解　説

情報資源辞書システム(IRDS)†

溝　口　徹　夫††

1. はじめに

過去2年あまりにわたって情報資源辞書システム：IRDS Information Resource Dictionary System の標準化作業が ISO の SC 21/WG 3 の作業の一環として実施されており，今後も継続した活動が予想されている．本解説では過去の活動を通して，IRDS の標準化作業の内容を説明し，また今後の動きや日本の立場についていくつかの予想や意見を混ぜて説明をしてみたい．

2. IRDS の標準化の背景

そもそも情報資源辞書システム（以下 IRDS と略す）という言葉は多くの読者にはなじみは少ないと思える．そこでまず，IRDS とはどのようなものか，その標準化の利点は何か，標準化がなぜ今必要なのか，標準化することでどのような影響が生じ得るのかなどについておおまかに述べてみたい．より細かい説明については後述する．

2.1 IRDS は概略どのようなものか

IRDS は DDS：Data Dicitionary System とか DD/DS：Data Dictionary/Directory System とかいわれているものに当たる．そこで IRDS を説明するには，データディクショナリシステム：DDS とはどのようなものかについての説明をする必要がある．

開発されたソフトウェアにはプログラムの中の手続きとプログラムの中で利用されるデータ，ファイルやデータベースシステムで利用されるデータの定義の2種類の表現が存在する．ソフトウェア開発者が生成するこの2種類の表現の，主にデータの定義を管理する機能をデータディクショナリ/ディレクトリ機能と呼んでいる．

ここでデータの定義の管理機能とはどのようなことを指すのかというと，おおまかにいってデータが定義されたことの登録，参照（どのプログラムがどのデータを使っているかなどの問い合わせ回答），抹消を行うものである．データディクショナリ機能は対人間用（ソフトウェア開発者，プロジェクト管理者らのため）の管理機能と考えられるのに対して，データディレクトリ機能は対機械の管理機能と考えられている．しかしこの二つの差は本解説では区別せずに話を進めることにする．

たとえばソフトウェア開発のプロジェクトが大規模化，複雑化するに従い，システムの要素がどのように作成され，利用されているかを管理する機能が必要になる．

このようなプロジェクト管理などのような応用において，必要とされる要素として定義データの管理，すなわちデータディクショナリ機能への言及がしばしばなされる．しかもこの機能は人手によるものでなく，自動化されていることが望まれる．したがって DDS は単に機能があるということだけでなく，ソフトウェアパッケージとしての存在が求められるわけである．

以上述べたソフトウェアパッケージの普及などを目指した標準化の動きが IRDS の国際標準化活動のきっかけとなっている．

IRDS 標準化のきっかけとなっている，データディクショナリシステムは十分な普及が行われているといえないのが現状である．普及が十分でない理由はいくつかあると思われるが，その一つはデータディクショナリ機能単独では，たとえばプロジェクト管理を実施するには十分でないこと，またデータディクショナリ機能を活用するにはデータの設計などの高度で複雑な問題に関わることになるからである．

どのような性格の IRDS を対象としているかをもう少しはっきりさせないとその技術的困難度，利用の効果などが明確にはならない．

IRDS が対象とするものは何かをここではっきりさせておく．プログラミング言語やデータベース管理システムはプログラムが実行され，データが処理される

† Information Resource Dictionary System (IRDS) by Tetsuo MIZOGUCHI (Financial Systems, 1st Systems Engineering Center, Mitsubishi Electric Corp.).

†† 三菱電機(株)情報通信第1 SE センタ金融流通システム部

ためのものであるのに対して，IRDS ではプログラミング言語によるプログラムの中で定義されたデータを扱う．つまり通常のプログラムの処理の対象はデータそのものであるが，IRDS の処理の対象は定義データ（メタデータとも呼ぶ）である．

従来データディクショナリ機能の性格付けとして，能動/受動の区分をする．データディクショナリが実稼動環境を能動的に支配するのか，それとも受動的に受け止める文書化の役目としてのものかの区別がされる．能動的なデータディクショナリがいかに困難な，複雑なものであるかはその影響度を考えると想像ができる．では次になぜ IRDS の国際標準化が検討されているのかについて触れてみたい．

2.2 IRDS 国際標準化の必要性

すでに IRDS がどのような性格のものであるかは説明した．IRDS は特に定義データを扱うものである．ソフトウェア製品には必ずというほどデータ定義が付随している．各種のソフトウェア製品，たとえば言語プロセッサ，データベース管理システムなどではおのおのその製作者が独自のデータ定義や形式を採用してしまうと，問題となる．したがって言語などの標準ではその点に配慮が払われる．ここで IRDS の標準化について，いくつかの課題を示してみたい．

（1） IRDS 標準はどの範囲の機能をもつのが望ましいのか？

（2） IRDS 標準は他の標準，たとえばデータベース言語標準で代用できないのか？ IRDS 標準の独自性は何なのか？

（3） IRDS 標準はどのような効用が予想されるのか？

これら課題に対する答として，以下に理想的な IRDS 標準の姿を示してみたい．

a) IRDS 標準はあらゆる種類の定義データに対して，格納，検索などの管理機能をもたねばならない．たとえば対象定義データの分類，名称管理，関連付け管理などである．（定義データの写像，一意性管理）

b) IRDS 標準は定義データを使っての各種適用（たとえばプロジェクト管理，ソフトウェア開発管理，品質管理）に十分有効な基本的機能を備えていなければならない．例としてソフトウェア開発フェーズの定義機能とフェーズ間移行の機構の提供などがある．（対象適用情報システムのライフサイクル管理）

c) IRDS 標準は定義データそのものの進化に対する管理機能，たとえば改訂管理などがなければならない．（対象定義データのライフサイクル管理）

d) IRDS 標準は定義データを扱う上での必要な機能，たとえば保全性に関するものを備えていなければならない．

e) それ以外に，分散環境（たとえば OSI 環境）での IRDS 機能の提供，他のソフトウェアとの連動機能（能動的 IRDS 機能や DBMS などとの協同）などがあげられる．

特定の適用に対して特筆すべき効用を IRDS 標準に望むというよりは，共通的（generic）な機能の標準化がこの標準の主旨といえる．

2.3 IRDS 国際標準化の影響

次に IRDS 標準化によって何にどのような影響があるのかについて述べてみたい．ただしここに述べるものは一面的，個人的見解も含まれるので，今後標準が普及した場合に振り返ってみたい．

まず IRDS 標準の利点をあげてみたい．

（1） データディクショナリ管理機能の体系化の利点

これまでなかったデータディクショナリ管理機能の体系的な整備によって，何が必要でどのように使うかを明確にできることで，過去の経験を集大成できる．また各種ソフトウェア製品はこの標準に合わせて定義データの管理機能を一本化し，他のソフトウェア製品との定義データ交換の道が個別的でなく開ける．

しかしこの利点の中にもいくつかの問題点を含んでいる．すでに他の各種ソフトウェア標準での機能の重複，たとえばデータベース言語標準でのスキーマ操作機能などが存在するし，単に定義データの登録管理など一般的機能では十分な IRDS の機能性が得られていないという問題もある．

これら問題点以外に次のような不安が IRDS 標準には存在する．

（2） モデル，方法論の固定化による進歩の停止や柔軟性の欠如の危険

もちろんどのような範囲の標準が設定されるかによるが，IRDS 標準は特定のデータモデルに支配される危険性をもつ．また特定の方法論（たとえばデータベース分析，設計方法論）に支配されるものであってはならない．

そのためにはデータモデルはできるだけプリミティブであること，方法論とは独立していることを保証する標準でなくてはならない．

完全無欠のものが存在しない以上，このような目的

を達するのは容易ではなく，多大な労力と時間を要すると思われる．

IRDS 標準をとりあえずの便宜的なものとするのか，根本的な問題に立ち入ったものにするのかという立場の取り方の選択が必要になる．

プログラミング言語やデータベース言語は複数存在することが許されても，IRDS は違ったタイプの標準が複数存在することはその性格上からありえない．とりあえずのものが標準化されることの利点，欠点はすでに過去のいくつもの経験が語っているところである．

過去の行き詰まった技術で将来を束縛してはならないことはだれしも認めるところであるが，過去の成果が有効であると主張するあまりに標準としての将来性に影を落とす危険がある．

2.4 ANSI（標準化原案提案元）の提案根拠

そもそも IRDS の標準化を検討し始めたのは ANSI からの標準化提案があったからである．当然提案をする側では，具体的な標準化文書の原案を用意し，標準化の必要性と文書が標準として適切であることを示す．

ここで ANSI の提案した IRDS 標準原案はどのような提案根拠に基づいているかについて述べてみたい．

米国政府機関でのデータ処理の中で，データディクショナリシステムの利用についての問題提起がなされ，もしもデータディクショナリシステムの利用者インタフェースを標準化できれば，利用者はいろいろの違ったインタフェースを知る必要がなく，費用削減につながるとの結論から，データディクショナリ機能の米国政府機関での標準の検討が開始され，また ANSI としての標準の検討が進められてきた．

この標準案の特徴は，利用者インタフェースの標準を狙っていること，したがって標準の対象はデータディクショナリ機能を実現するためのコマンドの形式である．

この標準化原案は既存のデータディクショナリシステムの機能の中からまとめあげたものとの印象を与える．既存のデータディクショナリシステムの機能が適切なものであるのかどうかの議論が十分されたか否か，その検討過程での議論の内容は不明である．成果としての標準化文書が提出されたという状況である．

標準化文書の妥当性については，十分な経験に裏打ちされていること，ANSI 内での公開レビュを行い，提出された意見は十分盛り込んで文書化されている旨説明が ANSI からされている．

なぜ今 ANSI の提案による IRDS を標準化する必要があるのか，技術的にその内容は妥当なのか（ANSI がいうことは一つの国の標準としては妥当であっても国際標準として妥当かどうか）の議論は十分にできているとはいえず，むしろ各国の，特に主要国の反応は消極的または否定的である．

ANSI 提案に対して積極性を示せない理由は，第1には原案文書の量（本文 1091 ページ）と文書構成の混乱があり，標準文書としての適性に欠けること，第2にはモデルを規定して（ER モデルと ANSI は呼んでいるが，必ずしも精密にモデルを記述していないとの批判を受けている），利用者インタフェースの標準を狙っていることである．利用者インタフェースが標準として必要，あるいは妥当かについては疑問が多く，むしろより内部のデータディクショナリサービスインタフェース（言語プロセッサなどとのインタフェース）の標準が望まれるとの意見が各国から表明されている．第3にデータベース関係の標準化にも関わらず，取り扱うデータ構造の概念が整理され，明示されていないことである．

3. IRDS 標準化作業とその経緯

IRDS 標準原案は ANSI によって提案されたが，すでに述べた事情から必ずしも歓迎されず，議論の初期に参加した英国，カナダ，日本からは消極的態度が表明され，特に英国 (BSI) は ANSI 原案の再構成を申し出た．以後二つの案が議論の対象になっていたが，その時点では ANSI 原案も BSI 再構成も正式な文書として各国に採否の投票を依頼する状態にまで達してはいなかった．その後次のような妥協が成立し，作業を進めることになった．

ANSI 原案は利用者インタフェース (Command and Panel Interface) の標準である．

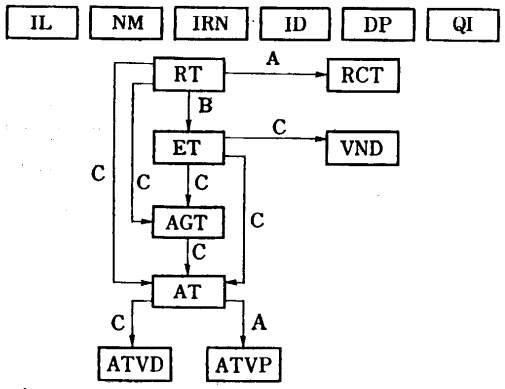
BSI 原案はサービスインタフェース (Service Interface) である．

この二つの標準を関連付けるために IRDS 枠組み (Framework) を設定する．

以下に ANSI 原案の具体的内容，ISO 標準としての作業の状況の説明を行い，その過程における日本を含めた各国の立場について述べてみたい．

3.1 ANSI の提案の概要

千ページ以上にもおよぶ資料を簡潔に説明するには



略 語

AGT	ATTRIBUTE-GROUP-TYPE	
AT	ATTRIBUTE-TYPE	
ATVD	ATTRIBUTE-TYPE-VALIDATION-DATA	
ATVP	ATTRIBUTE-TYPE-VALIDATION-PROCEDURE	
DP	DICTIONARY PARTITION	
ET	ENTITY-TYPE	
ID	IRDS-DEFAULTS	
IL	IRDS-LIMITS	A (0, n : 0, 1)
IRN	IRDS-RESERVED-NAMES	B (0, n : 2)
NM	NAMES	C (0, n : 0, m)
QI	QUALITY INDICATOR	
RCT	RELATIONSHIP-CLASS-TYPE	
RT	RELATIONSHIP-TYPE	
VND	VARIATION-NAMES-DATA	

図-1 ANSI 原案 IRD Schema の構造

困難がともなうが，ANSI 提案のデータ構造を IRD Schema として**図-1** に示す．ANSI 原案で規定されるのはディクショナリデータの操作（コマンド）であり，コマンドは2種類 (Schema Command と Dictionary Command) のタイプが存在する．そのコマンドの位置付けは**図-2** に示されている．図-2 は ANSI 原案の文書構造を反映しており，Part 1 はコアで，IRD Schema と IRD Dictionary Data からなり，Part 2 以後は選択追加可能である．コアは IRDS 動作のための機構を提供するもので，データディクショナリの言葉（たとえばデータ項目，ファイルなど）は Part 2 で定義される．その他の追加可能なものとして保全性 (Part 3) と拡張ライフサイクル (Part 4) がある．図-2 の箱の右肩の数字はそのタイプに属する値の数を示す．

図-1 に示されている 14 個の箱は図-2 における IRD Schema の中の Meta-Entity-Type(RT, ET, AT など）である．図-1 の 10 個の矢印は図-2 の Meta-Relationship-Type である．この 10 個の Meta-Relationship-Type で結合されている図-1 での二つの Meta-Entity-Type（たとえば RT と RCT）が接続されているということは図-2 の Meta-Relationship-Type（たとえば RT member-of RCT ; RT は RCT のメンバである）から Meta-Entity-Type（たとえば RT と RCT）への2本の矢印として表されている．

図-1 の ET (Entity Type) としての Meta-Entity-Type (ET は 14 個の Meta-Entity-Type の一つ) の内容である Meta-Entity (図-2) には，Dictionary-user, Schema-View, Dictionary-View が IRDS コアとして設置されている．それ以外に図-1 に示したおのおのの Meta-Entity-Type にその内容として総数 27 個の Meta-Entity が与えられている．

データディクショナリ機能はコアの IRD Schema の構造に Schema Command を使った IRD Schema の定義の追加（当然追加，変更には制約が課せられ，その負担はコマンドのセマンティクスとして与えられている）を行い，その IRD Schema 構造の下に IRD Dictionary Data を扱うことになる．

たとえば図-2 において，Schema Command は追加可能モジュールである保全性 (Part 3) の管理の下に，Dictionary-user として登録されていることの関門をくぐり，Schema-View としての権限などの関門をくぐって，さらに追加可能モジュールである，拡張ライフサイクル (Part 4) での Schema Structure（これはコアでは定義されず，Part 4 の中で定義されるもので，任意の Entity とその間の Relationship を Schema Structure として切り出すもの）の構造を経て IRD Schema の構造に変化を与える．また Schema Command とは別に Dictionary Command を用意し，Schema Command と類似であるが異なる機構を経て，データディクショナリデータの参照，追加，変更などが可能である．

3.2 ISO 内での標準化作業の状況

ANSI 提案による IRDS 原案を受けた以後の ISO としての IRDS 標準に対する作業の状況を説明したい．昭和62年5月時点での状況はすでに報告したが[2]，昭和62年10月の会議でいくつかの方向付けに関する議論がされた．これらの会議の方向付けは決定的なものではなく，上部機構によって正式な承認を必要とするものである．したがってここに示す内容は確定的ではないが，少なくとも各国の専門家の間で多数意見を得たものである．

また IRDS 標準はデータ管理参照モデルの標準化の影響，たとえばデータレベルの概念や名称などの統一的使用，が大きく，データ管理参照モデルの標準化

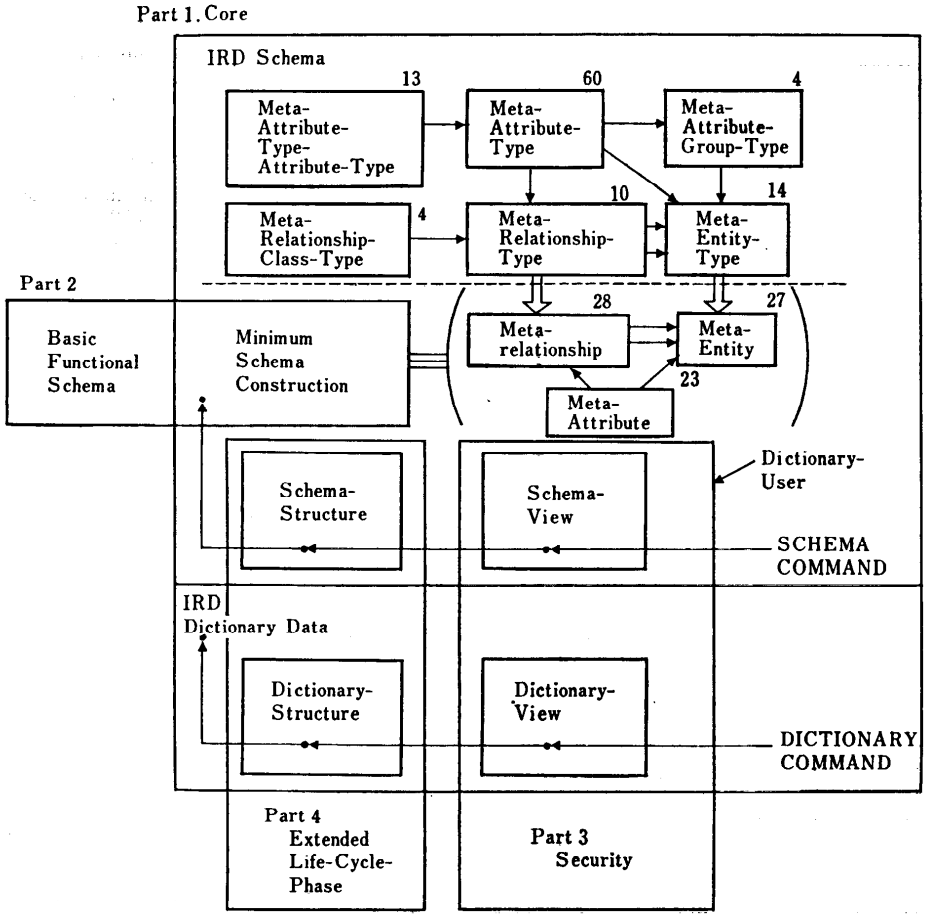


図-2 ANSI 原案[1]の構成

の動きに合わせて用語などの統一をも図らねばならない.

前述のように, ANSI 原案を IRDS 利用者インタフェースとして, また BSI 再構成案を IRDS 内部インタフェースとして位置付けたこと, およびデータ管理参照モデルとの整合性から, IRDS の枠組みに関する規定が標準の一部として必要との合意を得た.

ここではまず枠組みについて述べ, 次に内部インタフェースについて触れる. また利用者インタフェースについても触れるべきであるが, 内容は ANSI 原案であるので省略する. むしろ現在内部インタフェースと利用者インタフェースは文書上での食い違いがあり, その点が今後の課題として残っていることを指摘しておきたい.

3.2.1 枠組み Framework[3]

ISO 標準のための文書作成の手引に合わせて, 次のことが枠組みの目次としてあげられている. 標準の範囲, 参照資料 (特に他の標準), 定義と記号, IRDS データ, IRDS インタフェース, IRDS 機能性, IRDS 標準の構造, 標準の準拠.

定義と記号ではデータ管理参照モデルとの整合と内部インタフェースと利用者インタフェースで使われる用語の定義が与えられる. IRDS データについては IRDS が扱う定義データの理解を助けるための表現やデータレベルの概念の説明が行われる. **図-3** にその説明案を示す. この図の提案者は筑波大・穂鷹教授である.

データのレベル(縦軸)は 4 段階であり, この 4 段階のデータを使って, 三つの異なる定義—値の対(横軸)が存在する. 左の対 (Application Level Pair) は通常のデータベースシステムであり, Schema-Data の対である. 中央の対 (IRD Level Pair) が通常, データディクショナリシステムと呼んでいるものである. 右の対 (IRD Definition Level Pair) は標準の基礎を

Fundamental Level			Types: Entity type Relationship type Attribute type
Dictionary Definition Level		Types: Record Type Data Item Table	Instances: Entity "Record Type" Entity "Data Item" Entity "Table"
Dictionary Level	Types of Data Item for Record "Employee" Employee-name Person-no Department-no	Instances: Record Type "Employee" Data Item "Person-no" Data Item "Department-no"	
Application Level	Instances: Employee-name "J. Smith" Person-no "12345" Department-no "101"		
	Application Level Pair	IRD Level Pair	IRD Definition Level Pair

図-3 データレベルとレベル対の例

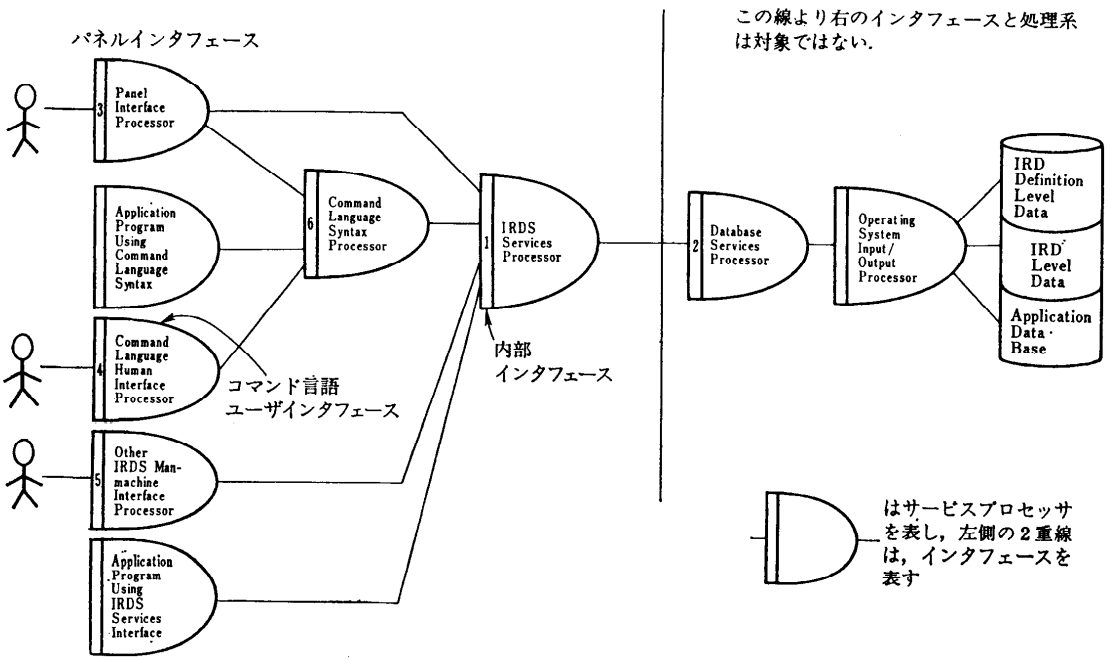


図-4 主な IRDS インタフェース

与えるもので，データディクショナリ実現者の理解を助けるものである．図の中の縦の矢印は Schema-Data の関係を示すが，横の矢印は右のデータが左の定義データとして内部的に重複していることを示している．自然なデータの定義の流れとしては，右上から左下に向かって定義が進められる．

また IRDS インタフェースでは，標準化すべきインタフェースの説明を行う．図-4 にその原形案を示す．この図において内部インタフェースと利用者インタフェースの位置付けがされている．

図-4に示された各種インタフェースのうち，インタフェース1は内部インタフェース(Service Interface)，インタフェース3と4を合わせて利用者インタフェースと本解説では呼ぶ．またこれらのインタフェースが標準化の対象であることが合意されている．

図-3，4ともに資料の一部のみの表現であり，今後検討の過程で表現の方法については改訂されるものであることに注意を要する．IRDS 機能性についてはIRDS 特有の機能性に必要な概念を説明する．

現在の標準文書の完成度は不十分で，かつ後で述べる2種類のインタフェースに関する標準の整合をとるに十分なものを提供していないという問題をもっている．

3.2.2 内部インタフェース Service Interface[4]

もともと内部インタフェースというものの標準提案があったわけではなく，すでに述べた経緯から，内部インタフェースの標準化の方が利用者インタフェースの標準化より重要だという考え方が出てきた（ANSIは賛成しなかったが）．内部インタフェースが重要であるとの根拠は図-4 に示した各種インタフェースの存在からも伺える．

内部インタフェースの標準（あるいは少なくともなんらかの記述をすること）の意義は，今 IRDS 標準の対象としているもののデータ構造がどのようなものかを明示的に記述することである．これは ANSI 原案が手順中心のもので，しかも利用者インタフェース(データベース的表現でいえば，外部スキーマからみたインタフェース)であることへの補完的意味もある．

内部インタフェースに関する標準の文書の構造は，Abstract Data Structure, Service Data Structure, Service Protocol の3種類が主なものである．次に順次説明をしていきたい．

(1) Abstract Data Structure

前述したように，IRDS の処理の対象となるデータの構造を明示的に行うことが主旨である．表現の手段として ISO 標準として作業が進められてきたデータベース言語 SQL を採用している．SQL を採用している理由は次のとおりである．

表現のあいまいさをなくすためにも，多くの関係者が誤解しない表現言語を用いなくてはならないこと，

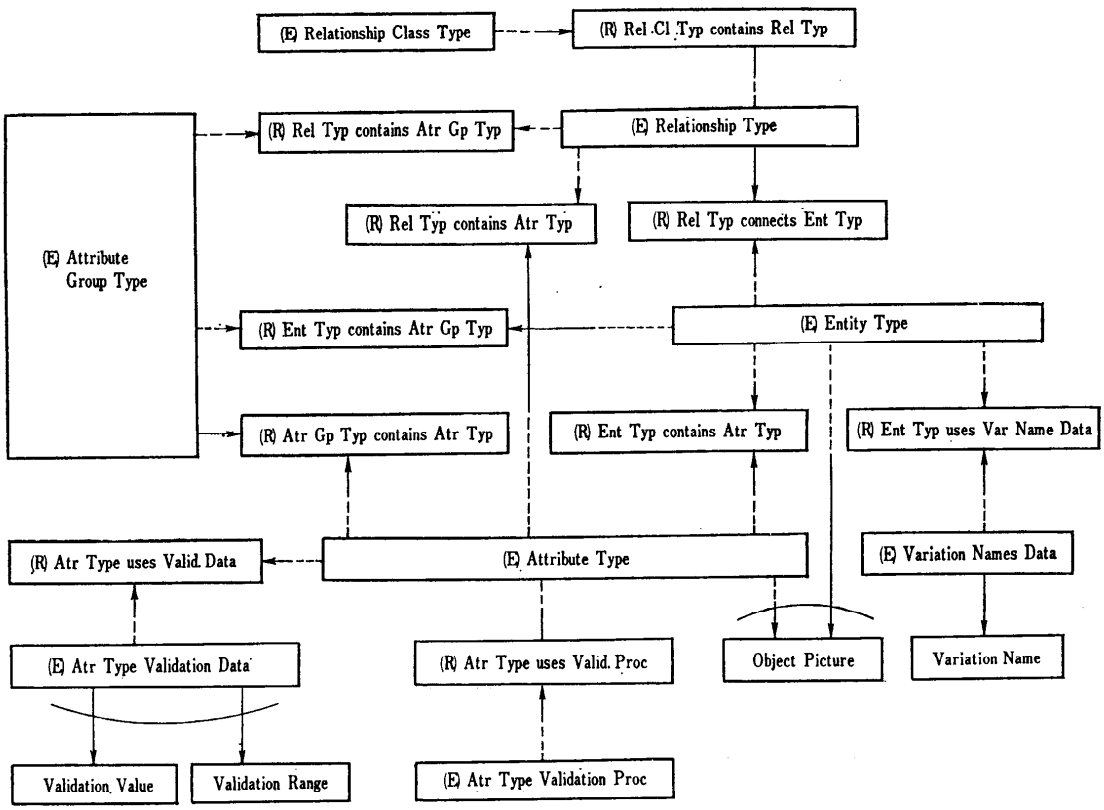


図-5 Abstract Data Structure. IRD Definition Level

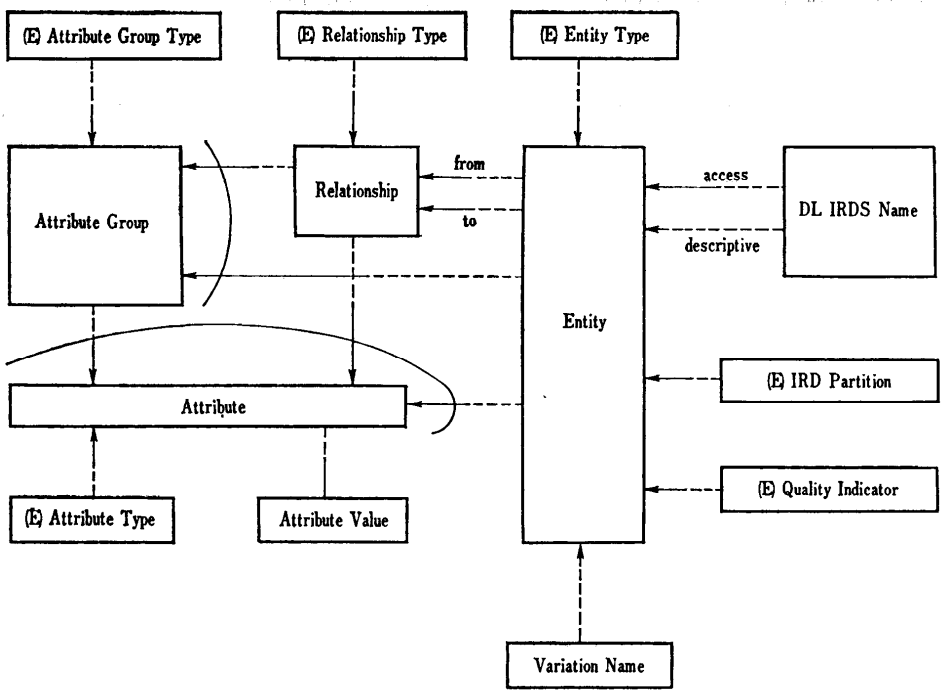


図-6 Abstract Data Structure. IRD Level

SQL は ISO の標準として関係者での誤解は少ないことの二つによる．SQL の採用は IRDS が SQL で実現されることを示唆しているかのごとくみえるかもしれないが，Abstract Data Structure は標準として規定されるインタフェースそのものにはまったく現れないものであり，SQL の表現はなんら標準を規制するものではない．Abstract Data Structure の表現にはいずれにしてもなんらかの表現言語が必要であり，たまたま SQL を選んだこと，それも図-4 に示したように標準の対象となるインタフェースからみて奥にあるデータの表現に記述を限定している．

図-1 の表現に対応する図式表現を**図-5** に示す．図-5 の表現は，図-3 における IRD Definition Level Pair におけるデータ構造を表している．図-5 が図-1 と異なる点は，箱とその間の矢印でデータ構造を表現し，矢印には属性をもたせていないことである．

さらに**図-6** には図-3 における IRD Level Pair におけるデータ構造を表している．ただし図-6 の中では図-5 で示されているものをも使った表現となっている．

図-5，6 に現れる言葉は ANSI 原案のものを使い，できるだけ用語の相違などによる発散を防いでいる．

図-5，6 の正確な表現は，箱の中身については，SQL の Column 表現を用い，矢印は SQL での参照制約 (Referential Constraints) によって表現している．

Abstract Data Structure の表現の一部分を**図-7** a) に示す．

（2） Service Data Structure

Service Data Structure は図-4 における内部インタフェースの位置からみたデータ構造（コマンドのパラメータなど）を表現したものである．**図-7** b）にその一部分を例として示した．これら Data Structure は次の Service Protocol によって使用される．

（3） Service Protocol

Protocol という表現が適当かどうかは疑問があるが，内容としては ANSI 原案における Schema および Dictionary Command と同じである．Command の種類としては，Open Cursor, Add, Modify, Delete, Retrieve, Close Cursor などであるが，おのおの IRD Difinition Level では Meta-Entity, Meta-Relationship, Meta-Attribute ごとに別々のコマンドが用意されている．

例：Add Meta-Entity, Add Meta-Attribute など．

また IRD Level も別個のコマンドが用意されている．

例：Add Entity, Add Attribute など．

具体的表現の一部分を**図-7** c) に示す．

```
CREATE TABLE ENTITY_TYPE

(ENT_TYPE_NUM INTEGER NOT NULL
CONNECTABLE_ENT CHAR
CHECK (CONNECTABLE_ENT IN ('Y', 'N'))
ENT_CLASS CHAR
CHECK (ENT_CLASS IN ('D', 'E', 'P', 'S'))
IMPLEMENT_LOCK CHAR
CHECK (IMPLEMENT_LOCK IN ('Y', 'N'))
MAX_ASN_ACC_LEN INTEGER
CHECK (MAX_ASN_ACC_LEN >= 31)
MAX_ASN_DSC_LEN INTEGER
CHECK (MAX_ASN_DSC_LEN >= 31)
CHECK (MAX_ASN_DSC_LEN >= MAX_ASN_ACC_LEN)
MIN_ASN_ACC_LEN INTEGER
CHECK (MIN_ASN_ACC_LEN > 0)
MIN_ASN_DSC_LEN INTEGER
CHECK (MIN_ASN_DSC_LEN > 0)
NUM_INSTANCES INTEGER NOT NULL
CHECK (NUM_INSTANCES >= 0)
START_NAME CHAR (irds_ment_aa_lim)
SYSTEM_GEN CHAR
CHECK (SYSTEM_GEN IN ('Y', 'N')))

PRIMARY KEY (ENT_TYPE_NUM)

FOREIGN KEY (ENT_TYPE_NUM)
   REFERENCES META_ENTITY (META_ENT_NUM))
```

a) Abstract Data Structure の表現例
Entity_Type の SQL による表現

```
Meta_Entity Record

IrdsMetaEntRec  = record
   MetaEntType          : IrdsMentAaName;
   MentAsnAccName       : IrdsMentAaName;
   MentAsnDscName       : IrdsMentAdName;
   RevisionNum          : integer;
   AddBy                : IrdsEntAaName;
   DateTimeAdd          : IrdsDateTime;
   ModBy                : IrdsEntAaName;
   DateTimeMod          : IrdsDateTime;
   TimesMod             : integer;
   ScmLcp               : char;
case IrdsScmMentType of
IrdsNullMent: ( );
meta-entity-type-1:
   (meta-ent-rec-1    : record
      meta-attributes for meta-entity-type-1
   end);
        :
meta-entity-type-n:
   (meta-ent-rec-n    : record
      meta-attributes for meta-entity-type-n
   end);
end;
```

b) Service Data Structure の表現例
Mete_Entity Record の Pascal によるデータ表現

```
Protocol        Retrieve Meta-Entity Service

procedure RDRETME
      (var Rpl              : IrdsRpl;
       var MetaEntCursorId  : IrdsCursorId;
       var MetaEntRec       : IrdsMetaEntRec);
```

c) Service Protocol の表現例
Retrieve Meta_Entity Service の例

図-7 Service Interface の表現例

3.3 日本の立場や他国の立場

IRDS 標準に対して日本委員会はどのような立場で臨んでいるかについて触れてみたい.

日本委員会での IRDS 標準に対する反応は, 次の意味で消極的である.

一般にデータディクショナリの利用が十分浸透していない現在なぜ標準化を急ぐ必要があるのか, およびANSI 原案のような形での標準は当面のいくつかの役には立つかもしれないが, 将来的に望ましいものというよりは一層複雑な世界にひきずり込まれるのではないかという疑問があるためである.

一方情報資源管理という課題はこれからのデータベースに関わるものの中では最も重要なものであり, この課題について, 構造や利用面での要件の整理を行うことは実りの多いことであるとの考え方をもっている.

したがって, 日本委員会では BSI による IRDS の対象とするデータ構造の明示的表現の作業を行うこと, またその表現に用いられるデータモデルをできるだけ単純化することの提案を終始支持してきた.

その意味で英国と日本はほぼ同じ立場をとっている. 当然米国は自国の提案の成立を主張しており, 米国内での標準 (ANSI としての) と国際標準が食い違いの生じないための主張 (内部インタフェースの標準には興味がないし, 利用者インタフェースは図-4 に示す形で内部インタフェースを利用する形で実現されるのではないとの確認を常に求めていること, 利用者インタフェースは内部インタフェースと食い違ったものであっても構わないとの了承を得たいなどの要求) を行っている. カナダ, フランス, 西独は定常的に同

一人物が会議に出席することはなく，会議のたびに意見は異なるが，少なくとも一貫していることとして，利用者インタフェースにはあまり興味はなく，内部インタフェースの標準化を希望していることである．西独は最近になってようやくメンバの固定がなされたようで，今後意見を固めていくであろう．

ただし米国内でも必ずしも同一意見であるかどうかは疑問があるところで，その点については残された課題で触れてみたい．

3.4 ANSI 内での作業の進行状況

ANSIの立場は国内，国際の標準はまったく同一との希望および主張であり，文献1)のタイトルにも見られるように ISO 標準の資料でありながら，国内標準の資料であり，文書編集上での問題である．ANSI内での標準文書はすでに完成したとのことで，ISO 標準との食い違いの発生しつつある現状に対して不満の意を表しているが，その主張に対して ISO の専門家会議での賛同を得られなかった．もちろんこの会議では決定的に結論を出す権限はない．少なくとも現在ではIRDSの標準は発散したものであってはならないということであり，ANSIが独自で国際標準を自国の標準に合わせることも許されていない．

今後技術的な面よりも政治的な面での決着が上位レベルでなされなければ解決が困難な可能性をもっている．

4. 残されている課題

以上の説明はすでにある程度文書化されたものでの標準化作業の状況を述べたものであるが，次に最近の会議の席上で合意されたこと，今後の課題としてこれから具体化せねばならないことをいくつか述べてみたい．

項目としては，データモデルの改変，名称に関する拡張，フェーズ構造に関する問題，保全性についての問題が主なものである．

データモデルについては ANSI のいう ER モデルではなく，もっと単純なモデルを採用しようとの意見が多数を占めた．このことから利用者インタフェース[1)]に修正が求められることになる．

名称 (Name) に関する拡張では，現在の ANSI 原案では IRDS の名称は単一の空間しか許していない．(個人的な会話で，ANSI の原案を作るときの最初の要件はまず「名前ありき」であったと聞いた) しかし ANSI のメンバの中にも，多重名称空間を許すとか，たとえば COBOL, PL/I などでの名称をそのまま IRDS に取り込みたいとのごく自然な希望がある．名称を対象のデータと密着させるより表現されるデータとその名称を区別することは重要なことであり，日本委員会でもその点に拡張を期待していた．

ライフサイクル (情報システムとしてのものと，データ実体としてのものの2種類がある) が ANSI 原案の形でよいかどうかについて正直なところ議論はされなかった．ANSI 原案の文書という形で突きつけられたわけであったが，理解が進むにつれて，疑問が投げかけられている．ANSI原案提案者に言わせれば，ANSIではすでにそのような疑問の対策を行い，その結果が現在の姿となったとの説明であるが，具体的にどのような代替案があったかは明らかにされていない．

保全性については，IRDS 独自の考えで提案がなされているが，SC 21 全体としての保全性に関する統一したアプローチが上位組織で求められており，他との整合を考慮せねばならない．

5. おわりに

限られた紙面の中で，IRDS 標準化の現状の説明を行った．このテーマはかなり複雑な性格のものであり，また挑戦に値するテーマでもある．専門家会議における議論は密度の濃い，またかなり高度の問題を扱ったものである．このようなテーマでの貢献ができることが本当の意味での技術の向上の印でもあると思う次第である．IRDS 標準の検討に対し，貢献いただいた委員各位に感謝の意を表したい．

参考文献

1) SC 21/WG 3/NI 66 R 1
Proposed Draft American National Standard IRDS Part 1 Command Language and Panel Interface (Nov. 1986).
2) 溝口徹夫：情報資源辞書システム (IRDS)，情報処理学会データベースシステム研究会資料(1987. 5).
3) SC 21/WG 3/N 389 (ISO TC 97/SC 21/N 2131) : IRDS Framework Working Draft Rev. 4 (1987. 9. 27).
4) SC 21/WG 3/N 390 (ISO TC 97/SC 21/N 2132) : IRDS Service Interface Working Draft Rev. 4 (1987. 9. 27).
5) Gradwell, D. J. L.: Development of Data Dictionary Standards, Computer Bulletin, pp. 33-38 (Sep. 1987). (昭和62年12月1日受付)