DEIMS－2について

鈴木 健司
（日本電信電話公社 横須賀電気通信研究所）

1．はじめに

DEIMS-2（DEndenkosha Information Management System）は横須賀電気通信研究所（以下通研と略す）において実用化されたDBMSである。

通研におけるDBMSの研究実用化は1972年ごろより開始された。当時、CODASYLより提案されたデータベース言語仕様は、実用レベルの具体的提案として高く評価され、また国際標準化の方向としても注目されていた。これらの点からCODASYL仕様にそったDBMSの開発を行ない、試作を行なった。その試作経験をふまえ、1975年より実用化への検討を開始し、DIPSプロジェクトとしての商用ベースのDBMSであるDEIMS－2を開発した。

以下、DEIMS-2の構成、特徴的機能等について述べる。

2．開発の背景・方針

電電公社データ通信サービスの分野はDEMOS、DRESS、バンキング・システム等のサービスから、医療、行政等のナショナル・プロジェクト関連システムへと拡張されつつある。ナショナル・プロジェクト関連システムでは、ファイル中心形のシステムが多数存在し、DBMSの実用化が必要となっている。

これらの背景からDBMSの実用化にあたっては以下の方針で進めた。

（1）オンライン実時間データベース・システムの実現

DIPSの104オペレーティング・システム（OS）のもとで、実時間制御プログラムRTP（Real Time Package）と有機的なつながりをもった高トラヒック・オンライン実時間データベース・システムを可能とすること

（2）大容量データベースの実現

大容量に伴なうファイル効率のよいデータベースの構成を可能とすること

（3）信頼性の向上

大容量データベースに対する信頼性をはかること

（4）性能に対する配慮

CODASYL仕様の豊富な機能に対し、性能上の配慮をはらうこと

上記の方針のうち、性能に関する問題がDBMSの実現上の課題ともいえる。一般に専用ファイル・システムで構築した場合に比べデータベース・システムではファイル概念と処理を仮想化し、機能レベルは汎用化されているため性能上のオーバヘッドが問題となる。特に、高トラヒック、大容量なオンラインへの適用にあたっては、性能上のオーバヘッドをいかに解決するかが重要である。そこで従来の専用ファイル・システムと同様な性能を維持するためのアプローチとして、CODASYL仕様の機能を制約したDBMSをDEIMS－2プロジェクトの一環として開発した。この場合には、従来ファイル・システムの10％程度のオーバヘッドに目標値を設定したが、これを下まわる6％で実現することができた。[10]

本稿では、CODASYL仕様のほぼフル機能をもたせ開発したバージョンについて主に述べることにする。

3. ソフトウェア構成概要

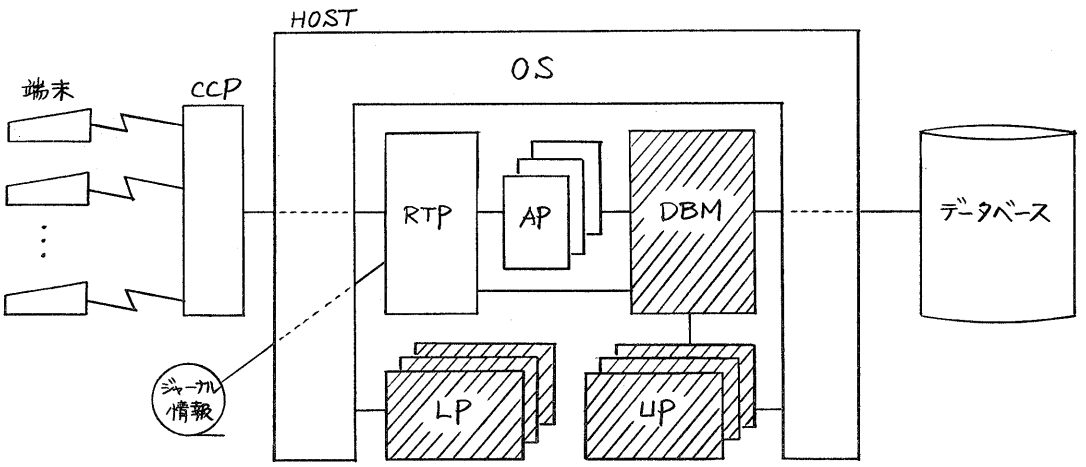
DEIMS-2はCODASYL提案のデータベース仕様をもとに開発したが、CODASYL仕様そのものはデータベース定義および操作の言語仕様であり、DBMSのもつべき機能は概念のみが述べられている。そこでDEIMS-2では、DBMSを実現するのにあたり、以下の3つのソフトウェア群から構成した。

(1) データベース言語処理プログラム（LP）
CODASYL仕様のデータベース定義言語および操作言語を処理するプログラム群
- スキーマ／サブスキーマ・ジェネレータ
- プリコンパイラ

(2) データベース・モニタ（DBM）
業務プログラム（AP）からのデータベース・アクセス要求を果す実行時制御ルーチン群

(3) データベース・ユーティリティ・プログラム（UP）
データベース・システムの運用管理機能としてのユーティリティ・プログラム群
- アロケータ
- 創成ユーティリティ・プログラム
- 再編成ユーティリティ・プログラム
- 再構成ユーティリティ・プログラム
- 統計解析ユーティリティ・プログラム
- 復元ユーティリティ・プログラム
- DEIMSジェネレータ

これらのソフトウェアとOS，RTPとの関係は図.1のようになる。



(注) ▨ DEIMS-2のソフトウェア群
CCP：通信制御処理装置

図.1 DEIMS-2とOS，RTPの関係

4. 機能概要

4.1 データベースの構成

DEIMS-2のデータベースは論理データベース，格納データベース，物理データベース（および仮想データベース）の概念に階層化されている（DEIMSデータ・モデル）.[8] このように階層化したねらいは、物理的データ独立性の達成のためである。DEIMS-2の開発の背景にある大容量データベースを実現するにあたり、以下の点を考慮する必要があった。

（1）データベースの成長特性やデータの地域的局所性等に伴なうデータ格納制御やデータ格納域の再構成等の柔軟性の必要性

（2）高ファイル効率，および高トラヒック・トランザクションに対する高アクセス効率の必要性

これらを実現する上では、当初のCODASYL仕様のスキーマでは、データ構造と記憶構造に関する要素の記述が混在しており、上記（1），（2）からくる機能要求を吸収するには物理的データ独立性が不十分であった。そこで、以下の基準にもとづき，多階層化データベース、即ち多階層化スキーマに拡張した。

（1）データ構造の要素として、性能に関する要素は含むべきではない．即ち、性能に関する要素はAPに影響を与えることなく変更できることが望しい。

（2）データ構造におけるある要素が、実現上、データの独立性や完全性をそこなうものであってはならないし、そのような要素は含むべきではない。

（3）さらに（1）（2）に該当する要素はOSとは独立とする。

これらの基準にもとづいたDEIMS-2データ・モデルとCODASYLデータ・モデルを比較したのが図.2である。なお，最近のCODASYLの動向でも、スキーマから前述の要素が除かれ、スキーマの記憶表現を表わす記憶スキーマ（storage schema）の概念があり、これを記述する言語は「データ記憶記述言語（Data Storage Description Language（DSDL））」と呼ばれている。

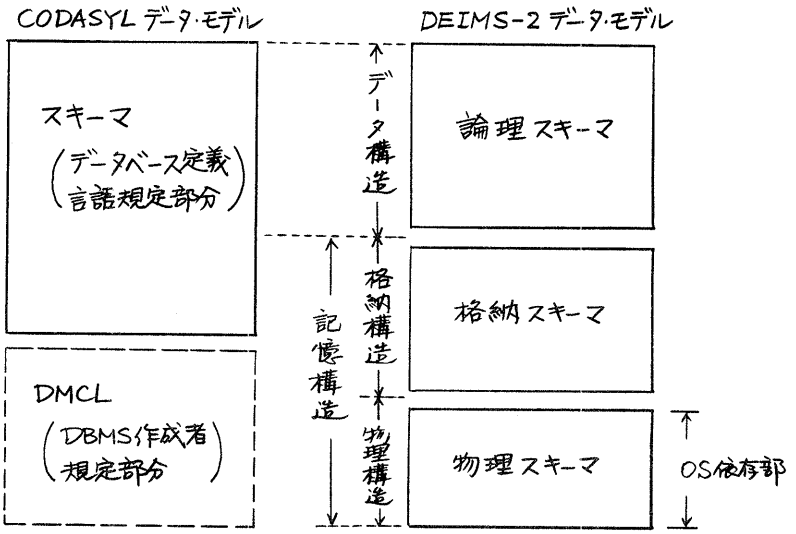


図.2 DEIMS-2とCODASYLデータ・モデルの比較

DEIMS-Zの各々のデータベースの概念は以下のとおりであり、それらの関係を図.3に示す。

(1) 論理データベース

論理データベースはデータベースの論理的な側面，即ち

① データの持つ特性（名前、属性）

② データの論理的包含関係

③ データとデータ間の階層関係

でとらえた概念である。この論理データベースの構造をデータ構造という。

(2) 格納データベース

格納データベースはデータベースを論理的な格納空間としてとらえた概念であり、その構造を格納構造という。この格納空間はOSのデータ編成（ファイル編成）やアクセス法からは独立した、DBMS固有のデータ管理空間である。従って、

① 格納空間の大きさ、構成の方法

② データの格納方法

③ データ間の関係の実現方法

等を，性能を考慮して、論理データベースとは独立に構成できる。

(3) 物理データベース

物理データベースはデータベースを物理的な側面からとらえた概念であり、その構造を物理構造という。DEIMS-Zでは主にOSとの構成要素と対応づけている。

(4) 仮想データベース

仮想データベースはAPに必要な論理データベースの一部を切り出し、それをあたかも一つの独立した論理データベースであるかのようにとらえた概念であり、その構造をサブデータ構造という。仮想データベースはCODASYLのサブスキーマと同一の概念である。

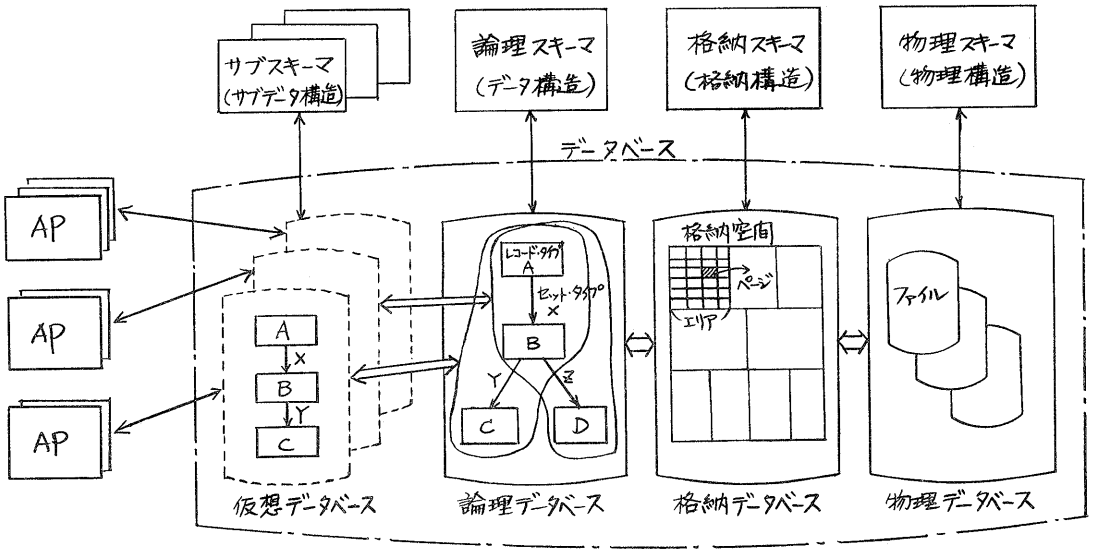
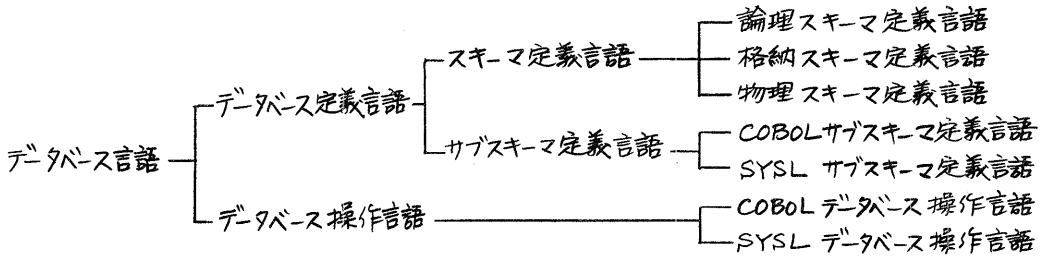


図.3 DEIMS-Zのデータベースの概念

4.2 データベース言語

DEIMS-2はCODASYL仕様をベースにほぼフルセットの仕様に近いが、一部仕様については実現していない。また、前節で述べたように、スキーマを多階層化したことにより、CODASYLの記述形式に準拠しているが、記述内容は異なっている。DEIMS-2のデータベースの言語は以下の体系となっている。



(1)論理スキーマ定義言語

論理スキーマ定義言語は論理データベースを定義する言語である。

1974年CODASYL DDL JODとDEIMS-2の論理スキーマ定義言語の相違と、現状の実現範囲を示したのが表.1である。表中、備考欄で*印の表示があるものは、セキュリティあるいはデータベース管理者のための管理機能であり、現状では実現していないがデータベース定義言語とは別の言語体系として位置づけるべきと考え、除いている。

(2)格納スキーマ定義言語

格納スキーマ定義言語は格納データベースを定義する言語である。この格納スキーマ定義言語に前述の物理的データ独立性に関する要素を移行させており、このことによりまた大容量データベースに関する配慮も吸収することができた。

格納スキーマ定義言語の

表.1 CODASYLとDEIMS-2のデータベース定義言語の比較

CODASYL データベース定義言語	DEIMS-2 データベース定義言語		備考
	論理スキーマ定義言語	格納スキーマ定義言語	
スキーマ記述項	○	○	
CALL節			*
PRIVACY節			*
SCHEMA節	○	○	
エリア記述項	○	○	
AREA節	○	○	
CALL節			*
PRIVACY節			*
TEMPORARY節			
レコード記述項	○	○	
CALL節(DATA,RECORD)			*
CHECK節	○*		
Data-Base-Data-Name節	○		
ENCODING/DECODING節	○		
LOCATION節		○	
OCCURS節	○		
PICTURE節	○		
PRIVACY節(DATA,RECORD)			*
RECORD節	○	○	
RESULT節	○*(ACTUAL/VIRTUALのみ)	○*	
SOURCE節	○*(同上)	○*	
TYPE節	○		
WITHIN節	○	○	
セット記述項	○	○	
CALL節(MEMBER,SET)			*
DYNAMIC/PRIOR節		○(DYNAMICは○*)	
KEY節	○		
MEMBER節	○	○(LINKED OWNER)	
ORDER節	○		
OWNER節	○		
PRIVACY節(MEMBER,SET)			*
SEARCH節		○*	
SELECTION節	○		
SET節	○	○	

(注) ○ : 現状のサポート
○* : 現状ではノンサポート
* : 他言語によるサポート想定

記述項と記述内容を表.2に示す。

格納データベースの格納空間は基本的に格納エリア、ページ、レコードの要素から構成され、格納エリアは固定長サイズのページから、ページは可変長サイズのレコードから構成される。この格納空間では、データベースに格納されるすべてのレコード(オカレンス)を一意に識別するものとして、データベース・キーがある。このデータベース・キーはエリア番号，ページ番号，レコード番号から構成され、このレコードがデータベースから削除されるまで不変である。

大容量データベースに対し、セットの構成法とインデックスの構成法を考慮した。セットの構成法は、ポインタ・チェーン法と物理順配列法のいずれかをオプションとしている。ただし、物理順配列法のセットは木構造であるという制約がある。

インデックスの構成は、直接対象レコードを位置づけるレコード・インデックスと、対象となるレコードを含むページに対し、位置づけを行なうページ・インデックスがある。ページ・インデックスのインデックスはエリア毎に閉じて構成される。レコード・インデックスはORDER SORTEDを持つ任意の階層のセットで可能である。

表.2 格納スキーマ定義言語

記述項	記述内容
スキーマ記述項	・スキーマ名定義
エリア記述項	・エリア名と属性(データ用,インデックス用)定義 ・ページ数，ページ・サイズ定義 ・オーバフロー・ページの構成定義 ・初期創成時の格納許容率定義
レコード記述項	・レコード名定義 ・ロケーション・モード(CALC, VIA)定義 ・IDENTIFIERの実現法定義
セット記述項	・セット名定義 ・セット構成法定義 ・ORDER SORTEDのインデックス定義 ・PRIOR/OWNERのポインタ定義
インデックス記述項	・インデックス構成法定義

(3)物理スキーマ定義言語

物理スキーマ定義言語は物理データベースを定義する言語である。物理的要素のうち、OSに依存する要素を定義する。従って、格納データベースの格納空間とOSの管理空間と対応づけることを行なうが、その他運用管理に関する付加的なものの定義を行なう。

表.3に物理スキーマ定義言語の記述項と記述内容を示す。エリアとファイルの対応関係は1対n、n対1が可能である。また、ファイルと記憶媒体、装置との割当て、対応はジョブ制御言語(JCL)によりなされる。

表.3 物理スキーマ定義言語

記述項	記述内容
スキーマ記述項	・スキーマ名定義
エリア名記述項	・エリア名定義 ・ファイル定義 ・ファイル内のブロック数割当定義 ・ジャーナル取得単位定義

(4) サブスキーマ定義言語

サブスキーマ定義言語は仮想データベースを定義するものである。DEIMS-2ではCOBOLとSYSL(SYStem description Language)をホスト言語としたデータベース機能を可能としていることから、COBOL, SYSL用のそれぞれのサブスキーマ定義言語がある。両者の機能上の相違はほとんどなく、同一データベースに対し、各々のホスト言語で書かれたAPのアクセスが可能である。両者の間のデータ属性の違いはDBMにより変換される。

(5) データベース操作言語

データベース操作言語(DML)はAPによりデータベースにアクセスするときホスト言語の中で使用される。COBOL, SYSLのホスト言語に対し、表.4に示すDMLを提供している。

表.4 DML一覧

検索系DML	更新系DML	制御系DML	
FIND*	CONNECT	ACCEPT	IF
FETCH*	DISCONNECT	ATTACH*	READY*
GET	ERASE	DETACH*	RESCUESTART*
	MODIFY	FINISH*	RESCUEEND*
	STORE		

(注) *: DEIMS-2で追加したDML, 4: DEIMS-2で拡張したDML

(i) 追加DML

DEIMS-2では4.4節で詳述するように、データベース・アクセス単位を宣言するためにATTACH/DETACH命令を、また障害の救済単位を管理するための宣言としてRESCUESTART/RESCUEEND命令を追加している。また、FIND命令とGET命令を複合したFETCH命令を追加し、APの記述性、操作性の向上をはかっている。

(ii) 拡張DML

DEIMS-2ではレルム名に対する変数指定(一意名指定)を可能とし、APの記述性の向上をはかっている。このレルム変数はREADY, FINISH, FIND, FETCH命令のレルムに関する操作において使用できる。その他、データベース・キーの概念をAPから直接みえないようにしたことにより、それにかわる検索キーとなる項目(IDENTIFIER)による直接検索をレコード選択式に設けた。また、READY命令ではINITIAL-LOADモードを追加し、初期創成モードの機能を与えている。

4.3 言語処理機能

4.2節で述べたデータベース言語を処理するプロセッサとして、ジェネレータとプリコンパイラがある。

(1) ジェネレータ

データベース定義言語に対するプロセッサとして、論理スキーマ・ジェネレータ、格納スキーマ・ジェネレータ(物理スキーマ定義言語処理を含む)、COBOL

サブスキーマ・ジェネレータ、SYSLサブスキーマ・ジェネレータがある。それぞれ、スキーマあるいはサブスキーマを作成し、必要に応じデータベース管理者用のリストを出力する。スキーマとサブスキーマはデータ・ディレクトリに格納され、ジェネレーション時、プリコンパイル時、DBMの実行時に参照される。

(2) プリコンパイラ

DMLに対するプロセッサとして、COBOLプリコンパイラ、SYSLプリコンパイラがある。DEIMS-2では、ホスト言語のコンパイル処理前に、プリコンパイラにより、データ・ディレクトリに登録されているサブスキーマを参照して、DMLをホスト言語コンパイラの処理可能な命令に変換する。また、サブスキーマで定義されたレコードをAPで使用できるように利用者作業域(UWA)の展開と、AP-DBM間で使用するシステム通信域(SCA)の展開も行なう。

これらのプロセッサの処理の流れを図.4に示す。

4.4 データベース・アクセス制御機能

オンライン実時間データベース・システムにおいて、データベースへのあらゆるアクセスを制御する実行時ルーチンの集まりがDBMである。DBMは多階層化されたスキーマに対応した処理を行ない、各々の要素が他に影響を与えないようなプログラム構成をとっており、今後の記憶構造の多面化に対し、対処しうるようになっている。DBMの主機能はDMLにもとづく、データベースの検索、更新処理であるが、それらの処理におけるデータベースの完全性を保証するために以下のような機能を持っている。

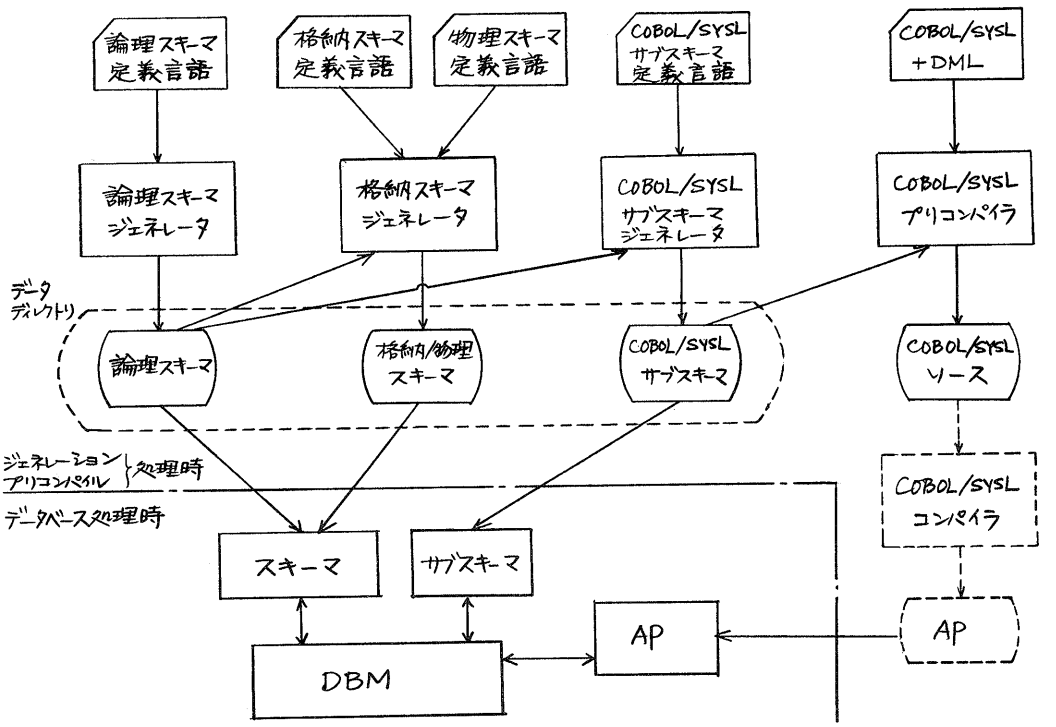


図.4 プロセッサと処理の流れ

(1) データベース・ユニット管理機能

DBMからみたデータベースへのアクセス管理単位を設け、この単位により、同時走行の制御、入出力の管理等を行なう。この単位をデータベース(アクセス)ユニット(DBU)という。このDBUはATTACH命令により確立され、DETACH命令で消滅する。この間でスキーマ,サブスキーマとの対応づけがなされ、READY命令でレルムをレディすることにより、他のDMLが使用可能となる。

(2) レスキュー・ユニット(救済単位)管理機能

DBUのデータベース更新処理において、障害やAPの論理矛盾が発生すると、データベースの破壊、汚染が生じうる。このとき他のDBUに影響を及ぼさず、そのDBU内の処理を無効とし、回復可能とする単位がレスキュー・ユニット(RU)である。RUはRESCUESTART命令により確立され、RESCUEEND命令により解除される。RUによる回復処理は、ジャーナル情報にもとづくロング・リカバリとロールバックによるクイック・リカバリがある。

DBUとRUの基本的な関係を図.5に示す。

(3) 排他制御機能

DBUのコンカレント処理範囲として、レルム単位とページ単位の排他制御機能がある。これは、DBUのREADY命令のレルム使用モードにより決定される。

(4) バッファ管理機能

データベース処理において、データの入出力回数を最小とするバッファ管理をLRU法をベースに行なう。

(5) データ・ディレクトリ管理機能

スキーマ,サブスキーマのロード、アンロードの効果的管理を行なう。

(6) サポート機能

障害事前処理としてのジャーナル情報やロールバック情報の取得、障害時処理としての通知、閉塞、障害事後処理としての回復、解除等のサポート機能を行なう。

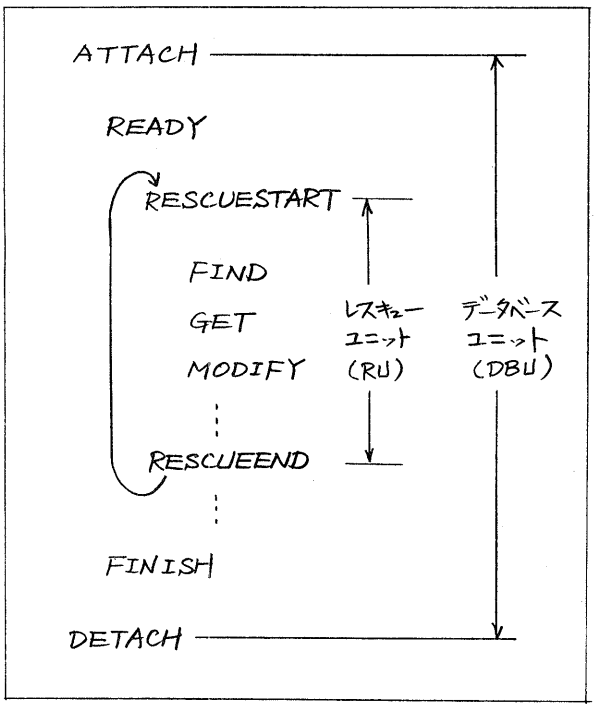


図.5 DBUとRUの関係

4.5 運用管理機能

運用管理のための機能として3章で述べたUPがある。これらについては特に、大容量データベースの運用管理という面から、①多量の原データからの効率のよい創成、②再編成・再構成の局所化等を考慮し、UPの多重走行、I/O回数が最少となるようなバッファ管理,処理途中の障害時の対処などをはかっている。

5. おわりに

本稿では、通研で実用化したCODASYL形DBMSであるDEIMS－2について、その概要をのべた。DEIMS－2により、現状のハードウェア上で、CODASYL形DBMSによる大容量オンライン・データベース・システムを構築可能とする技術を確立することができた。今後は、リレーショナル・モデルにみられるような問合せ言語インタフェース、分散形処理等の機能拡充を予定している。

参考文献

〔1〕 Programming Language Committiee : DBLTG 73001.02 - The COBOL Data Base Facility, 1973.

〔2〕 CODASYL DDLC : CODASYL Data Description Language, Journal of Development, 1974.

〔3〕鈴木・山多：CODASYL形データベースの記憶構造に関する一考察，情報処理学会第17回全国大会，1976.

〔4〕藤・河津・高橋：CODASYL形データベースのオンライン大容量システムへの適用法，情報処理学会第17回全国大会，1976.

〔5〕佐藤・北村・川手：COBOL，SYSLをホスト言語とするデータベース用言語について，情報処理学会第17回全国大会，1976.

〔6〕岡本・加藤：大容量データベースにおける救済方式についての一考察，情報処理学会第17回全国大会，1976.

〔7〕岡田・佐藤・伊藤・田中：CODASYL形データベースにおける運用管理機能の一体系，情報処理学会第17回全国大会，1976.

〔8〕 Suzuki, Toh, Kawazu : Multi-level Structures of the DBTG Data Model for an Achievement of the Physical Data Independence, Proceedings Very Large Data Bases, 3, 1977.

〔9〕鈴木・佐藤・田中・川手：CODASYLデータ・モデルの多階層化，情報処理学会第18回全国大会，1977.

〔10〕岡田・中川・木戸：データベース管理システムにおける性能評価に関する一考察，情報処理学会第19回全国大会，1978.

〔11〕杉浦・高橋・藤・益井：データベース管理システムの実用化，通研実報，27，No.4，1978.

〔12〕河津・佐藤・南・北村他：DEIMS－2の言語方式，通研実報，27，No.4，1978.

〔13〕岡本・鈴木・中川・加藤他：DEIMS-2のデータベース制御方式，通研実報，27，No.4，1978.

〔14〕岡田・石賀・伊藤他：DEIMS-2のデータベース運用管理方式，通研実報，27，No.4，1978.

〔15〕植村：CODASYLデータベース用共通言語の進展'78，情報処理学会データベース研究会資料5－3，1978.

〔16〕鈴木・田中：多階層化スキーマにおける格納構造の構成法，情報処理学会第19回全国大会，1978.