TM-1239

次世代データベースに向けて

横田　一正

November, 1992

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

次世代データベースに向けて

琵琶湖ワークショップ'92 (ICOT NDB-WG 合宿) —

次世代データベース・ワーキンググループ
(財) 新世代コンピュータ技術開発機構 (ICOT)
(横田 一正 (編))

1992 年 11 月 6 日

1 はじめに

1991 年度の ICOT の次世代データベース (NDB) ワーキンググループの活動のまとめとして 1992 年 3 月に琵琶湖のほとりで合宿 (ワークショップ) を開催した。本稿はそのワークショップでの発表と議論の内容を、編者の責任で以下の項目にまとめたものである。

- データと情報
- 情報の表現
- ビューと状況依存
- オブジェクト識別子
- 質問言語
- 永続システム
- 分散と開放
- その他
  - 並列
  - トランザクション
  - オブジェクト指向データベース
  - ユーザ・インタフェース
  - 応用

なおワークショップの内容は 付録 として付加しているので、詳細はそれを参照されたい。

2 データと情報

理想的には、少ないデータ量で多くの情報量が表現された (つまり効率的に表現された) データを持ったデータベース (DB) に対し、高機能な操作が提供され、それらが効率的に処理できることである。

1) 情報の階層

マルチメディア情報や科学データを考えれば、DB に蓄積すべき一次データ (一次情報) は爆発的に増えている。ここで考えなければならないのは

- n 次情報から $n+1$ 次情報へと次々に新しい情報が作ることができる
- n 次情報から複数の $n+1$ 次情報を作ることができる

ことである。理想的には $n+1$ 次情報は n 次情報のより効率的な表現になっていることであるが、現実的には、ビュー、要約、メディア変換、仮説、などさまざまな場合が考えられる。これらを可能にし、統一的に管理するDB である必要があるだろう。最近科学データではこれらをメタデータの階層として総称することもある。

2) 情報の品質

情報の品質が高いことが望ましいのはいうまでもないが、一般的には低品質か高品質かを決める判断基準はない。ただし、特定の問合せシステムに応じて決めることはできるだろう。一般的により良い品質を保つためには

- 内容のチェックが可能な状態にしておく
- 一貫性制約を入れておく
- 冗長な情報を発見し削除する

などが必要となろう。

3) データの高品質化

大量のデータを対象とするには、DB の設計者 / 利用者がその品質を維持するだけでなく、それを支援するシステムが必要となる。それには以下のものが考えられる。

- n 次情報から $n+1$ 次情報を自動生成する。これには要約や抽象化が考えられる。
- データから一貫性制約を発見するための知識発見。これは不要データの混入を防ぐ。
- データからルールを発見する知識発見。これはデータ量を削減に貢献する。
- オブジェクトの同定によりデータ / ルール間の冗長性を発見し、不要データを削除する。
- オブジェクト間の包摂関係によりデータ / ルール間の冗長性を減少させる。
- データ間の類似性等からオブジェクトやスキーマを進化させ、DB を自己組織化する。

4) 知識アーカイブ

この言葉はまだ熟していないが、必要条件のひとつとして、上記のような情報の階層を含み、漸次的なスキーマ / アクセス・メソッドの構築の可能性を持つことを考えることができそうだ。

3 情報の表現

DB の応用は拡大しており、今後の DB を考える際には、対象とするオブジェクトをいかに考えるかが重要な問題となってきている。そのために、どんな情報を表現すべきかの議論と、情報をいかに表現し取り扱うかについての2つの提案と議論があった。

1) 何を表現するか

以下の表現の必要性が議論された。

- 部分オブジェクト　　部分情報、制約表現、選言、曖昧さ、矛盾などを含む
- 類似などの曖昧な関係
- 自然言語の意味情報
- 状況依存情報
- ユーザ情報 — ユーザ・モデルも DB の一部
- 時空間情報 — 地図のような情報には必要

2) 情報構造の統一理論 (大堀)

DB 理論、ドメイン理論、自然言語理論、状況理論の4つに分野における情報構造 / 表現が類似していることは従来から指摘されており、これらのいくつかを統合しようとする研究はこれまでにも多くあった。これらをにまたがる統一的な情報理論を作ろうという問題提起と動向の紹介が行われた。これに関して以下の指摘と議論が行われた。

- 数学的な土台としてはカテゴリ理論が考えられる。
- さまざまな抽象化のレベルを取り扱う必要があるので新しい理論が必要となる。
- たとえば集合の扱いについてもさまざまあり、意味論の整理が必要である。
- 多相型を含むモジュール理論との関連が必要である。
- *QUIXOTE* はこの先駆けになっているのではないか。

3) 意味構造の表現 (清木、北川)

情報の多くは自然言語で表現されているが、自然言語の意味も含めて考えるのは、従来の DB 処理では困難である。そこで情報を自然言語の意味空間として表現し、質問時にその解釈を動的に行うというモデルが提案された。これに対し以下の議論が行われた。

- これは状況理論の欠点 (空間概念、距離の動的な変化がない) を補っている。
- 現在は検索に限定しているが、DB 全体を考えると、情報のデータ構造と意味構造の関連付けが必要となる。
- 分脈の切り替えにはさらに観測者の時空間の認識が必要となる。
- 解釈が動的に行われるとしても意味空間の定義は事前に行う必要があり、これが現実的かどうかを考える必要がある。

4 ビューと状況依存

ビューとはオブジェクトに対するある視点からの見方であり、状況依存オブジェクトとは状況によって値や意味を変えるものである。もしビューが、オブジェクトのデータ構造に対する視点だけでなく、意味内容に対する視点をもちかつそれがオブジェクトを作るものであれば、ビュー(・オブジェクト)は状況依存オブジェクトを包摂しているといえる。

ビューに関連して以下の議論が行われた。

1) オブジェクトの抽象化

ビューの機能としてオブジェクトの抽象化があるが、これには階層化を考える必要がある。抽象化はもともとプログラミング言語 (PL) を起源としているもので、階層化を考えるには、基本型と関数型による階層化を行っている型理論が参考になろうとの指摘があった。

2) ビューと 3 層スキーマ

ビューは古典的な 3 層スキーマの中では外部スキーマとして位置づけられているが、3 層スキーマがどれだけ有効かについて、

- 3 層に分けるのは現実にあっていない
- 3 層スキーマは非常に柔軟であり拡張性に富んでいる

などの賛否両論あった。

3) ビューと DB

生データからのオブジェクトの定義 (情報の切出し、記号表現) は、従来の DB ではこれはプリミティブとして唯一つ与えられていたが、コンティニュアス・オブジェクトなどでは複数種類のオブジェクト定義が必要となろう。これをビューと捉えれば、DB はビューの集合となるが、3 層スキーマの範囲外でもある。ただしこの場合、オブジェクトの同定の問題が生じる。

4) ビューの機能

ビューの基本的な機構として、仮想化、多重化、階層化の 3 つが考えられる。演繹 DB (DDB) のルールもこの分類では仮想化と多重化に当たっている。しかしビューは便利な言葉で何にでもこの言葉で修飾することがで

き、意味的な多義性で混乱を生じることが多い。たとえば他に、文脈依存性、時空間依存性、仮説推論なども考えられ、これらの混乱を避けるために整理する必要がある。またオブジェクト指向 DB (OODB) のメソッドについては、情報隠蔽と関連させる必要もあるかについての議論もあった。

5) ビューの課題

新たなビューの機能としては、状況依存オブジェクトのように、DB の意味そのものを動的に変えることが考えられる。この枠組としては DB のメタ定義が必要である、との問題提起があった。

また3層スキーマ、ビュー、抽象データ型の3つの概念の融合が必要であるとの問題提起もなされた。

ビューが本質的となる応用としては、対話理解、ロボット社会、Adaptive システム、開放システムなどが挙げられた。

5 オブジェクト識別子

実世界のオブジェクトを表現するために、値に基づく (value-based) アプローチと (オブジェクト) 識別性に基づく (identity based) ものがある。最近では両者の統合についての研究もあるが、ここではオブジェクト識別子 (OID) そのものについて以下の側面から議論が行われた。

1) OID の役割と表現

OID の役割と表現を分けて考える必要があることが議論された。役割としては、OID に基づいたオブジェクト表現のためであるが、表現としては、利用者に見えないポインタとしての表現と値に基づいた表現の2種類を考えることができ、役割と表現を混同すべきではない。

2) OID の生成

何をオブジェクトとするかはユーザの責任である。したがってユーザが生データからオブジェクトを定義する (OID を割り付ける) 枠組を与える必要がある。DBMS は、生データと定義されたオブジェクトの両者を管理する必要がある。例として、地理情報や遺伝子データ、ハイパーメディアが取り上げられた。また生成のための一貫性制御の必要性も議論された。

3) OID と質問言語

OODB における質問言語を OID という視点から考えたときの問題点が議論された。

- OID をポインタとして表現したときの質問言語の体系化
- 質問処理時に新しい OID を副作用として生成する場合の最適化

ポインタ表現と副作用の両者を避ければ、質問言語の体系化と最適化は容易だとの予測も示された。

4) OID とシステム

OID をもったオブジェクトを管理するシステムの問題点が議論された。

- 分散 / 開放システムで OID はどの程度大域的か
- OID 自体とクラスタリングを関連づけられるか
- OID とロックの粒度は関連させられるか

6 質問言語

1) 質問言語の特性

関係 DB (RDB) については質問言語に関し、代数、論理、最適化可能なクラス、現実の質問言語 (SQL) の4つがほぼ一致しており、特徴としては以下のものがあった。

- 宣言的集合操作
- 結果の閉包性
- 適切な表現力 (関係完備性)

新しいモデルを考えるときこれらをどの程度継承すべきか、を考えることが必要である。

2) DDB の質問言語

RDB の拡張としての DDB では Datalog¬ を中心にした言語の階層があり、宣言的集合操作と閉包性は満たしており、関係完備性は階層に応じて拡張されている。ただしこれらは質問言語の能力としての研究であり、ルールで表現されたものが質問言語かどうかには疑問があるとの問題提起もなされた。

3) OODB の質問言語

まず、現状の商用 OODBMS の言語は内部スキーマ記述言語に過ぎず、質問言語という問題が看過されている問題が指摘された。次に、OODB に RDB の質問言語の特性を継承することについては

- ポインタとしての OID の扱いの難しさ
- 新しい OID は副作用となるので最適化が困難なこと
- 結果の閉包性を諦めた研究もあり、閉包性の重要性を再考する必要性
- 適切な表現力のクラスを探ることの重要性
- ナビゲーションを宣言的言語に埋め込めるか

などの問題点があることが議論された。この議論でのポイントは、OID の扱いと表現力のクラスであった。

4) ユーザ・インタフェース

質問言語のインタフェースとして、RDB における QBE の成功と対比させながら、GUI、オブジェクトの視覚化、ハイパーメディア、さらに OODB における視覚化インタフェースなどについても議論された。

将来的には単なる GUI ではなく、マルチメディアや自然言語を使い、ユーザ・モデルを組み込んだ形でのインタフェースの必要性も議論された。

5) ハイパーメディア

地図 DB を例題に、ハイパーメディアに関するデータの視覚化の他に、質問作成のための視覚化の問題が提起された。

7 永続システム

DB の重要な特徴の一つに永続性がある。このために実装技術を中心にして問題提起と議論を行った。

1) 永続システムの現状

現在進行中のシステムには以下のものがある。

海外: Bubba/FAD (MCC)、Cricket (U of Wisconsin)、Eos (INRIA)、Mneme (U of Masachusetts)、Napier88 (U of St. Andrews, U of Adelaide)

国内: DSR (東大; 加藤)、Muse (ソニー CSL; 天満)、PCOB (IBM; 三ッ井)、P3L (東大; 鈴木)、WAKASHI (九大; 牧之内)、Machiavelli (沖; 大堀)、*Quixote* (ICOT; 横田)、SMASH (筑波大; 清木)

2) 永続システムのキー・テクノロジー

永続システムを実現するためのキー・テクノロジーとして以下のものが挙げられ、議論された。

- 永続ヒープ — 永続 root からの到達可能性による永続性
- ポインタ・スイズリング — 「大規模」永続空間から「小規模」仮想記憶空間への写像

- 仮想記憶マッピング — 下記によって効率が得られる
  - 「二重バッファリング」が不要
  - フォーマット変換のない統一的なオブジェクト表現
    ソフトウェアによるアドレス変換不要 (ポインタ・スイズリング不要)
- シャドウ・ページ — 安定性と回復力のため

3) 今後の課題

今後の永続システムのための問題点として以下のトピックについて議論を行った。

- 分散環境下での永続性の扱い

  この検討すべき問題点として、共通の型システム、並行オブジェクトの永続性と共有、名前空間の管理、異種計算機間での整合性、DBMS としての機能の保持、などが挙げられた。

- 記憶域管理

  単一記憶ストア、オブジェクトの消去、ポインタ・スイズリング、クラスタリング、などが今後とも重要であるとの指摘があった。

8 分散と開放

開放分散システムは DBMS にとって今後の大きな課題である。開放と分散という 2 つについて以下の議論があった。

1) 分散システムとは何か

分散処理とは、複数の場所、マシン、言語にまたがって、データ、オブジェクト、メソッド、型付けなどを分散させることである。CPU、記憶域などを分散させた並列処理と異なるのは、分散処理では

- 通信コストが高いこと
- 開放システムが考えられること

などである。また本質的ではないが、分散処理では異種マシン & ソフト間の通信を考えることが多い。

また、すべてのノードが対等な分散システムの他に、サーバ / クライアント・システムが大きな流れとしてあるとの指摘もあった。

2) 分散システムの現状

現在進行中の分散システムとして、

- ストリームに基づいた関数型パラダイムにより、単純データを分散させるアーキテクチャ (清木)
- 異なった型付けの言語間で永続データを共有するシステム (加藤)

の紹介の他に、異種分散システムにおける並行制御のサーベイがおこなわれた。

3) 開放 (分散) システムとは何か

開放システムとは、C. Hewitt & P. de Jong (1984) によれば

- 透明性概念の否定

  大規模分散環境
  - 地理性・位置性 (トポロジー・局所性) は、概念モデルとして与えられなければならない。
- デュープロセス
  仮説と目標を区別し、双方を別物として推論を行なう。
- 自己参照、自己知識、自己進化
  (他者との) 交渉の機能と自分の歴史、能力を検索できる機構が必要である。

の要素を考えなければならない。

4) 開放(分散)システムの問題点

開放システムを考える上での問題点として以下の議論が行われた。

- 大域的な制御が不可能であるが、一方では OID のようにある程度の大域的な仮定が必要となる。
- 透明性の概念の否定は、従来の分散システムの方向を大きく覆す挑戦的な試みだが、所の MD-Computing のように地理性を導入するとか、新しい概念が必要となる。
- 推論や無矛盾性の検査は閉じた系の中で行われるもので開放性とは両立しない。これを可能にするためには、動的に閉鎖システムを仮定し、動的に公理系を選択するなどの機構が必要となる。知識ベースの場合、大域的には矛盾を含むが局所的には無矛盾という知識の扱う必要があるが、この場合にも利用できよう。
- ユーザ自身がどこにいるかをシステム中に配慮する必要がある。ユーザがコンピュータ内にいるという概念であるが、これはハイパーテキストでのガイド(cf. guided-tour of a manshon)、位置、軌跡などという概念と対応するかも知れない。

5) マルチエージェント・システム

OO パラダイムではクラス構造が固定であるとか厳格過ぎて、開放システムには向かないように思える。協調計算、交渉、ボトムアップでの構築などを考えれば、エージェント指向パラダイムで考えたほうが良いかもしれない。

9 その他

9.1 並列

並列 DBMS について以下の議論を行った。

- 並列マシンの分類としては、スーパスカラ、共有メモリ、分散メモリの基本的なタイプと、それらのハイブリッド型が考えられるのではないか。
- 並列 DBMS では、記述、データ配置、データ複製、フォールト・トレランンス、ネットワーク形態などを考えることが必要。
- 並列処理ではスケーラビリティが重要だが、並列 DBMS ではデータ配置と関連しているので難しさがある。
- 現在の OODB には基本演算がないので OODB 用並列計算機の設計は難しい。並列 OOPL の成果が利用できるのだろうか?

9.2 トランザクション

トランザクション管理および並行処理制御に関しては、原子性と直列可能性という二つの概念を中心に論じられてきた。しかし、応用分野の拡大にともない、OODB が使用されるような環境下では、これらの概念をいかに緩和するかが重要である。

- 原子性
  原子性を保証する処理の単位を論理的に小さくするために、入れ子型トランザクションモデルのサブトランザクション単位での原子性を考えることが提案されている。
- 直列可能性
  データ操作の意味やユーザの協調作業環境を反映した制御法を考慮することが重要である。西尾の最近のモデルでは、DB 内の部分データ(領域)に課せられた一貫性制約と協調作業環境を反映することによって、領域データの直列可能性を中心とした並行処理制御方式となっている。

さらに検討すべき点として以下の指摘がなされた。

- OID によってアクセスされる集合が予想可能で、かつその情報をスケジューラが活用できるなら、DB の並行処理性能は向上する。
- OODB では、協調作業に必要なデータ全体を最初に二次記憶から共用のメモリキャッシュにチェックアウトして、そのデータを利用する形態が多い。その場合は、メモリキャッシュのデータを対象した並行処理制御がなされ、作業完了後のデータは、新たなバージョンとして二次記憶にチェクインされる。
- OODB におけるオブジェクトへの操作の意味を考慮した並行処理制御法が限定的ではあるが提案され始めている。
- DDBMS ではルールを含めた「管理システム」という意識が薄かった。今後並行処理制御方式をしっかり考える必要がある。

9.3 オブジェクト指向データベース

教科書に書かれるべき DBMS 技術という点から考えると、現在の OODB は、内部スキーマの記述に相当するものばかりで、概念モデルに関する決定的なものがない。以下の研究が必要であろう。

- スキーマ間のオブジェクト・マイグレーション
- スキーマ、オブジェクトの進化
- ビュー (4 節参照)
- 質問言語 (6 節参照)
- 制約の表現と処理

  Weldon のキー PFD の研究が役に立つか？
  そもそもキー自体が有効か？

9.4 ユーザ・インタフェース

GUI を始め、ハードウェアの進化に伴って人間と計算機の間のチャネルのバンド幅が拡大している。今後これをさらに拡大するためには以下の考慮が必要である。

- 通信媒体 / インタフェース

  マルチメディア
  自然言語
  人工現実環境

- ユーザのモデル化

  ユーザに特化された質問事項・語彙
  - 各利用者の知識レベルに応じた質問機能
  ユーザ (・モデル) 自体が DB の一要素

- 計算機が利用者を教育していく

  有能な秘書
  - 有能なチュータ
  - 思考実験の環境

9.5 応用

応用とデータモデルの発展の関係を、地理情報処理を例にしながら議論を行った。

- 応用が先かデータモデルが先かは決められるものではなく、相補的な関係で発展するのではないか。
- 地理情報処理に適したデータモデルはまだないといってよく、既存のデータモデルの写像するのではなく、何が既存のデータモデルに不足しているのかを明らかにすることが先決ではないか。その結果必要であれば新しいモデルを作れば良い。
- 地理情報処理では質問自体が複雑なので、質問そのものを視覚化することが必要である。

付録: ワークショップ (合宿) 概要

日時・場所

1992 年 3 月 10,11 日
滋賀厚生年金基金保養センター

参加者

有川 [九大]	大堀 [神]	加藤 [東大]	北川 [筑大]	清木 [筑大]
鈴木 [東大]	田中 [神大]	天満 [ソニー]	西尾 [阪大]	箕原 [慶大]
横田 [ICOT]	吉川 [京産大]	渡 [ソニー]		

プログラム

3 月 10 日	13:00 ~ 15:15	セッション 1　(座長: 横田、記録: 加藤)
		テーマ別総括 (問題点、…)
	13:00 ~ 13:45	応用: 有川
	13:45 ~ 14:30	並列: 鈴木
	14:30 ~ 15:15	質問言語: 吉川
	15:30 ~ 17:45	セッション 2　(座長: 清木、記録: 有川)
		テーマ別総括 (問題点、…)
	15:30 ~ 16:15	ビュー: 渡
	16:15 ~ 17:00	分散: 大堀
	17:00 ~ 17:45	実装: 加藤
	19:00 ~ 21:00	セッション 3　(座長: 吉川、記録: 渡)
		総括を踏まえて自由討論
3 月 11 日	9:00 ~ 12:00	セッション 4　(座長: 大堀、記録: 箕原)
		5 ~ 10 年後に核となる技術 / 残る文献
	13:00 ~ 15:00	セッション 5　(座長: 西尾、記録: 天満)
		-- 上記を踏まえた自由討論
	15:15 ~ 17:00	セッション 6　(座長: 田中、記録: 鈴木)
		自由討論 & まとめ

A　3 月 10 日 13:00 ～ 15:15 セッション 1 (座長: 横田、記録: 加藤)

A.1　応用 (有川)

基盤技術としてのデータベースの 3 要素

- 有川: (1) 高速化、(2) 一貫性保持、(3) 高機能化の 3 つが DB の基盤技術だ。
- 横田: (1) の中に入っている集合操作は、重要なことか?
- 大堀: 少なくとも形式化を考える上で集合演算は重要である。ストリーム、遅延評価などの技術を考えると、(1) と (3) は互いに関連している。

DB モデルの受動的、後追い的輸題から能動的、主導的発展へ

- 有川: DB モデル (および技術) の能動的、主導的な発展はできないか? 従来は、新しい応用からの要求を捉えて、それを満足するような DB モデルを作っていた。それとは逆に、DB モデル、DB 技術を先に作り、それから、新しい応用を生み出したり、他の分野への「輸出」ができないものだろうか。
- 加藤: DB のみならず、計算機科学全般が、ニーズからトップダウンに研究テーマ設定をしているのではないか。

データベースの研究の方向の健全性

- 有川: 現在の研究動向・手法は一つの方向である。
- 加藤: いや、現在でも、たくさんの研究者がいろいろなことを考えながら研究してきた。

地理情報処理における課題

- 有川: 次のような研究課題がある: 境界、補間、高速化、時間空間、地図 = 1 つのビュー、機械可読データと人間可読データ
- 加藤: 計算幾何学上の問題とはどこが違うのか。
- 有川: データが 2 次記憶上にあるという点が違う。
- ?: 地図情報処理にデータモデルはあるのか?
- ?: 最近の OODB ではプリミティブが低くなっており、従来の高水準データモデルに対応するものは確立していないのでは。
- ?: 地理情報処理における集合演算とは?

A.2　並列 (鈴木)

並列データベース処理計算機の分類

- 鈴木: 牧之内による分類
  - 低レベル並列計算機: VLIW、スーパスカラ
    共有記憶並列計算機
  - 分散記憶並列計算機
- 清木: この分類に意味はあるのか?

- 鈴木: 異なる粒度の並列性を抽出するためのハードウェア構成方式を列挙したものであり、意味はある。もちろん、並列性の抽出が処理の異なるレベルにおいて可能であるために、各々の方式を組み合わせた方式が取られるのが一般的である。また、もっと細かく分類することも可能にちがいない。

SIMD マシン (CM, MasPar)

- 加藤: スーパコンピュータなどのベクトル計算機はどうか?

各タイプの並列計算機と、汎用性、スケーラビリティ、等との関係

- 清木: スーパスカラ、共有記憶計算機、分散記憶計算機は互いに反するものではない。これらを組合せて混合型 (ハイブリッド型) の並列計算機を作ることができる。だから、このような比較をしても意味はないのでは?
- 鈴木: もちろん、混合させることはできる。ここでは、それぞれのタイプの特徴を上げてみただけ。

並列化にあたって考えるべきこと

- 鈴木: 記述、データ配置、データ複製、フォールト・トレランンス、ネットワーク形態。

OODB の (並列) 処理をどうするか

- 鈴木: 基本演算がないので、OODB 処理用並列計算機の設計が難しい。RDB よりも低いレベルにプリミティブを設定し、そのプリミティブを高速化するのが一つの方法。もう一つは、低レベルの意味しか書かれていないプログラムから、並列処理に適した高レベルの意味を抽出する。
- 加藤: 並列 OO 言語の成果を取り入られないか。

A.3 質問言語 (吉川)

OODBMS の質問言語を考える視点

- 吉川: RDB の質問言語にある有用な概念の継承:

  宣言的集合操作
  結果の閉包性
  適切な表現能力 (関係完備性)

  これらのうち、あとの 2 つは現在の商用 OODBMS に完全に継承されているとはいえない。
- 大堀: 関係完備に相当するような、OODB における質問言語の「適切な表現力」を探ることは重要な研究テーマだ。

オブジェクト指向データモデルとプログラミング言語を結合するためのアプローチ (Bancilhon and Maier による)

ObjectStore の質問機能: 例

- 吉川: ObjectStore では、ある限られた構文の部分のみを抽出して質問処理を行なっているようだ。
- ?: OODB における質問の最適化は、メソッドが入っているために難しいのでは。
- 大堀: メソッドのパワーが強いからといってあきらめるのは早い。適当なコンビネータの集合を選べば、最適化の可能性はある。

- 吉川: 集合演算とナビゲーション演算のうまい使いわけが重要だ。
- 吉川: ハイパーテキストでは、質問者がデータの中に入る。このとき「ガイド」という概念が重要だ (cf. guided-tour of a manshion)。また、質問者の「位置」、「軌跡」という概念がある。

B　3 月 10 日 15:30 ～ 17:45 セッション 2 (座長: 清木、記録: 有川)

B.1　ビュー (渡)

- 渡: ビューというものの中に、観測者をどう入れるか?
- 渡: ビューは、2 次的なものではなく、最も重要な基礎的概念である。
- 渡: 今までのプログラミング環境では、矛盾をするものを避けてきた。
- 渡: 一時的なオブジェクトを扱う機構が必要。
- 横田: 同一オブジェクトかどうかを決める手段は?
- 渡: 人が明示的に決める。
- 渡: Nifty のアドレスと Junet のアドレスは、視点の違い。
- 渡: ビューとは、共有を表す。何を共有するかどうかをプログラマにやらせないといけない。
- 清木: 制約自身も人により違うので、更新の派生もビューで定義できるはず。
- 渡: OO は、実世界をシミュレーションするために作られた。

B.2　分散 (大堀)

分散とは何か

位置やアーキテクチャ、言語にまたがって、データやオブジェクト、メソッド、型付けを分散させる

分散並列環境にけおける DB アーキテクチャ (清木)

ネットワーク上にデータと低レベル・オブジェクトを分散させる

内容: 分散 DB を実装するためのソフトウェア・アーキテクチャ (プリミティブの集合)

- 大量の (相対的に) 単純なデータ
- ストリームに基づいた関数型パラダイム

問題点:

- 複合オブジェクトの分散: ID を持った複合構造
- 高レベル操作

複合言語永続システム (加藤)

言語にまたがってオブジェクトと型付けを分散させる

内容: 異なった型付言語で永続オブジェクトを共有する方法

- 高階関数共有するためのシステム的方法

- 共有の型の健全性

問題点:

- 多相 / パラメータ付データの共有
- ビューの共有
- クラス階層の共有

分散永続ヒープの簡単な実現と課題 (牧之内)

- C++ ファミリでデータを分散させる

内容: 記憶マップを使ってデータを共有する簡単な方法

- C++ に基づいた言語の階層
- ページングの明示的制御

問題点: DB のための仮想記憶サポート

異種分散データベースシステムの並行制御 (西尾)

メソッドを分散させる

内容: トランザクションの大域的無矛盾性のサーベイと議論

問題点:

- オブジェクトのためのトランザクション理論
- トランザクションの型理論

その他

- OID のレベルの分散処理? 並列処理?
- カテゴリ理論の応用 (または適用)
- 閉鎖的な分散 DB (Multiple DB)
- 開放的な分散 DB (Heterogeneous DB)
- もともと別に作られた DB をどうするか?
- 渡: 長期トランザクションに対する話題は?
  ロックで解決するのは、むごい。(商用) バージョンに任す。マージは、ユーザに任す。
- OID とクラスタリングの関係 (OID) の割当て)
- 戦略: オブジェクトのマイグレーションができる
- 複製
- 渡: ナビゲーションに向くなど、応用に向くクラスタリング
- 大堀: 型理論の応用として考えることができる。ただしサイトにまたがる参照は、高価であるので避けたい。ロングポインタを避けるというのを型に入れることもできる。
- 複製を adaptive にやる?

B.3 実装 (加藤)

進行中の "永続システム" 研究

- Bubba/FAD (MCC)
- Cricket (U of Wisconsin)
- Eos (INRIA)
- Mneme (U of Masachusetts)
- Napier88 (U of St. Andrews, U of Adelaide) と Wilson のアプローチ
- DSR (東大: 加藤)
- Muse (ソニー CSL: 天満)
- PCOB (IBM TRL: 三井)
- P3L (東大: 鈴木)
- WAKASHI (九大: 牧之内)

その他に以下のものもある:

- 大堀(沖)他らのプロジェクト
- DOOD システム: *QUIXOTE* (ICOT: 横田)
- 分散質問処理システム: SMASH (筑波大: 清木)

永続システムを実装するキー・テクノロジーと NDB

- 永続ヒープ — 永続 root からの到達可能性による永続性
- ポインタ・スイズリング　「大規模」永続空間から「小規模」仮想記憶空間への写像
- 仮想記憶マッピング　下記によって効率が得られる

  「二重バッファリング」が不要
  フォーマット変換のない統一的なオブジェクト表現
  ソフトウェアによるアドレス変換不要 (ポインタ・スイズリング不要)

- シャドウ・ページ　安定性と回復力のため

商用システムの現状 (続)

ObjectStore の場合: C++ に基づいた DBPL の特徴をもった永続言語を提供。永続性は他の特徴とは直交している。

よく使われる実装技術

- 索引付け (B-木とハッシュ・テーブル)、
- 質問最適化を持つ連想質問処理
- 二相ロックによる悲観論的同時実行制御 (page grain)
- ログに基づいた障害回復
- 二相コミット
- SQL (近い将来)

新しい実装技術

- サーバ・クライアント構成 (ほとんどはクライアントで行われる)
- 記憶マップ・アーキテクチャ (最新の UNIX の機能)
- オブジェクトの再配置ポインタ
- cashe coherence プロトコルを持つクライアントによるページ・キャッシュ
- 版管理に基づく長期トランザクション
- ハードウェア・アーキテクチャの分散 (まもなく現れる)
- スキーマ進化 (まもなく現れる).

その他　加藤の意見抜粋

- 仮想記憶を利用 (主記憶、CPU、OS 仮想記憶が盛ん)
- ページ・フォールトが起こったときの対処:

  カーネルから、ページを要求する
  カーネルから、追い出すページを指定する

- ページを捨てることは、むかしほど、重要ではない。— むしろ主記憶をふやすべきだ。
- ほとんどの永続システムは、仮想記憶を使おうとしている。
- 難しい問題

  ポインタ・スイズリング
  アドレス変換

- 記憶マッピング・アーキテクチャ — 単一記憶ストア

  プログラムがしやすい
  効率が良い — 二重バッファリングの問題がなくなる

- copy on write
  仮想記憶上に残ったページを write 時に、copy して、dirty なページと pure なページを作る (利用者には見せない)
- 1 つのアドレス空間にするか? 複数のアドレス空間にするか?
- 清木: 複合オブジェクトが扱えるということは、単純なオブジェクトを扱うときのオーバヘッドがあるのではないだろいうか?
- 加藤: ノー
- ページ・レベルの同時実行制御
- 鈴木: プロダクツとしては、ObjectStore が良い — 速い、安い (、うまい?)

C　3 月 10 日 19:00 ～ 21:00 セッション 3 (座長: 吉川、記録: 渡)

C.1　「ワークショップの報告」の行方

ワークショップの報告を SIGMOD Record などに投稿して発表したいという件については、以下の議論があり

- 大堀: ワークショップ全体で方向性がはっきりしていないことと、議論が散々しているということから SIGMOD Record の趣旨にそぐわないのではないか

- 田中: 全体の報告をまとめる作業を行なっていく内に実は方向があることに気が付くこともある
- 横田: とりあえずまとめた形にして、その結果どこに発表するのが適切かを検討すべきだ

ということで、まとめを行うことになった。そして

- 横田: SIGMOD Record がだめでも最終的には ICOT のテクニカルメモにするという手段も残されている。(その場合、海外の数百の研究機関に自動配布されるという利点もある。)
- 大堀、田中: 国内の学会誌に投稿することも考えられる。

ということで、最初は各セッションの議長と書記が議事録、資料などを基に叩き台を作成し、それを参加者全員がメーリングリストをとおして検討していくという手順をとることになった。

C.2 自由討論 — OODB の質問とは何か

まず、吉川から「質問言語」のセッション (A.3節) での議論を引き継ぐ形で以下のような問題提起があった。すなわち、OODB の質問機能を考える場合、RDB の質問言語にあった、

- 宣言的集合操作
- 結果の閉包性
- 適切な表現能力 (関係完備性)

などの点を継承すべきか否か、また継承するならどのような形で行なうかを明らかにする必要がある。「適切な表現能力」というのは、RDB における関係完備性に相当するようなものが OODB の質問言語において存在するか? という問題である。また別の観点からは、OODB の質問言語は SQL から逃げることができるか? という問題もある。

OODB の質問 (これはデータの検索であって、計算ではないことに注意) で使われるであろう操作を抽象化した体系を確立することは重要な課題である。また、実現に当たっては、意味がはっきりしている計算完備な PL の (seamless な) 副言語として定義する必要性を強調。等式論理を備えた副言語だと SQL みたいに最適化が図れ、またインピーダンス・ミスマッチの問題が元から解決されてるであろうことが挙げられた [大堀]。

横田は論理型言語のパラダイムなら、Datalog、Datalog$^{\neg}$ などのきれいな階層があり、上記のような副言語の例であることを指摘した。つまり、大きな枠組として論理プログラミングがあり、それの副言語として Datalog$^{\neg}$ があり、それのまた副言語として SQL があることを指摘した。

問題はそのようなきれいな副言語の階層を OODB の世界に写像することが困難なことである。RDB の場合は、

- 代数
- 論理
- 最適化可能なクラス
- 現実の質問言語 (SQL)

という 4 者について、前の 2 つについては厳密に、また後の 2 つを含めてもほぼ一致していた。しかし、これらのクラスを OODB の世界に写像、拡張しようとすると、ズレが生じる。[吉川]。

大堀は OID を入れなければ、代数、論理などの成功例があり、わりとできるが、OODB で一番問題なのはポインタ (値指向でないことから) と OID の扱いであり、OID を入れると難しいだろうという予測を示した。すなわち、"クラス名 new" という表現が 2 つあれば、それらが同じオブジェクトを生成するか否かを判定するのは困難であり、そこから OID を入れたときの難しさが生じる。この問題については、OID 生成は副作用であり、それがために最適化が困難となる [清木]、質問の中でメソッド呼び出しをしても、それが副作用を持たなければ、最適化は可能であろうと予測される [大堀]、などの議論があった。

結果の閉包性については、田中より、閉包性はどれ程重要な問題か? 商用のOODBMSや研究(たとえばDOOD'91におけるM. Scholl(ETH)の発表)などでは閉包性をあきらめている、との指摘があった。

また、田中より以下の指摘があった。OIDは、利用者が決めるものであるが、OODBの下ではOIDのふり方に制限を付ける必要がある。すなわち、OIDのふり方に関する一貫性制約の議論が必要。質問言語については、DMLのみならず、DDL(ビューを含む)、データ制御(一貫性、認可など)も含めて考慮する必要がある。さらに、(ちょうど、RDBでのQBEのように)OODBのためのグラフ質問言語により新しい利用者層を開拓する必要がある。

D　3月11日 9:00 ～ 12:00 セッション4 (座長: 大堀、記録: 箕原)

「次世代のデータベースに向けての技術、あるいは将来にわたって残るであろう現在の技術、文献」といったテーマで5件の発表と討論が行なわれた。

D.1　加藤

発表の要旨

主要な学会誌で、"Should the DBMS area be declared solved?"というテーマが議論される(DB Systems: achivements and Opportunities. CACM 34(10):110-120)ことからも分かるように、DBMSの技術は成熟期に入ったといえる。しかしまだ未解決な問題も多い。特に以下のものが、今後の研究課題として重要である。

- システムの多言語化
- 複雑なスキーマを持つDBへのアクセスを容易にするスキーマpilot技術
- 物理的領域管理技術と記憶マップ技術との統合
- 三層スキーマやビューとADTの概念との融合

質疑応答

- 田中: 以上の項目は、OODB実現上重要といわれているものとほとんど同じで、新たなインパクトを持つものはないように思われる。

D.2　田中

発表の要旨

教科書に書かれるべきDBMS技術という点から考えると、現在のOODBは、内部スキーマの記述に相当するものばかりで、概念モデルに関する決定的なものがない。以下の研究が必要であろう。

- スキーマ間のオブジェクト・マイグレーション
- スキーマ、オブジェクトの進化
- ビュー
- 質問言語
- 制約の表現と処理

このうちOODBの制約に関しては、WeddellのキーPFDの研究が、この分野の最初のもので注目に値する。

質疑応答

- 加藤: スキーマの設計論も必要ではないか?

- 大堀: キー PFD が関係モデルの FD の一般性ある拡張になっているかは疑問がある。より形式的なアプローチとして、ドメイン理論やカテゴリ理論を用いた研究がすでにある。
- ?: OODB におけるキーとは何か、どの程度有効かを検証する必要がある。
- 横田: キーと OID を同一視するというのはどうか?

D.3 横田

発表の要旨

5 年程前から現在までの流れ

$$NR + DDB \Longrightarrow DOOD(F\text{-}logic, DOT, QUIXOTE)$$

を考えた近い将来の (*QUIXOTE* の) 研究テーマとしては、

- 質問最適化
- メタ表現 $\Longrightarrow$ プラットフォーム言語
- 他言語システム
- 分散システム $\Longrightarrow$ 協調システム
- 論理的拡張
- 並列処理
- DB エンジンとのより密な結合
- 移植性
- *QUIXOTE* 抽象マシン

を考えている。

より広い視点では、以下の各分野の研究との融合が将来重要となろう。

- DB、PL
- AI (非単調推論、蓋然性、ファジー、知識発見)
- 形式的意味論
- 数学 (統計学、…)
- センス

質疑応答

- 加藤: 言語の設計にはセンスと同時にコンセンサスが重要と思われる。
- 大堀: メタ言語は、プラットフォームであって、理路整然と理解できるレイヤとして考えられる。いろんな、アプローチがあって、そのためいろんな flavor のメタ言語が出てくる。別の flavor においては、制約をキーワードとしないものも出てくるのではないか?
- 清木: 蓋然性に関して、ICOT では、そのような研究がやられているのか? 応用から考えれば必要と思われる研究である。自己組織型の研究も確実にやってよいであろう。

D.4 清木 - 北川: 数学モデルによる検索

清木 - 北川の両者から、数学モデルによる検索が、10 年後の核となる技術として多くの時間を使って、その基本アルゴリズムについて詳細な発表があった。

既存の検索の問題点

- **パターンマッチング**
  異なる表現形態であるが、同一の意味を持つデータや近い意味を持つデータによる検索を行なうことはできない。また、特定の文脈に依存する近さではできない。
- **シソーラスも用いて同義語を照会する方法**
  同義語は、設計時に決定されてしまうものである。同義であることの定義が明確でない。ある特定の意味において同義あるいは近いという判定を検索時に動的に行なうことはできない。
- **情報の共有**
  情報の構造化と抽象化は、DB の設計時に決定され、静的に固定化される。よって、その構造化および抽象化をの枠組を理解せずに DB を操作することは困難となる。

意味の数学モデル

- 空間の設定: 距離、近さ (言葉と言葉の近さとは、"類似性" である)。
- 言葉と言葉の関係 (近さ) は、刻々と変化する文脈 (状況) に応じほぼ無限に変わりうる。
- 「文脈に応じ、意味を決定するメカニズムの構成」が必要。

意味の数学モデルへの試論

空間設定	基底・距離、動的に作られる
文脈に応じた意味	部分空間の選択、射影子の決定
解釈	選ばれた部分空間における元の最良近似を求める

すなわち、イメージ空間内のどの部分空間を選択するかによって、言葉の意味が異なる。つまり、意味の決定はとは、文脈に応じた射影子の決定を指す。

応用

- 曖昧語 (多義語) の解釈
- DB の関連検索・最良近似検索

  探索キーワードがない場合
  キーワードの集合と探索キーワードの差分ベクトルに関して意味射影関数を適用し、そのノルムが最小のものを求める。
  関連検索
  意味射影関数を適応した時に、探索されたキーワードからのベクトルのノルムが、ある一定のノルム以下であるようなキーワードの集合を求める。

- 仮名漢字変換 (点列?) など。

自然語の会話の解析に限定して

- 照応、近さ
- 原存在が多様な領域で使われる
- 多義的な言葉の決定

アルゴリズム

これらの問題を解決するために、次のようなアルゴリズムで意味解釈を行ない、検索に応用する。

- 空間の設定　モデル化する。イメージ空間 I を以下のように構成する。

  各単語の特徴を列挙して、ベクトルする。そのベクトルの集合を行列 A とする。
  A の特徴付けに関する相関行列 $A^T A$ を作る。
  $A^T A$ を固有値分解し、固有ベクトルを求める。
  求めた固有ベクトルを基底とするとしたイメージ空間 I を構成する。このとき、基底は正規直交基底となる。

- 言葉と言葉の間の近さを決める。文脈は射影を示すが、それを以下のように構成する。

  意味射影集合 Π_p を固有値に対する固有空間への射影から全ての次元について、$0, 1, \ldots, n$ 次元までに関して構成する。
  Π_p の要素の個数は、2^n 個であり、これは 2^n 個の意味の表層があることを示している。

- 意味解釈

  射影を 1 つ決める。意味解釈オペレータ S_p も定義する。
  Fourier 展開をする (正規直行系なので内積を取れば良い)。
  単語列の意味重心を求める。
  文脈と意味素の相関を求め、相関が高いものをしきい値を決めて引っ張りだし、射影値の和和を求める。

議論

- 横田: 状況理論との関連性は? 特に、Barwise が状況理論を提唱しているがそれとの関連性を知りたい。また、ICOT の橋田が提唱している力の場 (ポテンシャルエナジー) とは、どのような関連があるか?
- 北川: 状況理論とは、相補的な関係にある。
- 渡: 言葉の距離は元から設定しておくというが、それでは直観に合うような行列が得られるのか?
- 清木: キーワードと features の値付けを行なう。
- 横田: 分野によってテーブルが変わるはずではないか?
- 清木: これも動的に切替えることで対処する。
- 渡: micro features を始めに DB に入れるにはどうするのか?
- 清木: 今は、事前に与えられていると仮定している。
- 田中: 文脈に依存するキーワード設定のように思える。DB の演算において、文脈を切替えるのは何に対応するのか?
- 清木: multiple DB において、発表したようなキーワードから構成されるメタ DB を用いて、特定分野の DB を探しにいくことに対応する。ある文脈の中で、キーワードの検索を行なう時、全空間を見る必要性もある。ニューラル計算は、空間選びに使えるが、そこでの空間は直交していない。
- 田中: 類似性の概念が有効。文脈の動的切替えは、東芝のワープロでもやっている。
- 横田: 視点が必要であろう。また、観測者がどこにいるかという概念も必要と思われる。
- 渡: 加えて、どこかにいたときに次にどこにいるかも必要でないか?
- 北川: 状況意味論では、次のような欠点がある。

  - 空間概念がはいっていない。
  - 動的に近さが変わるということが表現できない。

  また、ニューラル計算では、空間の設定が直交基底ではない。以上のことによりこのアプローチが優れている。

D.5 渡: 21 世紀の開放システムに向けて

開放システムを前提として考えた場合、今後どのような技術が研究されなければならないかという主題で、特にビューの技術に焦点を当て、発表が行なわれた。渡の提唱している Morphe と呼ばれるモデルが発表の基盤となっている。

研究動向

- 開放システムを前提としたとき、何が問題か?

  大域的な制御が不可能であること。
  計算の最中に起こる予期せぬ入力。
  通信遅延。

- どのような方向に向かうべきか?

  部分情報の統合化を簡易に行なえること。
  計算内容および環境の動的な変更を行なえること。
  矛盾を含むような知識を管理できること。

- そのための技術 / 形式化とは?

  閉鎖的な仮説を仮定しない分散制約解消系。
  環境もまた first class のオブジェクトにする。
  * 状況理論?
  * 自己反映計算の理論?

  paraconsistent logic?

ビュー

- ビュー とは何か? (Morphe の観点)

  ビュー = 世界のある視点からの見方。抽象化の表現そのもの。
  抽象化されたシステムの表現は、システムとオブジェクトと状況を組み合わせて、部分・全体の構造を表現する。
  視点とは、システム内のポイント (パスと組み合わせて)

- 抽象化の次元とその表現

  モデリング・スペース
  * 水平軸: 多重表現 — 対象物の全体的な表現 (属性の集まり) のある視点に対しての写像。
  * 垂直軸: 抽象化のレベル (情報の隠蔽)— 部分・全体のグラフ上の深さを示す。

  対象物のダイナミックス
  * "時間" 軸: 状態および構造の変化 — システム (あるいはサブシステム) のバージョン

- 多重表現

  - 既存の表現のコピーの局所的変更。
  - 表現の共有: 共通部分とそうでない (局所的な) 部分とをいかに分離するか。
    アクセスパスによって表現の "型" が異なる。

ビューが必要とされる理由

- 新しい応用

  対話理解
  ロボットの社会
  Adaptive System(モデルの更新・構築)
  経済学、政治学、社会学、心理学
  開放的な情報システム (Hewitt)

- 複数のデータ・知識ベースの統合化

  CAD
  エキスパート・システム

- システム開発の過渡期

  漸次的なソフトウェア開発
  一時的な版
  デバッグ

10 ～ 20 年後の主要な技術

- Adaptive (Evolutionary) 計算

  遺伝的アルゴリズム
  Evolutionary Stable Strategy (ゲーム理論)

- Symbolic Categorization

  Phase Transition
  パタン認識

- 交渉戦略

  個人 — 組織

議論

- 横田: 開放システムにおいて大域的な仮説はどの程度あるか?
- 渡: まちまちである。ある程度の共通知識を仮定するものから、まったくないものまである。
- 横田: もし、何らかの大域的仮説がなければお互いに認識することもできないだろう。
- 加藤: DB 方面との関わりについてもう少し聞きたい。
- 渡: 知識ベースを考えた場合、異なった命題で表された同一実体を扱う必要がある。そして、そういったことを行なうとした時、無矛盾な知識を仮定しなければならない。
- 田中: 情報検索において、ユーザ自身が自分の視点を持っていないのではないか? また、ユーザが環境をどのように記述するのか?
- 渡: ユーザは、基本的にはコンピュータ・システムの記述の中に含まれている。つまり、ユーザがコンピュータ内にいるという概念である。そして、ユーザは今どの時点 (地点) にいるのかがわかっている。例えば、歴史を追えば、現地点を追随することができるだろう。

D.6 箕原: 開放システム入門

発表の要旨

Carl Hewitt と Peter de Jong の提案している開放システム (1984) の概念が将来の DBMS にとって重要である。彼らの提案の中の主要な概念は

- 透明性概念の否定
  - 大規模分散環境
  - 地理性・位置性 (トポロジー・局所性) は、概念モデルとして与えられなければならない。
- デュープロセス
  仮説と目標を区別し、双方を別物として推論を行なう。
- 自己参照、自己知識、自己進化
  (他者との) 交渉の機能と自分の歴史、能力を検索できる機構が必要である。

である。そしてまとめとしては「万物流転の法則」になるだろう。

質疑応答

- 加藤、横田: 人によって開放の意味、大域的なものは何かについてのとらえ方が違うと思われる。
- 西尾: 開放的な環境において、無矛盾性はシステム全体では発見できるか?
- 箕原: 命題間の無矛盾性として、発見が可能と思われる。
- 横田: 簡単にできるとは思えない。一般的には、矛盾しているかどうかはわからない。
- 横田: ここで仮説とは何を指すか? つまり、仮説や目標を推論するということはどういうことか?
- 箕原: 仮説や目標は、計算に応じて動的に変わっていくものと考えられる。それらは、はじめから定まっているものではなく、推論過程において徐々に決定されていくものなのではなかろうか?
- 西尾: 交渉は人間が行ない、システムは関わらないのか?
- 渡: 本質的には、人間が行なうものであるが、ドメイン特有なものについては戦略は決定できることが多い。
- 横田: 推論の概念は公理を前提としており、開放系と相容れないのではないか?
- 渡: 動的に公理系そのものを選択するというものが考えられる。
- 清木: この点は重要である。現在文脈を動的に選択するとういことはできないが、今後は重要となろう。
- 加藤: 開放 = 予測不能ととらえられるが、人間は未知のもの対応する能力があり、そこが計算機システムとは違う点と思われる。
- 箕原: 未知のものに対する能力、それが計算機と人間との違いでないか?
- 渡: 人間もある意味で未知のものに対して予想しているとも考えられる。
- 清木: 計算の複雑さを考慮に入れた計算モデルを考えることも重要。
- 田中: 透明性の概念の否定は、透明性の概念を元に発展をしたこれまでの分散処理システムの研究の方向を大きく覆す、極めて挑戦的な試み。
- 加藤: 本当は、透明性の実現がその根底にあるとも考えられる。
- 箕原: 概念レベルで、地理性が入っていることが重要なのである。たとえば、所の MD-Computing は、その一例である。
- 吉川: MD-computing は、情報のソースのことを考えているのではないか?
- 渡: ちょっとちがう。斥力・引力の関係。

D.7　吉川: 高品質のデータの蓄積

大規模環境における検索の困難性という観点から、高品質なデータを如何に獲得し蓄積させるかという技術が今後重要になるだろうという発表がなされた。

情報の収集と生産

- 開放分散環境からの情報の収集。
- 利用者自身による情報生産技術。

  データ量を増やす技術は、既に充分なレベルにある。
  データ量を減らしと情報量を増やすことが大切である。
  ごみデータからのデータの淘汰や知識獲得技術が必要である。

- 収集データからの本質的な情報の抽出。
- 低品質の情報は簡単に生産させない。
- 段階的抽出化、利用者の興味の動的変化に対応すべきである。

人間と計算機間のチャネルのバンド幅の拡大には?

- 通信媒体

  マルチメディア
  自然言語

- 利用者のモデル化

  特化された質問事項・語彙 — 標準語
  各利用者の知識レベルに応じた質問機能

- 計算機が利用者を教育していく

  有能な秘書
  有能なチューター

議論

- 渡: 以下に示す2つの方向性について言及されたが、

  情報の選択
  情報の生産過程への制限

  情報の選択だけが必要だと思われる。生産過程に制限を加えることなどできない。また、自由に生産が行なわれるべきである。検索者が、自分に合うような情報を見つけるには、創造的に新しいものを模索するしかない。不必要な情報は、選択していく過程でなくしていくしかないだろう。
- 吉川: 創造的に模索するというが、チューターが考えていなかったような検索はできないかも知れない。
- 箕原: 低品質か高品質かを決めるのに、一意的な判断基準はない。
- 田中: 加えて、特にマルチメディアにおいては、データの品質というのはデータを蓄積する段階ではわからない。質は、後で決められるものである。例えば、そこで考えられる技術としては、
  - 後から内容のチェックができるようにする、
  - 一貫性制約を入れておく

が考えられる。また、その両者を融合したものとして一貫性獲得の方法論が作れないだろうか。例えば、要約DB のようなものである。これは、DB の抽象化技術に貢献するだろう。

- 吉川: 逆に、このようなことをやりたいから、OODB がやりたいのである。品質というのは、確かに視点に依存するし、何らかの制約が必要だと思われる。
- 西尾: 知識発見は、一貫性獲得との関連があると考えていいのか?
- 吉川: ビジネス分野においては、数値データは統計情報によって扱うことができる。非数値に関しては、属性(サンプリングされたデータ)間の関数従属性を見つけ出すことによって、一貫性を得ることができるだろう。特に、クラスタリングや要約の技術は重要である。
- 渡: 知識のナビゲーションとかは、既に行なわれている研究である。また、パタンマッチよりも高度なものを、MCC のウェナー博士によって、百科辞典 Cyc 上におけるルール抽出において研究されている。
- 吉川: 特に、個人化することがその上で重要と考えられる。

E　3 月 11 日 13:00 ～ 15:00 セッション 5 (座長: 西尾、記録: 天満)

「次世代のデータベースに向けての技術、あるいは将来にわたって残るであろう現在の技術、文献」といったテーマで 3 件の発表と討論が行なわれた。

E.1　天満: 計算環境の継続と永続性並行オブジェクト

天満から、並行性オブジェクトシステムのテーマとして、計算環境を分散環境下で継続させるといった問題が提案された。そのためには、名前付け、異種計算機、並行オブジェクト群の保存と、整合性を保ったままの移動、などが要求させる。横田から開放型システムにおける、名前空間の管理問題が指摘された。また、西尾から並行オブジェクトを DB として扱う問題があげられた。並行オブジェクトの永続性やそれを共有する、問合せる、などの問題がある。

E.2　有川: ユーザ・インタフェース

有川から、表示されたものを実際にさわれるといった特性を持つ地図 DB を例として、ハイパーメディア に関する提案がなされた。そこでは、ハイパーメディア と質問の作成のための視覚化の問題がある。質問自身のデータ化、ハイパーリンク化を含めたデータの視覚化の問題が解決されなければならない。

笠原から情報の視覚化の認知科学の必要性が指摘された。視覚化を科学的に扱うためには、視覚化の良否を判断する基準が必要である。その観点から、横田は、1 つの方法として Xerox での子供を利用した方法を例示した。加藤によると、商用の分野では QBE を除いて、視覚化の成功例が少ないことが示された。田中から直接操作を含むグラフ・ユーザ・インタフェースから、データあるいはデータモデル視覚化が区別されるべきであるといった問題提起がなされた。QBE の成功の要因は、データモデルに視覚化、つまりテーブル表現が整合したためであるが、OODB における同様な良い視覚化方法は未だ現れていない。

また、OO データと視覚化との関係を探るために、渡から地図 DB を発展させ、OODB と合成させ、シミュレーションなどを含む地球 DB を作成する問題が出された。

E.3　鈴木: 記憶域管理

鈴木はオブジェクト記憶に関する問題点の指摘を行なった。まず、オブジェクトの消去に関して到着可能性に従って消去する方法でも明示的に消去する方法だけでも不十分であると行った指摘がなされた。また、今後も基本となる技術として、ポインタ・スイズリング、struct space、クラスタリング、GC が示された。また、代表的文献や技術として、

- ポインタ・スイズリングに関する Wilson の論文
- 記憶域の階層的分解と ハッシュ・テーブルの GC に関する Wilson の論文
- Unger の世代別 GC
- LOOM によるポインタ・スイズリング
- PS-Algol
- Moss による持続性 Modula-3

が紹介された。

E.4 大堀: 情報構造の数学的側面

大堀により、

- データモデル理論
- ドメイン理論
- 自然言語理論
- 状況理論

などの各分野の理論の類似性が示され、これらを統合して「情報構造の統一的な理論」を作る、といった問題提起がなされた。

統一的な理論を作ると行った観点から、数学的な事象を扱うカテゴリ理論 との比較において、ここではデータ、言語、人間の知識などのドメインを扱うため、抽象化のレベルが異なり新しい技術が必要であるとの指摘がなされた [大堀]。これらの理論の共通点として集合に対する意味があげられる。さらに、異なる部分をも含めるような新しい共通の形式体系をまず作りあげる必要がある。ICOT の *QUIXOTE* はこの先駆けではないかという指摘がなされた [横田]。

複合オブジェクト の扱いに関して [田中]、集合の扱いはデータモデル理論での知識の蓄積がある。集合に関連して、渡から選ぶと集合との対応とその意味が指摘された。横田によると、選択の扱いには 2 通りあり、論理型言語では 述語 となる場合があることが指摘された。

また、PL のモジュール理論との関連が指摘された [横田]。特に、多相型やパラメタ化モジュールの研究が多くなされており、それは整合しない知識を DB に格納するといった例において有効であるかもしれない、といった指摘がある。上記項目のドメイン理論に多相型が書き加えられた。

E.5 西尾: トランザクション

西尾から、トランザクション管理および並行処理制御に関する指摘と将来のテーマについての指摘が行なわれた。過去においてこれらの問題は原子性と直列可能性という 二つの概念を中心に論じられてきた。例えば、よく知られた初期の文献として、次のような論文がある。

- 原子性
  - Bjork, Davies 72, Bjork 73
    Chamberlin, Boyce, Traiger 74
- 直列可能性
  - Gray et al. 75: degree 3 consistency
  - Eswaran et al. 76: 2 相ロック
  - Papadimitriou 79: 理論、計算量

などの代表的な研究がすでになされている。

しかし、応用分野の拡大にともない、OODB が使用されるようなシステム環境下では、長時間トランザクション、協調作業支援など、従来の制御方式をそのまま適用するとシステムの性能上問題があったり、新たな要請を反映しきれない場合がある。また、システム・アーキテクチャ的にもクライアント / サーバモデルのもとで、メモリキャッシュを有効に利用した制御が重要となる。特に、これらの新たな要請が顕著なエンジニアリング DB などでは、今後、従来の原子性と直列可能性の概念をいかに緩和していくかが重要な課題になると思われる。

まず、原子性に関しては、最近、入れ子型トランザクションモデルをベースとし、サブトランザクション単位での原子性を考えることがいろいろと提案されている。つまり、原子性を保証する処理の単位を論理的にできるだけ小さくすることが重要である。一方、直列可能性に関しては、データに対する操作の意味を考慮した制御方式や協調して作業をするシステムのユーザのインテリジェンスを反映した制御法などを考慮することにより、従来の直列可能性の概念を緩和することが重要である。特に、西尾は、最近、DB 内の部分データ (領域) に課せられた一貫性制約を陽に反映し、協調作業環境を考慮した並行処理制御方式を考えている。このモデルでは、DB 内のデータ全体ではなく、領域データを中心とした直列可能性を考えることにより、並行処理性を向上させることが可能である。

また、OID とロックの粒度の関連が指摘された。これは、システム性能と関連しており、アクセスされる集合が予期可能であり、かつその情報をスケジューラが活用できるなら、DB の並行処理性能は向上する [西尾]。

OODB における並行処理制御の特徴として、協調作業をするユーザがアクセスするデータ全体を、最初に二次記憶から共用のメモリキャッシュにチェックアウトして、そのデータを利用して作業を行なう形態が多い。その場合は、メモリキャッシュのデータを対象した並行処理制御がなされ、作業完了後のデータは、一つのバージョンを形成して二次記憶にチェクインされる。

渡からOODB におけるオブジェクトへの操作の意味を考慮した並行処理制御法の可能性の指摘があったが、メソッド中の記述を利用したそのような制御法は、現存の商用 OODBMS には存在しない。ただし、限定的ではあるが、操作の意味を考慮し、それらの可換性をベースにした制御法が提案されている。横田から入れ子型トランザクションの話題と関連して、DDBMS における並行処理制御方式を今後考える必要性に関して指摘があった。

F　3 月 11 日 15:15 ～ 17:00 セッション 6 (座長: 田中、記録: 鈴木)

今回の議論の内容

- 田中: このセッションでは、WG 全体のまとめを意識して討論していこう。この 2 日間は昨年の Obase/ICOT の合同ワークショップと重複する対象を議論してきたが、皆の発言内容はかなり変化したようだ。昨年のキーワードとして、「OID, 型とモデル, DDL, 言語、質問言語、ビュー, DBPL の再検討」などがあげられたが、今年の議論を象徴するキーワードをあげると次のようになろう。
  - **部分オブジェクト**　不完全性、遅延、曖昧さ、矛盾
    情報構造　高水準モデル、設計論と方法論
    類似性 — 曖昧な関係
  - **制約** — 制約に基づいた計算、一貫性制約、制約に基づいたトランザクション制御
  - **知識獲得** - 抽象化、要約、ルール / 制約の発見
  - **自己組織化** — オブジェクト / スキーマ進化
    ビュー
  - **ルール** — 推論、ルール管理
  - **開放性・分散・統合** — 協調計算、交渉、同時実行制御
  - **文脈依存 & 切替え** — 文脈 / 環境 / 状況 / 位置の動的な切替え
  - **質問言語** — GUI、オブジェクトの視覚化、ハイパーメディア、質問構築ツール、完全質問言語 (DDL, DML, ···)

- **言語** -- DBPL、メタ
- **空間 / 時間オブジェクト**　地図
- **OID 管理** -- OID 操作演算、OID 生成制御
- **実装** -- ポインタ・スイズリング、クラスタリング

- 渡: 昨日と今日は似た議論も多かったが、異なる趣旨をもって行なわれた。昨日は、本年度の WG の議論のまとめであったし、今日の午前は、今後の研究方向や技術について話しあった。

ルール

- 田中: 昨日は、統括だった?
- 横田: まとめた人と、発展的な方向で話しをした人 (発散した人) がいたが、昨日議論になったことは、上記リスト中のキーワードとしてあげられている。
- 萬原: 暗示的ににルール、演繹が根底にある議論が多かったが、それらのどのキーワードに対応するものか。
- 横田: 5 年〜 10 年後の見通しのなかで議論されたが、演繹、OO がはっきりとはとらえられず、どう使われるべきなのかわからないうちに、議論の中で消えていってしまった。
- 田中: 制約に対応するんじゃない?
- 横田: ルールと制約は、ちょっと違う。
- 田中: (上記リストに

  ルール　推論、ルール管理

  を追加。)

質問言語

- 加藤: キーワードをみると、質問言語が特に取り上げられているが、質問がより重要だという意味か? GemStone, ObjectStore などの商用 ODBMS では質問は 3 層スキーマの最下層に位置する DB 構築よりのお話と位置付けられている。そうみるとキーワードのサブ項目として取り上げる程度重要なのかわからない。思ったことは、商用 ODBMS は DB 屋の目からみて、DB ではなく、OOPE ととらえるのがよいだろう。商用 ODBMS の質問はサブ項目として取り立てる程どではない。
- 田中: それは DBPL の研究はもう終ったということか?
- 加藤、大堀: いや、そんなことはない。
- 清木: OODB のデータ独立性とは?
- 加藤: 明確な定義は存在しない。その意味で OODB の研究はこれから始まるのだろう。あるいはそもそも研究対象として存在しないのかも。
- 吉川: 商用 ODBMS が、かくのごとく質問を看過しているのは何故だろう?
- 加藤: *QUIXOTE* の位置付けは?
- 横田: 話す場所によって違う。(笑い)
- 加藤: DBPL と質問言語は違うし、PL も別物だ。
- 吉川: ObjectStore とか O++ の (質問は) 低レベルで内部スキーマと密着する。概念スキーマが必要だ。
- 加藤: OODB の概念スキーマとは何かが問われている。

- 田中: Serendig という O_2 のメンバが O_2 Technology から離れてやっているプロジェクトがあり、型とビューを基本モデルとしている。一見意味モデルへの回帰のようでもあり、面白い。
- 清木: データの独立性と関連した話しだ。OODB は内部スキーマの記述言語に過ぎない?
- 大堀: 型システムも含めて?
- 田中: 商用 OODBMS では、そうだ。プログラマだけが喜ぶような OODB ではだめだ。

多層スキーマ

- 加藤: 仮想記憶の技術を駆使し、主記憶の表現とディスク上の表現を単一にする方式の実装が、MCC のプロジェクトの Cricket によって採用されている。これは性能の向上を計るとともにプログラマに単一の視点を与えるものだ。それは、スキーマの階層化を計ったのとは別の方向だ。どちらが正しいのか?
- 田中: Serendig では 2 つの抽象レベル (型 (下) とビュー (上)) を設けている。
- 大堀: それは、型システムがプリミティブだからだろう。
- 渡: 去年も田中先生は、今後も残る伝統的な DB 概念として「3 層スキーマ」をあげたが、私はずっと反対している (笑い)。OODB といったとき、皆 PL と関連づけているが、OO の技術は DB だけでなく OS とも融合されいて、DB と OS で共通の問題が解決を計られている。Hydra がもたらした新しい概念は、完全な階層化はよくないということだ。それには、保護のために階層的なメカニズムが用いられたが、オブジェクトの ADT としての性質に基づいて保護が扱われ、階層は排除されている。なぜこの方式が有効かといえば、オブジェクト自体に抽象化が入っていて、階層に分けなくとも、ビューを通してやればいいからだ。
- 加藤: 視点を設けたら、それは階層を入れることになる。Hydra が移植可能であることは、内部スキーマを分離しているということではないか。Hydra は 3 層スキーマの有用性を示した例だ。
- 田中: やっぱり層による抽象化は必要だろう。
- 渡: Hydra は 階層化よりオブジェクトによる抽象化が良いことを示したのだ。
- 加藤: でも、そのオブジェクトは 3 層構造になっている。抽象化の重要性は認識は言語を起源として広まった。DB からの貢献は、それが 2 層でなく、3 層である必要性を示したことだ。
- 大堀: 先程どのコメントを捕捉すると、型理論では基本型と関数型を基底として、上位の層を構築するという手法をとる。例えばクラスとは上位の層に位置するだろうし、またその上にビューを構築することも可能だろう。だから、抽象化を何層にもして、その各層をわかりやすくつなげるには型理論が参考になるだろう。
- 渡: 3 層スキーマは RDB から生じたのか?
- 清木: いや、RDB とは独立な概念だ。
- 渡: 3 層スキーマでの分割は、DB が事務処理程度の利用のためだったからだ。そもそも単純なデータだから、単純に分割できた。しかし、OOPL で OO システムを作る時にもそれが可能か?
- 西尾: ビューに関していろいろ議論されているが、OO には構造的側面と動的な側面 (メソッド) がある。ビューはメソッドにも適用されるものなのか? 静的でないメソッドをどう扱ったらよいのか? データとメソッドを切り離すことを目標とした時にはビューや 3 層スキーマで物理レベルを論じることには意味があったが …。
- 田中: PL 屋はデータの物理構造を見せることをしない。オブジェクトへのインタフェースはメッセージの送受信のみである。しかし、質問やインデックスを考えると DB 屋は中身が気になる。
- 渡: C++ も内部構造を外部に見せることができる。
- 加藤: 状態に基づく検索は、中身を見ないとだめだろう。

- 田中: OODB の質問はカプセル化とナビゲーションの間のジレンマに陥っている。非正規 DB も入れ子のアクセスによって同じ問題に直面している。
- 加藤: すべてメッセージによって行なうという制限を受け入れるだけでよいのでは?
- 田中: それでも、ナビゲーションはしなくてはならないでしょう。
- 加藤: 質問言語はできるでしょう。
- 田中: やってもいいけど、使えないでしょう。ネットワーク DB では、パスがわからないと質問が書けない。
- 加藤: OODB は、本質的なリンクのみを辿るという点でネットワークとは違う。本質的なリンクをアクセス器を使ってナビゲートすればよい。記述面では問題ないはず。
- 笠原: 田中先生は OOPL の OO モデルと違うらしい。なにが必要なのか?
- 田中: クラス構造は厳格すぎる。それに、制約まで張り付けたら使えない。
- 清木: 大堀さんの型理論では クラス階層の明確化をおこなうのでしょう。OODB では、DB の持っている物理的な意味を記述している。この意味を理解できない / 知らないユーザには DB は使えない。もっと柔軟な DB をモデルのレベルから考えないといかないかも。
- 加藤: PL 屋の抽象化と DB 屋の抽象化はちがう。前者はカプセル化だし、後者は隠すというより、ものごとを整理して合理に構成すること。
- 清木: データ独立性は隠すことでしょ。
- 加藤: 2 つの意味を含んでいるでしょう。概念スキーマのレベルでも構造を持っている。
- 大堀: PL のカプセル化も見えなくてはいけないものだけ見せる。正に同じではないか。
- ?: DB は構造が変わる (別のモデリングをする) ではないか。
- 田中: DB はないものを有るようにみせることもある。
- 清木: オブジェクトの作成者と利用者の間のコンセンサスを前提とした PL とすべて誰にもみせる DB とは違う。
- 加藤: 動機は同じではないか。
- 笠原: 加藤さんは抽象化のレベルの合わせ方が違うといっていたのでは。
- 清木: 実世界のマッピングをおこなって利用者に提供することを目的とすること (DB) と利用者の利便を考えてモジュールを作成すること (PL) が差異であろう。
- 大堀: (sort モジュールの例をあげて違いを見つけようとするが、) プログラミングのスタイルにもよる。
- 横田: 使い方の結果は少しずれているような気もするが、カプセル化、抽象化の精神は同じ。
- 渡: OODB のビューの話しがまだでてこない。「OODB のビューとは何か」をふまえて討論を深めよう。DB では概念スキーマで ビューが表されている。再利用性を高めることができない理由に、スキーマ設計者が抽象化したときの環境と、利用者の環境のずれがある。RDB では再利用性など考えないから、別にいい。OOPL となると、例えば SmallTalk80 では、サブクラス化による拡張を推奨していて、存在するクラスの変更はやめさせるようにしている。OODB では、既存のスキーマをも自分の環境として構築、変更できるようにすることがビューだ。
- 田中: それは、去年も話した。
- ?: 型理論は、OODB のビューにどう貢献するのか? OODB のビューの働きとして
  - 無いものを有るようにみせる。

多重性(同じものを別のように見せる)。
概念の階層化などがある。OODBでは仮想化の対象が増える。

- 大堀: 私もわからない。全部をカバーすることはできない。メッセージの送受信による状態変化、クラスへの帰属などをモデル化する必要がある。ビューはクラスのように振舞うはずだ。ビューを定義し、クラスと同じように扱うことができる? クラスの上にビューをつくり、それへのメソッドの作用としてモデル化する場合には、その作用可能性を定式化するのは(型?) 理論の得意な分野。拡張はもっと深刻。静的には決定不能。
- 加藤: メソッドの作用可能性については、SIGMODのMachiavelliのペーパーにあった?
- 大堀: あれはあらかじめ実行可能性を説明したものだ。
- 横田: 田中先生の3つのビューの働きをもっと分類したほうがよい。仮想性、多重性にもいろいろある。例えば時間的な視点。混乱をさけるためビューの働きを限定していく必要がある。
- 大堀: 仮想クラス(ビュー)を完全にクラスと同じように働かせるのは無理。
- 横田: 3階層モデルはむり。
- 田中: 3階層モデルはもともと柔軟。OODBは複雑化を、ビューによって解決しようとしている。この分野での日本からの貢献が望まれる。文脈、制約などが入ってくるので大変難しい。

OID管理、知識アーカイブ

- 横田: 応用はどうなっているのか? 良い応用はDBの発展に貢献するはずだ。
- 有川: ハイパーメディアのノードはオブジェクトかビューか?
- 田中: ハイパーメディアの構成時には、データを入れてしまってからリンクを張る。
- 渡: よくわかんらないんですけど。
- 田中: とりあえず入れた文書とハイパーメディアのノードとは異なるということだ。
- 横田: 知識アーカイブという言葉で指示されるように、とりあえず格納する情報(一次情報)というものがある。
- 大堀: 構造化しないデータをアーカイブしては、すぐに扱いきれなくなる。
- 横田: まずデータを入れないことには何も始まらないし、一次情報の形態でも全く使えないわけではない。
- 田中: (上記リストに

  - 応用 - 知識アーカイブ、漸次的なスキーマ / アクセス・メソッドの構築

  を追加。)
- 渡: この問題は更新とからんでいる。追加とどう違うのかをはっきりさせる必要がある。ことなるものだか、同一視する必要があるというのがこの例だ。オブジェクトの等価性とは…。
- 吉川: 話が混乱しているようだ。ハイパーメディアはOODB上のビューである。
- 大堀: 生データではないオブジェクトのOIDの問題だ。
- 渡: ビューは柔軟でいろいろなものに写像できるのがいい。
- 有川: OODBはハイパーメディアに使えるようなものでなければだめ。今のハイパーメディアはアドホックな作り方をしてしまっている。
- 横田: これはOIDの管理の問題だ。グルーピング / オブジェクト同定を行なった時にはOIDをふる必要がある。新しいOIDを与えるべきかどうかはユーザ(遺伝子情報の場合は生物学者)が決定すべきこと。

北川先生感想

昨日話し合われたことに関して

- 面白い構造とは?
- 高速化、高機能化
- 結果の完備性、閉包性
- 共通の仮定 (見えなかった)

今日話し合われたことに関して

- 情報構造理論
- 抽象化、理論化を歓迎
- (意味の?) 理論を構築中
- 何を捨て、何を拾うかの選択が難しい
- 理論を作るアプローチとして、確立論、統計学、幾何学、解析学的な視点をいれることで、指数オーダを、多項式オーダに。
- 他分野の参入への配慮が欲しい
- 人のため世のためになるよりも、自分が面白いことが大事