。

ダイアグラムは修正とコメントを反映して変更される。詳細化の過程でモデルへと追加されたダイアグラムもまた評価と変更が行われる。このようにして最終的に出来上がったモデルは、与えられた視点 (viewpoint) での、与えられた目的の為のシステムの表現物 (representation) として同意されたものとなる。この表現物は、初期 (initial) プロジェクトとは関係ない人でも簡単に読むことが可能であり、簡単な報告会 (short stand-up briefing) や 1 週間をかけたウォークスルー (? walkthrough、予行演習) でのシステム定義の説明と、システム変更作業の新しいプロジェクトの編成に使用されている。

3. IDEF0 ダイアグラムの理解

モデルはシステムを表している。どのようなシステムであるか、そして何をしているかもしくは何が動いているのかということが、モデルに書かれている。

システムは接続もしくは相互依存した、一緒に作用することで使いやすい機能を実現する、部品群によって構成されている。部品には機構 (machinary) や情報 (information)、オブジェクト (objects)、プロセス (process)、ソフトウェア (software)、人 (people) といったものが含まれている。IDEF0 は、システムやシステムの部品によって実行される機能と、機能インターフェースを通して移動する情報や物の記述に使用する事が可能である。

IDEF0 は、ダイアグラム、説明文 (text)、用語集 (glossary) によって構成されるモデルによって、システムを表現している。ダイアグラムは箱と矢印だけで構成されている。こうしたダイアグラムでは、箱は活動 (activity) を、矢印はシステムによって処理される物を表現している。

3. 1 IDEF0 のシンボル

3. 1. 1 ダイアグラム

モデルは、複雑な主体を構成部品へと分解する支援的記述を伴った、一連のダイアグラムによって構成されている。初期ダイアグラム (initial diagram、最初に書かれる根幹となるダイアグラム) は、最も一般的なダイアグラムか、もしくはシステム全体の概略ダイアグラムになっている。このダイアグラムは、主要な構成物を箱として表示している。主要な構成物のそれぞれの詳細は、複数の箱で構成された他のダイアグラムで表現されている。こうした箱は、ダイアグラムへと分解して行くことが可能であり、その結果、システムは必要な詳細階層に至るまで記述してゆくことが可能となる。

詳細化ダイアグラム (detailed diagram) は、より一般的なダイアグラム上にある箱の分解物 (decomposition) である。それぞれの詳細化 (分解) の過程において、一般的なダイアグラムは詳細化ダイアグラムの「親」であると言え、そして子である詳細化ダイアグラムは、親の「内部」に収まっているべきである。(図 3-1 参照)

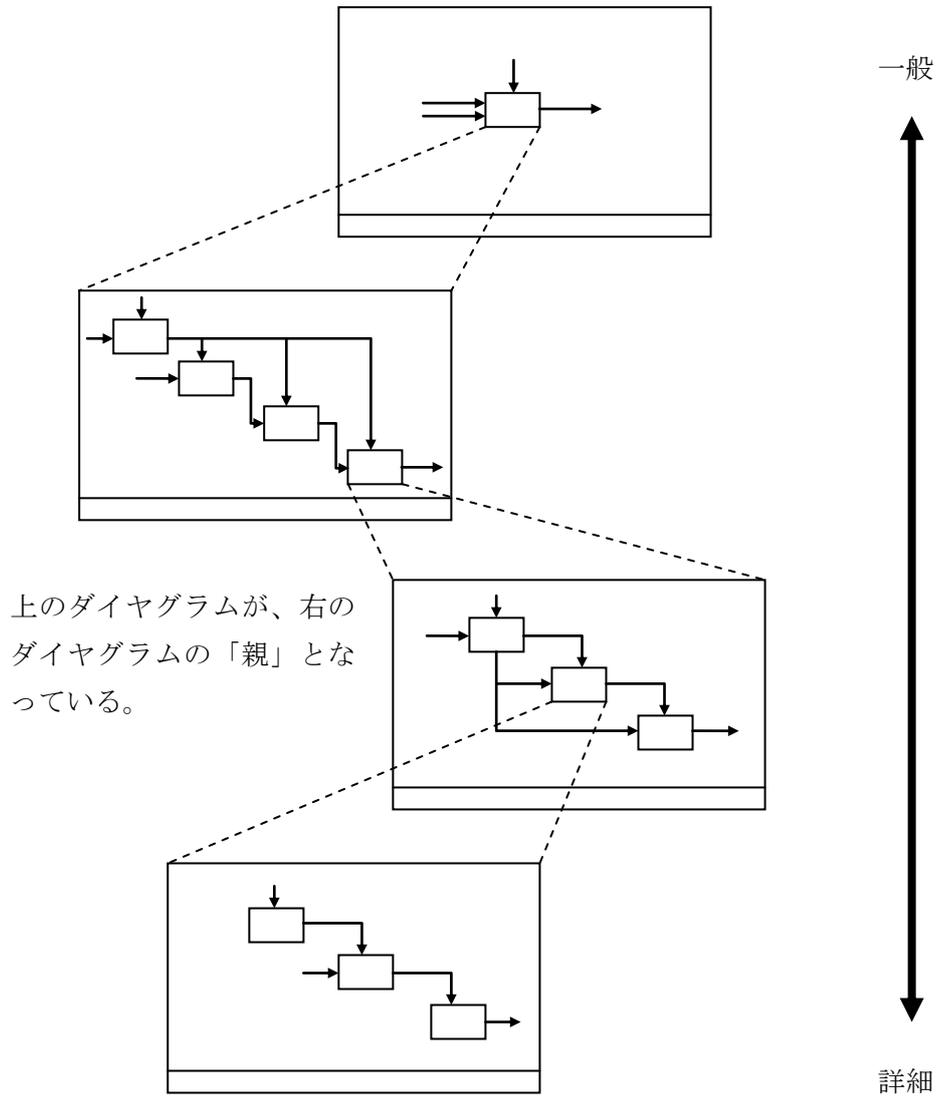


図 3-1 ダイアグラムの分解

箱はシステム機能（活動（activities）、行為（actions）、プロセス（processes）、運用（operations）の何れか）を、また矢印はデータ（情報もしくはオブジェクト（objects））を、それぞれ表している。上層のダイアグラムの箱は、下層のダイアグラムで複数の箱と矢印によって、その詳細が表されている。上層ダイアグラムの箱への入出力矢印は、そのまま下層ダイアグラムへの入出力矢印となるが、これは上層の箱と下層のダイアグラムとが、システムにおいて全く同じ部分を表しているからである。

IDEFOの基本原則では、ダイアグラム内の箱の数は3個から6個までと決められている。これは、継承階層毎にシステム的に開示される詳細さを定式化したものである。箱数の上限が6個である理由は、生理学上の実験（physiological experiment）から、人間は5～7個以上のコンセプトを同時に把握することが困難であるということが判明しているからである。また箱数の下限が3個である理由は、1回分の分解（詳細化）作業が有意義なものとなるように、そこでの詳細化の十分さを保証する為である。上層のダイアグラムは幅広い詳細範囲を包含していることから、ダイアグラム内の箱と矢印に使用される言葉は、概略で一般的なコンセプトを説明するものでなければならない。下層へとダイアグラムが継承されて行くに従い、段階的に専門的用語の使用を増加することで、次第に詳細を明らかにしてゆくのである。

3. 1. 2 箱 (boxes)

ダイアグラム上の箱は、機能（function）を表している。この機能（箱）は、（実行する為の）必要物や（実行）手段といった（実行に）必要な側面を他に識別することなく、何が実行されるべきか（のみ）を示している（※）。機能は、能動詞句（active verb phrase）によって箱の中に記述される。1つのダイアグラム内にある全ての箱には、右下に通し番号が1から6までふられている。

※訳注：実行に必要な物や実行手段は、下層ダイアグラム内で説明し、上層の箱には、こうしたものは説明しないようにすることで、判り易くしているのではないと思われる。

機能は、能動詞句（active verb phrase）によって命名可能なものである。これには、以下のような具体的な物から概念的なものまで（from the concrete to the conceptual）が含まれている：

締める (tighten) 組立てる (assemble) 分類する (classify) 適用する (adapt)
取付ける (attach) 複写する (transcribe) 建設する (construct) 解決する (resolve)
計測する (measure) 評価する (evaluate) 解く (solve) 開発する (develop)

こうした機能は、ある一定期間 (period of time) に発生するものである。機能は名詞 (noun) であってはならない。例えば「維持管理 (maintenance)」等 (訳注: この例では日本語にすると違和感が無くて困る。英語だと maintain と maintenance とで言葉が明確に分けられている)

3. 1. 3 箱と矢印の関係

箱に接続される矢印は、機能によって必要となる、もしくは生成される、オブジェクト (object、物体) もしくは情報を表現している。このオブジェクトもしくは情報となるものは、矢印の下側に、名詞として記入される。「データ (data)」とは、名詞として記述可能な、情報やオブジェクトといったものである。矢印は、機能の連続や流れではなく、箱を定義するものとして規定されている (図 3-2 参照)。

矢印が出入りする箱の側は、それぞれが入力 (input)、制御 (control)、出力 (output) という矢印の役割を示している。入力矢印は箱の左側へと入るが、これは機能を実行する為に必要となるデータを示している。箱の右側から出ている出力矢印は、機能を実行する事によって生成されたデータを示している。左手から右手へと、機能がデータを変換するのである。入力されたものが、機能によって出力へと変換される。入力・出力という用語を用いることで、箱が処理前の仕事の状態 (state of affairs) を処理後の仕事の状態へと変換しているという観念を示している。

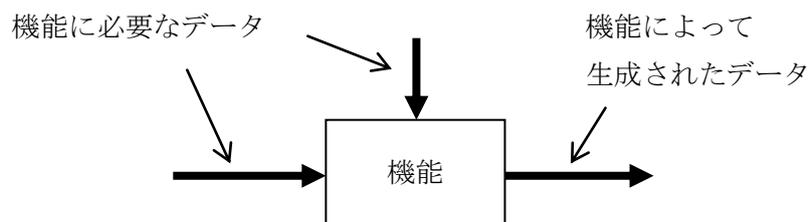


図 3-2 矢印は、各箱の意味を明確で限定なものにする

制御（control）は、機能を支配（govern）する状態や環境を示している。制御矢印の役割は、入力矢印とは異なっている。この制御矢印と入出力矢印との区別は、システムの運用を理解する為に重要である。「ある矢印が単なる入力としてのみの役割でない」と明確にわかるようであれば（訳注：入力以外の役割も持っているようであれば）、その矢印は制御である」と仮定することができる。全ての機能箱は1つ以上の制御矢印を持っている。

箱の下の矢印は、機能を実行する人もしくは機器といった、機構（mechanism）を示している。入力と出力が機能によって「何（WHAT）」が処理されるのかを示しているとする、制御は「何故（WHY）」処理されたのかを、機構は「どのように（HOW）」処理されたかを示している。（図 3-3）

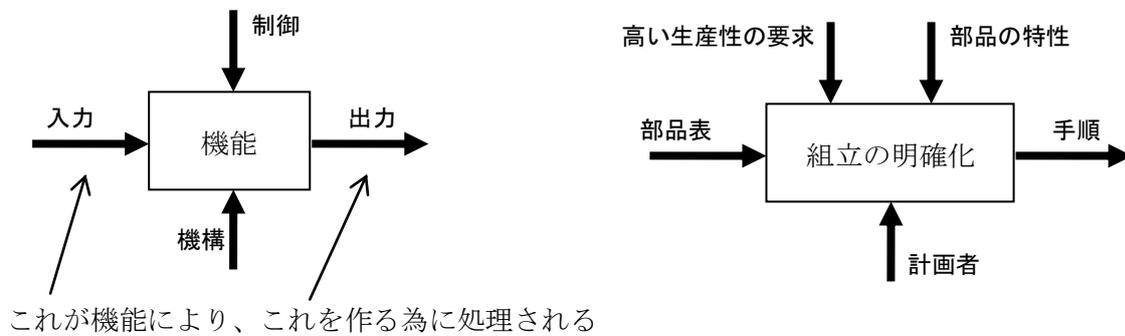


図 3-3 箱と矢印の関係

複数の箱は固定された活動のみを実行するだけでなく、関連性を持った機能群をも表現している。1つの箱は異なった環境の下で、異なった入力と制御の組み合わせによって異なった出力を生成することにより、その機能の持つ色々な部分を実行している。これは箱の相違活動（different activations）と呼ばれている。箱の辺上に複数の矢印が存在しているということは、相違活動が存在するという事を示している。

ある言葉が名詞（データ）か動詞（機能）か不明瞭である場合には、言葉に(n)か(v)を付けて区別する（訳注：名詞は noun、動詞は verb、それぞれの頭文字）。例えば「記録(Record)」という言葉には、記録という意味と、記録するという意味とがある。この時、「記録(n)」なら前者に、「記録(v)」なら後者という意味で使用されるという事になる。

IDEFO ダイアグラム上の矢印は、データの拘束（data constraint）を表しているのであって、流れや順番を示しているのではない。ある箱の出力の、別の箱の入力もしくは制御への接続は、拘束を示しているのである（図 3-4）。データを受けている箱は、そのデータが別の箱で生成されることで使用可能な状態になるまでは機能する事が出来ないため、拘束を受けていると言えるのである。ある箱に入っている矢印は、機能を実行する為に必要な全てのデータを表している。

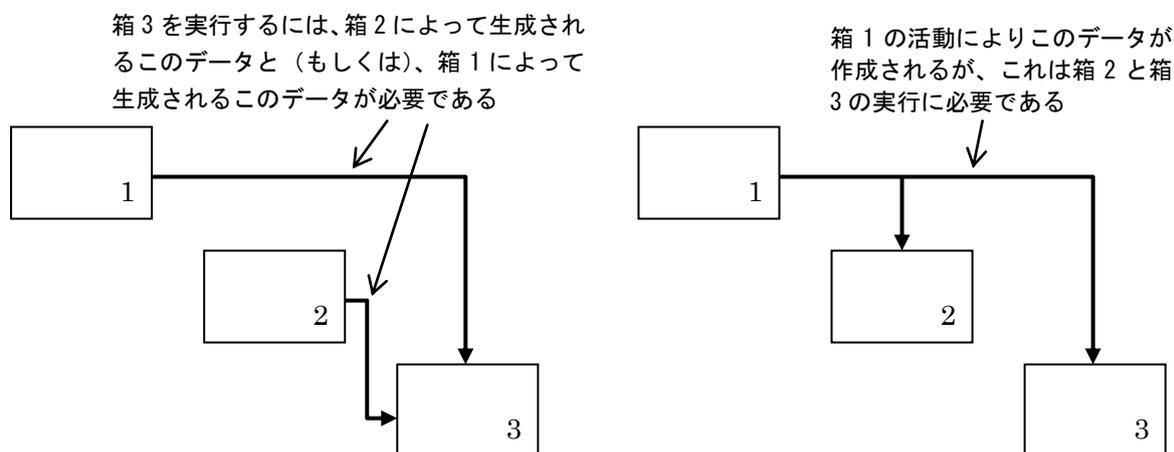


図 3-4 拘束の意味

1つのダイアグラム上にある複数の機能は、必要となる拘束が満足されれば、同時に実行することが可能である。矢印によって箱が接続（connect）され、1個の箱の出力が、1個以上の他の箱に必要なデータの一部を供給する（図3-5）。

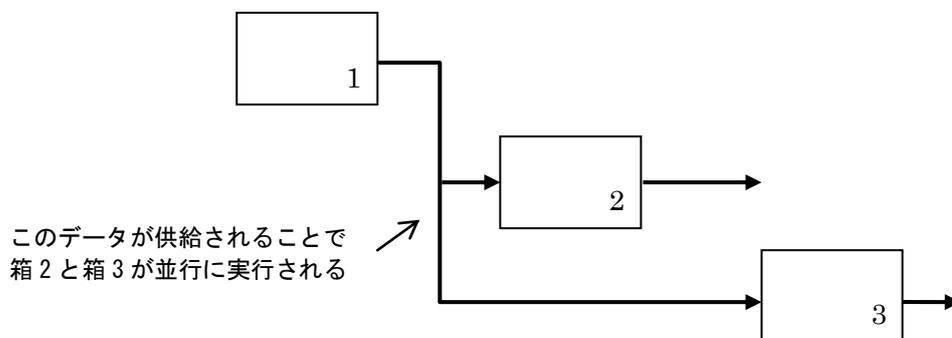


図3-5 同時行動

IDEFOダイアグラムは、順序も時間も不明瞭である。フィードバックや繰り返し、連続プロセス、(時間の)重なりのある機能も、矢印によって簡単に図示できてしまう（図3-6）。箱の左（入力）か上（制御）に付いた矢印は拘束されており、その機能のある部分を実行する為に必要となるデータもしくは物体を表している。例えば、見直し（review）に出されたシステムの仕様案（draft system specification）は、受諾されて完成仕様となるか、もしくはコメントと、新しい仕様で適用すべき要求とを付けて送り返される。送り返された場合には、設計機能がもう一度実行される。システム要求に関しては、設計と見直し（review）の両方で実行されている。

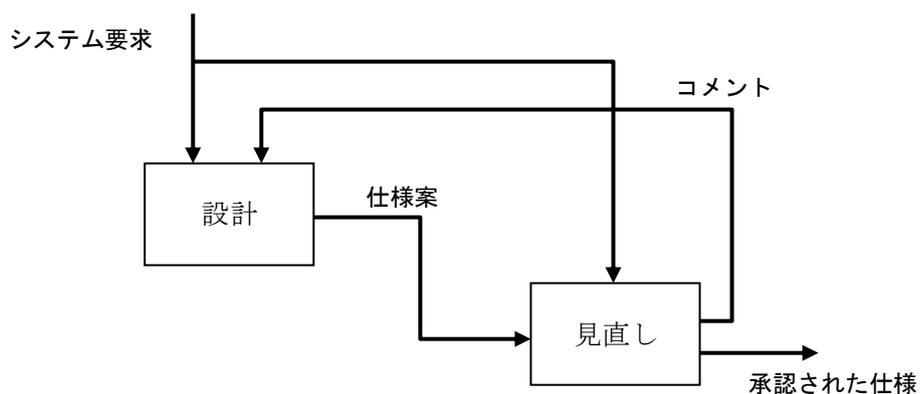


図3-6 フィードバックの例

矢印は、分離 (**branch**※、同じデータが 2 つ以上の機能で必要となることを意味する) や合流 (**join**、同じクラスのデータが、2 つ以上の機能で生成されることを意味する) を行うこともある。分岐は、それぞれが同一物、もしくは同じ形式 (**same general type**) だが異なった物を表している。分岐と合流に付くラベルは、直下のダイアグラムで箱の詳細が説明されるのと同じように、より一般的な矢印で詳細が説明される。

(※訳注：別のページでは、分岐 (**fork**) と合流 (**join**) を併せて枝 (**branch**) とされている。)

データ矢印も機能箱のように、種類 (**category**) を表している。上層でのデータ矢印を、「パイプライン」や「電線管 (**conduits**、複数の電線を通す管)」として考えると便利である (訳注：どちらも上層では 1 本ものとして扱うが、下層では別々の複数の線・管の集合となっている)。上層の矢印は一般的なラベル (**Label**) をもっている。矢印が分岐する場合には、それぞれの分岐先に、より明確なラベルを持たせている (図 3-7)。矢印のラベルは、作図者の意図が読者に伝わるようにしなければならない。矢印を少なくすることで、混乱を減らし、ダイアグラムを読みやすくすることができるが、意味が正確に伝わるようにラベルの言葉を慎重に選択しなければならない。

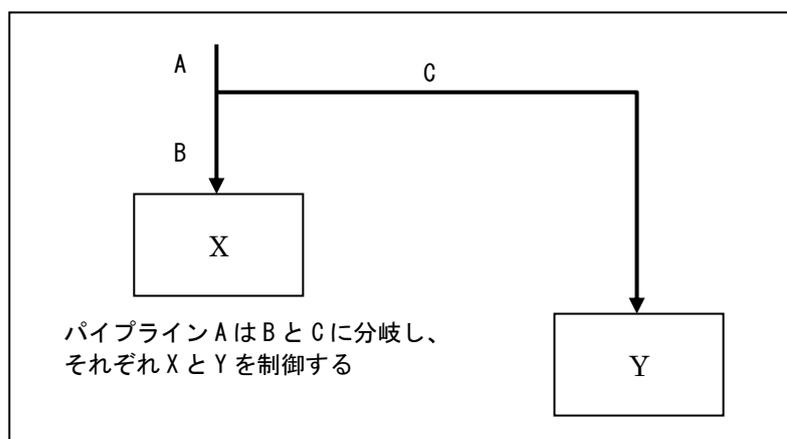


図 3-7 矢印分岐の例

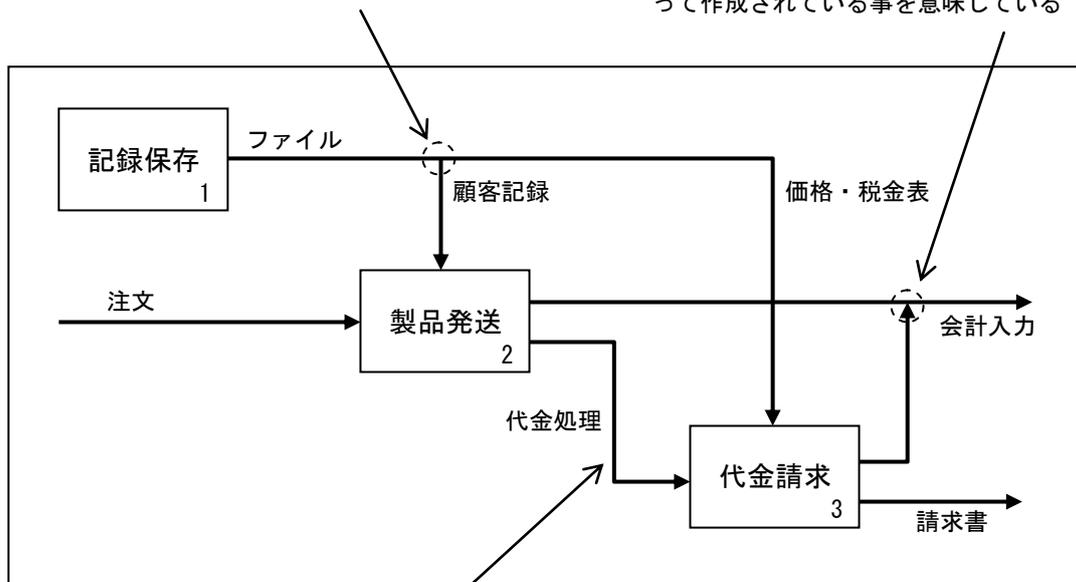
どのようなダイアグラムであっても、データは内部矢印 (**internal arrow**、始点終点が共にダイアグラム上の箱に接続されている) か、もしくは境界矢印 (**boundary arrow**、端点の 1 つが箱に接続されず、その非接続部分がダイアグラム外にある機能によって生成されるか、もしくは利用されるということを意味している) になっている。

3. 1. 3. 1 箱間の矢印の接続

完全なダイアグラムとするためには、3個から6個の箱を描き、それらを入力、出力、制御の矢印で接続する。出力矢印は、一部もしくは全てが他の箱の入力もしくは制御（もしくは機構）矢印となる。全ての種類の矢印の分岐と合流は、箱の関係性を表現している。出力矢印は分岐することで複数の箱にデータを提供している。片方の端点が接続されていない矢印は、ダイアグラムの範囲外から提供される、もしくは範囲外で消費されるデータを表している。

この分岐は、(箱1によって生成された)
「ファイル」が、(箱2で必要な)「顧客情報」と
(箱3で必要な)「価格・税金表」とで構成
されているということを意味している

この合流は、「会計入力」が製品発送(箱2)
の出力物と代金請求(箱3)の出力物によ
って作成されている事を意味している



この入力矢印と出力矢印の繋がりは、製品発送(箱2)の「注文」が「代金処理」として記録され、そしてそれが箱3で代金請求されることで「請求書」へと反映される、ということを意味している。

図 3-8 矢印の接続

3. 1. 3. 2 機構矢印 (mechanism arrow)

箱は機能を表している。左からの入力データは、出力データへと変換され右から出て行く。上から入る制御は、機能の実行を制御する。下から入る機構 (mechanism) は、機能が実行される際の手段を示している (図 3-9)。「機構」は、機能の実行を補助する人やコンピューター、機械、その他機器類である。入力、出力、制御を伴った箱そのものは、システムが「何 (WHAT)」をするのかを示している。それに対して、機構は機能がどのような (HOW) にして実行されるのかを表しているのである。機構を省いたダイアグラムは、システムが何の機能を実行しなければならないかを示している。機構は、そうした機能がどのように実行されるのかを明確にしてくれる。

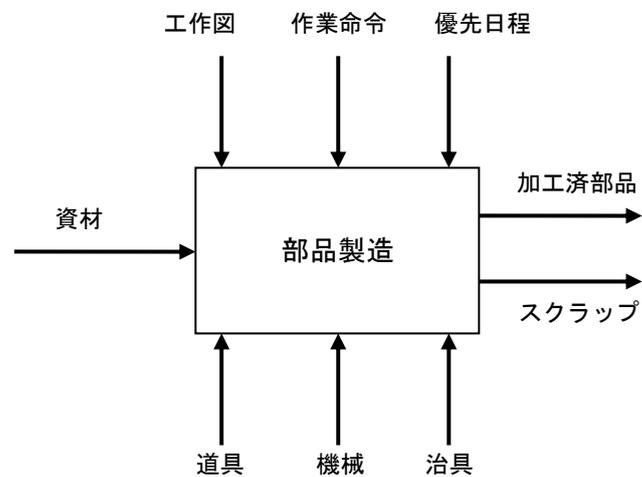


図 3-9 機構の例

下向きの機構矢印 (「呼び出し (call)」と呼ばれている) は、箱の機能を完全に実行する「システム」を指示している。この更なる詳細が必要な場合、機構そのものである別のモデルを参照することになる (図 3-10)。

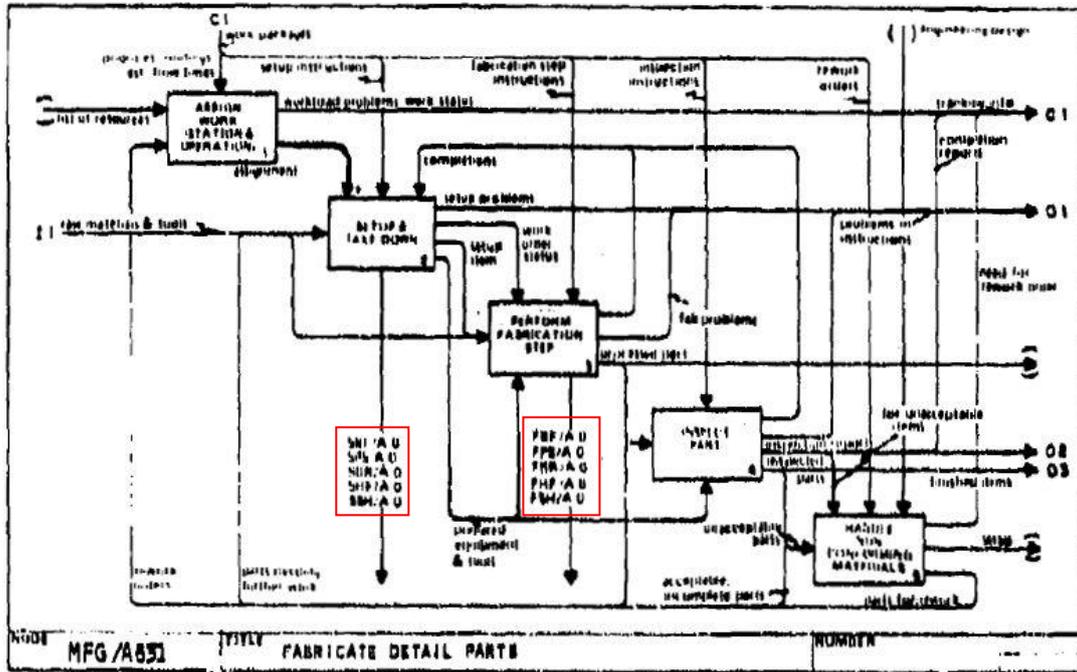


図 3-10 機構モデルの呼び出しの記述（赤四角部）

もしも別の箱が機器を作成・準備する機能であれば、機構矢印は、その機能の出力となることもある（図 3-11）。

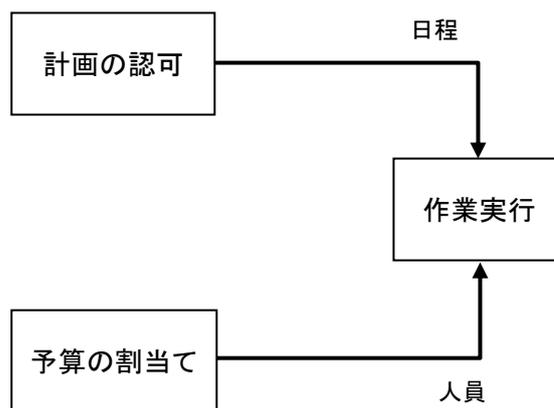


図 3-11 出力が機構になる例

3. 1. 4 IDEF0 ダイアグラムの例

図 3-12 は、「製品製造 MFG0」の機能モデルから取り出してきた、完全な IDEF0 ダイアグラムである。

A612 (図 3-13) の 3 つの箱は、A61 「生産命令の制御」の箱 2 を分解したものである。生産要求と現在の工場の負荷を基にして、適切な作業負荷を予測している (A612 の箱 1)。そして①適切な作業負荷、②工場能力、③具体的な工場命令の 3 つから、箱 2 で日程の調整が必要であることが判断され、箱 3 で実際に日程調整が行われている。

注目すべきは、個々の箱は、それぞれ異なった環境下で機能を実行しているという事である。箱 1 は、箱 3 からの「変更命令発行日 (? revised order release dates)」が出ていない場合か、もしくは「生産要求」と「工場負荷状況」が変更されなくとも「変更命令発行日」が出ている場合に実行される。このような、環境によって変化する挙動は、箱の相違活動 (different activation) として知られている。相違活動は公的に明示されるものであるが、しかし多くの場合、箱と矢印の表記に詳しい人に依って、自然に、直観的に、理解されている。例えば、箱 2 は「停止命令と資材不足の報告」が無い場合に実行される。また箱 3 は、どのような場合でも「問題」は発生しない。

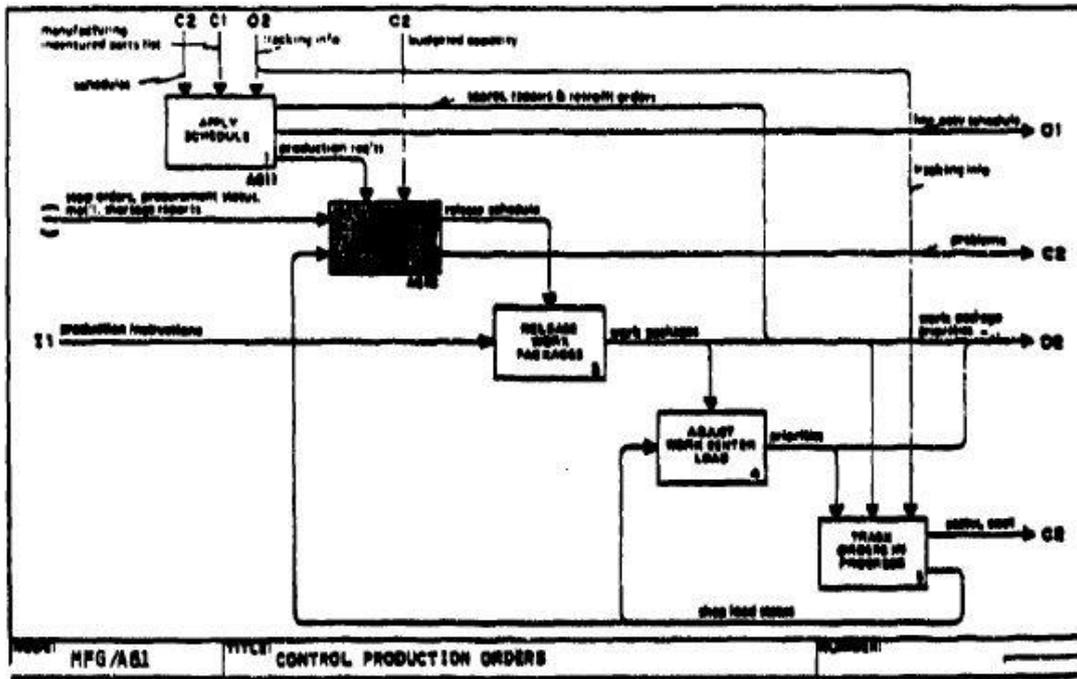


図 3-12 生産命令の制御

(訳注：文字が潰れて読めないなので、原資料の図をそのまま掲載)

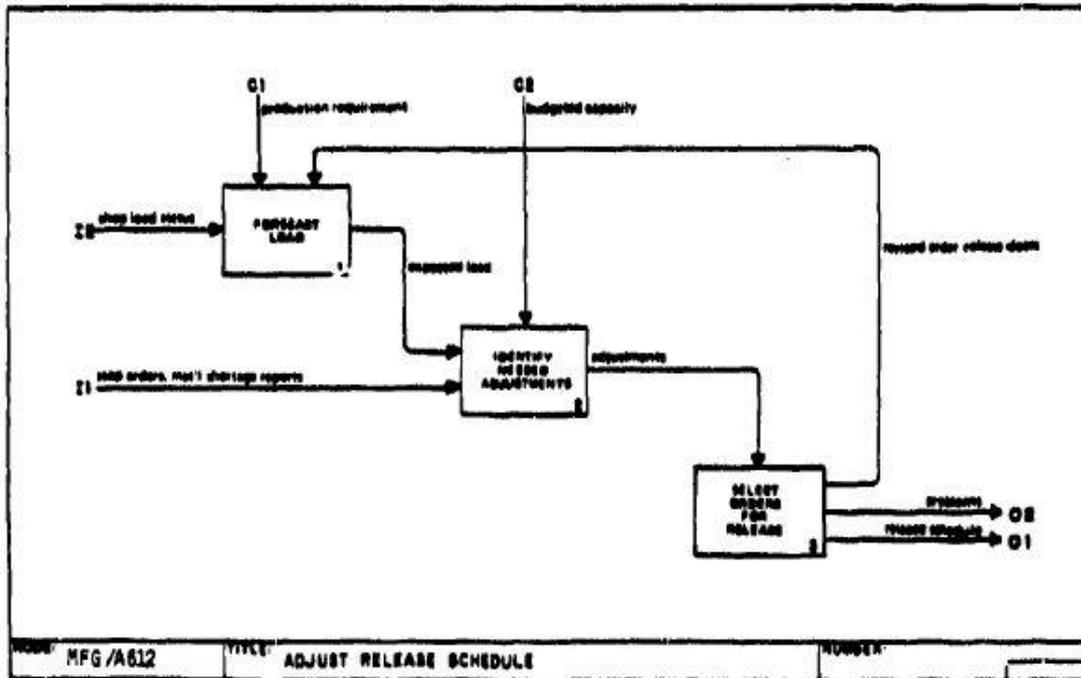


図 3-13 ダイアグラム A61 内の箱 2 の分解

(訳注：文字が潰れて読めないなので、原資料の図をそのまま掲載)

3. 2 追加シンボル

わかりやすい整然とした (coherent and consistent) モデルを作るには、ダイアグラムの構成に更なる表現方法が必要となる。

ダイアグラム、FEO (3. 2. 1. 1 参照)、文章、用語集で構成されるモデルを作るには、以下の物が必要である：

- ・それぞれのダイアグラムのモデル内での位置と、ダイアグラムに関する補助記述の指示。これは参照表現として実現される。
- ・親ダイアグラムの矢印に接続している境界矢印の指示。これは ICOM コードによって実現される。
- ・不必要な詳細の省略。これはトンネル矢印で実現される。

こうした取り決めにより、ダイアグラムが現実のシステム機能を正確に反映することが可能となる、シンボルのセットが補完されている。

3. 2. 1 参照表現

3. 2. 1. 1 ノード番号

3. 1. 1 章で説明したように、個々のダイアグラム内の箱の数は 3 個から 6 個に制限されているが、それらのダイアグラム内の箱には、各々に番号が振られている。どのダイアグラム上の箱であっても、その詳細が下層にあるダイアグラムで複数のダイアグラムにより詳細が記述されて行く。こうして、ダイアグラムは階層構造を成すのである。

ノード番号は、全てのダイアグラムと箱の、階層構造内での位置の指定に使用されている。例えば、A21 は A2 ダイアグラム内の箱 1 の詳細ダイアグラムのことである。同様に A2 は、最上層のモデル全体を説明したダイアグラムである A0 ダイアグラム上の箱 2 の詳細ということになる。この階層構造は、ノードツリーと呼ばれるダイアグラム名とノード番号の図として表現されている。図 3-14 は典型的なノードツリーである。

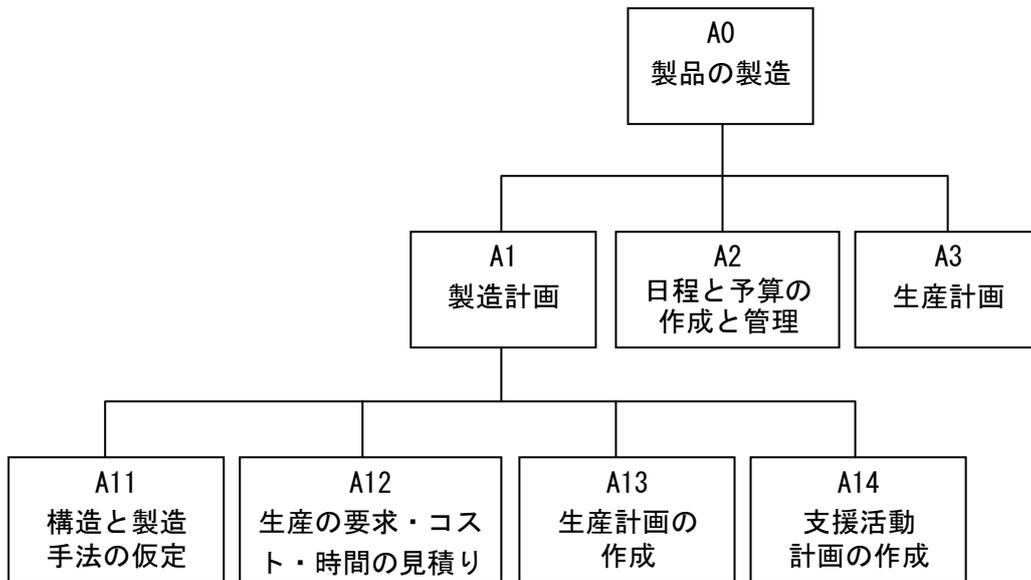


図 3-14 ノードツリーによって表現されたダイアグラムの構成

IDEF0 ダイアグラムの全てのノード番号は、A の文字から始まっているが、これはダイアグラムが「活動 (Activity)」であるか、もしくは機能ダイアグラムであることを意味している。箱が 1 個のみのダイアグラムは、「状況・背景・前後関係 (context)」であるか、もしくはモデル全体の親ダイアグラムである。慣例により、そうしたダイアグラムは「A-0 (A0 ではなく、A-0)」というノード番号を持っている。もしも A-0 全体の背景 (context) を説明する為に完全なダイアグラムが必要な場合には、これは「A-1」という番号になる。必要ならば、更に追加する事が可能である。モデルのトップが A-0 であるにもかかわらず、実際には「A-4」から始まる複雑な対象をしたモデルも存在する。この例は、レポートの第 7 部 (Volume VII) 「製造製品」(MGF0) の機能モデルの中に見る事が可能である。(訳注：「[IDEF0_MGF0.pdf](#)」)

別のノード番号が使用される場合もある。「FEO」(For Exposition Only、説明専用) ダイアグラムは、厳密な階層構造から外れたダイアグラムである。FEO ダイアグラムには、要点を説明する為に、6 個を超えた箱、部分的な矢印構造などが含まれることもある。この FEO ダイアグラムのノード番号には、「F」の文字が付けられている (A2F 等)。用語集の定義によりダイアグラムを支援する場合、用語集を含むダイアグラムのノード番号には「G」の文字が付けられている (A1G 等)。同様に、テキストには「T」の文字が、関係するノード番号の末尾に付けられている (A2T 等)。2 ページ以上の FEO ダイアグラムや用語集も

作成することができるが (A2F1、A2F2…)、しかし説明文については、関係するダイアグラムに対して1ページを越してはならない。説明文、FEO、用語集については、第6章で説明する。

ノード番号はまた、1つのダイアグラム内にある箱の分解先を指示する為にも使用される。もしも箱が分解されている場合には、分解先を示すダイアグラムのノード番号を、箱の外の右下に記述する。図3-15では、箱1と箱2に、分解を意味する参照表現がなされている。

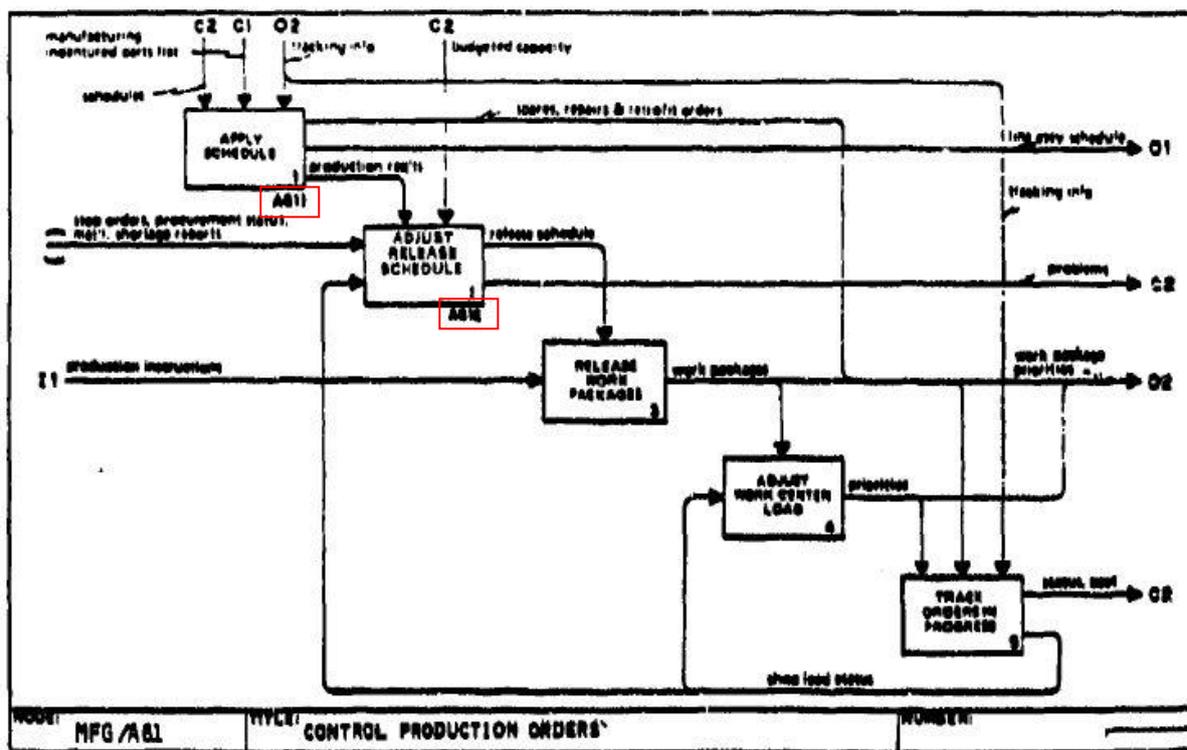


図 3-15 参照表現 (赤四角部)

3. 2. 1. 2 モデル名とノード番号

モデルはそれぞれ、他のモデルと明確に区別可能な名前を持っている。モデル内にあるダイアグラムへの参照は、モデル名に「/」(スラッシュ)とノード番号を加えたものを用いる。モデル内の各ダイアグラムの完全なノード番号は、この形式で記述されている。

モデル名の例: TOPIC

ダイアグラム名の例: TOPIC/A3

3. 2. 2 ダイアグラム境界を超える継続矢印 (continuing arrow)

矢印は、同じダイアグラム上の箱の間に接続した物もあれば、片方の端が未接続であるものもある (図 3-16)。この未接続の矢印は、親箱の入力、制御、出力を表している。こうした境界矢印 (boundary arrow) の源、もしくは行き先は、親ダイアグラムを調べることでのみ判明するようになっている。

非接続端部は親ダイアグラムの矢印と一致させなければならない。全ての境界矢印は、モデルが完全に構成されるまで、親ダイアグラムへと続けてゆかなければならない (図 3-17)。

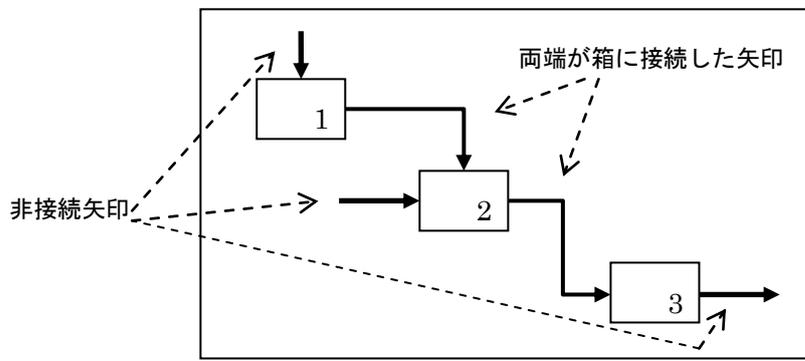


図 3-16 境界矢印は親ダイアグラムに由来している

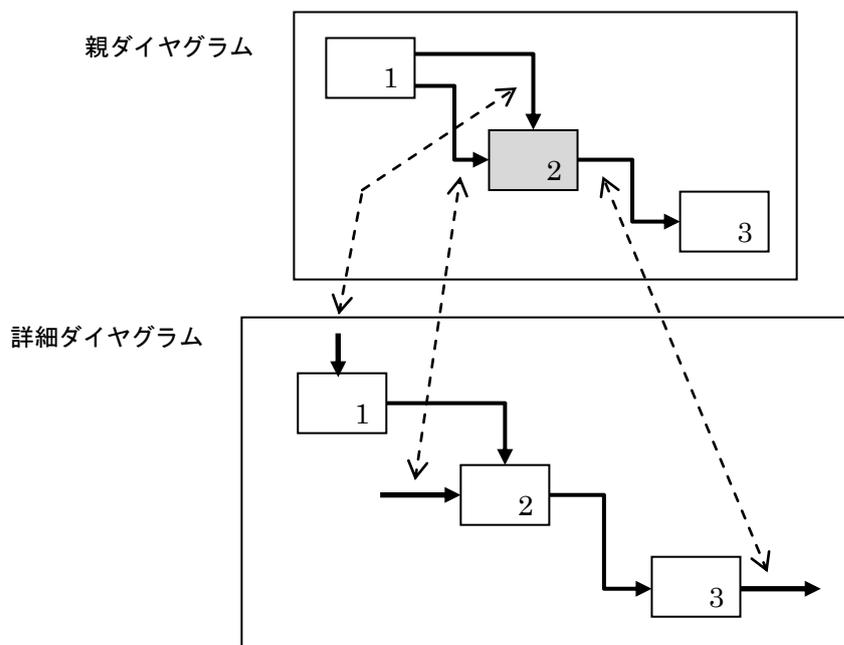


図 3-17 対応は連続・完全でなければならない

A-0階層の境界矢印は、外部矢印と呼ばれている。これは、A-0ダイアグラムは、自分の背景の外もしくはモデルの範囲外にあるモデルと自分との間に存在する全ての矢印と、自分自身とによって構成されている背景（context）についても定義しているからである。

3. 2. 3 境界矢印へのコード付与（Coding Boundary Arrows）

ICOM コードと呼ばれている特殊な記述法により、矢印の接続先を明確にしている。詳細ダイアグラム上の境界矢印の非接続端部に、I、C、O、M の文字を記述するが、それぞれは、ダイアグラムの親箱の入力（Input）、制御（Control）、出力（Output）、機構（Mechanism）を表している。この文字には数字が添えられているが、この数字は矢印が接続する親箱の番号（ダイアグラム内で左から右、上から下へふられている）である。例えば詳細ダイアグラムの矢印上に「C3」と書かれていれば、この矢印は3つ目の親箱の制御矢印と一致しているということである。

このコード番号によって、ダイアグラムはそれぞれの親へと関係付けられる。詳細ダイアグラムが分解されることで親ダイアグラムになると、新しいコードが付与される。この文字と番号による組み合わせ体系（scheme）を使用する事で、親ダイアグラム上で制御もしくは入力を示している矢印は、詳細ダイアグラム上で同じ役割に縛られずに済む。図 3-18 において、親箱の制御矢印 C2 は、詳細ダイアグラムでは箱 1 への入力矢印となっている。モデルの最上層ダイアグラムとトンネル矢印を除いた全ての境界矢印の非接続端部に、ICOM コードを記述されなければならない。

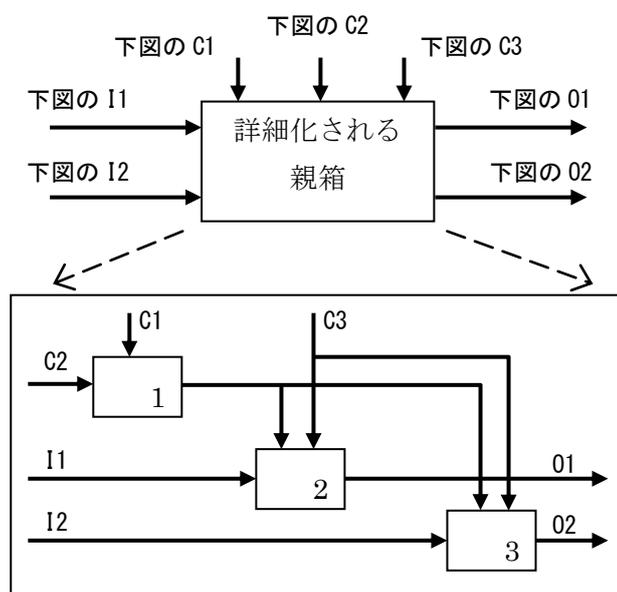


図 3-18 詳細ダイアグラム上に描かれたコード

3. 2. 5 トンネル矢印 (tunnelled arrows)

トンネル矢印は、この矢印によって移動するデータが、特定の階層の詳細とは関係していない事を示している。

矢印の、箱に接続された端部がトンネル化されたものは (図 3-19)、移動するデータが次の階層での分解には必要ないという事を示している。

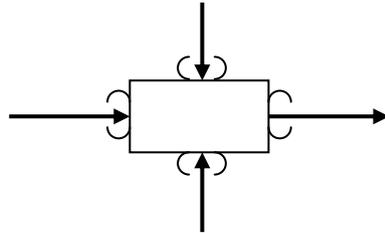


図 3-19 接続端部がトンネル化された矢印

矢印の、非接続端部がトンネル化されたものは (図 3-20)、この矢印で移動するデータが親ダイアグラムとは関係ないか、もしくは親ダイアグラムから供給されていないという事を示している。

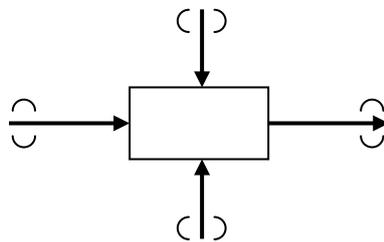


図 3-20 非接続端部がトンネル化された矢印

非接続端部にある括弧は、「この矢印は親ダイアグラムには出てこない。ICOM コードも無い」ということを示している。また箱に接続された端部にある括弧は「この矢印は詳細ダイアグラムには出てこない。また ICOM コードもここからは追跡できず、また明確に参照されることも無い」ということを示している。ある矢印の矢印部分を括弧で囲むことで、1つ以上の詳細階層を飛ばし、ある特定の詳細階層で括弧を使い、再登場させることが可能である。発生元・起源が判明している場合には、括弧の横にそれを記入するなどの、適切な参照表現によってそれを記述しておくべきである。

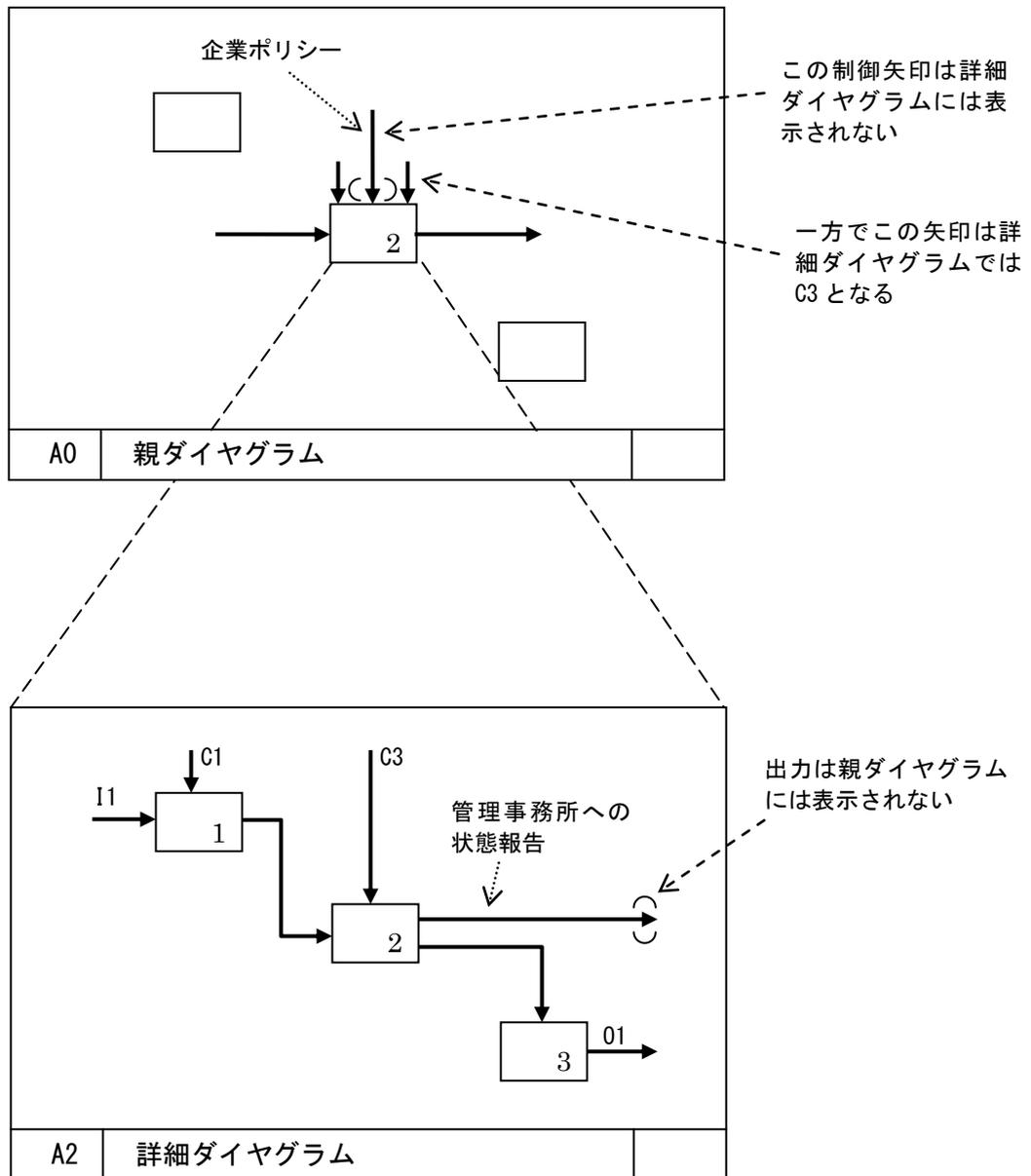


図 3-21 トンネル矢印の例

3. 2. 6 分解の例

「製品製造」(MFG0) という機能モデルを使用する事で、以下のような従属機能で構成された「製品製造」機能全体を見る事が可能である：(訳注：「[IDEF0_MGF0.pdf](#)」)

- ・ 製造計画 (Plan for Manufacture)
- ・ 日程と予算の作成と監督 (Make and Administrator Schedules and Budgets)
- ・ 生産計画 (Plan Production)
- ・ 生産リソースの供給 (Provide Production Resources)
- ・ 製造資材の入手 (Obtain Manufacturing Materials)
- ・ 製品の生産 (Produce Product)

これらの機能は、更に分割されてゆく。「製造計画」の場合、以下のような従属機能へと分割される：

- ・ 構造と製造手法の仮決定 (Assume a Structure and Method of Manufacture)
- ・ 生産の要求、コスト、時間の見積り (Estimate Requirements, Cost, Time to Produce)
- ・ 生産計画の作成 (Develop Production Plans)
- ・ 支援活動計画の作成 (Develop Support Activities Plans)

これらの従属機能は、使いやすさや知識、時間といった制限に行き着くまで、繰り返し分解されてゆく。

この機能と従属機能の構造は、図 3-22 のようなノードツリーによって表されている。

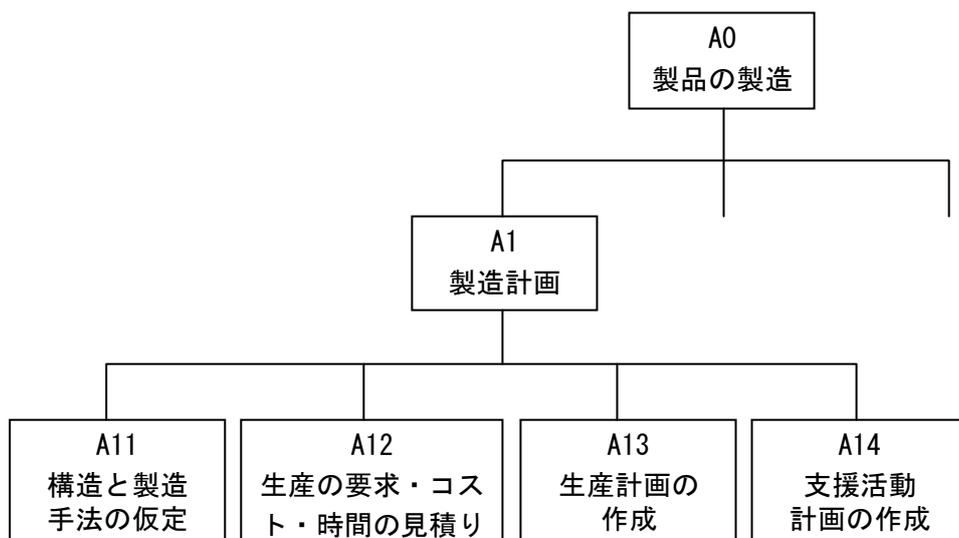


図 3-22 ノードツリー

機能と従属機能の構造は、「ノード見出し (node index)」によっても表現することが出来る (図 3-23)。この見出しの形式は、内容表 (table of contents) の形式と、製造とエンジニアリングで用いられている「インデント化された部品表 (indentured parts list、タブでずらされている部品表)」(資材表 (bill of material)) の形式と似ている。

- A0 製品製造
- A1 製造計画
 - A11 構造と製造手法の仮定
 - A12 生産の要求、コスト、時間の見積り
 - A13 生産計画の作成
 - A14 支援活動計画の作成
- A2 …

図 3-23 ノード見出し (node index)

以下に挙げる IDEF0 ダイアグラムは、先ほどのノードツリーの頂上にあるものと同じ対象から始まっている。この図 3-24 の A-0 ダイアグラムは、「製品製造」というモデルで意味し、規定している情報とオブジェクトとを表している。A-0 ダイアグラムは「製品製造」を記述してゆく為の背景 (context) を形作っているのである。

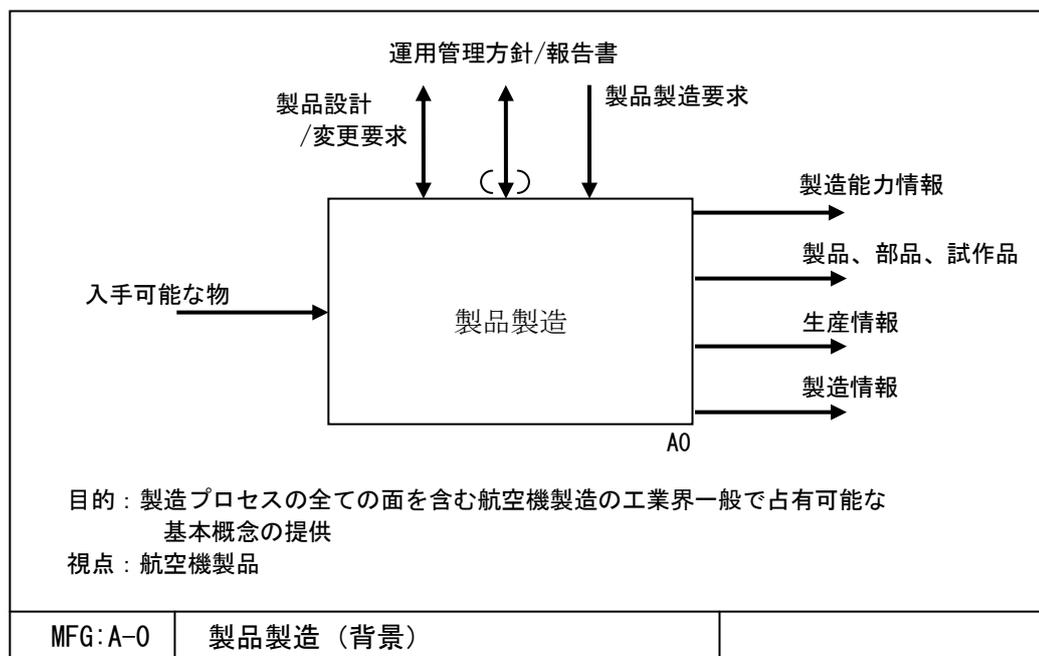


図 3-24 A-0、製品製造

「製品製造」の部分構成する各機能は、製造の複雑な構造（fabric）から選択された詳細な決定と行動を認識可能なように分別したものである。機能の分別は、機能間を移動する情報とオブジェクトであるところの、データの関係性によって定義されている。こうしたデータの関係性により、機能の境界と、各機能の命名に使用される用語（terminology）とが定義されている。

A0 ダイアグラム上の全ての箱は、他のダイアグラムによって更なる記述が為されて行く。A0の最初の箱は、A1「製造の為の計画」ダイアグラムで詳細が記述される。このA1ダイアグラムは、A0上の矢印で定義された「製造の為の計画」に関する部分の、機能とデータのみを表している。

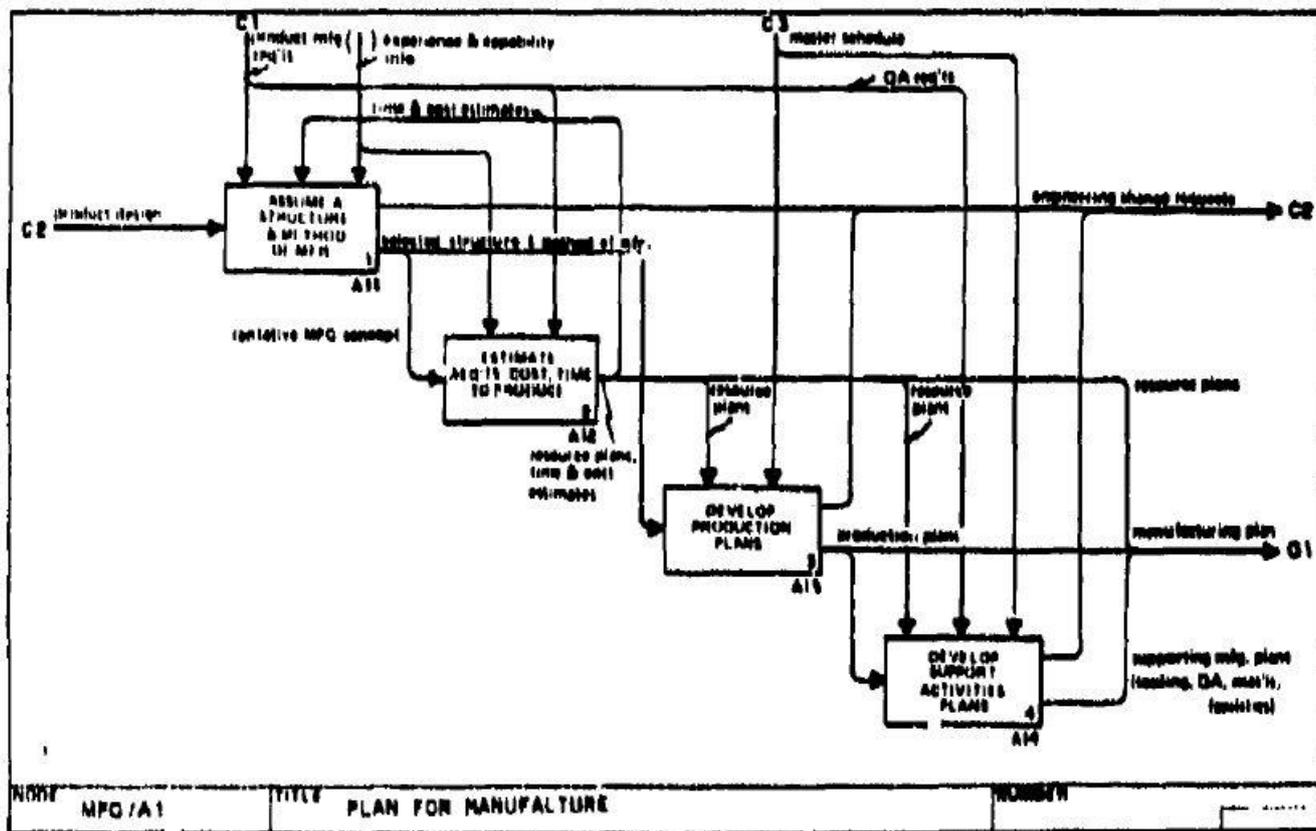


図 3-26 A1 ダイアグラム 製造の為の計画

第4章 IDEF0 ダイアグラムの読解

モデルは、階層構造 (hierarchical manner) で構成された、一連のダイアグラムと関係する資料 (material) によって作り上げられている。ノード見出し、もしくは内容表 (table of contents) も準備しなければならない。ダイアグラムを階層構造順に並べる事で、モデル全体を見渡せるようになり、またどの部分をも、すぐに見に行くことが可能となる。

読解はトップダウンで行われる。最上階層を読んだ後に第1階層のダイアグラムを読み、それから第2階層のダイアグラムを読む、という感じである。もしもモデルのある特定の詳細が必要となれば、ノード見出しを使用して必要となる詳細のある階層レベルへと素早く移動することもできる。

公布する際には、モデルは「ページ隣接 (page pair)」形式で、「ノード見出し (node index)」順でまとめられる。「ページ隣接」形式とは、各ダイアグラムと、それに関する全てのテキストとが、下図のように見開きで並べられたものである。

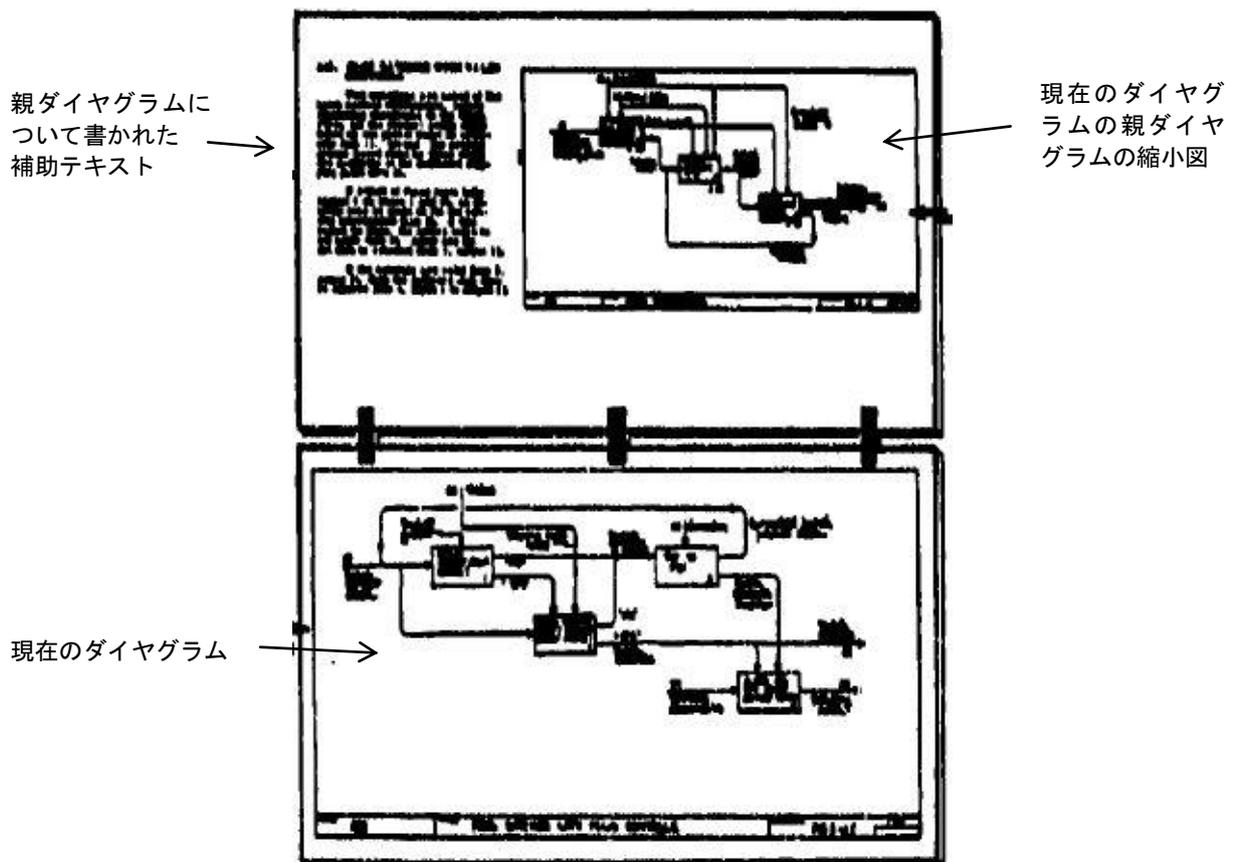


図 4-1 ページ隣接形式

「ノード見出し (node index)」順では、ダイアグラム上の、ある箱 (機能) に関する全ての詳細ダイアグラムを、その箱の次の箱の前に全て記載する。このような並べ方により、通常の内容表 (table of contents) で使用されているものと同じ様に、関係するダイアグラムと一緒にまとまることになる。

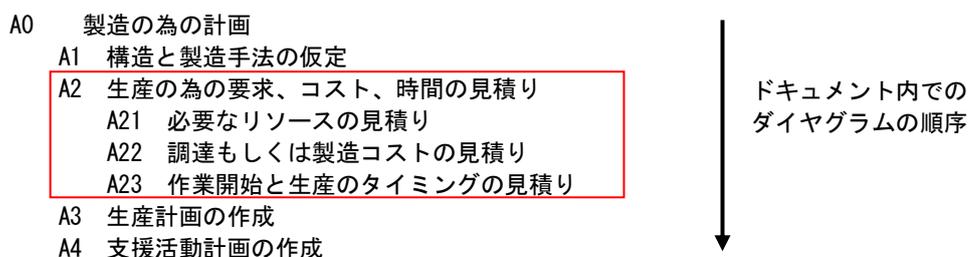


図 4-2 ノード見出しは、ダイアグラムの順序を示している

4. 1 モデルへのアプローチ (Approaching a Model)

モデルでは、システム全体の概要と特定の対象の詳細とが、一緒になっている。全体的な概要を読むには、見出しを使って高階層のダイアグラムを見つけなければならない。

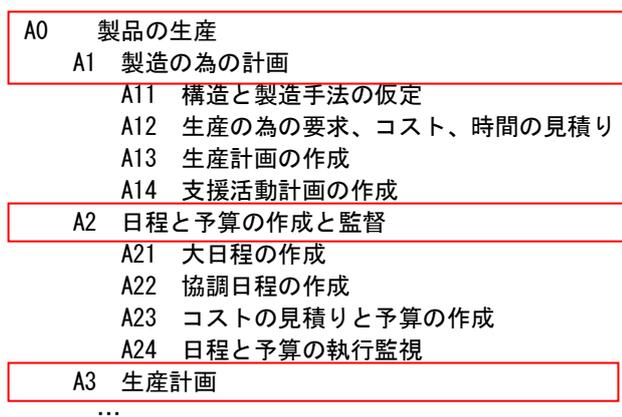


図 4-3 ノード見出し (赤線枠) は、概要ダイアグラムを示している

モデルの詳細を読むには、対象となるものと関係する全ての詳細ダイアグラムを、見出しから見つけ出す。

- A0 製品の生産
 - A1 製造の為の計画
 - A11 構造と製造手法の仮定
 - A12 生産の為の要求、コスト、時間の見積り
 - A13 生産計画の作成
 - A14 支援活動計画の作成
 - A2 日程と予算の作成と監督
 - A21 大日程の作成
 - A22 協調日程の作成
 - A23 コストの見積りと予算の作成
 - A24 日程と予算の執行監視
 - A3 生産計画
 - ...

図 4-4 ノード見出し（赤線枠）は、特定の詳細ダイアグラムを示している

モデルの更なる詳細を調べるには、箱番号の真下にある参照記号を見る。この記号はその箱についての詳細ダイアグラムのノード番号もしくは記載ページ番号を示している。もしもここに何も書かれていなければ、その箱についての詳細ダイアグラムがまだ作成されていないという事である。例えば、ダイアグラム A24 上の箱 3 に右下に A243 と書かれていれば（下左図）、箱 3 についての詳細ダイアグラムのノード番号が A243 ということを示している。

詳細は、モデル内、もしくはモデル間で共有される場合もある。この場合、下向き矢印で、共有詳細がどこにあるかを示している（図 3-10 を参照）。下右図の場合、箱 4 の詳細は、MQ モデルのダイアグラム A4 として書かれている（この下矢印は、「呼び出し (call)」と呼ばれている）



4. 2 ダイアグラムを読む手順

ダイアグラム群の中に、システムについての正確な情報が存在している。それを読み取るには、以下のような手順が必要となる：

- (1) ダイアグラム上の箱を流し読みし (scan)、何が書かれているかの印象を得る。
- (2) 親ダイアグラムを参照し、対象のダイアグラムへの矢印の接続に注目する。この時、「最も重要な」入力、制御、出力の各矢印が何であるかを認識するようにする。
- (3) 対象ダイアグラムの矢印について考える。その時、「最も重要な」出力・制御矢印と、「最も重要な」出力矢印との繋がりによって構成される、主要経路が存在するかどうか見極める。
- (4) 主要経路を中心にしながら、左上から右下まで、ダイアグラムの流れを追う。この時、主要経路以外の矢印が、箱に対してどのように関係しているかに注目する。副次経路の存在の有無についても見極めを行う。取り扱われている状況がどれだけ現実に近いものであるかを考慮しながら、ダイアグラムによって表される流れをチェックする。
- (5) 関係する「FEO」(For Exposition Only、参照専門) ダイアグラムが存在しているかどうかチェックする。
- (6) 最後に、説明文 (text) と用語集 (glossary) があれば、これも読む。

この一連の作業により、それぞれのダイアグラムの主要構造に注意を向けるようになるのである。説明文 (text) により、著者 (author) が何に比重を置いているのかに注意を向ける事が可能となり、また用語集 (glossary) は、ダイアグラム内で使用している専門用語の著者による翻訳となるのである。

ダイアグラムは、それぞれが中心となるテーマを持っているが、そのテーマは最も重要な境界入力矢印から始まり、最も重要な境界出力へと走っている。この、箱と矢印によって形成されている主要経路は、ダイアグラムの主要な機能の概略を示している。ダイアグラムの他の部分は、主要経路の次に来る、限定化された状態もしくは代替状態を表している。

主要経路を追跡する事で、システムの運用（operation）を頭の中で思い描くことが可能である。ある種類のデータ入力、エラーの取扱い、代替可能な出力といったものを加える事で、システムの流れをより詳細化することができる。こうしたワークスルー（walkthrough）により、ダイアグラムの理解をより進めることができる。

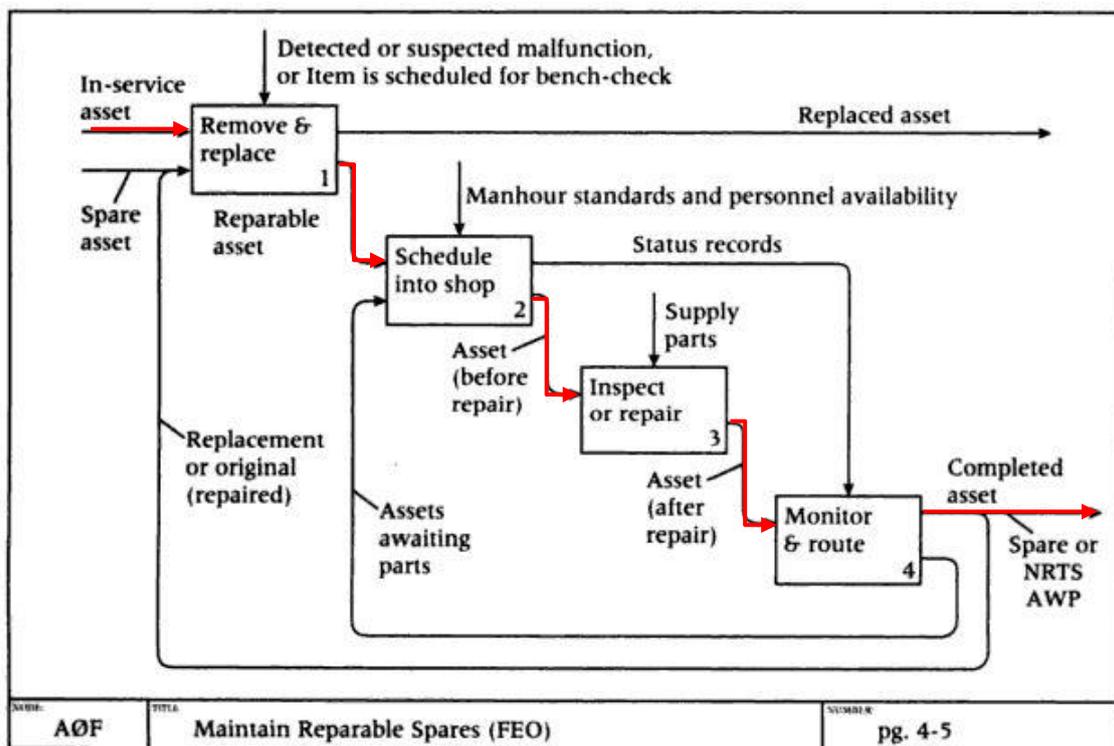


図 4-5 主要経路の例（赤太線）

4. 3 箱と矢印の意味論 (Semantics)

どのようなダイアグラム、もしくはダイアグラムのセットの解釈において、従わなければならない基本的な概念 (fundamental notion) は、次のものである：

明確に示されているものだけが、必然的に暗示されている

(Only that which is explicitly stated is necessarily implied.)

このことは、拘束されたダイアグラムの本質そのものからもたらされている。明示されていない拘束を (勝手に) 仮定してはならず、必要な拘束 (のみ) が明示されるべきである。その為、必然的に以下ようになる：

明確に禁止されていない部分の更なる詳細化は、暗黙的に容認されている

(Any further detailing not explicitly prohibited is implicitly allowed.)

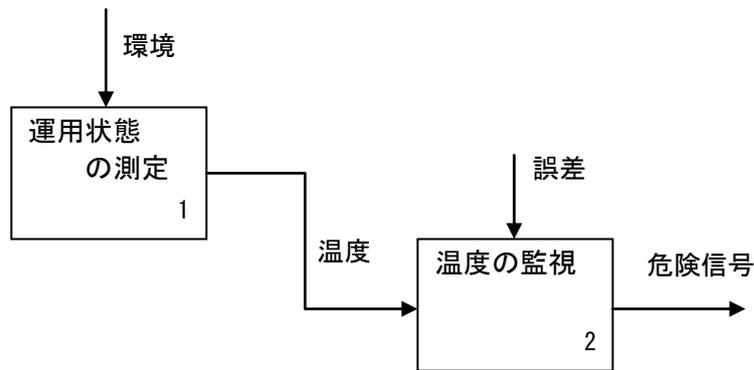


図 4-6 拘束の例

図 4-6 のダイアグラムから、次のような仮定を導き出すことができる：
危険信号を「十分に早く」出すことが可能なような「頻度」で温度が計測され、誤差も「適時」変更されている。

このような仮定を生み出した直観的理解は、以下のような、その後続く詳細化とは矛盾していない：

- a. 温度は定期的な抽出により、計測されている
- b. 現在の誤差は、温度がある一定量よりも増加した時にのみ要求される

- c. 温度変化の割合が誤差以内かどうかを判断する為に変化の割合を検査する箱 2 は、箱 1 で生成された温度の値を連続して保存している
- d. その他

ダイアグラムの図形的表記法は、それ自身が抽象的であるが、それと同時に重要な基本的区別を成している。図形的表記法の抽象的本質が、許容された解釈の幅を損ねるようなことがあるべきではない。

訳注：

以上、非常に分かり辛い。

複雑なシステムに関する詳細な説明を、厳密な拘束が形成されるまで行い続ける事は、非常な時間と労力とを伴う。その為、システムの、厳密な拘束条件が足りない部分の解釈の仕方を、

明確に禁止されていない部分の更なる詳細化は、暗黙的に容認されている

と定義する事で、システム構築中の過渡的な期間においても、システムの一貫性を保証しようとしているのではないかと思われる。

4. 3. 1 拘束による「どれだけ・いつ」の省略 (Constraints Omit How and When)

以下の2つの表現は、いずれも「箱2の活動は、箱1で作成もしくは更新されるdに依存している」ということを示している。



上のいずれの表現も、2つの箱の間の拘束関係を定義している。どちらの表現においても、中間にある矢印は、「箱2による活動は、箱1によって作成された矢印dによって呼び出される物を必要としている」ということを明白に示している。

ダイアグラムが、2つ以上の箱が1つの矢印の内容を求めて争うことになることを強く示唆することが多くある。図4-7における箱と矢印の意味するところは、箱1で作成された物が箱2と箱3とで必要であるということである。それは、矢印の「発生源 (source)」(箱1)の活動が、矢印の「行き先 (destination)」(箱2と箱3)のそれぞれの活動よりも前に行われているということ(「いつ」の省略)、そしてまた、発生源(箱1)の活動が、全ての行き先(箱2と箱3)の活動にとって十分なものであるということ(「どれだけ」の省略)を、それぞれ示しているのである。このように(いつとどれだけという)追加情報が無くとも、箱と矢印のみで、どちらの解釈も可能になっている。

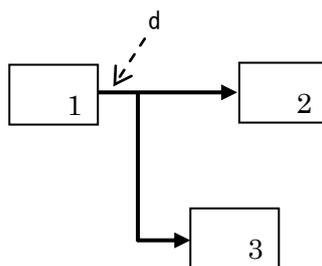


図4-7 1つの矢印の内容を求めて複数の箱が争っている

4. 3. 2 複数の入力・制御・出力 (Multiple Inputs, Controls, and Outputs)

下図に示された箱の基本的な解釈は、「出力{ O1、O2、O3 }の部分集合 (subset) を生成する為には、入力{ I1、I2、I3、C1、C2、C3、C4 }の部分集合が要求されることになる」というものである。これ以上の詳細化が無ければ、次のような仮定をすることが出来ない：

- a. 記述された全ての入力物が揃わなくとも生成可能な出力物が存在する
- b. どの出力も、生成には全ての入力物が必要となる

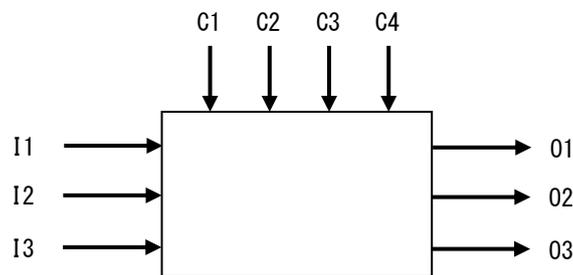


図 4-8 複数 ICOM

前出の箱の部分的詳細化 (これは FEO ダイアグラム等に出てくる、下図) により、O1 の生成に I3、C2、C3、C4 が不要である事が示される。このことから、以下の事が言える：

- a. 更なる詳細化が行われる事で、入力・制御と出力との実際の関係が明確にされる。
- b. そうした詳細化が行われるまでは、箱の「内部」の関係性についての制限的な仮定を行うべきではない
- c. ダイアグラムの読解は、いまだ暗黙である箱の中身よりも、むしろ明確になっている矢印に集中すべきである

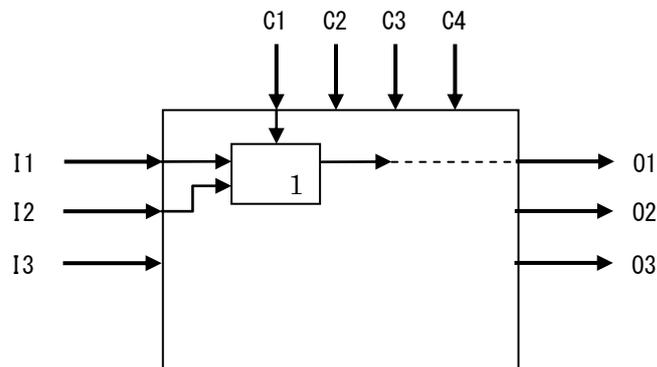


図 4-9 複数 ICOM の詳細を表現した FEO

第 5 章 IDEF キットの構築形式と手続き (IDEF KIT Cycle Forms and Procedures)

5. 1 IDEF チームワーク規律 (IDEF Teamwork Discipline)

どの IDEF モデル (IDEF0、IDEF1、IDEF2、訳注：1982 年当時) の開発も、2 人以上の参加者が必要な動的プロセスである。プロジェクトを通して、モデルの部分的な草案 (draft portion of a model) が著者 (author) によって作成され、他のプロジェクトメンバーに対して批評 (review) の為に配布される。このようなモデルの部分的な草案はキット (Kit) と呼ばれ、ダイアグラム、説明文 (text)、用語集、もしくは著者がモデルの開発に関係すると感じるその他の情報によって構成されている。

IDEF チームワーク規律 (IDEF Teamwork discipline) では、モデルの批評 (review) に参加する全ての人を、批評者 (reviewer) と規定している。中でも、キットについての評論 (critique) を書く立場にある批評者は、コメンター (commenter) と呼ばれている。情報としてのみキットを受け取る批評者は、キットについてのコメントの記述は求められておらず、読者 (reader) と呼ばれている。

また規律は、キットについてのコメントを期待されている人全員がコメントを書き、そしてキットの著者にそれを提出するように定めている。著者は、それぞれのコメントに対して同じ方法で対応する。この作業は、ある特定のモデルに関連する全てのキットを巻き込みつつ、そのモデルが完成し、公布が推奨される (recommended for publication) まで、繰り返される。

モデルの更新は、3 ヶ月毎にキットの形式で (最新の変更を行った) モデルを読者へと配布することで行われ、モデルに関する最新の情報維持に役立てられている。

組織化されたチームワークの下で、このプロセスが繰り返されることにより最終的に出来上がる IDEF モデルは、堅実で、かつ十分に検討が行われたことが確実に保証されたものとなる。そして、これこそがこのプロセスの効果なのである (訳注：かなり意識)。完成した IDEF モデルは、より多くのダイアグラムや説明文、用語集が追加されることで、より詳細なものとなり、更に、より多くのコメントが付けられ、より多くの変更が織り込まれている。最終モデルは、与えられた同一の視点と同一の目的の下でモデリングされたシステムという形となって顕れた、著者と批評者との間の合意なのである。

5. 2 IDEF キットサイクル (The IDEF Kit Cycle)

ドキュメントを作成する中で、著者によって記述もしくは収集された素材は、標準キット (Standard Kit) という形でコメントに対して配布される。コメントはこの素材を批評し、コメントを書きこむ。コメントがキットを著者に戻すと、著者はコメントに回答し、素材の批評もしくは素材の拡張に、そのコメントを使用する。そして著者による応答 (reaction) の付けられたキットは、コメントへと戻される。こうした一連の流れは、キットサイクル (Kit Cycle) として知られている。このキットサイクルは、以下のような段階を踏んで実行される：

- ・著者は素材を批評できるようにまとめ、標準キットを作成する。表紙を付け、コメント全員分と自分用のキットのコピーを作成し、配布する。原資料は参考物としてファイルへ保存する。
- ・ある決められた対応時間内に、コメントはキットを読み、配布されたコピーに直接コメントを記述して、著者へと返却する。
- ・著者は、コメントの戻したコピーに直接、コメントへの応答を記述する。コメントに同意する場合には著者は自身の作業用コピーにその旨を記述し、モデルの新バージョンへと反映する。コメントに不同意だった場合にはキットにその旨を記述し、コメントへと差し戻す。
- ・コメントは著者の応答を読み、それに満足するようであればキットをファイルへと保存する (コメントを記述したキットは、常にコメント側に残るようになっている)。コメントが著者の応答に満足できなかった場合には、著者との間の相違点を解決するための会議が開催される。会議を開催できない場合には、問題点のリストを適切な権威者に対して送付し、決定を仰ぐ。

このサイクルは、ドキュメントが全てのプロジェクトメンバーが十分に考慮し尽くしたものになるまで繰り返される。それに加えて、プロセスの全ての履歴も一緒に保存される。

サイクルが終了した時点で、キットは著者とコメントが寄与したものとなり、そして必要であれば、マネージメント活動 (上層部による調停) が必要となる項目のリストも加えられる。

サイクルを通して、プロジェクトの司書は、コピーの作成や配布、ファイルへの保存、

著者とコメントの間での資料の送付といった作業を行う。

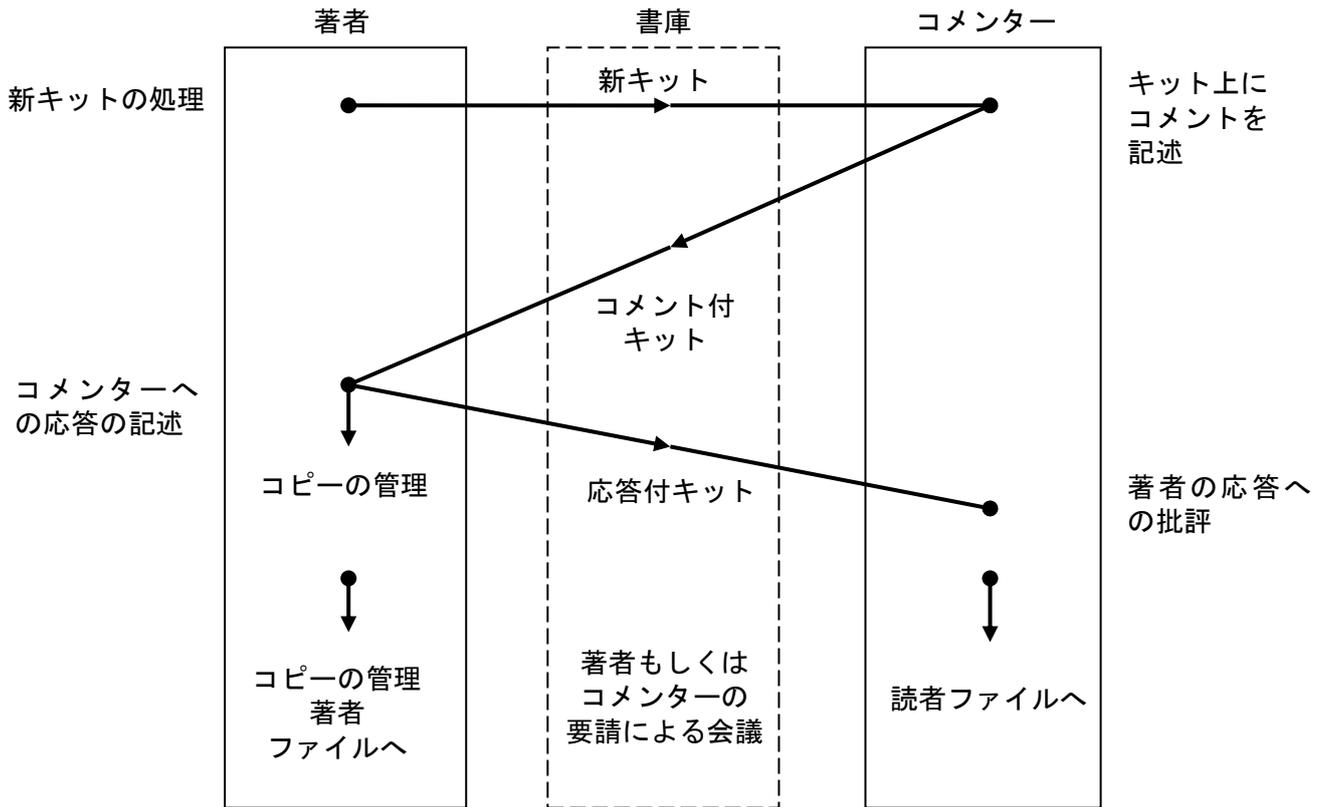


図 5-3 典型的な IDEF キットサイクル

5. 2. 1 各個人の役割 (Personnel Roles)

各個人の役割と (果たすべき) 機能は、以下のようになっている :

名称	役割
著者 (モデラー)	IDEF モデルを準備する
コメンター (専門家)	対象に関する知識を持っており、著者はインタビューにより関連する情報を収集する。また、定式化されたコメントを記述可能なように、IDEF 技術について十分な訓練を受けている※
読者 (専門家)	対象に関する知識を持っており、著者はインタビューにより関連する情報を収集する。参考としてドキュメントの評価を行うが、コメントの記述までは求められていない
司書	ドキュメントファイルの維持管理や、コピーの作成、キットの配布、記録保存などを担当する

※: コメンターと著者との間で取り交わされるコメントは、特権情報 (privileged information) とされている。その為、コメントされたキットは、プログラムに参加している他の人に対して複製・配布されることはない。また司書はコメントされたキットのコピーをファイルとして補完することも無い。

この「役割」は、個人の役職名とは関係なく、また場合によって同一人物が幾つかの役割をこなす事もある。その為、各個人の関わり合いは対象となるキットに依存し、キット毎に独特なものとなっている。

5. 2. 1. 1 著者 (Authors)

著者は、専門家にインタビューを行い、ドキュメントを作成する。この時、著者はドキュメントとしてまとめるべき技術的内容を、自らが持っても持っていなくてもかまわない。著者はただの技術記述者であっても、他の情報源から集めてきた記録物をまとめ直すだけであってもかまわない。著者は、専門家から得た知識の識別、整理、構成といった、著者的 (editorial) な役割を果たす割合が大きい。

5. 2. 1. 2 コメンター (Commenter)

コメンターは、著者が作成した素材を読み、それが技術的に正確であるかを確認する。コメンターは間違いを探し、改善を提案する責任と権限とがある (responsible)。このコメンターの役割は、結果の品質を高める為に重要である。コメンターは、著者が IDEF 技法に従っているかどうか、視点と目的とに忠実であるかどうか、そして間違いや見逃しが無いか判断し、それを著者に伝えなければならない。

5. 2. 2 著者とコメンターに対するガイドライン

5. 2. 2. 1 コメンターのガイドライン

対象の内容、形式、技術といったものは幅広く変化する為、コメント用の定型的な質問セットやルールは存在しないが、品質を向上させるためのガイドラインは存在する。品質についての主要な基準は：ドキュメントは、意図された相手に対して十分に情報を伝達できているか？、目的を果たしているか？、境界で囲んだ背景 (bounded context、切り抜かれた背景？) を与える事で、実際に正しく、正確なものとなっているのか？、といったものである。コメントに対する全体的なガイドラインは、以下のようになっている：

- ・注記は短く、完全で、明確である事。著者が理解可能な範囲で、精密さを捨てて簡略化を行う。これにより、情報交換 (communication) を容易にし、論点のブレ (less clutter、意識) を少なくする

- ・「①」という表記により、コメントを識別する。注記リストから次の番号を調べ、その数字を丸で囲み、対象箇所との間に引用線 (うねり線) を引く。(5.4 章の標準ダイヤグラム形式を参照)

- ・建設的な (前向きな) 批評を行う。解決法を提示し、否定的な不満を述べるだけにならないようにする。

- ・全体的なコメントの収集には時間をかける。全体的なコメントは表紙か、もしくは別紙に記載する。(ただし、詳細な点についてはこれには記述せず、それぞれのページに記載する)。著者とコメンターの間で行われる会議の協議事項 (agenda items) を要約しておく。協議対象を明確にしておく (Make agenda reference specific)。

批評にかかる時間の長さは、コメンターが記述内容にどれだけ詳しいか、これまでの批評の回数、コメンターと著者の経験の量、といった事によって異なる。コメントなしでキットを著者に返却するという事は、コメンターが著者を全面的に支持した、という事である。コメンターは、仕事の品質に関して、著者と責任を共有しているということ認識しておくべきである。

5. 2. 2. 2 著者とコメンターのやり取り (Author/Commenter Interchanges)

キットが返却されると、著者は「✓ (チェック記号)」もしくは「× (バツ印)」をコメントの番号 (①等) に付けて行く。チェック記号は、著者がコメンターに同意し、そのコメントを次のバージョンのキットに織り込むということを意味しており、バツ印は、コメントに対して著者は不同意であるということを意味している。この時、著者はコメントが書かれた部分に、不同意の理由を記述しなければならない。全てのコメントに対応した後、キットをコメンターへ返却し、コメンターはこれを保存する。

コメンターは、著者から返却された応答を読み、不同意な点が残っていれば、著者との会議の開催を要求する。この不同意項目が、会議の協議項目となる。

5. 2. 2. 3 会議のルール (Meeting Rule)

コメントとそれに対する応答がキット上に出そろうまで、コメンターと著者との直接対話は避けた方がいい。

両者の間で会議が必要となった場合、手続きは以下ようになる：

1. 会議 (meeting、会議全体) は時間を制限する事。
2. 会期 (session、項目別の小会議) は、考慮すべき協議事項を明確にする事から開始し、明確にされた項目に集中して、そこから逸れない事。
3. 会期 (session) は、会期での生産性のレベルが落ちたと参加者が同意したら中断し、

労力を無駄にしないようにする。

4. 会期 (session) は、明確な協議事項のあるフォローアップ会期の日程も含めた行動項目のリストに、出席者の合意が至れば終了すべきである。
5. それぞれの会期 (session) において「書記 (scribe)」は議事録を取り (take minutes)、行動や決定、議題 (topic) を記述する。
6. 解決不可能な深刻な乖離が存在した場合、双方の立場からドキュメントを作成し、専門的に取扱われるようにすべきである。

会議 (meeting) の結果、事案毎の解決策 (resolution) か、もしくは適切な経営者判断 (managerial decision) に委ねられるべき事案リストができあがる。解決策は参加者により更に詳しく調査されることで、形が整えられる。

5. 3 IDEF キット

キット (Kit) は技術ドキュメントである。キットにはダイアグラム、説明文、用語集、決定概要 (decision summaries)、背景情報等で構成され、批評やコメント向けにパッケージ化されている。

専用の表紙が付けられることで、素材はキットとして区別される。表紙には著者、日付、プロジェクト名、ドキュメント番号、表題、状況、注記といった項目が用意されている。

IDEF キットには、以下の2つの種類がある：

キット種類	説明
標準キット (Standard Kit)	全てのキットが、コメントの為に配布される。著者が担当するモデル全体の改良を補助する為の「作業用」という位置づけであり、最大で20ページまでに制限されている。
概要キット (Summary Kit)	最新バージョンのモデルを含んでいる。参考のみとして送付され、モデルの一部についてキットサイクルが行われている期間中に、モデル全体の現在情報の把握を補助する為に用いられる。

標準キットは、モデルの一部分のみであり、作業が進捗する度に提出される (submit)。標準キットは、IDEF キットサイクルの中で批評の為に提出され、また本文中で言及されているタイプのキットである。

概要キットは、3ヶ月毎に提出される。このキットには最新バージョンのモデルが含まれている。この概要キットは、受け取ったとしてもコメントをすることは出来ず、ただファイルに保存するだけである。概要キットについては、「ICAM 書庫 ユーザーガイド (ICAM Library User's Guide、別途存在するマニュアル?)」を参照のこと。

5. 3. 1 標準キットの表紙を埋める (Completing a Cover Sheet for s Standard Kit)

キットを提出 (submit) する度に表紙を作成する (再利用することはない)。表紙上の以下のような欄を埋めて行く (図 5-2)

(1) モデル/ドキュメントの記述 (MODEL/DOCUMENT DESCRIPTION)

表題 : キットを上手く説明するようなものを

ライフサイクル段階 : 「現品の (AS IS)」もしくは「将来の (TO BE)」

IDEF 手法 : 0、1、2 (IDEF の番号、1982 年当時は 0 から 2 まで)

システム : システムもしくはサブシステムの記号 (? **onym、読み取れず)

配布、形式 : 標準キット配布以外の物かを明確にする (※1)

※1: 配布の形式については、この報告書の第 11 章に記述

(2) プロジェクト情報

著者 : キットを提出する人の名前 (※2)

日付 : 書庫へ入庫された日付

会社 : キットを公布した会社名

空軍プロジェクト番号 :

作業番号 :

※2: 標準キットがグループ (委員会や共同執筆等) によって提出された場合には、グループの中の責任者を「著者」とする。

(3) キット情報

標準キット欄をチェックする。既存の標準キットの改訂版である場合には、
司書によって採番されたドキュメント番号を指示する。

(4) 評価サイクル (REVIEW CYCLE)

コメンターと著者によって批評が行われた後に署名と日付を入れる

(5) ノード見出し／内容 (NODE INDEX/CONTENTS)

ドキュメントの、ノード番号、題名、C 番号 (表紙を含む)

内容シート (CONTENT SHEET) : 図 5-4 では常にページ 2 となっている

(6) コメント／特記 (COMMENT/SPECIAL INSTRUCTIONS)

批評者に対する、その他情報。司書に対する、このドキュメントの取扱に関する
特別な指示にも使用する事が可能である。司書もまた、この欄をキットの批評者に
対する特別な指示に使用する事が可能である。。

5. 3. 2 標準キットの準備の仕方

見逃しを防ぐ為、キットの批評は、あたかも利用可能な情報であるかのように行う。誤
字も漏らさずチェックする。説明点を加えるが、キット上で短い注記となるように心掛け
なければならない。キットに出てくる用語を定義する用語集は、常に支援素材として付け
加えるべきである。

役立つ資材を収集し、これらをコメンターの利益となるように付加する。間違っても、
ダイアグラムで伝えるべき情報を、こうした補助資材によって伝える事が無いようにする。
読者がコンセプトを理解するにあたって重要となる詳細を示す為に用いる情報伝達手段と
しては、可能な限りダイアグラムを使用する。全ての資材を、表紙とノード見出し／内容
シートへとまとめたら、書庫へ公布する。

IDEF COVER SHEET FORM		MODEL DOCUMENT DESCRIPTION		PROJECT INFORMATION		KIT INFORMATION		REVIEW CYCLE	
TITLE Entity Class Def'n.		AUTOR D. Stone		DATE		STANDARD KIT		REVIEWER	
LIFE CYCLE STEP "AS IS"		COMPANY SofTech		TASK NO. II		SUMMARY KIT		DATE	
DISTRIBUTION TYPE		AF PROJECT NO. 112		TASK NO. II		SUPERSEDED OR REVISED		AUTHOR	
DISTRIBUTION TYPE		AF PROJECT NO. 112		TASK NO. II		DOCUMENT NUMBER		DATE	
COPY FOR		REVIEWERS		REVIEWERS		LOC		KIT CYCLE DATES	
NAME		NAME		NAME		FILE		RECEIVED BY LIBRARY	
CLASS-ART		CLASS-ART		CLASS-ART		AUTHOR		DATE TO REVIEWER	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		COMMENTS FOR BACK TO LIBRARY	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		COMMENTS TO AUTHOR	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		CUSTOMER RESPONSE TIME	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		BACK TO LIBRARY	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		CUSTOMER RESPONSE TO COMMENT	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		KIT CYCLE COMPLETE	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		COPYING INSTRUCTIONS	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		copies of _____ total	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		COMMENTS/SPECIAL INSTRUCTIONS	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		Returns Kit through ICAM Program Library	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		C-NUMBER DS 1094	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PAGE 1	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		DOCUMENT NUMBER	
PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		PROJECT NO. 112		7 2 3 4 5 6 7 8 9 10 11 12	

图 5-2 IDEF 表紙

5. 4 標準ダイアグラムフォーム (Standard Diagram Form)

ダイアグラムフォーム (Diagram Form) は最小限の構造と拘束とで構成されている。この書式は、構造化された分析の規律 (discipline) にとって重要な、以下のような機能のみに対応している。

- 背景の設定 (Establishment of context)
- ページ毎の相互参照
- 各シートについての注記

ダイアグラムフォームは、ファイルやコピーの利便性の良い、1 種類の標準サイズとなっている。書式は以下のような、3つの範囲に分割できる：

- 作業情報 (上段)
- 伝達欄 (Message Field、中断)
- 識別欄 (下段)

この書式は、最終的な「発行認可 (approved for publication)」バージョンになると、上段の作業情報の欄は切り離されるような作りになっている。ダイアグラムフォームは、記述された全ての物に使用される。

5. 4. 1 作業情報

「著者／日付／プロジェクト名」欄：

ダイアグラムを最初に描いた人と、最初に描かれた日時、その時のプロジェクト名を記述する。改訂した日付を追加する際には、元々の日付の下に記入する。表紙が再交付 (re-released) されても、変更が無ければ改訂日付は付加されない。

「注記」欄：

ダイアグラム欄に書かれた注記番号のチェックに用いられる。コメントを書いたら、その分だけ×印で消し込む。この欄の番号を消し込むことで、コメントの数を素早く把握すると共に、個々のコメントに独自の番号を割り当てる事が容易になる。

ST 2.2	USED AT	AUTHOR. PROJECT.	DATE: REV:	WORKING DRAFT RECOMMENDED PUBLICATION	HEADER	DATE	CONTEXT
	NOTES 1 2 3 4 5 6 7 8 9 10						
MODE	TITLE					NUMBER	

図 5-5 標準ダイアグラムフォーム

「状態」欄：

状態を分類する事で、認可の位置づけを知ることが出来る。以下のような状態がある：

状態	説明
作業中 (WORKING)	新しくダイアグラムが作成された時はもちろん、ダイアグラムが以前の状態から大幅に変更が加えられた状態
下書き (DRAFT)	ダイアグラムは以前の状態から小規模な変更が行われており、また、それに対して読者から同意をほぼ得られている。下書きダイアグラムはタスクリーダーによって提案されているものの、技術委員会もしくは技術連合会 (coalition) の評価会議では認可されていない状態。
推奨 (RECOMMENDED)	ダイアグラムとそれを補助する説明文 (text) の両方が技術委員会もしくは連合会の会議において評価ならびに認可され、これ以上変更される可能性が低い状態。
公布 (PUBLICATION)	このページは、このままの状態で最終的な印刷と公布とが行われる。

「読者／日時」欄：

これを読んだコメントのイニシャルと、読んだ日時を記入する。

「背景」欄：

このダイアグラムを解釈する為の背景を指示する。背景欄には、現在のダイアグラムの一つ上の階層のダイアグラムを記述する。背景となるダイアグラム内の現在のダイアグラムは強調する為に箱のまま、それ以外は楕円で描き、また矢印は省略する。そして背景となる1つ上の階層のダイアグラムのコード番号を左下に記述する (図 5-6)。背景ダイアグラム (A-0) の場合は、背景欄には「無し (NONE)」と、最上層ダイアグラム (A0) の場合には「最上層 (TOP)」とのみ記述する。

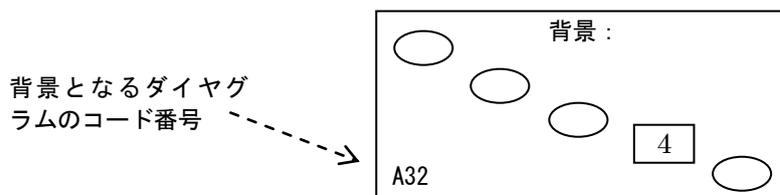


図 5-6 背景欄の書き方

「参照元 (Used At)」欄：

背景としてではなく、このページを参照しているダイアグラムのリストを記述する。

5. 4. 2 「メッセージ」欄

メッセージ欄には、伝えるべき主要なメッセージを記述する。通常はダイアグラムの作成に使用されているが、それ以外の用語集、チェックリスト、注記、スケッチ、といった目的にも使用可能である。著者はダイアグラムフォーム以外の別の用紙を使用すべきではない。

5. 4. 3 「題名」欄

ダイアグラムフォームに書かれている対象の名前を記入する。メッセージ欄にダイアグラムが含まれていれば、題名欄に書かれる内容は、親箱に書かれている名前に正確に一致させておくべきである。

5. 4. 4 「番号」欄

このシートを参照する為に必要な全ての数字を記載する。

C 番号：

C 番号は、著者のイニシャルから成る 2~3 の文字と、著者毎に連続的に割り当てられた番号によって構成されている。C 番号は、番号欄の左下に配置され、主にこのシートを参照する際に使用される。1 人の著者によって使用される全てのダイアグラムフォームに、固有の C 番号が付与されることになる。モデルが公布される時に、この C 番号は標準の連続ページ番号（例えばページ 17）へと置き換えられる。

ページ番号：

キットとしてのページ番号は、番号欄の右下に、司書によって付けられる。この番号は、ドキュメント番号と、そのドキュメント中のシートの識別番号によって構成されている。

5. 5 ファイルの保持 (Keeping Files)

プロジェクトに参加している全ての人は、受け取ったドキュメントをファイルに保存しなければならない。司書は、プロジェクトにおける一連の作業の中で、認可された全てのキットの原本ファイルと参照用ファイルを維持管理する。書庫ファイルについての完全な解説については、この報告書の第 11 章にある「ICAM プログラム書庫維持管理手続(ICAM Program Library Maintenance Procedures)」を参考の事。

ファイルへの保存方法は人に依って異なるが、保存されたファイルは維持管理されることが推奨されている：

- ・ 標準キットファイルは著者とコメンターによって維持管理されるべきである
- ・ 概要キットは、著者とコメンター、そして読者によって維持管理されるべきである。
- ・ 作業ファイルは著者によって維持管理されるべきである

5. 5. 1 標準キットファイル

このファイルには、作成された (issued)、もしくは受け取った標準キットが含まれている。キットファイルの全ての記録と、キットの内容にアクセスしやすくする情報も維持管理しておくべきである。

5. 5. 2 概要キットファイル

このファイルには、作成された、もしくは受け取った概要キットが含まれている。概要キットの全ての記録が維持管理されるべきである。

5. 5. 3 作業ファイル (Working File)

このファイルには、キットとしてはまだ認可されていない全てのドキュメントが含まれている。進行中の作業と注記とは、このファイル内にて維持管理されるべきである。

5. 6 IDEF モデルウォークスルー手順 (IDEF Model Walk-Through Procedure)

キットサイクルに加えて、ワークスルー手順も開発されている。この手順は、モデル構築の参加者が、一緒になってコメントを行う時に用いられる。

(1) ノード見出しを使用して、モデルを分析可能なようにまとめる。これはモデルの内容一覧であり、批評者に何がどうなっているかについての概要を素早く理解させることができる。

(2) 用語集をまとめる。これにより、批評者がそれぞれでバラバラに持っている言葉を、プレゼンチームが選択した共通の言葉へと置き換える事が可能となる。言葉の意味については、この時点では質問しないようにする。意味を変更すると、ダイアグラム内で多くの変更が必要となってしまう。

(3) 批評用に、各ダイアグラムをそれぞれまとめる。

ダイアグラムワークスルー手順は、通常はステップ毎に実行し、その都度質問を行い、ダイアグラム内に隠れている弱点を識別して行く。ステップは、以下に挙げるように6つの構成になっている。

ダイアグラムの修正は、ステップ毎に提案される。後で修正を行うか、直ぐに修正を行うかはステップ毎に決定する。

ステップ 1: ダイアグラムの読み取り (Scan the Diagram)

このステップで、読者がダイアグラムの中身について大まかな印象を得る事ができる。

一般には、読者は対象となるダイアグラムが1つの箱として表現されている親ダイアグラムの批評を行い、それから著者がその箱をどのように分解していったのかについて調査を行う。

容認の基準 (criteria for acceptance)

(1) ダイアグラムの分解が有意義なものであり、またダイアグラムの親箱の背景内に完全に収まっていること。下層にある全ての機能やデータが、それぞれの箱の下に明確に分類されていること。

(2) 批評者にとって、ダイアグラムで、モデルの目的を基にした適切な視点が反映されていること。

(3) 批評者にとって、親箱の理解を深める為に十分なだけの新しい情報が存在していること。ダイアグラムが複雑になり、読解が困難となるような詳細化が行われていないこと。

問題があまり明確にならない場合には、批評はステップ3まで延期するが、最初に受けた印象を黒板やフリップチャートに書くなどして、問題が解決されるまで残しておかなければならない。

ステップ2： 親を見る (Look at the parent)

読者が現在のダイアグラムの分解状況について理解したら、互換性を確認する為に親ダイアグラムを批評するべきである。

容認の基準：

(1) 批評者が親ダイアグラムを読む際に準備していた全ての要点が、分解の中で網羅されていること。

(2) 対象のダイアグラムの元である親ダイアグラム内の箱を、批評者自身でも分解し、それが対象のダイアグラムと一致するかどうかを確認する (意識)。もしそうならない場合には、抜け落ちた詳細を注記しておく。

このステップでは、親ダイアグラムへと少しの間だけ戻り、分解されたものを見る事で得られた知見を基にして、新しい注記（○に番号）を付加するか、既存の注記を確認して行くことが重要になる（訳注：相当に意識）。

ステップ 3： 親箱と詳細ダイアグラムとの接続

このステップでは、親子の間で矢印のインターフェース接続を確認する。

容認の基準：

- (1) インターフェース矢印に、消失した物や余分な物が無いこと
- (2) インターフェース矢印に、正しい ICOM コードが付けられていること
- (3) 子の矢印のラベルが、それに相対する親の矢印と同じラベルか、もしくは詳細化されたものであること
- (4) 親ダイアグラムの中で矢印の接続が問題ないか調査する。（余分なインターフェースは、親ダイアグラムから変換されるメッセージを誤解させる原因になる（かなり意識））

親箱の 4 辺から出ている矢印を時計回りに調べることで、親矢印の ICOM コードと合致しているかどうか規則だ正しくチェックする事が可能である。

ステップ 4： 内部の矢印パターンの調査（Examine Internal Arrow Pattern）

箱と矢印のパターンによって、作成中のモデルの主要な部分が表されている。

全ての箱がノード番号順に並んでいるか、それぞれの箱の矢印が ICOM 順に並んでいるかを調査する。この作業が終わったら、批評者は自分が慣れた状況の結末へと行き着くかどうかを確認し、そしてこのダイアグラムによって既存の関係性を再現可能かどうかテストする為に、ダイアグラムの流れを追うべきである（かなり意識）。

容認の基準：

(1) ダイアグラムが見た目に散らかっていないこと。矢印の交差や曲りの数が、最少であること。

(2) 箱は、詳細化の点において、バランス良く配置されているべきである。またそれぞれの箱に含まれる詳細の量は等しくあるべきである。ただし、明快さを維持する為には、その限りでもない。

(3) ダイアグラムが、対象に対する批評者の経験と知識に一致していること。フィードバックとエラー状態も、批評者の期待通りになっていること。

ステップ 5： 支援ドキュメントの読解 (Read the Supportive Documentation)

このステップでは、著者が説明文、用語集、そして FEO によって強調されている点を調査する。

容認の基準：

(1) 説明文と、ダイアグラムそのものの調査から得られた解釈とが一致していること

(2) 通常の流れ (normal-paths)、フィードバック、エラー処理等の、説明文によって提起されている特性 (feature) が、ダイアグラムや FEO (説明専用) ダイアグラムの中に、見受けられること

(3) ステップ 1 から 4 の間では判らなかつた重要なダイアグラムの特性が、説明文、用語集、FEO に見受けられること

(4) ダイアグラムへの参照が十分に詳細化され、説明文や用語集、そしてダイアグラムの特定の部分に対する FEO へと接続可能であること。

ステップ 6： ダイアグラムの状態を設定する (Set the Status of the Diagram)

- (1) このままで良い
- (2) 変更が必要
- (3) 下書き (draft)： 変更が余りにも多い、書き直しが必要、再度批評し直す必要がある
- (4) 不認可： 完全な再分析が必要である

第6章 著者の為の、IDEF0 ダイアグラム作成ガイド

(Author's Guide to Creating IDEF0 Diagrams)

どのような IDEF0 ダイアグラムを作成する時でも、以下の要求を満たしていなければならない：

- a. ダイアグラムの目的と視点は、モデル全体で定められた目的と視点とに合っていないなければならない
- b. ダイアグラムの境界矢印は、親ダイアグラムの境界矢印と一致していなければならない
- c. ダイアグラムの内容は、親箱そのものでなければならない

6. 1 編集の基本ステップ (Basic Steps of Authoring)

ステップ毎の編集規則を守る事で、使いやすく、かつモデルと矛盾しないダイアグラムを作成する事が可能になる。従うべき規則は、以下の通りである：

- a. 機能箱の題名よりも、より精密に主体物の境界を引く。機能によって実行・処理されるデータ（物体もしくは情報）のリストを作りつつ、これを行う。
- b. 境界を引いた主体物のセットを研究し、全体機能に対する従属機能が存在し得るものになるように形成する。
- c. こうして作成した従属機能間で、自然な接続パターンを探す
- d. 従属機能の分割と接続 (connect) を行って他の箱を作成する。
- e. 配置と明確さに注意を払いつつ、ダイアグラムの最終版を作図する

6. 1. 1 背景、視点、目的の選択 (Selectong a Context, Viewpoint, and Purpose)

どのようなモデルから始めるとしても、始める前にモデルの方向性を決定しておくことが重要である。この方向性には、背景 (context)、視点 (viewpoint) そして目的 (purpose) が含まれる。

背景により、全体の一部としてのモデルの主体を定める。外部とのインターフェースを記述する事により、外環境との境界を作成することになる。

視点は、背景の中で何を、どの見地 (slant) から見る事が可能であるかを決定する。視点 (がどういうものであるのかということ) から、著者の立ち位置が1人の観察者としてのものであるのか、それとも1人の参加者が、(自身の) 利益の為にシステムに参加しているものであるのかが、明らかになる。参加者としての立ち位置により (管理者、技術者、顧客、等)、対象の異なった面に比重を置いた、異なった視点が適用されることになる。

モデル毎に、ただ1つの視点のみ：

1つのモデルは、ただ1つの目的しか反映できず、またただ1つの視点でしか運用できない。

目的は、モデルの意図、もしくはモデルによって提供される情報伝達 (communication) の目標 (goal) を設定する。目的は、何故モデルが作られたのかという理由を具体化したものである (機能的仕様、適用設計、顧客運用、等)

こうしたコンセプトにより、モデルの製作が誘導され、かつ拘束される。編集作業により (as authoring proceeds) コンセプトが改善されて行く中で、モデルの立ち位置がはっきりと変わらず、ゆがめられない限り、コンセプトは1つのモデルの中で首尾一貫したものとしなければならない (意識)。

目的を透明に：

詳細化の作業において強く意識しておかなければ、本来の目的から逸れてしまうことになる。

全ての分析の開始時に、背景の範囲を明確にしなければならない。最上層の箱を作成する前に、何に焦点を当てるかを決定しなければならない。そして、この慎重に選択した

開始時の領域から、外れないように気を付けなければならない。

全てのステップにおいて、開始時の目的に合っているかをチェックすべきである。これから異なる物には注記しておき、後で適切な視点でモデリングできるようにしておく。

明快さ (clarity) は、詳細化を厳格 (rigor) にすることで得られる。どれくらいの距離があるか、いつ止まるべきか、いつギアを変えるべきか、どれくらい部品を揃えるか、ということを知るには、モデルを何のために作るのかという目的が常に必要となるのである。

6. 1. 2 背景ダイアグラムの作成 (Creating the Context Diagram)

モデルは、A-0 ダイアグラムを作成する事から始まる。これは、システムの全ての範囲を含有した機能の名前を持った 1 つだけの箱を描写したものである。箱から出入りしている矢印は、システムとその環境とのデータインターフェースを表現している。この箱 1 つのダイアグラムにより、モデル全体と背景とを区切り、これから行う分解作業の基本部分を形作るのである。

A-0 階層で 1 つだけの箱のダイアグラムを書く前に、A0 ダイアグラムをいきなり描く方が楽な著者も居るだろうが、A-0 と A0 ダイアグラムを何度も行き来し、分解作業を上手く開始できるようにすることが必要となる。

A-0 ダイアグラムの詳細化階層が余りにも低く (詳細に) 始まった場合には、A-0 箱を新しい階層の A0 ダイアグラムの基礎とし、その一つ上の階層に A-0 ダイアグラムを新しく作成する。そしてこのプロセスを、A-0 ダイアグラムがシステムの全ての面を含む、十分な範囲を持つまで繰り返す。(時には、選択した視点を明らかに超えた高い階層になってしまうこともある。もしそうなった場合には、A-1 複数箱背景ダイアグラム (multi-box context diagram) を作成し、A0 ダイアグラムを最初の意図通りに保持する)

6. 1. 3 最上層ダイアグラムの作成 (Creating the Top-most Diagram)

全てのシステム機能は、A-0 ダイアグラム上のたった1つの箱の内側に存在することになる。このダイアグラムにより、システムの背景が明確に分離される。A0 ダイアグラムはA-0 ダイアグラムを3つから6つの主要な従属機能へと分解したものである。

事実上の最上層モデルはA0 ダイアグラムである。そこには、最初の、最も重要な、モデルの視点が表現されている。選択された対象を完全に記述している事から、A0 ダイアグラムの用語と構造もまた、それに続く階層を明確に分離しているといえる。下層は、A0 ダイアグラム内の機能(箱)をそれぞれ描写していったものである。モデルの目的が達成されたならば、この繰り返された詳細化の繋がりは、ステップ毎に注意深く追跡して行けるものになっている筈である。最上層の始めの部分は、編集(autoring)のなかでも難しい所である。抽象化のレベルを維持するよう著者が強制されているのなら、モデルの深さを均一にし、詳細化をより下層へと追いやることになる。

6. 1. 4 それ以降に続くダイアグラムの作成 (Creating Subsequent Diagram)

ダイアグラム構造を形成するには、A0 ダイアグラム上の各箱を分解し、それを主要な部分としてゆく。親箱と同じテーマ(topic)を対象としつつも、新しいダイアグラムの形成は、より詳細に行われる。

それぞれの箱を3個から6個の箱へと分解する為には、追加の事実が必要となる。まずは、分解する箱の持つ全てのデータ項目と活動をリストにまとめながら、ダイアグラムを描いてみる。このリストは親箱のテーマ全体を含むようにし、分解の際に書洩らしが無いように注意する。このリストを基にして箱を描き、箱の間にインターフェース矢印を描いて行く。

より明確なダイアグラムを描くには、満足のいくまで、何度も修正や書き直しを行う必要がある。分割(1つの箱を2つ以上の部分へと分解する)と集合(2つ以上の部分を1つの箱にまとめる)を満足いくまで繰り返す。

明確にする必要がある部分を調査する為に、より詳細な階層のダイアグラムを部分的に作成する。1度に1つだけのダイアグラムを作るよりも、3つから4つのダイアグラムをセットで作成した方がいい。

6. 1. 5 支援材料の作成 (Creating Supporting Material)

説明文 (text) と用語集 (glossary)、そして恐らくは FEO により、それぞれのダイアグラムのページが最終的に完成する。A-0 ダイアグラムに関する説明文は、ダイアグラムが作成された時に記述され、モデルの方向性を完全なものにする。説明文は、モデルで定義された視点と目的を詳細に説明する事により、背景 (A-0 そのものがこれを説明しているが) を補完している。

A-0 以外のダイアグラム (A0 を含む) の説明文は、A-0 のそれとはかなり異なっている。説明文は短く、簡潔な流れを示しているだけである (tells a story)。ダイアグラムが既に表現している事を繰り返すのではなく、ダイアグラムによる表現だけでは足りない隙間を埋めている。それぞれの階層において、このようにして視点を捉えて行き、それにより目的が促進されている。

用語集は、ダイアグラム中の機能とデータを、著者がどのように定義しているかについて説明している。モデルで使用されている専門用語は会社毎に全く異なった意味で使用されている事がある為、こうした定義が重要になってくる。

FEO (For Exposition Only、説明専用ダイアグラム) は、あるダイアグラムの特定の興味対象や微妙な側面を強調したダイアグラムである。FEO ダイアグラムは IDEF 文法に従っておらず、対象を強調する為に部分的な矢印、注記等のみで構成されている。

6. 1. 6 分解する箱の選択 (Selecting a Box to Decompose)

親ダイアグラムを完全なものにしたら、次は上階層を固めて行くのだが、この時に局所的に詳細化を進めないようにする (意識)。つまり、A0 が出来上がったら、A1、A2、A3 の作成をまず行い、A1 を A11、A111 へと分解するのは、その後に行うべきである。これにより、上階層のダイアグラムでの作業のやり直しを防ぐ。

ただし、階層の深さを常に等しく維持しつつ作業を行うことは、厳密な決まり事ではない。階層の深さは、深くすることでより意味を成すかどうかという事に常に依存している。

A111のような下層ダイアグラムであっても、アイデアが新鮮な内にスケッチしてしまうべきである。重要な事は、そうした局所的なスケッチの深掘り（foray）を、「水平」に、同じ階層になるまで繰り返すという事である。その代り、上階層（A1、A2、A3等）間で矛盾が発生し、やり直しが発生する事を考慮し、下階層でも作業のやり直しをする準備をしておかなければならない。

どの箱を分解するかを決定する際には、以下の2つのガイドラインが便利である：

- (1) 「難しい部分」から分解を開始する。難しい部分とは、最も疎遠か、最も不明瞭な部分のことである。
- (2) その箱を分解する事で、他の箱に最も多くの情報が付与される箱を選択する。

6. 1. 7 編集作業 (Author Activities)

6. 1. 7. 1 データ収集フェーズ (Data Gathering Phase)

背景を読む (Read Background)

(訳注：context と background のニュアンスがわからない…)

著者 (author) は、資料情報を読むことで、対象物について情報を収集する。

インタビュー (Interview)

著者は、対象物の専門家にインタビューを行う。このインタビューは、複数人による立案作業や、批評会議の為に使用されるものではない。

思考 (Think)

実際にダイアグラムを作成する前に、資料とインタビューから得られた情報を十分に整理しておく。

箱の選択 (Pick Box)

得られた情報を基にした詳細化に適した箱を選択する。

6. 1. 7. 2 構築フェーズ (Structuring Phase)

作図 (Draw)

この作画作業は、ダイアグラムの生成という、実際の創造的プロセスを中心としている。作画作業は箱と矢印の描写だけに止まらず、それに必要となる、ランダムデータ要素の一覧表作成や、スケッチの作成と言った作業も含まれている。(訳注: random data の意味が良く判らない)

再作図 (Redraw)

再作図作業は、ダイアグラムを作成する為の熟慮作業 (消化ステージ、digestive stage) であり、文章 (verbal text) 作成における編集 (editing) と書き直し (rework) 作業にあたる。ここでは、新たな要素の追加を行うのではなく、より明確にするための図形的な編集と再配置を行う。

原図の修正 (Fix Master)

改善されたものを原図 (master drawing) へと織り込む。原図は別に保存し、コピー上で検討や変更を行っている事から、原図への織り込み作業は機械的なものとなる。

6. 1. 7. 3 表現フェーズ (Presentation Phase)

説明文の記述と編集 (Write and Edit Text)

ダイアグラムに付属する説明文 (text) を、より正確なものにする。編集作業においては主に、不要な詳細と余剰物の削除を行う。

組み合わせ (Assemble)

ダイアグラム、ノードツリー (node tree)、用語集 (glossary)、説明文 (text) といった各種素材を (冊子の形へと) 組み合わせて行く。完成した表紙も含まれる。

6. 1. 7. 4 対話フェーズ (Interaction Phase)

コメントへの返答 (React)

著者 (author) はコメントに返答を行う。コメントを読み、それに対する注釈をつける。

会話 (Talk)

著者とコメント者とが直接に会い、コメントに対する著者の返答について会話を行う。

グループ会議 (Group Meetings)

グループ会議を開き、進捗状況の批評や、次の段階の為のブレインストーミングを行う。グループ会議を行う事により、議論の対象がより明確になる。

6. 2 IDEF0 ダイアグラムの作図

ダイアグラムの作図作業は、モデリングプロセスの中でも最も主体的で、創造的な作業である。別々の複数の著者 (author) によって、幅広い種類のダイアグラムが作成される。全ての著者に対して等しく効果的なやり方というものはない。以下に挙げるやり方は IDEF0 ダイアグラムを初めて作図する著者を補助する為に作られたもので、実績のある順序で並べられている。

- a) 関係性をまとめる。ただしデータリストの構造化はまだ行わない。親箱の背景にある対象項目 (item) をリスト化し、それをまず頭に入れておく。可能であれば対象項目を仕分けし、それらの類似性を判り易くしておく。
- b) リストにまとめたデータに作用している機能に名前をつけ、名前を四角で囲んでおく
- c) 適切な矢印を描く。箱を別々に作画している時は、矢印は片側だけのままとし (stub、半券や切株)、箱の意味が判り易くなるようにしておく。ダイアグラムが何を示しているのが明確になった時に、矢印を接続してダイアグラムを完成させる。
- d) ダイアグラムの構成を下描きし、箱と矢印の配置がより明確になるようにしてゆく。構造が詳細過ぎるものになった時には矢印を束ね、本質的な要素のみを記述するようにし、必要に応じてダイアグラムを変更する。
- e) 必要ならば、説明文、用語集、FEO (説明専用の) ダイアグラムを作成し、重要箇所を強調する。必要に応じて、親ダイアグラムの変更を提案する。

6. 2. 1 機能箱の作成 (Generating Function Box)

機能箱は、親箱の主要な従属機能を用いて作成される。従属機能の名前を記述したら、それを四角で囲み、それを起点として実際のダイアグラムを作成してゆく。この段階では箱の数は重要でない。箱は集束と分割によって変更することができる。

集束 (Clustering)

集束は、2つ以上の箱を1つの箱へとまとめる。集束の目的は、関係した機能を1つにまとめ、機能をより一般化させることである。集束を行う事で早熟な詳細化を防止し、この階層においては不明瞭なメッセージを持ち込まないようにする。

分割 (Splitting)

分割は、1つの箱を2つ以上の箱へと分割する。集束と逆の作業である。分割の目的は、分解された主題 (subject) を十分に理解可能になるまで、詳細化を行うことである。

こうして出来上がった機能箱のセットを批評する。選択された要因 (factor) の間のバランスが良くなっているかを見る。より明確になる名前が他にあるかどうかを見る。専門用語や略語は、意図した読者 (audience) との情報交換を促進する為に必要である時だけに、そして詳細なダイアグラム階層である時だけに使用する。専門用語や略語は、最上階層 (A-0 と A0) では使用しない。専門用語は用語集の中で定義するように注意する。

全ての場合において、機能箱のラベルは動詞句 (verb phrase) とする。動詞と名詞と判断が付きかねない場合には、「(v)」という注記を入れ、動詞であるということを明記する。

箱 (Boxes)

(1) 殆どの場合、箱は左上から右下へ向かう対角線上に並べられる。著者の意図が明確となる配置であったとしても、箱が上下もしくは左右に並べられると、矢印が混み合い、構造化された分析形式 (structured analysis style) に成りにくくなる。

(2) 左上に配置された箱は、ここよりも右下にある箱を、それぞれに接続した制御矢印を通して「支配」する。この標準的な形式により、著者の意図を読者が理解しやすくなる。

(3) それぞれの箱の右下に番号を入れる。ダイアグラムの左から右、上から下の順番で番号をつける。この番号が、それぞれの箱のノード番号となる。箱の完全なノード番号の上の桁は、ダイアグラムのノード番号であり、下の桁は箱の番号である。図 6-1 の箱がダイアグラム A4 内に有った場合、この箱の完全なノード番号は A42 となる。

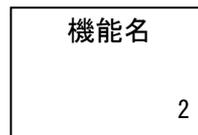


図 6-1 箱番号

(4) 作業中、もしくは下描きのコピーの場合、著者の C 番号を、分解中の箱の右下隅の下に記述しておく。

(5) 1つのダイアグラム内の箱の数は、最大で 6 個までとする。

6. 2. 2 インターフェース矢印の作成 (Creating Interface Arrows)

データインターフェース矢印を、それぞれの箱へと描いて行く。矢印の端を箱のどこへと接続するかにより、それが出力、入力、制御のどれを意味するかを表すようになっている。

入力データが機能によって変換され、出力になっているということを思い出してほしい。1つの矢印に入力データと制御データの両方が含まれている場合か、もしくはどちらか不明瞭な場合には、制御としておく。データのどの部分かわからない、それが必要かどうかもわからないといった場合には、その矢印は描かずにおく。

複数の出力矢印は、機能が実行し得る活動の全ての結果を表している。出力矢印の文法だけでは、どういう環境で、どういう出力矢印のパターンとなるかまでは表現する事ができない。もしもこれら複数の出力矢印の順番が重要であれば、別にパターンを **FEO** を作画することで説明を行う。しかし順番について心配する必要は無く、全ての重要な場合が、ダイアグラムによって許容されるようにしておけば良いだけである。(訳注：出力矢印が複数ある場合、状況によって色々な出力パターンを持つことになる。それを説明している)

可能な限り、関係する矢印は束ねてグループ化しておく。矢印を作成する際の最も一般的な間違いは、矢印の構造やラベルを詳細化させ過ぎることである。矢印の詳細化階層と箱の詳細化階層は、必ず相対させなければならない。上階層では、箱のラベルと矢印のラベルは概略的なものでなければならない。

最後に、データリストと全ての矢印とを比較し、全てのデータ項目が正しく表示されているかを確認する。抜けた要素があるということは、詳細化を行う階層が間違っているか、もしくは矢印を作成する際に見落とししてしまったかのどちらかである。

制御と拘束を考慮し、流れは考えない (Think control and constraint, not flow)

矢印構造の配置の為の基本的なルールは、「順序ではなく、拘束」である。これは、順序に関わらず正しくなるという関係性を、ダイアグラム構造で表現すべきであるという事である。

確かに何事も、ある段階からある段階へ進むことで何らかの期待された結果へと至っているのだが、ある結果を生み出す特定の段階の順番よりも、正しい状態となるべき時に満たされるべき拘束か、もしくは不変の特性を表現すべきである。

何故なら、全ての箱は並行に活動を行っているからである。その為、順序は意味を持たなくなるのである。

常に、順序よりも拘束に重点を置かなければならない。可能な限り、どの段階を最初に実行するかということを見捨て、ダイアグラムを作成すべきである。たった1つの順序のみに拘束されてしまうダイアグラムよりかは、その方がマシなのである。

簡単だからと、まず初めに、ある決まった順序でサブモジュール化された行動のセットを考え、取り敢えず何か紙に描いてみようとする人が多い。これは、取り敢えず前へ進めるには上手いやり方ではあるが、拘束構造へと変換する際に、必ず作り直さなければならなくなってしまう。

ラベル付けは慎重に (Label carefully)

(その階層では) 不要な詳細を下層へと移しこむことで、(残った方の) 意味のある詳細が、より明瞭なものになる。

過剰な量の情報や矢印で、ダイアグラムを読み難くしてはならない。

1つの階層に長い時間をかけてはならない。一度に全てのものを書き表そうとすれば、不十分なものや、誤解を招くものになってしまう。

何事も、段階的に行わなければならないのである。

1つのダイアグラムに詰め込み過ぎると、ダイアグラムが硬直し、(意味合いが) 弱くなってしまう。ダイアグラムの構造こそが、強さなのである。余分な詳細を従属機能に引き渡すことで、強いダイアグラムを実現する事が可能である。

詳細が全体的に行き渡るまで、上階層のダイアグラムと従属機能との間を行き来するのである。

不明な矢印は除外する (Leave out questionable arrows)

その矢印を引くべきかどうか、判断に迷う事が多い。

迷った矢印に対する最も簡単な対処は、「疑わしい矢印は除外する」というものである。矢印が主要な構造に必要な不可欠でない場合や不明である場合、その矢印は恐らく間違っているのである。

不明な矢印を削除することが誤りであったとしても、受けるのは一時的な損害のみである。矢印の必要性は、従属機能を考慮する際に明確なものとなり、そして ICOM 規則によって本来の階層へと押し戻されるのである。その際、矢印に対する疑問も解消されることになる。

6. 2. 3 努力の水準 (? Level of Effort)

ダイアグラムを作成する際の第一の目標は、文法を破ることなく定義したメッセージを表現した、明瞭なダイアグラムとなるようにして行くことである。ダイアグラムが出来上がったら、読解と批評を行う人向けの重要なガイドラインを作成することにより、最初が出来上がったダイアグラムを更に改良することが可能である。殆どのダイアグラムは、何らかの改良を施した第2バージョンへと進化させる事が可能である。逆に、一発で完成品が出来上がることは稀である。

技量が上がると、1回目のダイアグラムの出来も良くなり、**IDEF0** という仕組みにも、より満足できるようになる。ダイアグラムの描き直し作業は、必須なプロセスである。中心となる重要なアイデアは、批評を繰り返すことになり、紙上でより進化したものになる。当たり前の作業 (**ordinary advances**) をひたすらに繰り返して行くことで、全ての重要な面が確実に把握されて行くのである。

IDEF0 は、ダイアグラムを作成する為のただの技法 (**exercise**) ではなく、思考構成手法論 (**thought-forming methodology**) なのである。思考を紙の上に描き表し、表記法 (**notation**) と作業規則 (**discipline work**) に従う事で、満足の行く結果へと自然に導かれて行くのである。「完全な」答えが提供されることを期待するよりも、むしろ、良い疑問を持つ能力 (**ability to ask good questions**) に依存すべきである。

6. 3 IDEF0 ダイアグラムの再作画 (Redrawing an IDEF0 Diagram)

6. 3. 1 箱の改良 (Modifying of Boxes)

出来上がった1つのダイアグラムは、詳細の程度がほぼ同じである3~6個の機能箱で構成されている。集束と分割を繰り返し、より判り易く、機能箱間の関係がより単純な境界となるように調整して行く。

殆どの場合、集束と分割は同時に行われる。箱の分割でできた部品を、新しい箱へと集束し直してゆくことで、意図されたメッセージをより綿密に伝わるようにする。主題の範囲は分割・集束の前後で変わらないが、部品はより判り易い様にグループ分けされる。

分割と言換え (Split and rephrase)

重要な事は、1つのダイアグラム内の全ての箱は、ある一貫した匂い (**consistent flavor**) を持っているという事である。何かを変えたからと言って、ある従属機能を場に合わないものにしてしまうことはないのである。分割と言換え (**rephrase**) は、バランスを取り戻す為の作業なのである。

検討中のダイアグラムでは、時には、他の箱の流れの中に入らない箱も出てくることがある。

良くある問題が、別の面が変更されたり、純化されてしまうということである。最初の段階では良いアイデアであったものが、後々に間違った傾向 (**slant**) や匂い (**flavor**) になってしまうことがある。

邪魔な箱を分割して2つ以上の箱にしてしまっても、少なくとも1つの箱は、分割前の箱が持っていたアイデアの本質を残しているのである。

分割されることで、箱の文字が変化することに期待すべきである。分割により、新しいアイデアがより明確になり、関係する箱とより密接に調和するようになる。

集束と置換え (Cluster and replace)

(全体的に、おもっくそに意識、というか英文の意味不明)

成熟 (**solid**、固まったということだが、後の **premature** と対比させたいので、ここでは成熟とする) した抽象の方が、未熟な詳細よりも明確で、強力である。その理由から、関連した幾つかの箱を集束して、それらを包含した1つの箱へと置き換えるのである。

幾つかの箱を1つの箱へと集束することで、より概略的な視点へとまとめ、そして詳細化作業を一つ下の階層へと先送りすることにより、抽象化を更に上手く行う事が可能な場合が良くある。集束するものを線を引いて囲み、それを1つの箱へと置換え、新しい箱に適した名前に付け替える。階層を増やすことは、複雑さを増すことではない。構造がより明確になることを考えれば、より良い表現であると言える

こうした現象は、集束と分割の組み合わせの中で度々発生するものであるが、それは機能を説明する為の最も強力な手法の1つである。

6. 3. 2 矢印を束ねる (Bundling Arrows)

1つのダイアグラム上にある矢印と箱は、同一の抽象化レベルでなければならない。

抽象化レベルを揃えるには、以下の2つの方法がある：

(1) 矢印を束ねる

始点と終点が共に同じである矢印を束ねて1つの矢印にし、より概略的な名前をつける。

(2) 複数の箱の名称変更

分割と集束により、従属機能の分配をより良くし、それに伴い矢印のラベルを付け直す

矢印の本数が過剰であっても、それらが正確・精密であれば、誤りとなることは滅多にないが、矢印が過剰になることで意味が不明瞭になるのであれば、それは誤りである。(ただ、一般的には、) 読者の理解能力は使用されている矢印の本数に左右されるものである。

6. 3. 3 背景に対する改善提案 (Proposing Modification to the Context)

新しいダイアグラムの作成を通じて、より詳細な部分を理解したことで、親ダイアグラムにおける間違いや見落としが明らかになる場合もある。その時、親ダイアグラムを修正することは自然であり、予め考慮され得る出来事である。矢印構造を作成する際に「機能箱に必要かどうか不明な矢印は描かずにおき、その矢印が本当に必要かどうか判明した後で詳細化を行う」という決まり事があるが、そうした疑問を、憶測を通してではなく、明確な理由を基にして解決することこそが肝心なのである。

親ダイアグラムの変更は、場合によって難易度が変化する。直上の親ダイアグラムだけの変更は、離れたダイアグラムまでも関係する大掛かりな変更よりも、単純なものになる。ダイアグラムの変更が提案されたら、変更内容を十分に考慮し、その複雑性を評価する。大掛かりな変更が必要な箇所を簡単な変更で代用してしまう事は、ダイアグラム分解の質

を低下させてしまうことになる。修正が終わったら、全ての境界上の接続部で、ICOM コードが正しく表されているかを確認する。そして変更の行われたダイアグラムに関する範囲で作業中の他の著者に連絡を行う。

現在詳細化を行っている箱の親ダイアグラムのことを、常に頭に置いておかなければならない。そうすることが、ダイアグラム作成プロセスの補助になる。

詳細化したダイアグラムが背景と合わない場合には、作成したダイアグラムか、もしくは背景が間違っているという事になる。その場合、背景もしくは作成したダイアグラムを変更し、どちらもが合うようにしなければならない。

6. 3. 4 ダイアグラムを接続する為の ICOM 文法

(ICOM Syntax for Connecting Diagrams)

ダイアグラムを理解する為には、必要な事実を探し出して理解する能力が重要である。ノード番号は箱の分解構造を表し、そして矢印によるネットワークはインターフェースの接続を表している。

ダイアグラム上の全ての矢印の未接続端部には、ICOM コードが記載されている。こうした境界にある矢印は、別のダイアグラムへと接続されている。それぞれの境界矢印に記載された ICOM コードによって、親ダイアグラムのどの矢印に接続しているかが明確になっている。

6. 4 ダイアグラム図の構成 (Graphic Layout)

構造的拘束に従って箱を左上から右下へと斜めに配置し、フィードバック矢印を上と左へと伸ばす。この時点で、左から右へと箱に番号を付ける。

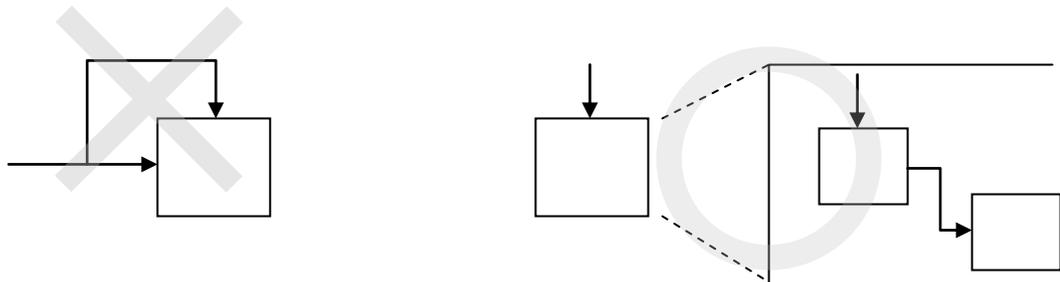
初めは、最も頻繁に使用する拘束矢印だけを配置し、使用頻度の低いものは後で加えるのが良い。一部の矢印だけを先に考慮することにより、箱の配置が決定しやすくなる。そして親ダイアグラムにある全ての境界矢印を描き、残りの矢印を加えて行く。

6. 4. 1 ダイアグラム上の拘束 (Constraints on the Diagram)

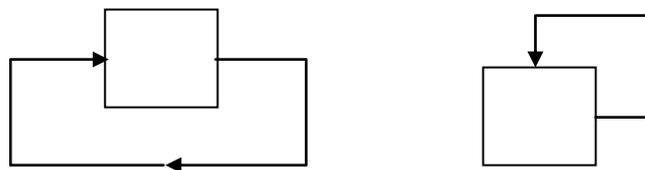
(1) 1本の矢印が、制御と入力の両方の機能性を持っている場合か、もしくはどちらかわからない場合には、制御として作図する。親ダイアグラム上で制御として描かれている矢印は、下階層では制御か入力、もしくはその双方として表現される可能性があるが、それはその階層の従属機能に対する関係性に左右される。

(2) 機能箱にとって、入力矢印は常に必要な物ではないが、制御矢印は必ず必要である。

(3) 通常、1本の矢印を、同じ箱に対する制御矢印と入力矢印とに分岐することは無い。何故分岐させなければならないのかという詳細は、下階層のダイアグラム上で、分岐の行き先と分岐の理由として図示することにより、説明されるものである。その際、2本の枝への名前付け方によって、何故分岐させたのかという決断を伝える事になる。



(4) IDEF0 ダイアグラムにおける循環プロセスもしくはデータの保存は、以下のようになる。



この書式は、「同一階層」上でデータ保存が考慮されている場合のみに用いる。そうでない場合には、箱の内部の次下階層のフィードバックループによって表現する。

(5) 以下のような冗長的表現は避ける：



この場合、箱のラベルと矢印によって伝えられる意味が、完全に重複している。少し考えるだけで、より良い箱のラベルを付けることが出来るものである。

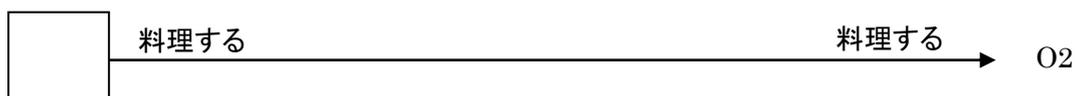
6. 4. 2 矢印の配置 (Arrow Placement)

(1) 矢印は水平、もしくは垂直方向に引き、斜めや曲線（角部を除く）にならないようにする。（訳注：元資料のダイアグラム図の矢印線には、角部と分岐には **R** が付けられているものの、ワードの図形機能ではこの表現が難しい為、この翻訳資料内では全て折れ角として作図している）

(2) 矢印の角や交差部、ラベルは、箱から合理的な距離を空けて配置する。

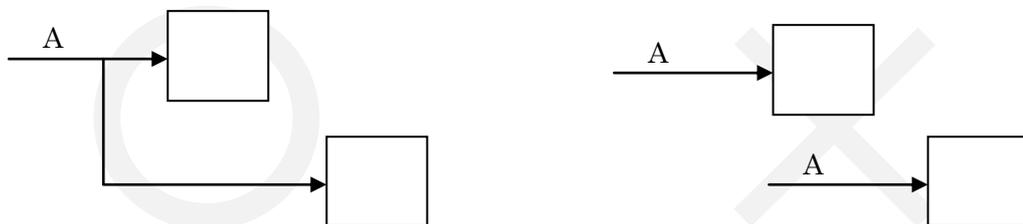
(3) どうしても必要となる場合でない限り、「データ」、「機能」、「入力」、「出力」、「制御」、「機構」といったキーワードを使用したラベルは用いない。

(4) 矢印が長い場合、ラベルは 2 か所に配置する。



(5) 矢印の非接続端部には、必ず **ICOM** コードを記載すること

(6) 全ての場所に影響していることを表すために、境界矢印の開放端は接続しておく。
バラバラのままだと、読者が別々の接続部だと勘違いしてしまう恐れがある。



(7) 矢印が、ダイアグラムシートの余白を超えないようにする



(8) 矢印が近すぎると、視覚的に追って行くことが難しくなるため、平行な矢印は十分に間を置いて作図する。



(9) 明快さを必要とする場所では、追加の矢印の頭を配置する。

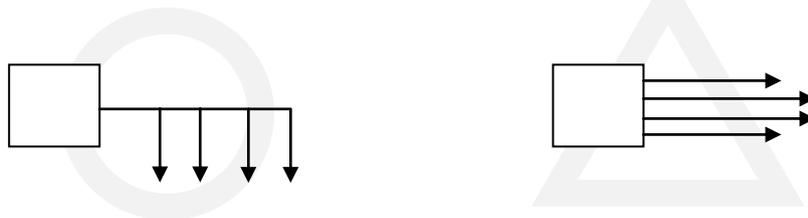


6. 4. 3 矢印の配置 (Arrow Layout)

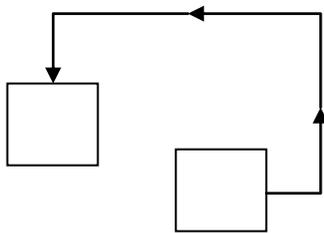
(1) 始点と終点が同一の複数の矢印は、まとめる事で明快さが失われてしまうような重要な物でない限り、まとめておく。



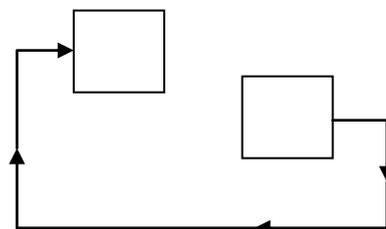
(2) 箱の1辺からは、4本以上の矢印を出さない。もしも4本以上の矢印が出る場合、幾つかを束ねて適切な概略のラベルを付け、終点近くで分離して、バランスが良くなるようにしておく。



(3) 制御へのフィードバック矢印は、制御であることが明確になる様に、上を回るようにする。また入力へのフィードバック矢印は、入力であることが明確になる様に、下を回るようにする。

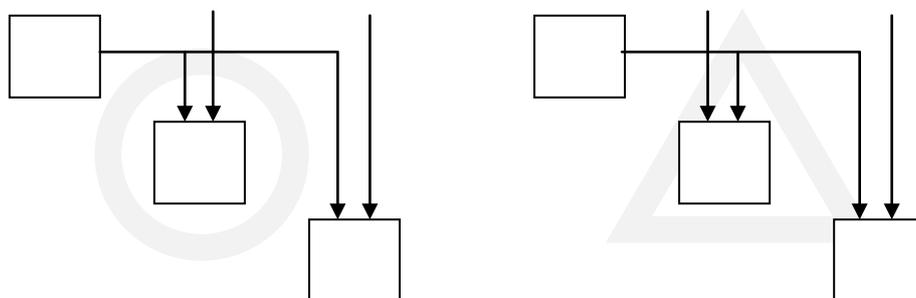


制御へのフィードバック



入力へのフィードバック

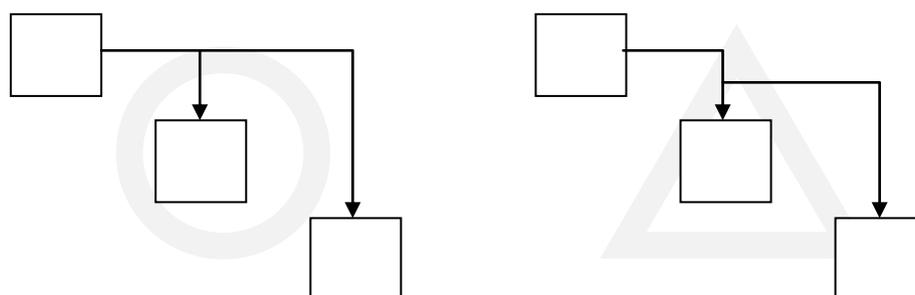
(4) 1本の矢印が分岐して複数の箱へと接続する際には、可能な限り、それぞれの箱のICOM位置の同じ側に来るように作図する。



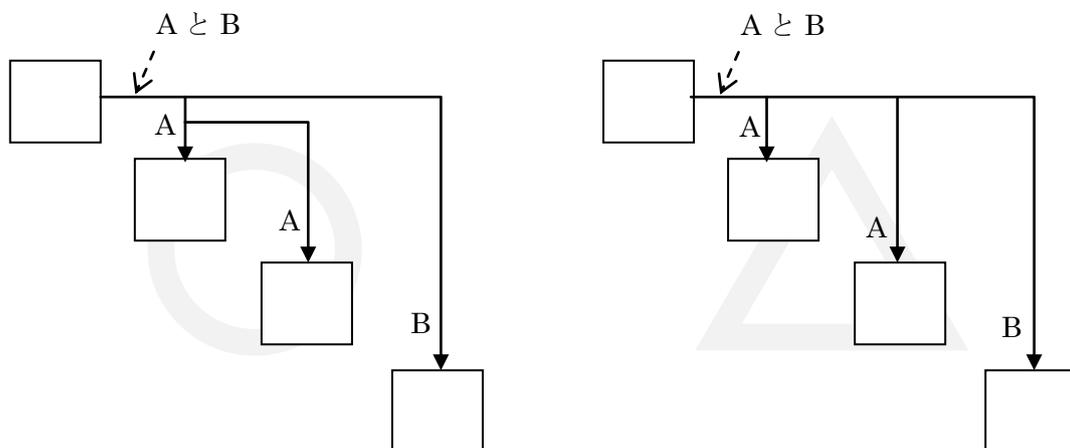
(5) 矢印は、接続しない交差部が最小になるように配置する。



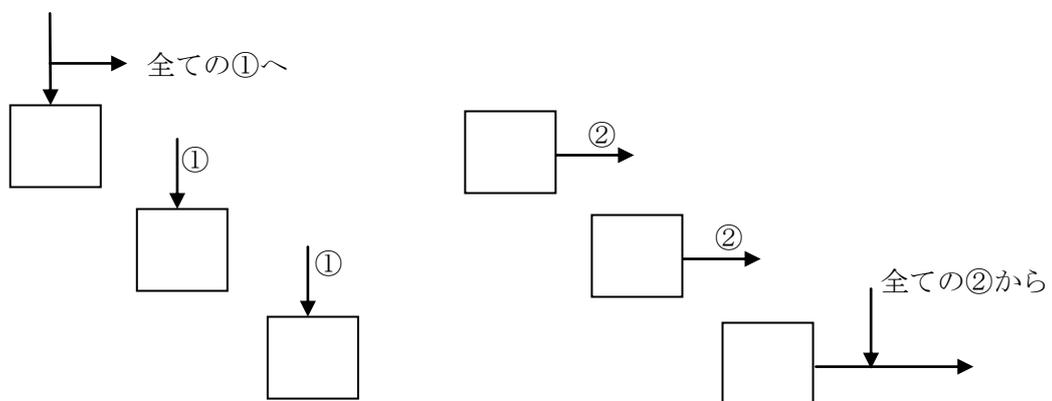
(6) 曲線と曲り部も、可能な限り最小になるように配置する。



(7) 必要な場合には、表現力に富んだ分岐矢印を使用する。



(8) ダイアグラムの混乱を防ぐ為、外部矢印が1つのダイアグラム上にある複数の箱へ入るか、もしくは複数の箱から出るかしている場合、以下のように記述する。



6. 5 説明文の記述 (Writing Text)

ダイアグラムに添えられた説明文 (text) は、そのダイアグラムの簡単な概略である。文量は必ず 1 ページ以内に抑える。説明文では、著者が特に興味を持ち、読者がダイアグラムから、その中に含まれる主要な考えを読解する中で特に気を付けて欲しい箇所を強調する。ダイアグラムそのもので表されていること全てを説明し直したものでは無い。その為、ダイアグラムそのものだけで伝えたい意図を十分に表現できる場合には、説明文は省略してもかまわない。

説明文を書くのは、そのダイアグラムが十分に高いレベルの評価と認可を受けた後である。評価と認可を受けるまで説明文の作成を待つことにより、ダイアグラムそのものだけでも、意図を十分に伝達できるようになる。十分に練られたダイアグラムを基にして書かれた説明文は、基にしたダイアグラムと同じくらい十分に構造化・組織化されたものになる。

ダイアグラムで使用されている専門用語を説明するには、用語集 (glossary) の定義を行う。一つの単語に内包された意味は、異なった会社間どころか、同じ会社の中でさえ異なっているのである。例えば「directive (指示的、指向、等)」という言葉は、設計部門や資産部門 (finance department)、統括 (government)、そして経営方針 (corporate policy) 等の中で、それぞれ異なった意味合いを持っている。

最初は FEO 無しで、出来の良い説明文を書いてみよう。ダイアグラムの微妙な部分を FEO で描き表すことで、そのダイアグラムの意図を明確にすべきではあるが、FEO に含まれるダイアグラムによって、本来のダイアグラムそのものが混乱し、意味がぼやけてしまう事になる。それでも FEO が必要となる場合には、FEO に添えた説明文で、関連するダイアグラムについて言及しておくべきである。。

6. 5. 1 A-0 説明文の記述 (Writing the A-0 Text)

A-0 ダイアグラムが作図されたら、A-0 階層の説明文を作成し、A-0 と共に公布する。全ての分解プロセスは A-0 ダイアグラムから始まっていることから、A-0 ダイアグラムの説明文の中で挙げられた全ての注目すべき事象は、そのままモデル全体へと伝わって行くことになる。

6. 5. 2 参照と注釈 (Reference and Notes)

ダイアグラムそのものと、ダイアグラムの有る部分についての説明文と注釈の記述には、短い参照言語が用いられている。それには以下のようなものがある：

表記	意味
O2	ICOM コード O2 の境界矢印
2I1	箱 2 の入力 1
2O2 to 3C1	2O2 から 3C1 へと向かう矢印
①、②...	注記 1、2、...

こうした表記は、現在のダイアグラム内にあるものを参照している場合には、そのままの表記で用いられる（注記や説明文内で、「①」のようにして用いる）。他のダイアグラムを参照している場合、更に別のモデルのものを参照している場合には、ノード番号やモデル名によって参照先がどこなのかを正確にしなければならない。「. (ピリオド)」は、ある特定のダイアグラムにあるものを、「/ (スラッシュ)」はある特定のモデルにあるものを参照しろという意味に用いられており、以下ようになる：

表記	意味
MODEL/A21.3C2	「MODEL」モデルにあるダイアグラム A21 の箱 3 の制御矢印 2
A42.③	現在のモデルのダイアグラム A42 の注記 3

全ての参照は、完全に一意性が満たされるに十分なだけの表記を行う必要がある。最も長い形式には、いかのようなものがある：

ACCT/A21(T56).1O2 to 4C3

これは「ACCT」モデルのダイアグラム「A21」のバージョン「T56」にある、箱「1」の制御「2」から箱「4」の「制御 3」へと延びている矢印、を示している。

しっかりとした完全な参照を、説明文と用語集、FEO へと加える事

読者の読み間違いを防ぐことは簡単である。

説明文を書いている時には、話の流れ (story) に興味が引かれるように、ダイアグラムを常に注視しなければならない。

参照の追加をためらってはならない (don't worry about)。参照の追加をためらうことで話の流れは悪くなり、文章がぎこちないものになってしまう。

説明文を書き終えたら、一通りチェックし、それから箱番号の参照を加えることで、説明文と実際のダイアグラムとを結びつけてゆく。

殆どの場合、単純な箱番号 (「3」) や、矢印の始点終点のセット (「O1 と O2」) の記述で事足りてしまう。読者が他のダイアグラムを参照する必要があるれば、参照言語の「. (ピリオド)」を使用する。

参照は明確化に用いられる。説明文の話の流れ (story-line) を書き終えた後で参照を加えることにより、参照を目立たないものにすることができる。

6. 6 モデル品質チェックリスト (Model Quality Checklist)

以下にて説明されているものは、IDEFO モデルの品質の構築とその評価の、定義可能 (? definable) なチェック手法である。

6. 6. 1 文法 (Syntax)

文法的拘束は、図形的に何が表現されているかということを論理的に記述しているという意味において、拘束であるといえる。IDEFO のダイアグラムの構成は、関数 (? primitives) であると考えることが可能である。すなわち、文法的拘束とは、許容されたもしくは有効な関係を明確にする宣言であり、こうした関数に影響を及ぼす運用なのである。判り易く言うならば、文法拘束は1つの基準 (criterion) であると考えることが可能である。なぜなら、もしも拘束が不十分か、もしくは図形表示として破綻している場合、それは基準を満たしておらず、その結果、ダイアグラムもしくはモデルが、その明確な拘束にとって不十分だということになるからである (訳注：半分意味わからない)。

少なくとも3つの異なる型の文法的拘束に分類されており、それらは分析の階層や図形

表示の階層にそれぞれ依存している。

(1) ローカル構文 (Local Construction)

このルールは、簡単な、関数の一次組合わせ (first-order combination) か、もしくは関数の関連付け (association) に適用される。これらは FEO ダイアグラムとダイアグラムの為に定義され、正当な構文に従わなければならない。下で挙げたもの以外の、構文が欠落しているか、もしくは関数間の関係性を示しているダイアグラムは「文法的に無効」となる。

(2) グローバル構文 (Global Construction)

このルールは、説明文と FEO 以外のダイアグラム全体に適用される。その為、幾つかの構文に跨って定義され、そしてダイアグラムが完成した後になって初めて確認する事が可能である。

(3) モデル構文 (Model Construction)

この分類での拘束は、ノードの配置と、ダイアグラム間に整列された階層構造を作り出す番号付け/名前付けの指示とを、明確にしている。

以下に挙げるチェックリストは、IDEF0 ダイアグラムとモデルの、最低許容レベルを判定する目安に使用されており、ローカルレベル、グローバルレベル、全てのモデルレベルで、不足箇所を的確に示す事が可能である。ここの全ての評価標準において「完全さ」と「正しさ」に合致していなければならない。IDEF0 文法の為の作業定義 (? working definition) には以下の物がある :

完全さ (completeness) :

定義

全ての必要な入力項目、ラベル、箱、矢印、識別表記が、ダイアグラムもしくはモデルに表示されている度合い

正しさ (correctness) :

定義

ダイアグラムもしくはモデルが正確に構築され、ラベル付され、識別されている度合い。文法的関係が、図形的に適切に表示され、拘束に従っている度合い。

計測

アイテム毎に、拘束への適合度をチェックする（6. 6. 1. 1 から 6. 6. 1. 3 までを参照）

6. 6. 1. 1 ローカル構文の文法 (Local Construction Syntax)

(a) 一方向の矢印線 (arrow segment) は、始点 (SOURCE) と終点 (SINK) の 2 つの端点の組み合わせで構成される：

始点 (SOURCE) とは、以下の何れかである：

- ダイアグラムの I、C もしくは M 境界に近い境界端点
- 箱の O 边上の端点
- 分岐終点 (fork sink) ※
- 合流終点 (join sink) ※

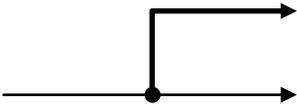
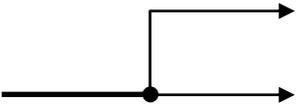
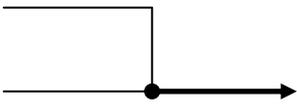
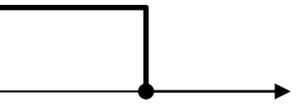
終点 (SINK) とは、以下の何れかである：

- ダイアグラムの O 境界に近い境界端点
- 箱の I、C もしくは M 边上の端点
- 分岐始点 (fork source) ※
- 合流始点 (join source) ※

境界端点同士の組み合わせを除く、全ての組み合わせが合法である。

訳注※：

sink と source の意味合いが今一つ分かり難い。下図のようなものか？

	始点 (source)	終点 (sink)
分岐 (fork)		
合流 (join)		

(b) 矢印ラベル (arrow label、矢印の名前) は、矢印線と結びつける。ジグザグ線 (line squiggle) を使う事で、関係性を明確にする。

(c) 親箱から来ている ICOM コードを、矢印の境界端点と結び付けなければならない。

(d) 箱は整数番号を持つことが出来る。箱番号は 1 から始まり、1 ずつ増加する。

(e) 箱の名前 (Name) は箱と関係付ける。名前は単独で配置できない。

(f) ダイアグラム境界上の端点もしくは矢印は、トンネル「()」もしくは ICOM コードの何れか片方のみを持つ。

(g) 箱边上の矢印の端点は、トンネル「()」を持つことが出来る。

6. 6. 1. 2 グローバル構文の文法 (Global Construction Syntax)

・完成した IDEF ダイアグラムは、A-0 ダイアグラムを除いて、3 個から 6 個までの箱で構成される。A-0 ダイアグラムは、1 個の大きな箱のみで構成されている。

・全ての箱には、名前と番号を付けなければならない。全てのダイアグラム毎に、番号は 1 から始まり 1 ずつ増加する。A-0 ダイアグラム上の箱の番号は 0 である。1 つのダイアグラム上の 2 つ以上の箱に、同じ名前を付けてはならない。

・全ての箱は、最低でも O 辺 (箱の右辺) 上に矢印の始点を 1 つ、C 辺 (上辺) 上に矢印の終点を 1 つ、それぞれ持たなければならない。

・A-0 ダイアグラムを除く、ダイアグラム境界にある矢印線の全ての端点は、ICOM コードもしくはトンネル「()」のどちらかを持たなければならない。A-0 ダイアグラムの場合は、ICOM コードもトンネル「()」も持たない。

6. 6. 1. 3 モデル構文の文法 (Model Construction Syntax)

IDEF0 の適正なモデル構文の為のルールと約束 (convention)

1つのモデルは、階層構造をした多数のノードによって構成されているが、この階層構造は継承的な洗練を表現している。階層構造上にある個々のノードは、1つ以上のダイアグラムとして詳細化される。モデルの頂点にあるノードの IDEF0 ダイアグラムは、A0 ダイアグラムである。箱は、その箱のあるダイアグラムのノード番号に箱番号を付け加えたノード番号を持っている。ダイアグラムのノード番号は、基になった箱と同じ名前を持っている（ノードが同じである）。ただし、箱番号が 0 の場合（A0 ノードから派生する物）は、この 0 を Null として扱い、ノード番号の中には入れない。例えば A0 ノードの詳細であるダイアグラム上にある箱 2 を分解したものの識別子は、(A02 ではなく) A2 となる。また、ノード A2315 の詳細であるダイアグラム上にある箱 1 を分解したものの識別子は、A23151 となる。（箱は最大 6 個までなので、階層毎のコードの増加分は 1 桁のみ）

ある IDEF ダイアグラムを説明している FEO の識別子は、そのダイアグラムの識別子の右端に F の文字を付けたものになる。ダイアグラムもしくは FEO について説明した説明文 (text) もしくは用語集 (glossary) の識別子は、同様に右端に T、G の文字を付加する。例えば、A21 ノードの箱 3 について説明した FEO の識別子は、A213F となる。A114 ノードについての説明文の識別子は A114T となる。また A31F という FEO についての説明文の識別子は A31FT に、A114 ダイアグラム上で使用されている用語を定義した用語集は A114G となる。更に、FEO や用語集で、2 ページ以上になるものについては、F もしくは G の右にページ番号を加える。A213 ダイアグラムについての 2 枚の FEO の場合、それぞれ A213F1 と A213F2 となる。

IDEF 機能モデルの頂点ノードは常に A0 であるが、背景 (context) と環境を示す為に用いられる、頂点よりも上階層のノードも存在する。A-0 (A マイナス 0) は、A0 の背景を示している。この A-0 の箱は、A-1 ダイアグラムへと書き下される。

親ダイアグラムと子ダイアグラムの間に跨る矢印の接続部は、ICOM コードを用いて管理する。ICOM 表示は、子ダイアグラムの矢印の境界端点上に明確に表記する。それに対する親箱の ICOM 表記は、矢印の関係点 (context point) の位置によって暗黙的に決定される。関係点のある箱の辺と一致する I、C、O、M の文字が割り振られる。この文字の前には、箱の同じ辺にある矢印の順番（上から下、左から右の順番に割り振られる）の数字が付けられる。親箱の上の矢印が追加、削除、移動される度に、この ICOM コードは変更される。

6. 6. 2 語義 (意味論、Semantics)

「語義 (semantics)」とは、要素に幅広く割り当てられた用語であり、その全てが IDEF0 内のラベル、ダイアグラム、そしてモデルの「意義 (意図、meaning)」に何らかの影響を与えている。モデルの評価に便利な基準を作る為には、IDEF0 の、データと機能の明確な命名と、そして構成に関する「意義」のそうした方向性 (aspect) を取り扱う必要がある。ダイアグラムもしくはモデルの全ての「話の流れ (story)」、「メッセージ (message)」そして「意味 (sense)」といったもの全体を評価する為の基準は、複雑性の取扱に関して言及される。

少なくとも以下の 5 つの型の語義について評価されるべきである：

- ①完全性 (COMPLETENESS)
- ②簡潔さ (CONCISENESS)
- ③一貫性 (CONSISTENCY)
- ④正確さ (CORRECTNESS)
- ⑤複雑性／判り易さ (COMPLEXITY／UNDERSTANDABILITY)

以上の評価基準は、それぞれ次のように考えられている。

完全性 (Completeness)

情報の包含範囲の適切性を評価する便利な計測として、倍率 (amplification) がある。この倍率は、親から子ダイアグラムを作る際に、どれだけの詳細が追加されたのかという度合いを示している。理想的には、情報内容において、親がより包括的なものとなるように十分に詳細化されるべきであるが、モデル全体が複雑で理解し難いものになってしまう過度の詳細化は為されるべきではない。そしてまた、機能箱のラベルによって伝達される詳細化の度合いと、データ矢印ラベルによって表現される詳細化の度合いとで、バランスが取れていなければならない。機能箱には非常に一般的で抽象的なラベルが付けられているが、データ矢印には極めて詳細なラベルが付けられているダイアグラムは、語義的に (semantically) バランスが悪いといえる。また、データ矢印のラベルをより詳細にする

には場所が狭い場合には、継承される子ダイアグラムでの倍率係数 (factor) は低くて良い。

完全性 (Completeness) の定義と計測：

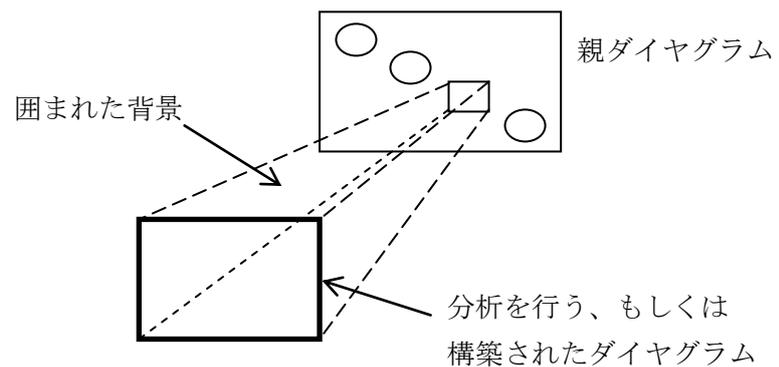
定義：

情報内容が主体対象 (背景) をどれだけ十分に網羅しているか

尺度とパラメータ：

ダイアグラムレベル：

最上層のものを除いた全てのダイアグラムにおける、親ダイアグラム上の囲まれた背景 (bounded context) に対する、情報包含の十分さの度合い



全モデルレベル：

箱の分解 (= 詳細化) の度合い

(モデルの目的に従って変化する。

モデルの目的を完全に伝達する為に必要となる、詳細さの総量)

モデルの最上層から最下層までの間で行われた、矢印の詳細化の度合い

(機能箱の詳細化と抽象化のレベルに従って変化する)

簡潔さ (Conciseness)

IDEF0におけるラベルと題名の「情報量 (information value)」は、究極的には、読者に対するダイアグラムの内容の潜在的妥当性 (potential relevance) と言える。この理由から、中でも生産環境における技術的なプロセスや実践 (practice) に関係しているのであれば特に、著者は共通に使用されているラベル名を選択すべきである。ダイアグラムと、そのダイアグラムに付属の説明文で使用されている単語もまた、参加している読者・聴者にとって自然なものであるべきである。例えば、会計士の読者 (audience) は、「間接労働 (indirect labor)」や「直接労働 (direct labor)」、「一般管理費、間接費 (G&A overhead)」といったラベルに関係しているが、プロジェクト管理者の場合は、「?後払い労働者 (billable labor)」や「?社内研究 (internally funded research)」、「?補助労働者 (support labor)」といった用語を用いる方が望ましくなる (訳注: 会計士でも経営者でもないので、どういふ言葉で訳すべきか良く判らなかった)。

1つ、もしくは連続したダイアグラムを簡単に流し読みするだけでは、「冗長性 (redundancy) の度合い」を測るのは難しい。2つのラベルが、同じ意味である別の2つの単語であったり、同じ名前だが異なった物であった場合には、著者が重要な用語に関する用語集を準備し、技術用語の特徴や特性を十分に詳細なリストにまとめることで、読者と評価者は、より客観的な決定手段を得ることが出来る。

IDEF0のラベルは、IDEF1 (訳者注: 情報モデルを表現する構造体) の実体クラス (entity class : 共通の属性を持つ実体の集合) 名と属性クラス (attribute class : ある実体クラスの持つ属性値の集合) 名の候補となりうる。もしも、ある実体クラス (entity class) が IDEF0の矢印ラベルに使用されている場合には、実体クラスの属性によって、矢印の内容を正確に理解する助けとなるとともに、矢印の理解や比較に使用することができるようになる。ダイアグラム内で命名されている実体の属性をリストにまとめたものを用語集の書式へと追加する事で、より便利なものになる (相当に意識)。属性リストがあると、属性セット間の重複 (overlap) の度合いを見る事で、「冗長性 (redundancy)」を評価する事が可能となる。

簡潔さ (Conciseness) の定義と計測 :

定義 :

- ダイアグラムもしくはモデルに含まれているか、それらによって伝達される情報の精度
- 用語論と記号論の妥当性
- モデルの方向性 (orientation) の周辺情報は含まれない (absence)

尺度とパラメータ :

- ラベル、題名、機能名の持つ情報量
- ラベル名の冗長性 (redundancies) の度合い
- 以下のものに対する用語の自然さ :
 - 主題 (subject matter)
 - 読者 (audience)

一貫性 (Consistency)

以前に挙げた文法のチェック作業は、あるモデルの階層構造を通じたデータ矢印の、ラベル付けと追跡可能性の一貫性の伝達を、図形表記法が補助しているかどうかの確認に役立っている。モデル内容の一貫性の度合いについては、むしろ批評者とモデル対象に関わる専門家の側が判断することとなる。

しかしながら、機能箱と矢印の呼び名 (designations) の場合、ダイアグラムもしくはモデルに付属する用語集の一部として属性リストが準備されることにより、命名の一貫性が促進されることになる。

(訳注：わけわからん。

- ・データ矢印命名と追跡可能性の一貫性 → 文法チェックを通して確認
- ・モデル内容の一貫性 → 評価者と主体に関わっている専門家がチェックして確認
- ・機能箱と矢印の名前の一貫性 → 用語集内の属性リストを準備して一貫性促進
という3つのバラバラな事を述べているだけなのか?)

一貫性 (Consistency) の定義と計測：

定義：

- ・内部：
(モデル内部での) 表記法、記号論、用語論の画一性 (uniformity) の度合い
- ・外部：
(ダイアグラムもしくはモデルの外での) モデル化されているシステムに対して
内容が追跡可能 (traceable) である度合い
(訳注：下にもあるが、トンネル記号による外部へのワープのことか)

尺度とパラメータ：

- ・ICOM ラベルの正確性：
階層構造上の配置と追跡可能性
- ・トンネル記号「()」が正しく使用されているか

正確さ (Correctness)

「正確さ」は恐らく、この章で説明されている語義の特性の中で、最も主観的なものである。その主な理由は、評価の為の最終的な標準は、モデル化されている主題に関して最も完全な知識を持っている人の意見に依らざるを得ないからである。その為、IDEFOのキットサイクルは、専門家 (technical expert) による批評だけではなく同輩からの批評を基にして「正確さ」の判断を可能にすることで、モデル構築を補助するように作られている。

正確さ (Correctness) の定義と計測：

定義：

- ・ダイアグラムもしくはモデルの中で表現されている情報が、モデル化されたシステムをどれだけ正しく記述しているか
- ・含有された拘束関係と、データ機能間のインターフェースが、妥当な関係性を表現している度合い

尺度とパラメータ：

- ・専門家によるモデルの批評
- ・用語と技術の参考になっている社内文書もしくはテキスト本のチェック

複雑性／判り易さ (Complexity/Understandability)

「判り易さ」は、モデルの品質評価基準として重要なものであるが、主にその抽象性と主観的暗示の為に、難しいものとなっている。更に「判り易さ」の計測が、文法的、語義的要因の強力な相互関係であることも、計測を難しくしている。この「判り易さ」を、モデルの内容が文法によって如何に良く描写されているかという機能として定義する事で、文法と語義の両面から適切な計測法を導くことが出来る。その為、語義の中の1つの特性だけでは、批評者がモデル全体を十分に理解可能となる指標にはならないのである。

一方で、「複雑性」の計測は「判り易さ」の計測よりも容易な作業である。その為、「判り易さ」と「複雑性」を対となる (dual : 二元的な) 計測法として見る事を提案している。すなわち、非常に複雑なものは理解し難く、また複雑性が低いものはユーザーにとって取り付きやすく、より簡単に理解することが可能である。

複雑性／判り易さ (Complexity/Understandability) の定義と計測

定義:

- ・モデルもしくはダイアグラムの目的が、どれだけ批評者に明確になっているか
- ・モデル／ダイアグラムの主張が、文法を通じて正確に表現されている度合い

尺度とパラメータ:

- ・「主要な流れ (main path)」のを見つけやすさ
- ・活動／箱の数
- ・潜在的なシナリオ (scenario) の数
- ・活動とシナリオの識別の容易さ (facility)
- ・「配置の改善 (refining the layout)」のやりやすさと成功しやすさ
- ・組み合わせと結束 (cohesion) の評価体系 (rating scheme)

IDEFO の著者とコメンターにとって便利な、もう一つの尺度は、ダイアグラム上の配置の改善の容易さである。ここで定義するところの「配置の改善」とは、機能箱とデータ矢印インターフェースの配置パターンによって伝えられる、ダイアグラムもしくはモデルのメッセージの曖昧さを軽減する為に必要な手続きである。モデルがより複雑に、そしてより理解し難くなるに従い、配置の改善作業もより困難になる。複雑さを低減する為に用いられるダイアグラムの配置を改善するルールとして、以下のようなものがある:

- ・2つ以上の ICOM コードを伴う境界矢印は、ICOM コード毎に矢印を割当てて

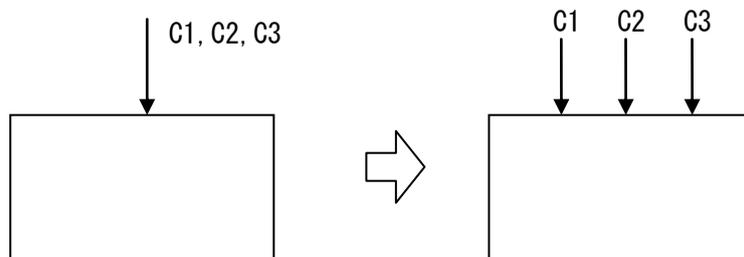


図 6-2 2つ以上の ICOM コードを伴う矢印

- ・同じ ICOM コードを持つ境界矢印は接続 (connect) する

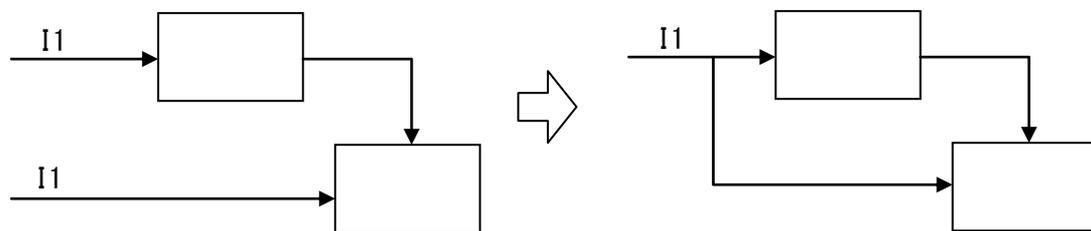


図 6-3 境界矢印の接続 (connecting)

- ・親ダイアグラムと子ダイアグラムの間で名前が異なるものの、同じデータを表現している矢印は、著者によって名前を一致させなければならない。著者は適切な名前を決定し、モデルを通して同一の矢印の名前が一致するようにする。

(訳注：以下の図 6-4 は文字が潰れて読めないなので、そのまま添付)

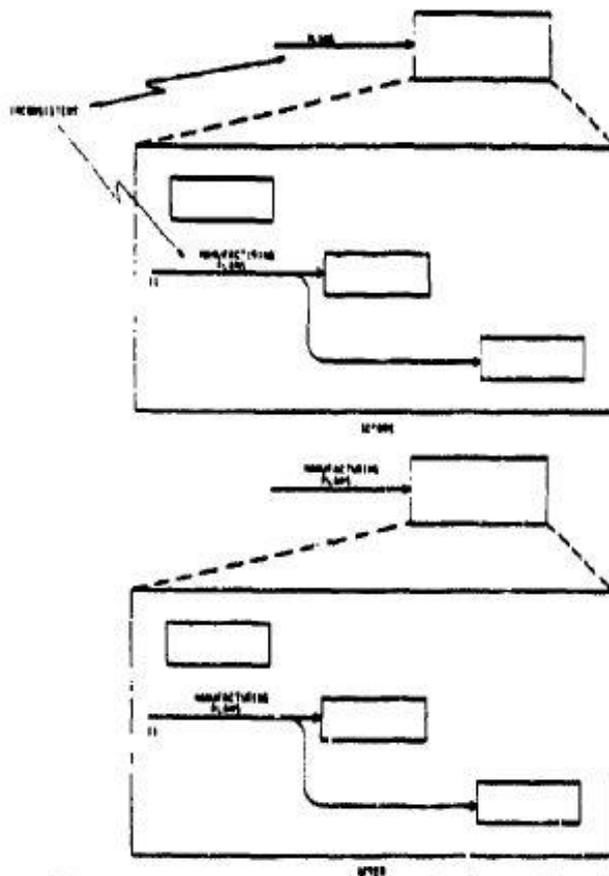


Figure 6-4. Arrow Labels Remain the Same

・矢印の名前が同じだが、それぞれが別の意味を持っている場合、著者が異なった名前を割当て直さなければならない。

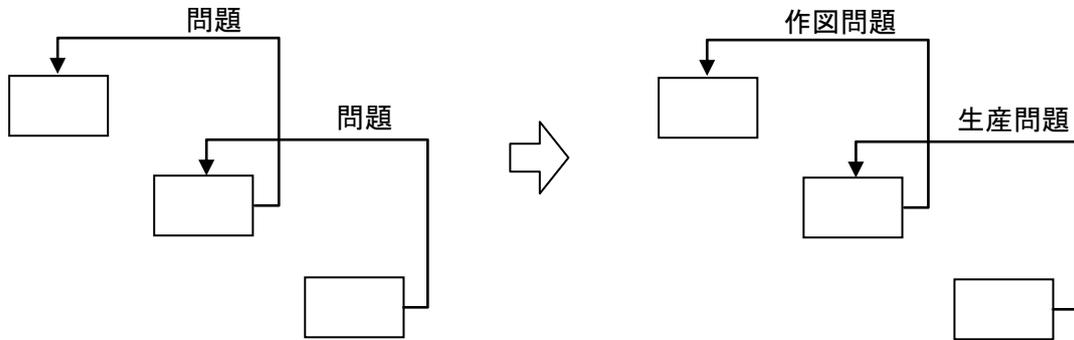


図 6-5 矢印ラベルの定義

・矢印を分割・合流させた時、無名の区間には、著者が明確な名前を割り振らなければならない。

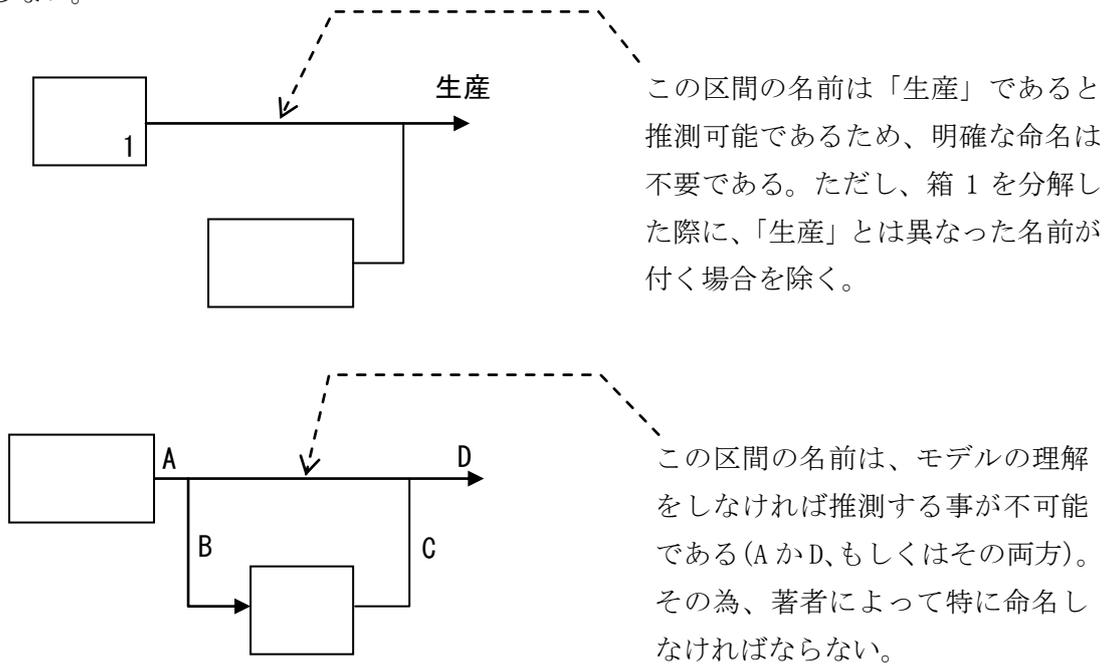


図 6-6 矢印ラベルの分割

6. 6. 3 結合 (Coupling) と結束 (Cohesion)

結合 (Coupling) は構成物 (components) 内での相互接続の度合いを表している。IDEF0 では、1つのダイアグラム上の機能箱間の接続の数と種類が、構成物内の相互接続となる。一方で結束 (Cohesion、分子の凝集力) は、与えられた部分内にあるサブ部分 (subpart) の近縁性 (relatedness) の度合いを表している。「結束」は、IDEF0 では1つのダイアグラム内にある機能のグループ化のやり方を評価する際に使用する、意味深い特性である。

複雑性の評価に用いるこれらの尺度の関係性 (relevancy) は以下のようになる：

「ダイアグラムもしくはモデルの構成物が、互いに多く結合する場合、モデルはより複雑なものになる。また、構成物が無関係 (=unrelated、無結束=incohesive) な要素で構成されている時にも、モデルはより複雑なものになる。」

このように、IDEF0 ダイアグラムは、その構成物の結合を緩やかなものにする事で、より理解しやすくなるが、その一方で機能的な関連可能性や結束パターンへと成り下がってしまうことになる。

6. 6. 4 結合と結束を基にした計量法 (metrics)

モデル内の結束を強く、結合を弱くすることにより、モデルの理解しやすさが向上し、複雑性が低減する。その為、結合と結束の種類を基にした計量法は、IDEF0 モデルの理解の容易さと複雑性の評価に便利である。また、結合と結束の種類を認識する為の尺度、もしくはコードを著者が持つことで、より理解しやすく、簡単なモデルの構築が容易になる。

6. 6. 4. 1 他のシステムエンジニアリング特性との関係性

(Relation to Other Systems Engineering Properties)

IDEF0 モデルの結合と結束の種類を評価する計量法を作り出す事により、追加の利点が幾つか出てくる。

結合数を削減することにより、同一の想定でありつつもモジュール数が最小となるように

モジュール接続の数が制限されることから、特に「あるがまま (? AS IS)」のシステム記述を「あるべき (? TO BE)」状態へと変換する事が容易となるのである。

(原文注：ここで言う所の「モジュール接続」は、**IDEF0** ダイアグラムでは機能/データのインターフェースのことである)。

遮蔽 (**hiding**) というシステムエンジニアリング原理 (**systems engineering principle**) もまた、同じ状況を表している。つまり、あるモジュールのセットだけに対して公開された情報は、他のモジュールから遮蔽されている (もしくは孤立させられている) と言える。結合の削減等による情報遮蔽の利点は、変更の必要性の余り無い他のモジュールや構成物の変更を行わなくても良くなる事から、1つのモジュール内もしくはシステム設計内での局所的な変更を促進する事である。その為、遮蔽と結合の間には直接的な関係が存在しており、ある特定のモデル内での結合の度合いを評価することで、情報遮蔽の範囲についての情報も得ることができる。

遮蔽 (**hiding**) が結合の同型物 (**isomorphic**) であるのに対して、局所化 (**localization**) が結束の同型物となる。局所化とは、物理的な近接範囲内を要素集めて、その価値を記述する原理であり、あちこちに同一の情報が重複して定義される事を防ぐことにより、冗長性が最小となる。

例えば、**ICAM** の「あるがまま (? AS IS)」の基本設計概念 (**Architecture**) (製造の機能モデル、**Function Model of Manufacturing**) から、一般的な製造サブシステムを識別するという潜在的な価値を考慮してみる。例えば、製造管理 (**manufacturing control**) と資材運用 (**material management**) に関する機能を、別々の機能モデルとして局所化する事により、適切な機能とデータとが、機能的・物理的近接範囲へと集められ、それにより統合されたユニットとしてではなく分離された状態で見に行くことが可能となることから、「あるべき (? TO BE)」環境への転換が促進されることになるのである。環境全体に渡った変化を予測することは、基本設計概念 (**Architecture**) に広がった機能に対して影響を持つ可能性のある全ての要素を追跡しなければならず、非常に困難な作業となる。

訳注：システムエンジニアリングにおけるモジュール化・局所化の原理や特徴は、プログラミング手法や、組み立て式パソコンの仕組みを参考にすると判り易い。階層的組織もモジュール化・局所化の原理が用いられているが、日本においては正しく機能していないものが多く、あまり参考にならない。

6. 6. 4. 2 結合の計測と種類 (Measures and Types of Coupling)

ここでは、IDEF0 の機能間に存在する結合を調査しつつ、結合の種類の違いを説明してゆく。

6. 6. 4. 2. 1 記述の視点 (Viewpoint of the Description)

システムエンジニアリング学 (system engineering literature) では、機能間の結合は、伝統的に 2 つの視点から分析する事が可能である。その 1 つは、結合におけるデータの性質 (nature) を考慮してゆくことであり、もう一つは、結合に存在するデータ内にある多くの顕れ (token) や例 (example) の間にある関係性を識別してゆくというものである。

下の図 6-7 は、この 2 つの視点からの分析について説明したものである。図内で使用されている用語はシステムエンジニアリング学 (literature) 内でも見受けられるものであり、その用語の技術的意味については、順次説明して行く。

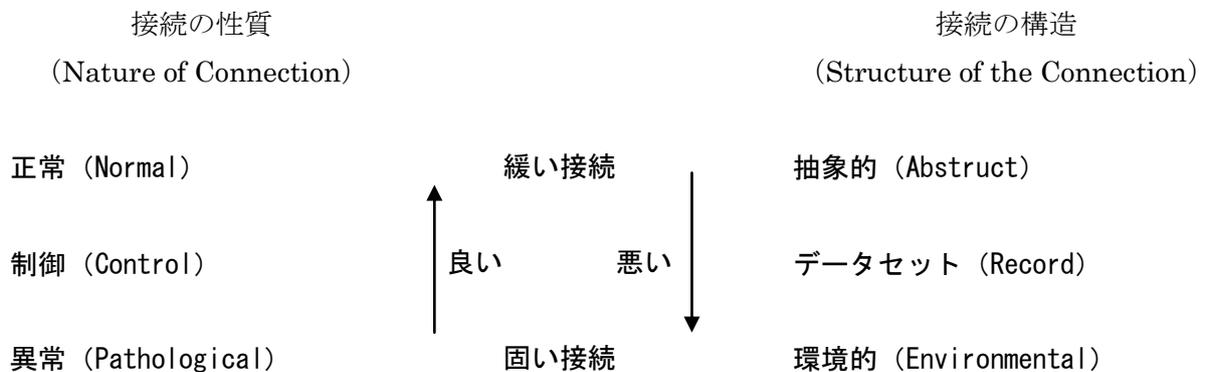


図 6-7 機能間の結合の分析

訳注：接続の性質

遮蔽原理を上手く利用することで、特に階層の左右方向での機能間の組合せが、単純で判り易いものになっているか

訳注：接続の緩さ・固さ

接続数が多く、また接続の仕方が複雑になる程、構成を動かさにくく、硬いものになる。それは同時に、構成が把握し難いということでもある。

訳注：接続の構造

抽象化階層を上手く利用することで、特に階層の上下方向での機能間の組合せが、単純で判り易いものになっているか

6. 6. 4. 2. 2 接続の性質 (Nature of Connection)

異常 (Pathological) : 最低レベル (The Least Desirable)

片方の機能がもう片方の機能の内容 (contents) を参照している 2 つの機能の結合は、「異常な結合 (coupled)」である。この状態は、遮蔽原理 (hiding principle) の適用に失敗した時に発生する。

下の図 6-8 のような例を考える。箱 1 と 2 の間のデータ矢印は「異常」であり、この定義により 2 つの箱は「内容結合 (content-coupled)」になってしまっている。箱 2 の効果により、箱 1 の活動で必要になる材料が削除されてしまうのである。このような種類の結合は、他の要因が条件 (qualification) を提供する場合を除き、通常は望まれないものである。

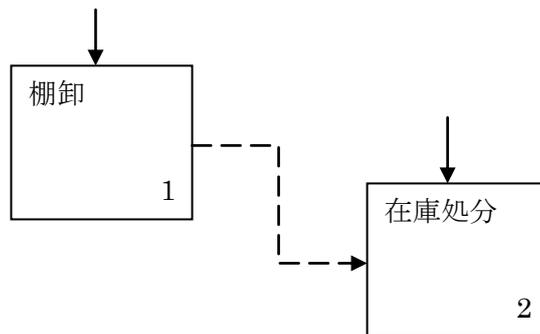
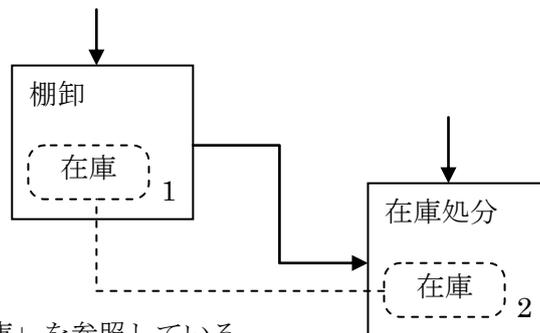


図 6-8 異常な結合



それぞれの機能内で同じ「在庫」を参照している。

「在庫処分」の結果、その中の「在庫」が削除されると

「棚卸」内の在庫も削除され、「棚卸」を実行できなくなる。

(訳注) 図 6-8b 内容結合？

制御 (Control)

2つの機能が「制御結合 (control-coupled)」しているというのは、片側から出力されたデータが、もう片方の制御の流れ (flow-of-control) に影響を与えている状態である。ただし、この制御の流れ (flow-of-control) とは、箱の制御矢印 (control-arrow) とは別のものである。ここで言う所の「制御 (control)」はデータの種類であり、ダイアグラムの箱の上の位置のことではない。ある機能が、もう片方の機能の挙動を制御する影響を与える時に「制御結合」が発生する。例えば「記録失敗メッセージ/印刷エラーメッセージ」のように、意図した機能と共に「フラグ (flag)」を生成する機能名 (function-names) に注意することで、この種類の結合を見分ける (spot) ことが可能である。

訳注：

例が不明瞭で、今一つピンと来ない。

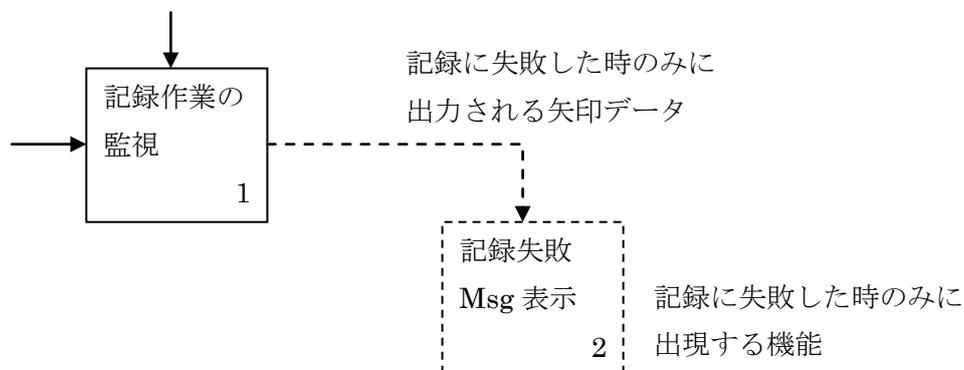
ある機能の出力結果によって、別の機能が異なった挙動を示す場合が「制御結合」状態ということらしい。

例にある「記録失敗メッセージ」の場合：

記録作業の監視

記録失敗メッセージの表示

という2つの機能が結合されることになるが、「記録失敗メッセージ表示」機能は記録作業に失敗した時のみに実行される物であり、成功した場合には実行されない。



正常 (Normal)

2つの機能が「正常に (normally)」結合しているというのは、2つの機能の間のデータ接続 (connection) が互いの内容と独立して存在しているか、もしくは互いに制御の流れに影響されていない、ということである。機能間の情報伝達がデータ経由で明確に行われ、かつ、互いの内容を暗黙的に参照していないということから、この種類の結合 (coupling) が、最も望ましい種類の接続 (connection) であるといえる。

訳注：

極言するならば、**IDEF0** で定められた箱と矢印のルールのみによって完全に定義された状態が、正常状態である。なぜなら、**IDEF0** のシステムはスマートな情報伝達を実現することを目指して作られており、スマートでない情報伝達を表現する手法については、最初から考慮されておらず、表現する事が不可能である。

ルールに従えない不細工なダイアグラムを描くことは可能だが、しかしそれは、改善を繰り返すことで、いずれはルール内に収まるように調整しなければならないものである。

6. 6. 4. 2. 3 接続の構造 (Structure of The Connection)

環境的構造 (Environmental)

2つの機能が、どちらも1つの一般的なデータソースを参照しているにも関わらず、それぞれの箱にデータのどの部分に関係しているのかが明確にされていない時、これらの機能は環境的に結合している。この状態は、下の図6-9にあるようなダイアグラムで表される。

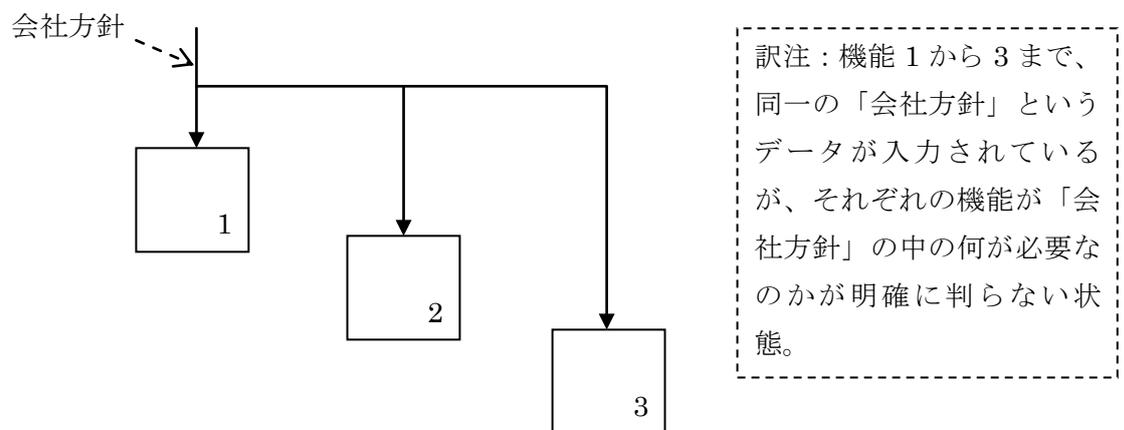


図6-9 環境的結合

訳注：環境的結合

複数の機能間の関係性をまとめて定義された状態。その為、個々の機能間の関係性は不明瞭になっている。

環境的結合による問題点は、以下の通りである：

- (1) 遮蔽原理 (**hiding principle**) が排除されてしまう。
必要なデータのみアクセスを制限できない。
- (2) 局所化が排除される。使用するユーザーに対してのみにデータを関連付けることができない。
- (3) エラーの伝播が潜在的に増加してしまう。
環境の変化によって、全ての機能が予測不可能なものになってしまう。

データセット構造 (Record)

訳注) コンピューター用語としての record :
互いに関係のあるデータ項目の集合で、1つの処理単位となるもの

2つ以上の機能が、グローバルで無い (non-global) 1つのデータセットを参照し、かつ、環境的結合のように発生し得るインターフェースを全て持つ必要が無い場合、それらの機能はデータセット結合である。IDEF0での例は、下の図 6-10 のようになる。

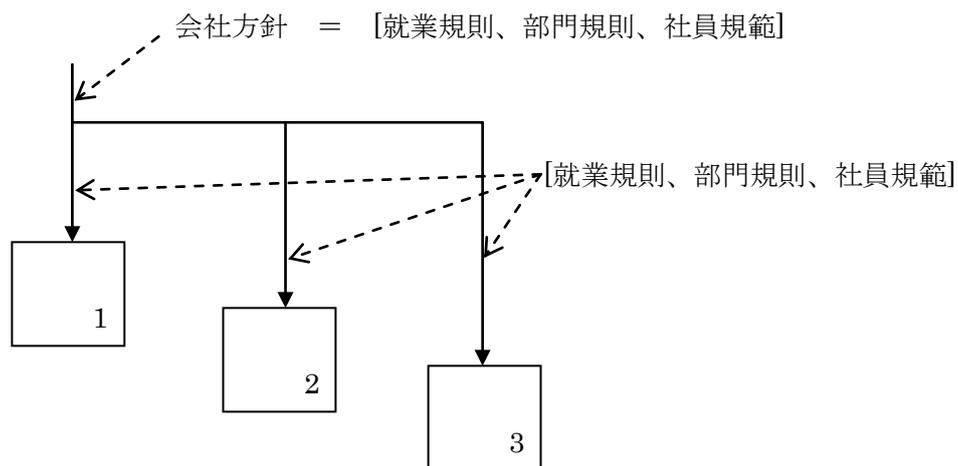


図 6-10 データセット結合 (訳注：元図から変更している)

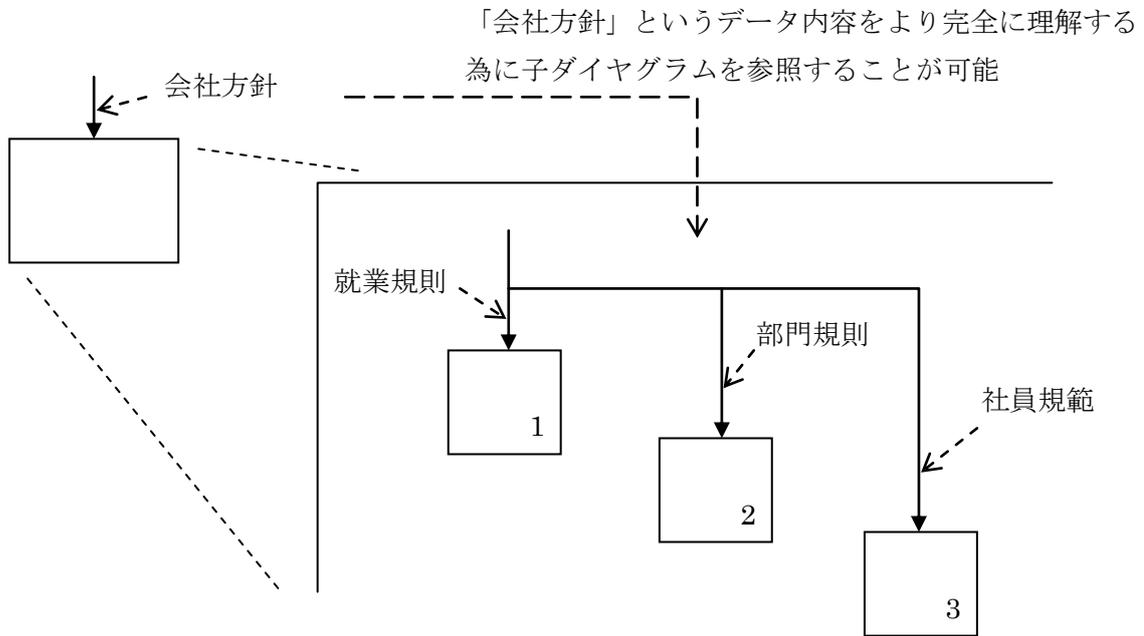
訳注：

環境的結合の場合、「会社方針」内に何のデータが含まれているか不明瞭であるため、「会社方針」に含まれそうな全ての可能性を考慮しなければならないが、データセットを介した結合の場合、「会社方針」が「社員規則」「部門規則」「集合規則」の3つのデータのセットであることだけは明確になっていることから、この3つのデータに対するインターフェースを準備していれば良い。

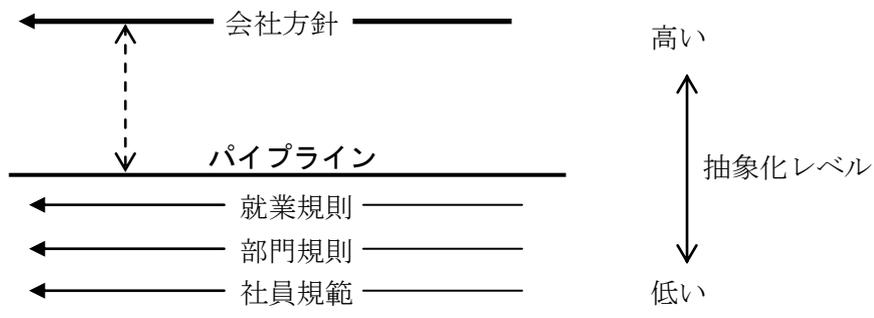
しかし機能 1~3 は、それぞれ必要とするデータは1つのみであり、他2つは不要となる。この点において、この後で述べられる抽象的構造と比べると、スマートではない。

抽象的構造 (Abstract)

ある特定の要素と、その要素の潜在的「構成 (constituents)」の関係性が明確になっている場合、2つの機能は抽象的に結合している。IDEF0において、親ダイアグラム上のデータ矢印が、子ダイアグラム上で詳細な構成物へと分解されているのは、抽象的結合である。親ダイアグラム上の矢印が、データ内容をより完全に理解する為に必要となる抽象化レベルへのアクセスを提供する為に「抽象化の力を使用」している限り、遮蔽原理 (hiding) は効果を持つのである。IDEF0での「矢印パイプライン」という考え方は、抽象的結合そのものであり、ダイアグラム構築の際には、これを最大限使用すべきである。



抽象的結合の例の予想図 (訳注)



パイプラインによる抽象化 (訳注)

6. 6. 5 結束の尺度と種類 (Measures and Types of Cohesion)

「結合 (coupling)」が、要素 (機能もしくはデータ) の間の接続 (connection) のことを指しているのに対して、「結束 (cohesion、凝集)」は1つのダイアグラム上にある要因 (factor) 間の、束縛 (binding) もしくは相互関係のことを示すために使用されている。

システムエンジニアリング学 (system engineering literature) では、束縛 (binding) は少なくとも以下の7つの種類に区分されている：



それぞれの種類について、IDEF0 の典型的なスケッチで説明してゆく。

(0) 同時的 (Coincidental) : 最低レベル

同時的結束 (Coincidental cohesion) は、モジュールの要素間に建設的 (constructive、構造的、積極的) な関係性が殆ど無い場合に発生する。この状況には、1つのダイアグラム上の IDEF0 矢印上のデータ名が、相互関係を殆ど作らない時にも含まれる。図 6-11 は同時的結束の極端な例を示している。

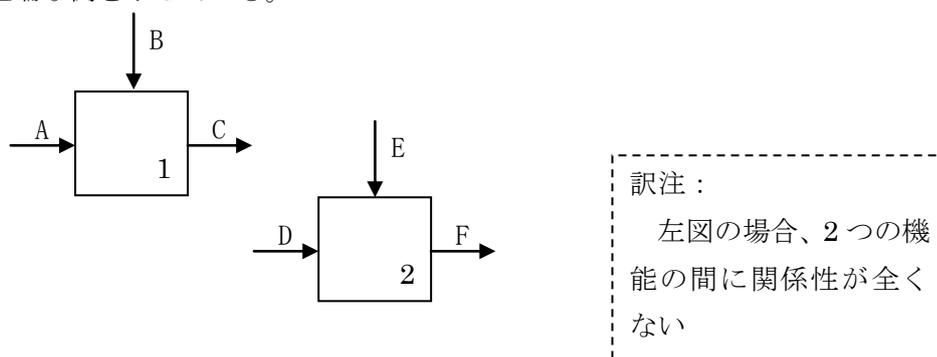
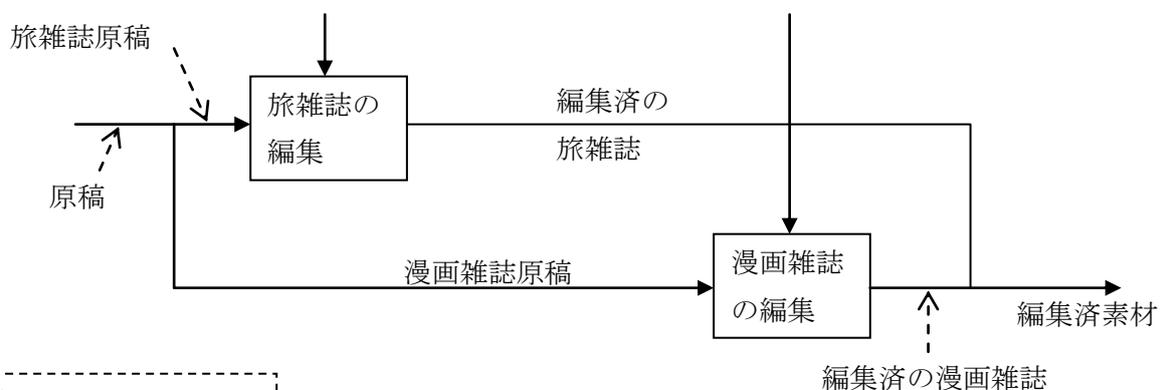


図 6-11 同時的結束

(1) 必然的 (Logical)

(訳注: 「論理的」という訳が一般的のようだが、意味わかんなくない?)

必然的結束 (Logical binding) は、共通クラスもしくは同じ要素セットとした方が合っている為にデータや機能をまとめてみたものの、その中に機能的関係性を見出すことができない場合に発生する。典型的な必然的結束を図 6-12 に示す。



訳注:
この場合、2つの機能を別々にしても、あまり変わらない

図 6-12 必然的結束

(2) 時間的 (Temporal)

データが同時に使用されているか、もしくは機能が直列ではなく並列に働くかすることで、その時点において機能が関連しているように見える為、時間的に束ねられた複数要素は集合しているかのように見える。ダイヤグラム内で良く見受けられる例としては、機能が全て「初期化 (initialize)」作業に関連しているものがある (図 6-13)。

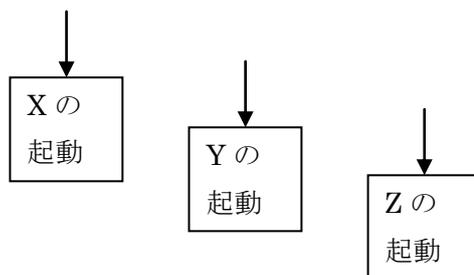


図 6-13 時間的結束

(3) 手続的 (Procedual)

手続的に束ねられた複数の要素は、あるプロセスもしくはサイクルの同じ部分で実行・運用されていることから、あたかもグループ化されているように見える。この結束の要となるプロセスの共通ユニットは、繰り返し (iteration) か決定分岐 (decision-branch)、または直線的に繋がった複数ステップ (a linear ordering of steps) であることが多い。手続的結束ダイヤグラムの例を図 6-14 に示す。

このダイヤグラムでは、出力 A と出力 B が下流の箱 3 で必要となる為、箱 1~3 がまとめられている。箱 1 と箱 2 は互いに直接的な影響を与えてはいないが、箱 3 のプロセスを通して間接的に関係している。

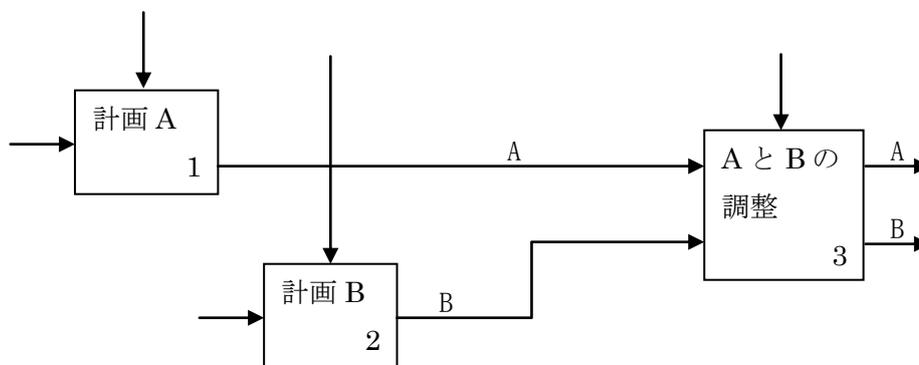


図 6-14 手続的結束

訳注：Communicational は一般的には「対話的」と訳されているが、ここでは関係性の特性から「伝達的」を用いている。

(4) 伝達的 (Communicational)

同一の入力もしくは同一の出力を行っている複数の箱をグループ化したダイアグラムは、伝達的な結束を表している。伝達的な結束は、これまで説明した結束の中で初めての、IDEF0で望ましいとされているレベルの結束である。その大きな理由の1つは、これまで説明してきたような結束レベルは、それぞれが何らかの問題を構造的に抱えているからである。伝達的な結束は、本質的な問題から独立した最低レベルの、プロセス要素間の関係性となる。図 6-15 に典型的な例を挙げる：

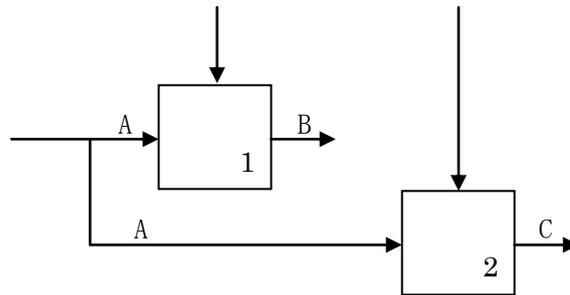


図 6-15 伝達的結束

(5) 連続的 (Sequential)

連続的な結束を持つダイアグラムでは、ある機能からの出力が、次の機能の入力データとなっている。連続的な結束を持つダイアグラム上の要素間の関係性は、データの因果関係 (cause and effect transforms) が接近したものである限り、これまで説明してきた結束レベルよりも相互関係性が高いものになっていることから、高く評価されている。図 6-16 は連続的な結束を示している。

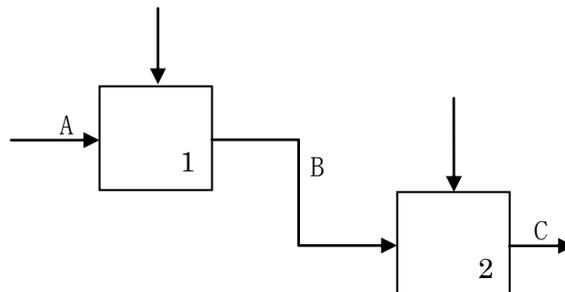


図 6-16 連続的結束

(6) 関数的 (Functional)

ある機能の全ての要素が、1つの機能もしくはその結果の性能 (performance) に寄与している時、そのダイアグラムは完全な関数的結束を示している。

純粹に関数的なダイアグラムには、連続的結束以下の弱い結束と関係する異質な要素が一切含まれていない。関数的結束を持ったダイアグラムを識別する為の方法の1つは、図 6-17 に示されたような、制御矢印を介して結びついた2つの箱を探す事である。

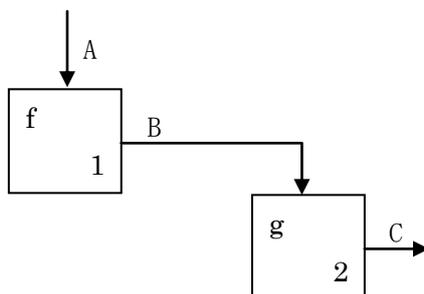


図 6-17 関数的結束

図 6-17 に示した、最も単純な種類の関数的結束に必要な条件を、数学的な表現で表すと、以下のような式になる：

$$C = g(b) = g(f(A))$$

別の便利な関数的結束のチェック方法として、消去法 (?clues-by-elimination) がある。以下のガイドラインは、著者と批評者が関数的結束とそうでないものを区別する際の助けとなるだろう。

- ・ダイアグラムを文章で記述する唯一の合理的手法が、重文 (訳注 : compound sentence、主語と述語が2組以上ある文) を使用する物か、もしくはコンマを含んだ文 (訳注 : コンマによって2つの文が連結されている) か、もしくは2つ以上の動詞を含んだ文を使用する場合には、そのダイアグラムは関数的結束未満の、恐らくは連続的か、伝達の、もしくは必然的結束となる。

- ・ダイアグラムを記述する文章に、「初めに」「次に」「その後」「そして」「始まる」「ステップ」「した時」「まで」「全てにおいて (for all)」といった時間に関する言葉が含まれて

いる場合、そのダイアグラムは恐らくは時間的結束もしくは手続的結束となるが、極稀にそうした言葉が連続的結束を指示する場合もある。

・ダイアグラムを記述した文章の述部 (predicate) に、動詞の後ろに 1 つの明確な単語が含まれていない場合 (訳注: 動詞の後ろに目的語が 2 つ以上か、目的語が無い場合) には、そのダイアグラムは恐らく必然的結束である。

まとめ (Summary)

これまでに説明した結束の種類を、図 6-18 にまとめている。重要なことは、IDEF0 開発者として「品質」のあるダイアグラムを作成するには、レベル 4 から 6 までの種類の結束 (伝達・連続・関数) で構成する事が最低限必要であるということである。著者は、これらの種類の結束の効果が最大になるように努力すべきである。

格付	結束種類	機能	データ
0	同時的	ランダム	ランダム
1	必然的	同一のセットもしくは種類の機能 (「全ての入力を編集」等)	同一セットか同一種類のデータ
2	時間的	同一の期間内の機能 (「運用の初期化」等)	ある期間内で使用されるデータ
3	手続的	同一フェーズもしくは繰返して発生する機能 (コンパイルの第 1 パス等)	同一フェーズもしくは繰返して使用されるデータ
4	伝達的	同一のデータを使用する機能	同一活動で作用されるデータ
5	連続的	同一データを連続的に変換する機能	連続した機能で処理されるデータ
6	関数的	合体して 1 つの関数として実行される機能	単一の機能に関係するデータ

図 6-18 結束のレベル

より望ましい

訳注: 理想的なダイアグラム

結合 (Coupling)

抽象化階層と遮蔽原理を用い、階層の上下左右で整理された判り易い接続が行われていること

結束 (Cohesion)

より機能的・有意的な結束を実現すること

6. 6. 6 IDEF での結合と結束の評価 (Assessing Coupling and Cohesion in IDEF)

ここで説明されている格付けの尺度 (rating scale) は、元々はシステムエンジニアによる設計選択の補助と、その選択肢となっている分解の相対的な望ましさを明確化する為に開発されたものである。IDEF0 では、これに似た尺度を、著者とコメンターによるモデル品質の保持を補助する目的で使用する事を提案している。

こうした目的を考慮すると、幾つかの追加の所見 (観察) が必要となる :

・ IDEF0 との妥当性において、「結合 (coupling)」と「結束 (binding)」は本は同一のコンセプトをそれぞれに別に解釈しただけのものであるが、モデルの異なった構成物に対して適用されていることから、違ったものになっている。

「結合 (Coupling)」は、ダイアグラム間、もしくは親ダイアグラムの箱間の、接続 (connection) の複雑性を比較する為に使用する事が可能である。

「結束 (Binding)」は、あるダイアグラム上に存在する箱同士の結びつき (link) の強さの評価や、あるダイアグラムの親箱の結束力 (cohesiveness) の計測に使用する事が可能である。

訳注 : わかりにくい、以下のような事ではないかと考えられる :

結合 :

ダイアグラム上の機能 (箱) 間の組み合わせの複雑性。組合せが判り易いかどうか

単純で判り易い組合せ :

- ・ 遮蔽原理 (モジュール化) の利用
モジュールで遮断することにより、直接的な組合せ数を制限する
- ・ 階層構造的抽象化
階層が進む毎に詳細化される構造を採ることで、組合せを下層へと先送りする
(モジュール化との合わせ技で実現)

遮蔽原理と階層的抽象化を同時に用いる事で、対象物、組織、ワークフローの 3 つを同じ構成にすることが可能となり、より一層判り易くなる

結束：

同一平面上の機能（箱）間の関係性が機能的、有意的であること

1つのダイアグラム上にある機能の関係性が機能的・有意的

ダイアグラム＝親ダイアグラム上での1機能が十分に整理されている

不要な機能や接続を、本来あるべき別のダイアグラムへと移動、

もしくは、不要な機能や接続を排除

・望まれない接続（connection）を識別するための簡単な手がかりは、親ダイアグラム上の結合（coupling）の固さに注目することである。親ダイアグラム上の結合が固い場合、通常は子ダイアグラム上での結束が低く（low-valued binding）、非結束（incohesion）になっている。著者は、親ダイアグラムを再分割し、複雑な結合を削除することで、モデル品質の改善と、親－子ダイアグラムの組み合わせをより明快なものとするのが可能となる。

訳注：

親ダイアグラム上で整理がされていないプロセスを分解すると、当然に下層は無意味な関係が多くなってしまふ、ということか。この事実の証明は難しいが、少なくとも親ダイアグラム上の結合が緩やかな場合、その階層のプロセスは十分に整理されている為、分解された子ダイアグラムも整理された、綺麗なものとなることは確かである。

・結合（coupling）と結束（cohesion）の格付けの値は足し合わせる事が可能である（additive）。多くの接続（connection）は、2つ以上の分野での格付尺度によって定義され、そして複数の値を付与してゆく事で接続の評価精度を高める事ができる。例えば、ある1つのダイアグラムで「連続的（sequentially）」結束と「伝達的（communicationally）」結束のどちらもの性質が見られる時、このダイアグラムの結束の品質は、単純な伝達的結束だった場合よりも良くなることになる。一方で、ある結合に「制御（control）」結合と「データセット（record）」結合のどちらもの性質が見られる場合、単純に制御結合もしくはデータセット結合のみである場合よりも、（より結合が強固なものになるこということで）悪い結合となるのである。

7. IDEF モデリングの為のデータ収集 (Data Collection for IDEF Modeling)

7. 1 はじめに

どのようなシステムの分析・設計においても、まずはシステムもしくは対象物の実態を手元に収集、確認する必要がある。実態に関する情報の入手先は、以下に挙げたものも含め、数多く存在している：

- ・ 内容一覧表や索引を使用して必要となる情報を追いつつ、既存のドキュメントを読む
- ・ 既に存在していれば、運用されているシステムを観察する
- ・ アンケート (questionnaire) 等の手段を利用して、幅広く情報を収集する
- ・ 必要な知識を持っていると思われる「専門家 (expert)」と話をする
- ・ 著者の既知の知識も活用する
- ・ 仮説を立て、これを読んでもらい、読者に事実に近くなるような情報を提供してもらう

こうした手段の中でも最も重要なものは、専門家と直接顔を合わせて話をするという事である。全ての情報が記述されている事は有り得ないし、アンケート内の予め考えた質問も間違っている事が多い。

専門家から得られた情報は、インタビュープロセスを通して定型化される (formalized)。このインタビュープロセスを1つ1つ段階を踏んでいく事で、インタビューアーの既知の情報が専門家に悪い影響を与えないように、インタビューを行うことが可能となる。

インタビューで重要となるのは、得られた情報を記録することである。記録は、正式ではないノートへのメモ、活動・データリスト、正式な機能マトリクス表、ダイアグラムスケッチ、等の手法で行われる。

7. 2 インタビュープロセス (The Interview Process)

インタビューの目的は、分析作業にとって重要な専門的知識 (**expertise**) を持っている個人から、情報を集める事である。IDEF プロジェクトの分析フェーズでは、以下の 4 つの種類インタビューが用いられている：

(1) 実態の探索 (Fact finding)

現在運用中の物を理解する為に、実態の探索を行う。このインタビューで、現在の運用モデル (**Current Operations Model**) の内容の作成や、既存環境の理解促進を行う。

(2) 問題点の識別 (Problem Identification)

将来の要求物の準備を補助する為に問題点の識別を行う。このインタビューで現在の運用モデルを確認し、将来の運用モデルを作成する。

(3) 解決法の議論 (Solution Discussion)

将来のシステムの能力 (**capability**) を考慮する為に、解決法についての議論を行う。このインタビューで、将来の運用モデルの内容を作成する。

(4) IDEF 著者/コメンター間の話し合い (IDEF Author/Commenter Talk Session)

このインタビューで、IDEF モデルの構築中に表面化した問題の解決を行う。

インタビューの種類を以上のように分類 (**identify**) する理由は、現実のインタビューにおいては、これらを組み合わせて行うことになるからである。インタビューの回答者は、対象となるシステムについての実態 (**fact**) を問題点という形でインタビューアーに伝えたり、問題点に対する解決策によって問題点を識別していることがある。インタビューアーがインタビューから情報を最大限に引き出すには、こうした回答者の意見を常に分類 (**classify**) しなければならない。

7. 3 インタビューキット (The Interview Kit)

インタビューの記録には、「標準的な」インタビューキットの使用が推奨されている。キットの記録はインタビューファイルとして保存され、関係者に配布される。配布先には分析チームの他のメンバーも含まれるが、内容の修正、追加、削除を行ってもらうために、インタビュー回答者にも送付されることがある。インタビューキットは以下の物で構成されている：

1. 表紙 (キットカバー)
2. インタビューと記録のフォローアップ
 - a) インタビューアーの名前 (IDEF 著者名)
 - b) インタビュー日時 (IDEF ダイアグラム日時)
 - c) インタビュー期間 (開始時間、終了時間)
 - d) 回答者名
 - e) 回答者の役職名、所属
 - f) 回答者の連絡先 (電話番号、内線番号)
 - g) 追加情報ソース
 - 1) ドキュメント名と保存場所
 - 2) 他のインタビューアー：名前、役職、所属、住所、電話番号
 - h) 情報の本質的要素：インタビュー対象の重要項目の概要
 - i) インタビューで対象にできなかった範囲もしくは質問についてのフォローアップ
 - j) プロジェクト用語集に追加すべき新しい用語
3. 活動 (activity) リスト・データリスト
4. インタビューの質問予定表 (Agenda)
(インタビューの準備中に作成した物、これについては後に触れる)
5. インタビュー中のメモ、ダイアグラムの粗いスケッチ

7. 4 インタビューのガイドライン

インタビューを上手く運ぶには5つの段階がある。各段階では最大の情報を引き出し、それを最短時間で記録するようにしなければならない。

- ・準備
- ・初期化
- ・インタビュー
- ・終了（? Termination）
- ・仕上げ

インタビューの各段階には、実行すべき明確な基本行動がそれぞれ存在する。それに加え、各段階毎に、インタビュー回答者との間で、専門的な話を行い易い雰囲気と信頼関係を作り出す助けとなる心理学的な支援（psychological aid）も存在している。

7. 4. 1 インタビューの準備

インタビューを行う前に、インタビューで聞き出すべきことが何かを明確にしておくことで、より整理された効率的なやり取りを行う事が可能になる。インタビューの準備には、以下のような作業がある：

(1) インタビューアーの選択

- a) 担当範囲から（area of responsibility）
- b) 他からの推薦
- c) 組織の様々な階層から：
上層の人は全体像を、下層の人は詳細な情報をそれぞれ提供し、
そして中層の人はそのギャップを埋めてくれる

(2) インタビューの約束をとる

- a) 短時間で（30分から1時間）
- b) 昼食の直前や、午後遅くは避ける
- c) インタビュー目的の説明
- d) インタビューアーの役割の説明

- (3) 仮の質問予定表の作成 (Establish Tentative Agenda)
 - a) 質問項目 (topical area) : インタビューの基礎として使用される
(「広範囲の一般的質問 (broad general question)」の準備に役立つ)
 - b) 具体的な質問 (specific question)
- (4) 背景情報が適用可能なものか、チェック (review)
- (5) 専門用語が適切なものか、チェック (review)
- (6) 他のインタビューアーとの間の協調
 - 過去のインタビューファイルをチェックし、回答者が以前にインタビューに回答しているかどうかを確認する。フォローアップインタビューの場合は、フォローを行うインタビュー結果を調査する。
- (7) インタビュー記録とフォローアップに足りない部分を追加する
- (8) インタビューの質問予定表としてまとめる

7. 4. 2 インタビューの初期化 (initialization)

この段階は、インタビューアーと回答者との間に、親しい関係を結ぶことを目指している。インタビューの開始時において、回答者に礼儀を示すことが可能な時間は通常は短いものであるが、回答者のやる気を向上させてインタビューを補助する為には、重要な時間である。この段階における主要な項目は、以下の通りである：

- (1) 回答者に対して、名刺などを用いて判り易く自己紹介を行う。(名刺により、インタビューアーの名前をどのように書くのか、もしくはどのように発音するのかといった疑念を取り払い、ありがちな回答者の困惑の原因を排除する)
- (2) インタビューの目的を設定する
 - a) 先に提供した情報を、更に詳細なものへと展開する。
 - b) インタビューの視点 (point of view) を設定する。
インタビュー種類 1 から 4 までの何れかを基本として用いる。
 - c) インタビューの目的を設定する。これはフォローアップインタビューの

場合でも行う。

(3) メモの許可をとる。回答者から機密保持を要求される可能性がある。

(4) 専門家と著者の間の関係を構築する。これにより、インタビューによりノウハウが流出してしまうとか、仕事が危険にさらされるといった、回答者の心配を解消する。

(5) 広範囲の一般的な質問 (**broad general questions**) からインタビューを始めることで、回答者の発言を促す。この質問は、質問予定表 (**agenda**) で識別したインタビュー質問項目 (**topical area**) を基にして行う。

(6) 回答者が、適切な情報を提供可能か評価する。準備しようとしている **IDEF** モデルの対象範囲 (**stage**) に対して、回答者の情報が一般的過ぎたり、詳細過ぎたりした場合には、回答者が寄与可能な能力を査定する。その結果、インタビュアーにとっても回答者にとっても時間の無駄にしかならない場合には、インタビューを中断する。

(7) 質問予定表 (**agenda**) を完全なものにするための具体的な質問 (**specific question**) をまとめ (**formulate**) 始める。

(8) 書くこと。無駄話をしない。 (**Write, don't talk**)

7. 4. 3 インタビューの実行

インタビューをしつつ質問を考えることは難しいが、考慮すべきガイドラインを認識しつつインタビューを行う事は可能である。1つ目のガイドラインセットは入手された情報の品質保証 (**qualification**) に関するものであり、2つ目のセットは情報の流れの促進 (**stimulate**) に関するものである。

情報の品質保証 (**Information qualification**) :

人間の理解力は、話す時の方が (聞く時の) 2倍になっている。インタビューにおいて危険な事は、聞き手が話されている内容でなく、返答すべき内容の方に意識を集中させてし

まいがちであることだ。

インタビュアーが何が話されているかについて考え、提供されている情報へと集中する為に役立つ、一連の質問を以下に挙げる：

- ・議論中の主要項目に対して、実態を支援する為に何が提供されているのか？
- ・情報はどれくらい新しいか？
- ・情報はどれくらい完全か？
- ・話されている事を本当に理解しているのか？
- ・詳細レベルはインタビューの目的に対して適切か？
- ・省略された範囲は存在しないか？
- ・この情報は、別の人と議論したものではないか？
- ・この情報の重要度は？
- ・テーマから外れたことを議論していないか？
- ・インタビューの視点が変化していないか？

情報の流れの促進 (Information Flow Stimulation) :

以下に挙げるガイドラインのセットは、回答者から信頼に足る情報を最大限に引き出す為に使用されるものである：

- ・無関係なコメントや雑談は最小限に止める。インタビューは情報を得るために行う為の物であり、友達を作ったり自身のアイデアを売り込むためのものではない。

- ・何が問題点かについて、回答者が正しく認識しているかどうか注意する。正しく理解できていないことが原因で、回答者がインタビューアーに対して打ち解けていない場合がある。

- ・回答者に考える時間を与える。こちらから答えを提案したり、別の質問を行わない。インタビューを一時中断することで、回答者に情報の重要な部分を思い起こさせる。

- ・思考の流れ (train of thought) を崩すような、無関係な気の紛らわしとなるものを避ける。可能な限り、インタビューは回答者の日常から切り離れた場所で行う事。

- ・インタビューに満足していない、もしくは打ち解けていないといった兆候である、回答者の気の散漫に注意する

- ・得られた情報が事実 (fact) なのか、意見 (opinion) なのかを判断してみる

- ・発言された情報を別の言葉で言い換えたり、まとめ直したりすることで、より精度の高い情報にする

- ・議論されている主題と、回答者の背景・関連との繋がりを確認する。回答者の組織と既存システムに対する関係性を知ること、回答者の意見 (remark) に対する貴重な洞察を得ることが出来る。

- ・皮肉 (sarcasm) やユーモア (humor) には立ち入らない、近寄らない。

- ・他の人とのインタビューについて、触れる事も議論する事もしてはならない。

- ・回答者によってなされた質問は全て記録する。ユーザー組織 (user-organization) の運営や計画、個人情報に関する物を除き、インタビューアーは回答者の全ての質問に答えるべきである

- ・回答者の話に興味を示すこと。

- ・議論している主題の、不明瞭で難しい面に集中すること。既に明確となっている事は回避すること。

- ・言葉の不一致、言葉の誤用には気を付ける。不明瞭な用語や、疑問の残る用語につ

いては改めて定義してもらい、プロジェクト用語集に登録する。

- ・話している内容に事実の裏付けが無くとも、回答者に反論しない。そうした不一致の解決にはキットサイクル (Kit Cycle) を用いる。

- ・謙虚 (humble) であれ。専門家は回答者であり、インタビューアーではない。

- ・同意された時間内で完全に対応できなかった項目については、先延ばしすべきである。インタビュー時間を延長するのではなく、別にインタビューの約束を取るべきである。

- ・同一の主題に対して異なった意見が存在することを理解するように努める。IDEF を用いることで、こうした異なった意見を示し、不一致を解決してゆく。

- ・適切な自由回答質問 (open ended question) を用いて回答を促進する。

7. 4. 4 中断 (Termination)

以下に挙げる 4 つの内の、何れかの状態に至った場合には、インタビューを中断すべきである。

- a) インタビューで得られた情報が適切な物で無かった
- b) 終了時間になった
- c) インタビューアーが情報を処理しきれなくなった
- d) インタビューアーと回答者とが喧嘩してしまった

以上のような理由でインタビューが中断した場合、中断期間中に以下の項目について考慮すべきである。

- ・インタビューは突然に終了すべきではなく、日常的な会話を数分交わした後に終わるべきである。
- ・インタビューの主要な点について要約する。
- ・延期された、もしくは対応できなかった範囲を、認識しておくべきである。
- ・必要なら、フォローアップインタビューを行うべきである。
- ・他にインタビューすべき人を回答者に推薦してもらう
- ・配布に先立ち、回答者にインタビューメモの見直ししてもらうのであれば、この中断期間中に行ってもらう
- ・忙しい中にも関わらず時間を割き、インタビューに参加してもらったことに対して、回答者に感謝すべきである

7. 4. 5 仕上げ (Finalization)

この段階では、インタビューによって得られた情報がしっかりと記録され、プロジェクトチームに対して伝達されているかを確認することを目的としている。インタビューの仕上げ作業にはインタビューキットを用いる。回答者によってメモを許可されなかった場合には、インタビューアーはインタビューの中断を待って速やかに議論された要点を書き出しておくべきである。インタビューの仕上げ作業には、以下のものが含まれている：

- a) 追加が必要な情報ソースの識別
- b) 情報の本質的要素の要約
- c) プロジェクト用語集に必要となる新しい用語の識別
- d) インタビューの延期、もしくはインタビューで対応できなかった場合に必要となる、フォローアップ用の質問と範囲をリストにまとめる。

- e) 活動リストとデータリストを完成させる
- f) 見直し期間中に、あらゆるメモを読み直し、あらゆる情報を思い出す。
- g) 得られた情報を反映した IDEF ダイアグラムを下描きしてみる
- h) 仮定のままのものや、不明瞭なものを識別する
- i) インタビューキットの公布と配布
- j) インタビュー内で触れられた人の名前、専門分野、電話番号、住所などを書き出す

訳注：本文はここまで

8. IDEF0 用語集 (glossary)

用語	(英語)	説明
矢印	Arrow	データを表現した線。発生源（点無し）と使用先（線分の端部に点）
著者	Author	IDEF モデルを作成する人
箱	Box	名前と番号を持った長方形で、活動（Activity）の表現に使用する
枝	Branch	分岐（fork）もしくは合流（join）
C 番号	C-Number	IDEF ダイアグラム用紙の右下隅に書かれた時系列番号で、これによって履歴と著者を追跡することができる。C 番号は詳細参照表現（Detail Reference Expression）として使用されている
呼び出し	Call	箱の下辺から外（下方向）へと出ている指示矢印（pointer）で、この箱が別の箱の分解によって詳細化されていることを表している
コメンタ ー	Commenter	IDEF 技法の訓練を十分に受け、付番システムを使用した定式的なコメントを提供し、時には技法そのものの適用内に有る欠点を指摘する人
背景	Context	モデルが運用される直接の環境；モデルの範囲。IDEF0 では箱、特に A-O ダイアグラム上の箱の周辺の矢印。また、IDEF フォーム上の、親ダイアグラムと親箱が識別できる小さな箱も。
制御	Control	IDEF0 箱の上辺に関係する矢印のクラス。変換に対するガイダンスを提供する
データ	Data	物や状態、情報といった、名詞句によって命名可能な物。通常はクラス（人、等）を参照するが、1つのインスタンス（山田太郎）を意味している。
詳細参照 表現	Detail Reference Expression	IDEF0 箱の下に書かれた C 番号もしくはノード番号で、その箱の詳細がどこに書かれているかを示している。
下描き	Draft	「作業中」と「推奨」の間の、IDEF ダイアグラムの承認レベル
専門家	Expert	モデル化された部分の、実世界のシステムに詳しい人。モデル化された部分情報源もしくは批評者の役割を負う

FEO	FEO	通常の文法が適用されていない、参照のみの目的で使用されるダイアグラム
分岐	Fork	IDEF0 の（発生点から使用点に向かう）矢印が 2 本以上に分岐する点
機能	Function	動詞句によって記述される活動（activity）であり、何が実行されるべきかを識別する
用語集	Glossary	IDEF モデルで要求されるもので、言葉や句のモデル上での使用方法を定義する
ICOM	ICOM	ICOM コードシステムのこと。 Input、Control、Output、Mechanism の頭文字を用いる。 矢印のラベルに用いる。
IDEF の役割	IDEF Role	IDEF プロジェクトでの位置づけ。著者、専門家、コメントー、読者、司書
入力	Input	IDEF0 箱の左辺に入る矢印のクラス。通常は出力と対になる。
合流	Join	IDEF0 矢印が他の矢印と合流し、1 本の矢印となる点
キット	Kit	データの一部もしくは完全なデータセット、見直しの必要なモデルを含む、標準的なダイアグラムパッケージ。キットサイクルを参照
キットサイクル	Kit Cycle	モデルの開発中に同僚もしくは専門家の批評を得る為に用いられる、公式の手続き
ラベル	Label	IDEF0 矢印の名前
司書	Librarian	キットの送付と追跡、プロジェクトファイルの取扱を担当する人
機構	Mechanism	IDEF0 箱の下辺に入る矢印のクラス
モデル	Model	システムについての質問に答える為に使用する事が可能な、あるシステムの表現物（意味不明）
モデル作成者	Modeler	著者（author）と同義
ノード	Node	従属部分（下層部・子）の基となる、もしくは中心となる点。 IDEF0 の箱もしくはダイアグラムに関係する番号。（個々の活動は、箱とダイアグラムに、それぞれ表される）
ノード一覧	Node List	多くはインデントを用いて表された、1 つの IDEF0 モデル内に有る全てのノードの一覧。（名前順ではなく）「構成（? outline）」順で書かれている。

ノード ダイヤ グラム	Node Diagram	1 つの IDEF0 モデルのノード間の関係性を表現した図
注釈	Note	通常の手法で取り扱われるコメント以外で、IDEF0 ダイアグラムに対して事実 (fact) を記録したコメント。もしくはダイアグラムの読者もしくはコメンターによるコメント
出力	Output	IDEF0 の箱の右辺から出ている矢印のクラス。IDEF0 変換の結果である。
親	Parent	子ダイアグラムによって詳細が説明されている箱が存在するダイアグラム
プロジェ クト	Project	IDEF モデルを作成する目的となる、組織化された任務
プロジェ クト 管理者	Project Manager	完成品について最終的な責任を負っている、プロジェクトのメンバー
公布	Publication	IDEF0 ダイアグラムの最高の認証レベル
目的	Purpose	モデルの作成目的を説明した短い文章。モデルの存在理由を明確にする。
読者	Reader	モデルの一部もしくは全体を見る、IDEF 技法の訓練を受けていない、もしくは部分的にしか受けていない人。読者もコメントを行うことがあるが、そのコメントは定式化 (structured) されていない。ダイアグラムのワークスルー (walkthrough) に参加する個人もしくはグループは、通常「読者」としてグループ化される。
推奨	Recommended	IDEF ダイアグラムの上から 2 番目の承認レベル
技術 委員会	Technical Committee	モデル開発を指導し、その内容を承認する役割を担ったグループ
説明文	Text	ダイアグラムフォームとは分けられている、IDEF0 ダイアグラムについての言葉によるコメント
トンネル 矢印	Tunnelled Arrow	IDEF0 矢印の端部の 1 つが、親でも子ダイアグラムでもない他のダイアグラム上の矢印に繋がっているもの
視点	Viewpoint	表現すべき背景 (context) 内で発生し得る事実のサブセットを、定義する為の試み (attempt)。描写されるものを認知する人の見地で表現されている事が多い。
作業中	Working	IDEF ダイアグラムの最低の承認レベル。全ての IDEF ダイアグラムは、この「作業中」の状態から開始される。

9. IDEF0 用語索引

(略)

添付資料 A

(略)

訳注：この IDEF0 資料の大本となる資料の概略。第 1 章から第 11 章まであり、IDEF0 は「第 4 章 機能モデリングマニュアル (IDEF0)」らしい。

添付資料 B

(略)

上記のドキュメントを請求する為の発注書。

翻訳後の印象、まとめ等

この翻訳は、1982年版の IDEF0 に関する資料である。最新の IDEF0 に関する資料は別
にあり、微妙に異なっている。といて改めて翻訳し直す気にはならない。

CMMI と同じで、理想的な構成を明示している。つまり、IDEF0 で定められたルールに
従った物こそが、理想的なシステムということである。また CMMI と同じく時間軸を持っ
ており、初期段階の出来損ないのシステムから完成した理想的システムに至るまでの様々
な段階についても定義可能になっている。いきなり完成品を作るのではなく、現在ある物
や取り敢えずでっち上げた下描きを、時間をかけつつ理想的な姿へと変化させて行くこと
を前提としている。ルールと言う形で理想的な方向を示しているので、それに従っていれ
ば、理想的な姿へと至るといいう仕組みになっている。

プロセスフローだけではなく、それを運用する組織、プロセスが対象とするオブジェク
トも同時に、同じ構成で構築される仕組みになっている。プロセスとオブジェクト、組織
が同一構成であるべきなのは、当たり前といえば当たり前だが。

複雑性をいかにマネージメントしてゆくかが、カギになっている。
色々な関係者全員が十分に理解できないものが、まともに運用できるわけがない、という
明瞭な視点に基づいている。またスマートな物は理解しやすいものであるということであ
る。

ただ一方で、自身のプロセスに関心の低い日本において使用するには、複雑で難しすぎ
る。もっと粗雑で良いので、もっと取っ付きやすいフローマネージメントシステムが必要
である。