2G−5

永続プログラミング言語INADAの分散共有データ操作機能

天野浩文　有次正義　寺本圭一　白光一　牧之内顕文
(九州大学工学部情報工学科)

1．まえがき

画像や音声のように長大なデータやＣＡＤデータのように相互関連の複雑なデータを扱う要求に応えるため，オブジェクト指向データベースシステムの研究開発が進められている．一方，複数のサイトに分散したデータを統合する必要性も高まっている．このため，データの分散を透明化することの必要性が以前から唱えられている．

しかし，一部のサイトに蓄えられたオブジェクト集合が新たな用途のために他のサイトのオブジェクト集合と統合されて使用される場合を考えると，サイトごとのオブジェクトの構造（クラス定義）が整合するとは限らない．また，それらの構造がどれも新しい用途に適していない可能性もある．

このため，上記のような使用法にも対応できるようなデータベースシステムを構築するためには，データの分散を透明化する機能の他に，既在のデータ構造の不整合をも透明化する機能が必要になる．

現在開発中の永続オブジェクト指向プログラミング言語INADA[Ma91, AA92]には，分散データを自サイトのデータと同様に操作する機能，条件検索の結果得られるオブジェクトを集合として扱う機能，および，すでに存在するオブジェクトに新たな型（クラス定義）を付与する機能を有する．本稿では，INADAにおけるこれらの分散共有永続データ操作機能の概要について述べる．

2．分散共有永続ヒープと永続オブジェクト

プログラム実行終了後もデータを存続させる場合，データをファイルから読み出し処理後書き戻す操作が必要になる．複雑なデータ構造をメモリ上に保持する場合には，ファイルのデータ形式とメモリのデータ形式との間の変換操作も必要となる．

INADAでは，仮想記憶空間の一部にファイルと１対１に対応する領域(**永続ヒープ**)を設ける．この領域に生成されたデータは実行終了後もファイル上に存続する．永続ヒープを利用するプログラムはファイルの入出力を意識する必要がない．永続ヒープは分散共有永続ストレージシステムWAKASHI[BM92]によって実現されており，INADAではこれを利用するシステム定義クラスを提供する．プログラムはこれらのクラスのインスタンスを生成するだけでよい．

メモリマップトファイルI/Oを利用したシステムには他にObjectStore[LL91]がある．ObjectStoreでは自サイト内のマッピングとサイト間のマッピングが異なる機構によって実現されている．一方，WAKASHIは両者を統合した外部ページャとして実現されている．

同一のファイルを異なる位置にマップできるよう，INADAでは，永続オブジェクトへのポインタの内部表現を間接参照形式にすることができる．しかし，この擬似的なポインタ（**永続オブジェクトポインタ**）は通常のポインタと同様の構文で使用できる．

永続オブジェクトポインタはINADA処理系によって永続オブジェクトポインタクラスのインスタンスに変更される．永続ポインタクラスも標準ライブラリに含まれる．この「ひな型」はC++によって記述され，プログラマによるカスタマイズも容易である．

3．集合体オブジェクト

大量の永続データを再利用する場合，必要なデータだけを選択的に操作しなければならないことがある．INADAでは，検索結果を保持する**集合体**クラスを提供し，その標準インタフェースを公開する．集合体を応用プログラムの側で作成することもできる．

集合操作は応用プログラムの一部ではなく言語の機能として提供する．このため，クラスをインスタンスの「倉庫」と見なす考え方を放棄し，クラスを「鋳型」，集合を「倉庫」とする考え方を採用することができる．この転換により，「ビュー」機能をクラス定義の変更ではなく集合体に対するファセット付加ととらえることができるようになる[AA92].

4．ファセット

複数サイトで独立に作成したオブジェクトを統合して再利用するとき，その型（クラス定義）を統一できなければならない．しかし，このための変更が既存のプログラムの動作に影響を与えてはならない．

INADAでは新たに必要となった見方(**ファセット**)を任意の時点で付加する機能を提供する．ファセットの定義は通常のクラス定義とほぼ同様である．これにより，オブジェクトの同一性を保ちながら，複数の「型」を持たせることができる．

複数の型を持つオブジェクトをモデル化する試みは，[RS91]，[SB86]，[IU92]などに見られる．しかし，[RS91]では，データ型を追加する際に旧データへのポ

Distributed Shared Data Manipulation Facility of Persistent Programming Language INADA
Hirofumi Amano, Masayoshi Aritsugi, Keiichi Teramoto, Guangyi Bai, and Akifumi Makinouchi
Department of Computer Science and Communication Engineering, Kyushu University

インタを内蔵する方式を採っているため，参照の切り替えは新しい型から古い型への一方通行である．[SB87]は揮発データのための枠組であり，永続データの複数の型の操作には適していない．また，[IU92]では新たな「視野」の付加が既存のオブジェクトの型を変更するので，その変化を吸収する機構をデータベース管理システムに備える必要がある．この方法はプログラミング言語での実現には適さない．

一方，INADAでは参照切り替えが双方向に行え，既存のプログラムの動作にも影響を与えない．

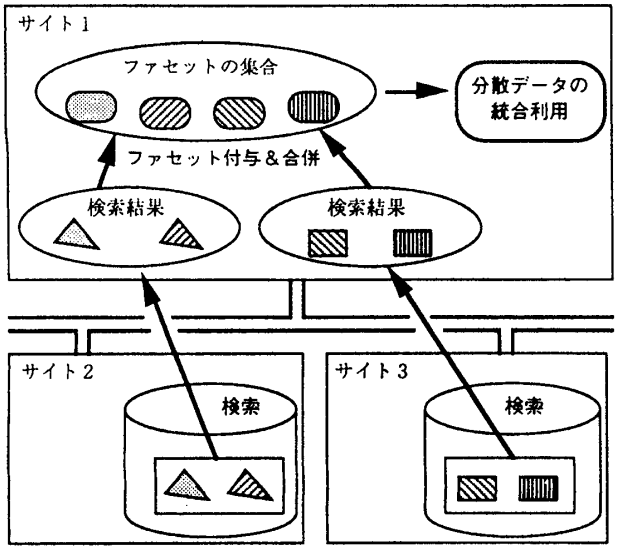

図1　分散データの統合利用

5．既存の分散データの統合利用

複数のサイトでそれぞれ独立に生成されたオブジェクト集合を統合して利用することを考える．前説までに述べたINADAの要素機能を用いれば，ファイル入出力・データの分散・データ型の不一致を隠ぺいしながら，個々のサイトで既に開発されたプログラムの動作に影響を与えずに，統一的なデータ型のもとで動作する新しい応用プログラムを開発することができる．

分散データの統合利用を行うINADAプログラムは，次のような手順を踏んで処理を行えばよい．

(1)各サイトに存在するオブジェクトを検索する．
(2)得られた複数のオブジェクト集合のそれぞれに，統一的なファセットを付与する．
(3)これらの集合体の合併を取る．

(1)の操作は，ファイルの所在さえ指定すれば，あたかも自分の仮想記憶空間内のデータ構造を探索するのと同様の記述が可能である．ファイル入出力はプログラマの目から隠ぺいされる．

(2)の操作では，既存のデータを破壊することなく，用途に適した新たなデータ型を付与することができる．また，単一のオブジェクト識別子に複数の型を与えることができるので，オブジェクトの同一性も損なわれない．すでにその型が与えられていれば，この段階は単なるファセット切り替えとなる．

(3)で得られる合併集合は，あたかも自分の仮想記憶空間内に存在する均質な集合体と同様に操作することができる．

すなわち，ファイル入出力，データのサイト間分散，および，データ型の不整合を隠ぺいすることができたことになる(図1)．このような性質は，製品の企画・設計の段階から製造・販売までを支援するコンピュータ統合生産(CIM)環境の構築にも有用である．

6．むすび

本稿では，拡張可能な永続プログラミング言語INADAの分散共有永続データ操作のための機能について述べた．分散共有永続ヒープはWAKASHIを利用する標準クラスとしてモジュール化されているが，これは，将来のWAKASHIの機能拡張や，新たな分散共有メモリ技術の成果の導入を容易にする．また，永続オブジェクトポインタの部分もクラスのひな型で定義されており，応用の性質に応じて改良することが容易になっている．これに加えて，集合体操作とファセット付与の機構を用いれば，分散サイトで独立に生成されたオブジェクトを統合利用するシステムを構築するのに役立つ．

この他，オブジェクト指向データベースシステムの記述に必要な機能として，トランザクション管理がある．現在，これをどのような形でINADAのプリミティブとして取り入れるか，検討を進めつつある．

参考文献

[AA92]　有次，天野，牧之内：「永続プログラミング言語INADAにおけるマルチタイプオブジェクト」，情報処理学会アドバンストデータベースシステムシンポジウム，平成4年12月．

[BM92]　Bai, G. and Makinouchi, A.: "Implementation and Evaluation of a New Approach to Storage Management for Persitent Data --Towards Virtual-Memory Database--," Proc. the 2nd Far-East Workshop on Future Database Systems, pp.211-220, April 1992.

[IU92]　石丸，植村：「データベースシステム「沙羅」の機能拡張」，情報処理学会データベースシステム研究会，90-3，平成4年9月．

[LL91]　Lamb, C, Landis, G. et al.: "The ObjectStore Database System," *Comm. ACM*, Vol. 34, No.10, October 1991.

[Ma91]　牧之内：「マルチメディアデータベースのためのオブジェクト指向永続プログラミング言語族」，電子情報通信学会データ工学研究会，DE91-28，平成3年7月．

[RS91]　Richardson, J. and Schwarz, P.: "Aspects: Extending Objects to Support Multiple, Independent Roles," *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pp.298-307, May 1991.

[SB86]　Stefik, M. and Bobrow, D.G.: "Object-Oriented Programming: Themes and Variations," *The AI Magazine*, Vol.6, No.4, 1986.