オブジェクト指向データベースの開発

6D－8

－C言語インタフェースの開発－

土屋 武彦　脇園 竜次　川村 敏和　田中 立二

(株) 東芝 重電技術研究所

1 はじめに

オブジェクト指向データベースは，複雑なデータを高速に操作することができ，プログラミング言語との親和性が良い．そのため，高速応答性が要求され，従来のデータベースでは適用が難しかったリアルタイムシステムへの適用が可能となった．

Odbはプラント監視制御などの産業用リアルタイム分野向けに開発したOODBで，OODB必須条件を満たし，高速で応用プログラムの生産性に優れたものをターゲットとして開発した．

Odbの特徴は下記の通りである [1].

- C++言語と透過なデータベース言語の提供
- 集合と外延の概念と操作性の統合

我々は現在，Odbの機能拡張を進めると共にOdbの適用を進めており，配電系統監視制御システム [2]，電力系統監視制御システム [3] 等への適用を試みている．

これらの分野におけるソフトウェア開発では，C言語を用いたものが主流であり，他の分野においてもC言語が広く普及している．

その中でOdbを活用するためには，C言語によるソフトウェア資産の継承が不可欠である．この時，C言語で記述されたソースプログラムのみならず，C言語を開発言語としているプログラマも考慮しなければならない．既存のCコードからOdbの機能を利用できるようにし，C++言語に習熟していないプログラマにもOdbを用いた応用プログラムを容易に開発できるようにする必要がある．

本論文では，C言語インタフェースの開発を行ったので，その概要を紹介する．

2 C言語インタフェース仕様

OdbはC++言語を基本としたデータベース言語を提供する．すなわち，C++のクラス宣言によるスキーマ定義，new()/delete()による永続オブジェクト生成/削除，C++言語の代入/参照構文による永続オブジェクトのデータベースへの書き込み/読み出しなどが行なえる．

Development of the OODBMS
Takehiko Tsuchiya, Ryuji Wakizono, Toshikazu Kawamura, Tatsuji Tanaka
Heavy Apparatus Engineering Lab., Toshiba Corp.

このデータベース言語はC++のクラス・ライブラリとして提供される．図1に示すようにユーザ・プログラムをC言語で記述する場合には，C言語からC++言語のクラス・ライブラリを利用できるようにするインタフェースが必要となる．そこで，C言語用のデータベース操作言語の開発を行なった．

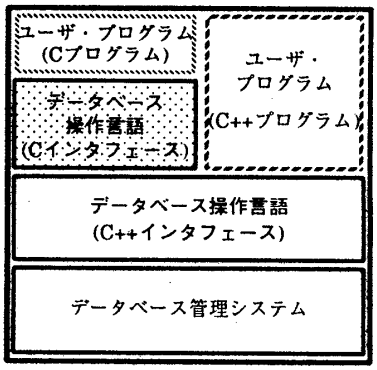

図1 C言語インタフェースの位置付け

2.1 APIクラス

C言語ユーザからは，C++言語で開発されたOdbのAPI(Application Program Interface)クラスへはアクセスできない．そこでC言語用に次のようなインタフェースを用意した．

(1) 各APIクラスごとに「odb_クラス名」というインタフェース用の構造体を提供する．
(2) APIのメソッドに対応して「クラス名_メソッド名(オブジェクト・ポインタ,引数1,…)」の形式のC関数インタフェースを用意する．
(3) APIのメソッドで多重定義のあるものは，引数に応じて別名のC関数インタフェースを用意する．

データベース処理，トランザクション処理に関するAPIの例を表1に示す．

2.2 ユーザ定義

2.2.1 データ(クラス)定義

C言語インタフェースを使ってデータを定義する場合は，構造体(struct)宣言によりスキーマ定義を行なう．スキーマ登録された構造体は，odb_new()インタフェースによって，永続オブジェクトとなる．永続オ

表1 C言語インタフェース仕様

機能	C言語インタフェース	C++言語インタフェース
スキーマ定義	struct, union	class, struct, union

データベース処理	odb_database* Database_open(char* db_name);
	int Database_close(odb_database* db);
	odb_type* Database_type(
	odb_database* db,char* struct_name);
		Database* Database::open(char* db_name);
		int db → close()
		Type* db → type(char* class_name);

トランザクション処理	odb_transaction* Transaction_start();
	odb_transaction* Transaction_startl(int lock);
	int Transaction_commit(odb_transaction* trsct);
	int Transaction_abort(odb_transaction* trsct);
		Transactiont* Transaction::start();
		Transactiont* Transaction::start(int lock);
		int Transaction::commit();
		int Transaction::abort();

オブジェクト生成/削除	void* odb_new(odb_type* type); void odb_delete(void* object);	void* new(Type* type); void delete(void* object)
オブジェクト参照/操作	car = (Car*)Type_find1(car_type, condition, key); car → color = bule;	car = (Car*)type → find1(condition, key); car → coler = bule;

ブジェクトはC及びC++プログラムのいずれからもアクセス可能である．C++言語でクラス定義を行なって生成した永続オブジェクトをC言語を用いて参照/更新する場合には，まずC++クラス定義のメソッド部分を削除した構造体(struct)を再定義する．この時，クラス名と構造体名は同じ名称とする．

OdbのAPIであるType_find1()に対し，クラス名を指定してオブジェクトの取り出しを指示すると，永続オブジェクトへのvoid型ポインタが返される．ユーザはこのポインタを構造体ポインタに型キャストして，参照/更新を行なう．

2.2.2 関数(メソッド)定義

C言語の構造体ではメソッドの記述が行なえない．そこで，構造体を扱う関数については,「構造体名_関数名」という名前の関数を定義することとする．

この規則により，1つのデータベースをC言語ユーザ，C++言語ユーザで共有する場合に，アプリケーションの記述形式を統一できる．また，C言語からC++言語への移植も容易に実現できる．

3 記述例

上記インタフェースを用いたアプリケーション・プログラムの例を次に示す．C++言語を用いて記述した例を図2に示し，C言語により記述した例を図3に示す．

```
 1  class  Car {
 2     char  name[10];
 3  public:
 4     char  *getName() {return name;}
 5  };
 6
 7  main() {
 8     Database   *db_a;
 9     Type       *typeCar;
10     Car        *car;
11
12     db_a     = Database::open("DB1");
13     Transaction::start();
14
15     typeCar  = db_a->type("Car");
16     car      = new(typeCar) Car;
17     printf("%s\n", car->getName());
18
19     Transaction::commit();
20     db_a->close();
21  }
```

図2 C++言語インタフェースによるプログラミング

```
 1  struct  Car {
 2     char  name[10];
 3  };
 4  char*  Car_getName(car)  struct Car* car; {
 5     return car->name;
 6  }
 7
 8  main() {
 9     odb_database *db_a;
10     odb_type     *typeCar;
11     struct Car   *car;
12
13     db_a     = Database_open("DB1");
14     Transaction_start();
15
16     typeCar  = Database_type(db_a,"Car");
17     car      = (Car*)odb_new(typeCar, NULL);
18     printf("%s\n", Car_getName(car));
19
20     Transaction_commit();
21     Database_close(db_a);
22  }
```

図3 C言語インタフェースによるプログラミング

上記プログラムを比較するとプログラムの記述の流れやフォーマットは同じである．インタフェースの書式に多少の相違があるのみであることがわかる．

4 まとめ

C言語によるソフトウェア資産を継承するためにOdbのC言語インタフェースを開発した．これにより，C言語ユーザに対しても，透過なデータベース言語を提供することが可能となった．

参考文献

[1] 川村敏和，脇園竜次，土屋武彦，田中立二：
高速性と言語透過性を重視したオブジェクト指向データベース，
情報処理学会第99回データベース研究会，1994.

[2] 脇園竜次，土屋武彦，川村敏和，田中立二：
配電システムへのオブジェクト指向データベースの適用，
平成7年電気学会全国大会，1995.

[3] 関知道，関俊文，土屋武彦，田中立二：
電力系統システムの柔軟性確立の研究
平成7年電気学会全国大会，1995.