データ・ベースへの要望について†

石　田　喬　也††

1. 背　景

1960 年代初頭にその源を見出すことができるデータ・ベースの思想は約 10 年のあれやこれやの模索の時を経て，ようやく有効なツールとして広く認識されるに至ったと思われる．そしてそのひと通りのパターンが計算機メーカやソフトウェア会社で作成された各種のソフトウェアに現われたと考えられる現在は，今後データ・ベース管理システムをどのような形に完成していくべきかを一度立ち止って考え直してみる絶好の機会となっているのではないだろうか．その動きのひとつの表われとして，ここ 2，3 年の間に既存の各種の代表的データ・ベース管理システムを比較して評価することを目的としたレポートがそれに種々の異なる名称を与えて多数報告されている[1),2),3)]．それらの中でも特に 1969 年 5 月に 9 種の代表的システムに対するサーベイをまとめ[4)]，2 年後にその続編として 10 種のシステムに対する機能別の詳細な比較解析レポートを公けにした[5)]，CODASYL Systems Committee の仕事は最も大がかりなものであり，非常に高い評価が与えられている．CODASYL Systems Committee のこの一連の仕事は，かつて事務処理用共通言語として COBOL を生み出し，1966 年に ANSI で標準言語としての認定を受けたのと同じように，データ・ベース管理システムについてもまず共通に受け入れられる言語ならびに機能を提唱し，ゆくゆくはそれが世の中の標準とされることを目的としたものである．

一方この動きとは別に CODASYL ではその Programming Language Committee の Data Base Task Group (DBTG) において，COBOL 言語をデータ・ベース処理の分野にまで適用するためにはその言語仕様をいかに拡張すべきかという問題提起に基づき，データ・ベース管理システムのあるべき姿の具体的イメージが検討されていることにも注目しなければならない．その検討結果の集大成は DBTG レポートとしてまずその初版が 1969 年 10 月に発表され，ついでその改訂版が 1971 年 4 月にまとめられている[6)]．この DBTG レポートが提唱するデータ・ベース管理システムは，基本的に COBOL 言語をホスト (Host) 言語とする手続指向 (Procedure Oriented) 型に偏したものである．また，IBM を代表する DBTG の委員が，このシステムはデータの独立性の実現が不完全であることを主要理由に挙げてこのレポートを公表することに反対の見解を表明している[7)]，というような複雑な事情もあるが，汎用データ・ベース管理システムに対する具体的仕様の伴った提唱はこの DBTG レポートがはじめてであり，今後の発展のひとつの方向を指し示すものとして高く評価されるべきであろう．

これら 2 種類のアプローチが汎用データ・ベース管理システムの実現を目指す言わば地道な積上げ努力であるのに対し，より未来に焦点をあわせてデータ・ベースに対するビジョンを描くことから始めて，それを実現する過程として現在を位置づけようとする動きが一方に見出される．著明な IBM ユーザ・グループである SHARE と GUIDE のそれぞれのデータ・ベース要望グループが共同研究の形で 1969 年 5 月以来おこなってきた仕事がその動きを代表する立場である．その研究成果は 1970 年 11 月にデータ・ベース管理システムに対する要望集という形でまとめられている[8),9)]．この要望集は，ユーザにとって，真に効果的なまた真に役立つデータ・ベース管理システムとはいかなる機能を持つものであるべきかを明らかにすることを意図したものである．これらの要望が単にひとりやふたりの思いつきの意見でなく，ほとんどあらゆる種類の企業や大学，政府機関をそれぞれ代表する 40 人以上のメンバが約 2 年間という長い時間を要して組織的に検討された結果の集大成であるという事実になによりも注目すべきであろう．すなわちこの要望集の中にはデータ・ベースに期待するほとんどすべてのユーザの声が凝集されていると考えなければならない．従来のアプローチがその時点のハードウェアならびにソフトウ

† On the Requirements for Data Base, by Takaya Ishida (Kamakura Works, Mitsubishi, Electric Corp.)

†† 三菱電機鎌倉製作所

ェア技術で可能な範囲内で実用可能なパフォーマンスを発揮できるもの，と限定してシステム作りをおこなってきたのに対し，この要望集はそのような制限にとらわれずむしろ，真に価値あるデータ・ベース・システムを作り上げるために必要とされる，ハードウェアを含めた技術水準の発展の方向を示唆する立場をとっている．また単なる要望の寄せ集めに留まらず，あるべき姿の論理的体系を提唱している点が重要である．そのために各種要望の記述は抽象的ではあるが，それがために要望しているシステムのイメージまではぼやけることはなく，その要望に基づく具体的なシステム設計を可能ならしめていると思われる．

以上に述べたデータ・ベース・システムの発展を目指す努力はもちろんそれぞれにおいて価値がある．今後もそれぞれの立場からのアプローチを併行して進め，それらの成果を結集することによりはじめて，真に望ましいデータ・ベース・システムの実現が達成されるものであると考えられる．しかしその中でも特に大切なことは，このようなデータ・ベース・システム発展の努力が正しい方向に向っているものかどうかという検証であり，そのためにはデータ・ベースに本当に要望されているのは何なのかを明確に把握しなければならないであろう．

そこで以下では上述の GUIDE/SHARE データ・ベース要望グループのデータ・ベース管理システムに対する要望集（以下では G/S 要望集と略記）を取り上げ，一体ユーザはデータ・ベースに何を要望しているかという問題を中心にして解説を試みることにする．

2. データ・ベース管理システムに期待する役割

G/S 要望集が将来のデータ・ベース管理システムに対して何を期待しているかは，下記の 7 点に要約することができる．

（1） アプリケーション・プログラムのロジックを，データがどこにどのような形で，またそのアプリケーションにとって用のないデータとどのような関係を持って，保持されているかということから完全に浮いたもの（Data Independence）であらしめること．その結果，データの物理的属性やフォーマット，データ構造等が変更されてもアプリケーション・プログラムを作り変えることは不要であり，またプログラマはデータのアクセス技法にわずらわされることなく，本来の仕事であるデータ処理手続にのみ専念することができる．

（2） データ間の複雑にからみあった論理的関係を，システムがアプリケーション・プログラムに代って自動的にさばいてくれるものであること．

（3） データの世界を論理的レベルと物理的レベルに分けることにより，物理的にはできる限りデータを冗長に保持しないよう管理すること．

（4） 保持しているデータの完全性（Integrity）――データの内容ならびに関係が常に正しく維持されること――を完全に保証する管理をおこなうこと．

（5） 保持しているデータの安全性（Security）――データのプライバシーの保護――を完全に保証する管理をおこなうこと．

（6） システム・パフォーマンスを監視し，測定する機構を持つことにより，システム効率の最適化を図り得るものであること．

（7） 現有の，あるいは今後の過渡的時期に作成される，プログラムやデータを新しいシステムでも極力そのまま使える互換性を有すること．

上記（1）から（5）までの項目はデータ・ベース管理システムの基本条件というべきものであり，従来のシステムでも一様に強調されてきた点である．しかし従来のシステムではいずれも，システムの実際構築上の妥協あるいはハードウェア上の制約のため，これらの基本条件をかなりの制約の下で満足しているにすぎない．G/S 要望集はそれを不満とし，これらの基本条件を完全なレベルで実現することを要求している点を注目しなければならない．

（6）番目の項目としてパフォーマンスの問題を重視している点は，充分なる実用システムとしての期待からは全く当然の要求とは言え，安易なシステム開発に対する警鐘となっている．まさにその点がこの要望集の要望を満足するシステムを近い将来（恐らく 1980 年頃まで）に作り得ることは不可能であろうという推察の最大の論拠となっている．

最後に互換性の問題が挙げられていることは，この要望集が長期的視野に立つことからくる必然である．この点も従来のシステムでは古いシステム（データ・ベース以前のシステム）との互換性の問題まで充分配慮がおよばなかったことを考えれば，きわめて価値ある指摘と考えられる．G/S 要望集を満足するシステムに移行できるまでには，非常に長期間にわたるステップ・バイ・ステップのアプローチがとられるであろう．

その間に旧システムでおこなってきた業務を中断することなく新システムへの移行をスムーズにおこなえるためには，新しいシステムに相当程度の高い互換性能力が必要とされる．さらにまた将来起り得るであろう技術革新に際しても既存のプログラムやデータに最小限の影響を与えるに止める配慮を要望している．

3. システム構成に対する要望

G/S 要望集では，図1に示すように，データ・ベース管理システムは機能的に，リクェスタ (Requestor) とデータ・ベース・アドミニストレータ (Data Base Administrator, 以後 DBA と略記) ならびにデータ・ベース・マネジャー (Data Base Manager, 以後 DBM と略記) という構成要素からなるという考え方を出発点としている．

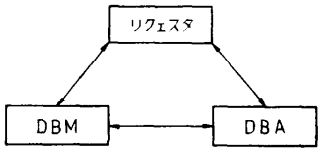

図 1 データ・ベース管理システムの機能構成

リクェスタはデータを使用することを望む各個人であり，DBA は各リクェスタのデータ使用に対する統括管理を受け持つ特定の人間あるいはそのグループである．一方 DBM はリクェスタならびに DBA から指示されたようにデータを受け渡しする役割を果す，ハードウェアとソフトウェアの融合体として定義される．データ・ベース管理システムがこの3者から構成されるという考え方は最近では一般的になりつつあるが[5),6),10)]，従来のシステムでは特にユーザをリクェスタと DBA に明確に分離する思想が充分でなかった[5)]と考えられる．その意味は DBA が果すデータ定義等の仕事をユーザの誰かがおこなう必要性は認識されているが，そのユーザをシステム内で特別に特権づけ専業化させるという配慮が欠けていることである．G/S 要望集では DBA 機能の明確な分離がデータ・ベース管理システムを成功させる鍵と考えており，今後は DBA の存在がシステム内で明確に位置づけられているもののみがデータ・ベース管理システムの範疇に入れられる方向に向うものと思われる．

一方 IMS[11)] においては DBA に相当するシステム・オペレーション (System Operation) 機能を司どる存在に加えて，システムの運行を絶えず監視し，システム故障発生時に適切なリカバリ処置を施すなどのシステムを円滑に運行せしめることを責務とするマスター・ターミナル・オペレータ (Master Terminal Operator) の存在を用意している．G/S 要望集においてこのマスター・ターミナル・オペレータの必要性が明確に打出されていないのは，IMS がデータ・ベースとコミュニケーションの統合システムであるのに対し，G/S 要望集 (ならびに従来の大部分のシステム[5)]) はデータ・ベース管理のみに注目している相違によると思われる．

3.1 リクェスタについて

リクェスタに相当するユーザをさらに，データ使用に際して手続指向型言語を使用するタイプをアプリケーションプログラマ，問題指向型言語を使用するタイプをノン・プログラマ (Non-Programmer)，トランザクション指向型言語を使用するタイプをパラメトリック・ユーザ (Parametric User)，という3種類に分ける見方があるが[5),10)]，G/S 要望集においてはリクェスタとして第1のタイプ，すなわち手続指向型言語を使うユーザのみが主として対象とされている．

リクェスタは COBOL, PL/I 等のホスト言語とデータ・ベース・コマンド言語 (Data Base Command Language, 以後 DBCL と略記) を使ってアプリケーション・プログラムを作成することにより，任意のデータ処理をおこなうことができる．DBCL は DBM にデータ・アクセス要求を出すためのコマンド群であり，従来の手続指向型システムにおけると同様，基本的には，ファイルのオープン／クローズ，ならびにデータの検索／更新／追加／消去，用のそれぞれのコマンドが用意されている．DBCL に対する要望として特に注目すべき点は以下に示す通りである．

（1） 上記基本コマンドを任意に組合せた複合コマンドを自由に定義して使用できること．

（2） データの選択条件を，NEXT, LAST, AND, OR, EQUAL TO, GREATER THAN 等々のシステム定義シンボルばかりでなく，任意の関係式や論理式あるいは検索プロシジュアを定義して使用することにより，全く自由にかつあいまいさがなく指定できること．

（3） DBCL コマンドの各パラメータのデフォルト (Default) 値を，システム生成時ばかりでなく，データを定義する時点，複合コマンドを定義する時点，コマンド実行時点等できるだけきめ細かいタイミングにて定義できること．

その他の注目すべき要望としては，リクェスタがオ

ペレーション・モード（テスト・モードあるいは本番実行モード等）を自由に指定できること（同一アプリケーション・プログラムでもモードを変えることにより異なったオペレーションをおこなわしめる）や，個人用のデータを一時的に論理レベルで定義する（後述する DBDL のサブセット機能を使用）能力を有すること，などを挙げることができる．

3.2 DBA について

DBA はデータの格納，更新，検索等のリクェスタのデータ・アクセス要求を全体として最も効率良くおこなうにはいかにすればよいかを専門に考え，データ・アクセスに際してどのような制御をおこなうべきかをデータ・ベース記述言語（Data Base Description Language，以後 DBDL と略記）を使って記述する．

具体的には DBDL ステートメントにより以下の定義がおこなわれる．

（1） データ・ベースの物理的ならびに論理的特性．

（2） データ・ベースを構成するデータ間の関係．

（3） データの論理的レベルと物理的レベルの間の写像ルール．

（4） データの安全性ならびに完全性保護のルール．

（5） システム・パフォーマンスを監視するルール．

DBM は DBA が作成した DBDL ステートメントを受けとると，それをデスクリプタ（Descriptor）の形に変えてシステム内のデータ・ベース・ディレクトリ（Data Base Directory，以後 DBD と略記）の中に記憶する．DBD は，従来のシステムはスキーマ／サブ・スキーマ（Schema/Sub-schema），ディクショナリ（Dictionary）等の名前で呼ばれているものに相当する．データ定義情報をこのような形でデータそのものとは分離して保持することはデータ・ベース管理システムの基本的特徴と言うべきものであり，DBD そのものは特に新しい概念ではない．

DBD は各リクェスタに対してデータ使用に関する論理的かつ物理的枠組を定義する情報を保持する．リクェスタはこの枠組が与えられたときのみ，その枠組をはみ出ない範囲でデータのアクセスをおこなうことができる．そのためにリクェスタはデータ使用に先立ち自分が必要とするデータの枠組を定義することを DBA に要請しておかなければならない．

DBD に対する要望として特に注目すべき点は以下の通りである．

（1） DBD 内では複数レベルのカタログ構造により個々のデスクリプタの識別をシンボリックにおこなわしめること．

（2） 各デスクリプタが複数の版（Version）を持つことができ，それぞれの版が使われる条件を指定できること．

（3） デスクリプタを段階的に作りあげることができること．たとえばコンパイル時にあるデータのデスクリプタのデータ属性部分のみを埋め，次に DBA がデータ安全性に関する要望を定義したときにデスクリプタのデータ安全性規定の部分を埋める，ということを繰り返してデスクリプタを完成させるという順序をとる．

（4） 段階的に生成したデスクリプタの内容のうち，ある特定の段階で生成した部分の内容変更を指示した場合には，他の段階で生成された部分のうちその変更によって影響を受ける部分にはすべて自動的に適切な変更が施されること．

また DBDL に対する一般的な要望として特に注目すべき点は以下の通りである．

（1） 従来のホスト言語とは独立した新しい言語であること．この要望は従来のホスト言語の拡張の形では記述能力に限界があるという判断に基づいている．

（2） 言語仕様を自由に拡張できる性格を持つタイプの言語であること．

（3） 記述型（Descriptive）言語であること．ただし記述型言語では定義することが困難な部分に対しては，COBOL，PL/I 等の手続型（Procedural）言語で記述したプロシジュアを自由に呼出すことができること．

（4） DBDL ステートメントのデフォルト値をほとんどすべての時点で定義あるいは変更できること．

3.3 DBM について

DBM はリクェスタから与えられるデータ使用要求（DBCL ステートメント）を解読し，そのリクェスタの要求に対してすでに DBA が定義しているデータの論理的物理的枠組（DBD）に従って，データの格納，更新あるいは検索を実際におこなう．したがってデータ領域の割りつけや適切なデータ構造ならびにアクセス技法の選択という仕事はすべて DBM にまかされる．データのアクセスに関する DBM のシステム内で

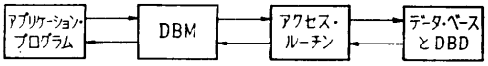

図 2 データ・アクセス時におけるデータの流れ

の位置づけは図2に示される．すなわち DBM はアプリケーション・プログラムとアクセス・ルーチンの間にあって両者の橋渡しの役割を受け持っている．DBM はこのような実際のデータ・アクセス作業を受け持つ一方，データ安全性ならびに完全性を保護するための各種処置ならびにシステム・パフォーマンスの改良のための各種統計量のカウントを併行しておこなう．

DBM に対する要望として特に注目すべき点は以下の通りである．

（1） データ・ベース以前のシステムにおいて作られたプログラムやデータを処理する橋渡しの役割をも果すべきこと．その理由はデータ・ベースもデータ・ベース以前のデータも併存せしめて処理するためには DBM がそれらのデータ・アクセスを一括して管理して，データの安全性や完全性の確保ならびに資源の有効利用を図らなければならないからである．

（2） 適切にサブセット化し，あるいは容易に再構成し得るようなモジュール構造で構成されること．これは当り前の要求であるが，実際になかなか実現し得ないところに問題がある．

（3） 新しいハードウェア技術やソフトウェア技術の採用に際して，またハードウェア構成の変更に際して，それに容易に追従できるだけの拡張性を持つものであること．これは（2）と同様，当り前でありながらその実現が容易でない点が問題である．

（4） 複数個の計算機から構成される計算機ネットワークにあっても，それぞれの計算機の DBM が他の計算機によって安全性や完全性を乱されることがないような管理をおこなうことにより，DBM ネットワークを構成し得ること．

4. データ・ベース構造体系の考え方

G/S 要望集の最もユニークな点はその提唱するデータ・ベース構造体系にあると考えられる．

物理的なデータ保持の体系と，ユーザがデータをアクセスするレベルでの論理的体系を明確に分離する考え方は従来のシステムにおいてすでに定着していると思われる．これはデータ独立性を実現するための基本的条件である．

G/S 要望集の斬新さは，従来のこの考え方をより発展させ，データ・ベース構造体系の中にエンティティ（Entity）の概念を組み込むことにより，従来不充分な程度にしか満足し得なかったデータ独立性を完全なレベルにて実現することを意図していることにある．ここで言うエンティティとは人や物，あるいはある種の観念のような抽象的なものであってもよい，一切の存在するものに対する総称であると定義される．エンティティの概念そのものはけっして新しいものではない．従来よりファイルとは特定のエンティティの特定のいくつかの属性（Attribute）に対する値（Value）の集合であると把握する見方が一般に存在している．たとえば従業員ファイルがある企業の全従業員それぞれの名前，年齢，学歴等を記録するものであるとすれば，このファイルは従業員というエンティティの名前，年齢，学歴等の属性に関する値の集合とみなされる．

この見方をより拡張した情報空間（Information Space）の考え方[12]はデータ・ベースをエンティティの概念に結びつけるものであり，G/S 要望集が提唱する新しいデータ・ベース構造系を理解するのに有効であると思われる．

情報空間の概念図は図3に示される．すなわち情報空間はエンティティ，属性，値の3軸からなる3次元

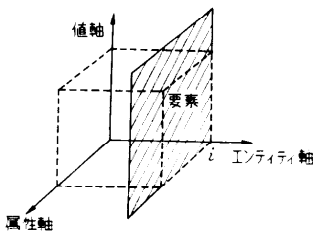

図 3 情報空間

空間であり，その空間内の一点を要素（Element）と名付ければ，それは特定のエンティティの特定の属性に対して特定の値を持つデータを表わすものである．したがって，この情報空間にあってはひとつのデータ・ベースは情報空間内の要素の特定集合として定義することができる．一方エンティティ軸上の一点 i を通る，属性軸と値軸が作る面に平行な面を考えると，この面上の任意のひとつの要素は，エンティティ i に関するレコードの特定の属性に対する特定の値の一対を

表わしている．G/S 要望集でエンティティ・レコードと名付けているものはこの属性軸と値軸に平行な面のひとつひとつに対応するものであると考えられる．

データがどのように使われようが，データがどこでどのように物理的に保持されていようがそういうことには無関係に，人や物のエンティティが存在する限り，その影の如き形でそのエンティティに関するデータ（具体的には属性と値の一対）は固有に存在する．これらのデータの特定のエンティティに関する論理上すべての集合がそのエンティティのエンティティ・レコードと名付けられるものである．

一方アプリケーション・プログラムの側ではエンティティ・レコードが保持するデータのすべてを知る必要がなく，またすべてに対して知る権利が与えられているわけではない．そこでアプリケーション・プログラムはエンティティ・レコードが保持するデータの中の必要とする部分のみを抜き出し集めたものをひとつのレコードとして使用したいと考えるであろう．このレコードはエンティティ・レコードとして固有に存在するものを，アプリケーションの都合にあわせて再構成して形成されるものであり，G/S 要望集ではこれをロジカル・レコード (Logical Record) と名付けている．ロジカル・レコードは情報空間の表現を使えば，特定のエンティティ・レコードに対応する属性軸と値軸に平行な面の中の要素の特定の部分集合に対応している．

以上はもっぱら論理的世界の中での記述であるが，現実のシステムとしてはエンティティ・レコードを実際にどこにどのように存在せしめるかという配慮は不可避の問題となる．この要望集ではエンティティ・レコードが実際に物理的に存在するものとしてのレコードをストアド・レコード (Stored Record) と名付けている．ひとつのエンティティ・レコードは通常複数のストアド・レコードに分けた形で保持されると考えられる．

上記3種類のタイプのレコード間の関係を写像体系の立場から表現したものを図4に示している．この図が G/S 要望集にて要望されているデータ・ベース構造体系を要約している．

図4においてロジカル・レコードとエンティティ・レコードの間の写像ルールをロジカル・マッピング (Logical Mapping)，またエンティティ・レコードとストアド・レコードの間の写像ルールをフィジカル・マッピング (Physical Mapping) と名付けている．従

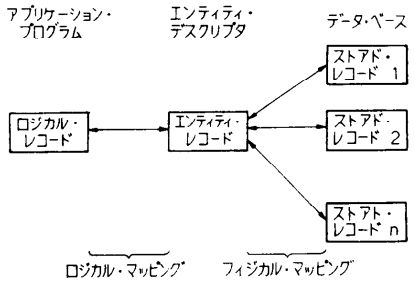

図 4　データ写像

来のシステムでは，上図に沿って表現すれば，ロジカル・レコードとストアド・レコードの間に1レベルの写像を考えるのが一般であったが，その写像が論理的レベルと物理的レベルの2者に分けられ，データ独立性が一歩進められた点に注目しなければならない．具体的表現をとれば，アプリケーション・プログラムの特定のロジカル・レコードに対する論理的データ特性を変更する必要が生じたときには，ロジカル・マッピング部分のみを変更すればよく，もちろんそれによってアプリケーション・プログラムを変更する必要はない．一方格納しているデータの物理的特性を変更（記録媒体の変換，データ編成の変更等）する必要が生じたときには，単にフィジカル・マッピング部分を変更するだけでよく，他に何らの影響をおよぼすことはない．

従来のシステムにおけるデータ・ベース構造体系においてこのような2レベルの写像の考え方が全くなかったわけではない．CODASYL-DBTG レポート[6]におけるスキーマ (Schema) とサブ・スキーマ (Sub-Schema) の考え方や IMS[11] における感度 (Sensitivity) の概念等には2レベル写像の考え方が漠然とした形で見出される．たとえばスキーマをエンティティ・デスクリプタとフィジカル・マッピングに，サブ・スキーマをロジカル・マッピングに，形の上では対応づけ得るが，エンティティ・レコードのような明確な概念を持たないため，論理的体系としてはあいまいである．その意味で2レベル・マッピングの思想は G/S 要望集で提唱された画期的なものとして高く評価できるであろう．

6. データ属性に対する要望

データ属性に関する G/S 要望集における個々の要望を記述するためには，データ・ベースのデータ構成単位に関する説明から始めなければならない．従来のシステムではこの種の用語は統一を欠いており，また

表 1　データ構成単位一覧

(ロジカル・データ列)	(エンティティ列)	(フィジカル・データ列)	
		データ・ベース (Data Base)	
	エンティティ (Entity) エンティティ・タイプ (Entity Type)		
	エンティティ・コンストラクト (Entity Construct)		
ロジカル関係 (Logical Relationship)	エンティティ関係 (Entity Relationship)	フィジカル関係 (Physical Relationship)	ロジカル・データ／エンティティ／フィジカル・データの構成単位間の関係
ファイル (File)		データ・セット (Data Set)	ひとつ以上のタイプのロジカル・レコード／ストアド・レコードの特定の集合
ロジカル・レコード (Logical Record)	エンティティ・レコード (Entity Record)	ストアド・レコード (Stored Record)	一つ以上のデータ・アイテム／エンティティ・フィールド／データ・エレメントの特定の集合
データ・アイテム (Data Item)	エンティティ・フィールド (Entity Field)	データ・エレメント (Data Element)	ロジカル・データ／エンティティ／フィジカル・データの最小構成単位

この要望集ではさらに新しい用語を創り出しているからである.

G/S 要望集にて使用する各種データ構成単位の用語は表1に要約される. この表はデータ構成単位を, まずロジカル・データ (Logical Data), エンティティならびにフィジカル・データ (Physical Data) の3種に大分類し, それぞれについて小さい単位から大きい単位へと, 各種別の対応レベルをそろえて表示している. ただしここで, ロジカル・データとは前述のロジカル・レコードのレベルにおけるデータの総称であり, フィジカル・データとは前述のストアド・レコードのレベルにおけるデータの総称である. この3種類のデータの基本概念については前節で説明した. また, 表1に各用語の概略説明が付加されているので, 以下の用語を除き, 個々の用語のこれ以上の解説は省略する.

○ データ・ベースとは関係しあうフィジカル・データ構成単位の特定の集合である.

○ エンティティ・タイプとは, 従業員, 部, 課等特定のタイプに分類されたエンティティである.

○ エンティティ・コンストラクトとは, エンティティ・レコードの特定のタイプのいくつかを特定のエンティティ関係により結合したものであり, これに特定の名前が与えられるとエンティティ・レコードの特定のひとつのタイプとなる.

これらのデータ構成単位に関して, G/S 要望集が要望しているものが多々あるなかで, 特に注目すべき点は以下に示すものであろう.

(1) データ・アイテムのデータ属性 (英数字, 10進数, 浮動小数点数等) としては, ホスト言語でサポートされる限りのものはすべて定義できる上に, オウンコードの組込により設置システムに固有の特殊なデータ属性 (特殊フォーマットの浮動小数点数等) を定義できること. また使われるときの条件次第で選択されるデータ属性も定義できること.

(2) ロジカル関係としては, データ構成単位のレベルの如何を問わず考え得る限りの全く任意の組合せの間で任意の関係 (階層構造, ネットワーク構造等) を定義し得ること. また静的な関係ばかりでなく, ある状態 (内容) のデータ構成単位間においてのみ存在し得る動的な関係も定義できること.

(3) 特定のロジカル・レコードが特定のファイルの構成要素 (Member) であるか否かを決めるアルゴリズムを任意に設定できること. また特定のレコードの追加, 更新, 消去等が他のファイルに対して波及的効果 (Propagation) をおよぼす場合にはそれを自動的に処置するよう指定できること. たとえば給与委員会の委員であるためには人事部に所属していることが条件である場合に, ある人が人事部から他部門に配置転換されたときには, その人を給与委員会の委員からはずすという波及措置が必要とされる.

(4) ロジカル・データの任意のデータ構成単位とエンティティの任意のデータ構成単位の間の写像

関係を表わすロジカル・マッピングを全く任意の形に設定できること.

(5) エンティティ・フィールドとして実体のある (Real), すなわち特定のデータ・エレメントが対応づけられているもの以外に, それ自身は実体を伴なわないがその値を他のいくつかのエンティティ・フィールドにあるアルゴリズムを作用させて得ることができる見かけ上 (Virtual) 存在するタイプのものを定義し得ること.

(6) エンティティ・レベルで, ロジカル・データならびにフィジカル・データに対してデータ属性やオーダ (配列順序) 属性等のデフォルト値を定義できること.

(7) データ・ベース内に計画的に冗長データを導入したり, データに変化が生じるときその変化を冗長データにまで正しく波及せしめるルールを任意に定義できること.

(8) データ・エレメントのひとつのタイプとして派生型 (Derived) のものを定義できること. 派生型データ・エレメントはその値がいくつかのデータ・エレメントに特定のアルゴリズムあるいはプロシジュアを作用することにより得られるものである. したがって派生型データ・エレメントの値はそれが関係するデータ・エレメントの値が変更されるとき, それに付随して自動的におこなわれなければならない. たとえば[9], 次の関係式,

$$X=A+B,\ Y=B\times C,\ Z=Y\times D$$

でデータ・エレメント間が関係づけられているとき (ただし, A, B, C, D は実値を持つ), Bの値の変化に際し, X, Y, Z の値は自動的に変更されなければならない.

7. データの安全性ならびに完全性の保護

同一のデータ・ベースを多数のユーザが共有することが基本前提であるデータ・ベース管理システムにあっては, 許可しないユーザが特定のデータを不正にアクセスすることを防止してデータの秘密を守る安全性保護ならびに, ユーザ間の不正な干渉によりデータが破壊されることを防止する完全性保護, の2種類のシステム制御機能はとりわけ重要とされており, 従来のシステムでもこの点につき種々の配慮がなされている[5].

G/S 要望集では従来のシステムにおけるこれらのデータ保護をより徹底したものとするよう要請している. その中で特に注目すべき点は下記の通りである.

(1) ロジカル・データ, エンティティならびにフィジカル・データの各レベルのそれぞれのデータ構成単位に対して, 各ユーザに対して, また各 DBCL コマンドやその各パラメータに対して, それぞれに固有の, 考えられ得る限り多様な安全性保護規定を定義できること. たとえばあるユーザはあるひとつの仕事に携わっているときには特定のデータをアクセスできるが, 別の仕事に従事している時はそのデータをアクセスできず, 他の特定のデータのみをアクセスできる, というような指定である.

(2) あるシステムに対する安全性保護を定義する仕事を, すでに特定の安全性保護機構が設定されているシステムの下でおこなえること. また安全性保護機構そのものが故障した場合に備えて, 安全性保護機構そのものを一時的に取りはずし得る特別の手段が用意されていること.

(3) ロジカル・データ, エンティティならびにフィジカル・データの各レベルのデータ構成単位に対して, できる限り多様な完全性保護規定を定義できること. もともとの完全性保護の発想は物理的レベルでのデータの保護であるが論理的レベルでの非矛盾性チェックも効果的である. たとえばある従業員の性別が男である場合, その人に最も近い親族は夫ではあり得ない, などである.

(4) データ構成単位の排他的使用あるいは共有使用をできる限りきめ細かいタイミングにて制御できること. またそのような制御をおこなう時間幅についても任意の形で指定できること.

(5) データ・ベースの一部分が損なわれた場合その部分の使用を, 修復される時まで一時的にロック・アウトした形でシステムの運行を継続できること. また故障時からの回復のために, ある与えられた任意の時点の, あるいは何時点か前の版のレコードの内容を再現できること. さらにまた, 緊急回復を要するものとそうでないものというようにデータに優先順序づけをおこなっておくことにより, 緊急性を有するものから第1に回復せしめ得ること.

8. むすび

G/S 要望集は難解なレポートである. 抽象的記述が多く具体的事例の引用が少ない上に, どちらともとり

がたい一見矛盾している感じの記述が混っているため，相当の部分を想像力で補わなければならない．

この要望集を受け取った IBM では，開発部門からマーケッティング部門にわたるまでのこの分野の専門家達によってその内容を検討し，その結果を非公式回答として表明している[13]．その回答はこの仕事に対して高い賛辞を述べている一方，この要望集の中の記述不充分なところ，ならびに矛盾する記述に対して種々の批評を加えている．それらの中で特に大きな問題点として，エンティティの概念の記述が不充分なこと，ならびに DBDL が既存のホスト言語の拡張でなく独立した記述言語でなければならない理由の正当性の記述が不足することとそれに関して矛盾する記述があること，の2点を指摘している．

しかし，たとえその内容にこのような不充分さがあるとしてもそれが故にこの要望集の価値がけっして下るものではない．その多くの将来を見通した示唆に富む内容は将来のデータ処理の分野の発展に多大なる貢献をなすものであることを固く信ずるものである．

力およばず充分なる解説とはなり得なかったが，この貴重なデータ・ベース要望集を理解する上で少しでもお役にたてば幸いである．

参考文献

1) J. P. Fry et al.: Data Management Systems Survey, The MITRE Corp., Washington D. C. (Jan. 1969).
2) J. B. Carolyn & B. S. Donald: File Management Systems; A Current Summary, DATAMATION, Vol. 15, No. 11, pp. 138-142 (Nov. 1969).
3) T. Angel & T. M. Randel: Generalized Data Management Systems, Computer Group News, Vol. 2, No. 12, pp. 5-12 (Nov. 1969).
4) CODASYL Systems Committee: A Survey of Generalized Data Base Management Systems, Available from ACM, New York (May 1969).
5) CODASYL Systems Committee: Feature Analysis of Generalized Data Base Management Systems, Available from ACM, New York (May 1971).
6) CODASYL Data Base Task Group Report, Available from ACM, New York (April 1971).
7) R. W. Engles: An Analysis of the April 1971 Data Base Task Group Report, A position paper presented to the CODASYL Programming Language Committee by the IBM Representative to the Data Base Task Group (May 1971).
8) The Joint GUIDE and SHARE Data Base Requirements Group: Requirements for a Data Base Management System, Available from GUIDE Secretary Distribution, Chicago and SHARE Secretary Distribution, New York (Nov. 1970).
9) A. C. Patterson: Requirements for a Generalized Data Base Management System, Proc. FJCC pp. 515-522 (1971).
10) T. W. Olle: MIS-Data Bases, DATAMATION, Vol. 6, No. 15, pp. 47-50 (Nov. 1970).
11) Information Management System/360, Version 2 System/Application Design Guide, IBM Corp. (Oct. 1971).
12) J. K. Lyon: An Introduction to Data Base Design, Wiley-Interscience, New York (1971).
13) J. C. Fensterstock & P. Y. Tani, IBM to GUIDE/SHARE an DBMS, Informal Response to GUIDE/SHARE Data Base Requirements Group by IBM Corp. (Jan. 1971).

(昭和47年8月10日受付)