データ構造変換向けクラスライブラリ生成ツールの開発

5J－7

熱田ミハル†　HongXin Huan‡　黒田清隆†　高野彰†
{kuboi, kuroda, takano}@isl.melco.co.jp
三菱電機(株) 生産性推進部 ソフト生産性技術センター†　University of Waterloo ‡

1　はじめに

我々はデータ構造変換向けクラスライブラリ生成ツールを開発している。本ツールはあるデータ構造から別のデータ構造への変換プログラムの自動生成を目的とする。UNIX ツールの yacc はパーサを生成するが、出力結果の形式を定める機能は持たない。従ってアクション部に出力形式を規定するプログラムを書かなくてはならない。言い換えれば yacc 記述は、yacc への入力は明瞭だが出力に関しては不明瞭である。本ツールは出力形式を記述することにより自動的に出力形式を整えるプログラムを生成する。これによりデータの入出力の見通しが良くなりプログラムの可読性の向上が期待できる。また、ソースコード中に散在しがちな出力構造の定義を一括して扱い、再利用することができる。

2　Ryacc の特徴

- yacc 風の入力のため、yacc ユーザには簡単に記述できる。
- 出力は C++ プログラムである。
- Ryacc ではデータの入力順序に関してある程度柔軟性を持たせる。
- 何段階かのデータ構造変換が必要な場合、yacc-Ryacc-Ryacc-...-Ryacc という風に組み合わせて使える。
- 出力構造を記述するために、低レベルのデータ構造まで定義できる（例えば、２５６バイトの文字列、等）。
- Ryacc は、文法記述に日本語名を使える。

3　Ryacc の機能

図1は Ryacc の機能説明図である。Ryacc への入力は Ryacc 記述である。

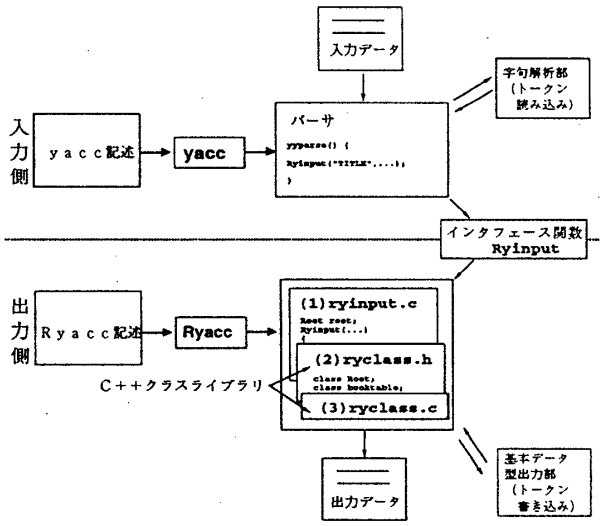

図 1: Ryacc の機能説明図

Ryacc 記述は yacc 風であり、ユーザの必要としている出力形式の仕様とアクションを記述する。Ryacc からの出力は、Ryacc 記述に従ったデータ構造変換をするための C++ クラスライブラリとインタフェース関数である。

yacc 側から Ryacc 側へのデータの受渡しは、グローバルなインタフェース関数 Ryinput によって行なわれる。Ryinput は Ryacc により自動生成される。パラメータは以下のようにデータのラベル名とデータ自身であり、yacc 記述のアクション部で呼び出される。

Ryinput("データのラベル名", データ);

データのラベル名は、そのデータの種類を示す。実際には Ryacc 記述に書かれた記号名である。例えば"著者名"という記号名で Ryacc 記述を書いたとき、これがラベル名となる。

Development of Class Library Generator for Data Structure Transformation
Miharu Atsuta†　HongXin Huan‡　Kiyotaka Kuroda†　Akira Takano†
†Mitsubishi Electric Corp.　‡University of Waterloo

図１では入力側にyaccを用いているが、必ずしもyaccを用いる必要はなくユーザがRyinputにより、データをRyacc側に渡すようなプログラムを作成すればよい。実際にRyaccが生成するファイルはC++で記述されている図１中の（1）、（2）、（3）の３つのファイルである。

4 言語仕様

Ryacc記述はyacc風である。以下に図書データのRyacc記述例の一部を示す。

```
ryout  :  図書テーブル  著者テーブル  出版社テーブル
       ;

図書テーブル
   | FILE* fp;
     fp=fopen("booktablefile", "a"); |
   : /*  empty  */
   | 図書レコード  図書テーブル
   | $1.output(fp);   $1.delete(); |
   ;

図書レコード  :  タイトル  著者index並び  出版社index  価格
              ;

タイトル  :  char[256]  ;
```

図 2: Ryacc記述例

Ryacc記述では初期手続きを書くことができる。初期手続きの中ではファイルのオープンや既存のC++クラスのインスタンス宣言などを書く。ターミナルはchar, intなどの基本型クラスとなる。アクション部にはユーザが任意のC++コードを書くことができる。

5 生成クラスライブラリ

RyaccからC++クラスライブラリという形をとっている。ここでいうC++クラスライブラリとは、図書データを例にとると図書データの構造変換をするクラス群である。各々のクラスを単独で用いることはできない。これらのクラス群で必要な出力構造にデータ構造を変換する出力構造ツリーを生成する。具体的にはRyacc記述で図書テーブル、図書レコード、タイトル、著者名、出版社名といった一連の出力データ構造の項目を記述するとこれら１つ１つの項目がクラスとして実現される。出力構造ツリーの１つのノードは１つのクラスのインスタンスに対応している。Ryacc記述中の"図書レコード"の定義に対応して"図書レコード"クラスが定義され、さらに"図書レコード"クラスは"タイトル"クラスなどへのポインタを持つものと定義される。このクラスの形式は図３のようである。

```
class t  |
   char*     label;  //  自分のラベル名
   下位クラスへのポインタ並び

 public:
   t();      //  クラスのインスタンス構築手続き
   ~t()      ||
   int       ryinput(char*,  void*);
   void      output();
   void      output(FILE*);
             //  もしツリーのターミナルクラスなら自分
             //  の型に合わせて出力、そうでなければ自
             //  分の下位クラスにoutput()を送る
   void      reduce();
   void      reset();
             //  もし、ターミナルなら記憶している値を
             //  リセット、ターミナルでなければ下位ク
             //  ラスへreset()を送る
   void      delete();    //  記憶しているデータ
                          //  をデリート
|;
```

図 3: 生成クラスライブラリの例

図３の下位クラスへのポインタ並びは、

```
a : b c d;
```

という記述において、クラスaはクラスb、クラスc、クラスdへのポインタを持つことを意味する。クラスaから見てb, c, dを下位クラスと呼んでいる。ルートクラスのインスタンスが生成されるとその下位クラスのインスタンスが再帰的に生成され、最終的に出力構造ツリーが生成される。Ryinputにより入力されたデータは、出力構造ツリーのルートに渡され自分が当てはまるノードを探索する。そして正しいノードに格納されたデータは、アクション部に記述されたタイミングにより出力データファイルに出力される。

6 おわりに

本稿ではデータ構造変換向けツールRyaccを紹介した。Ryaccを使用することにより、ユーザはyacc風の出力構造記述を書くだけでデータ構造変換プログラムを得ることができる。今後、本ツールの開発を通じてマイコン組み込みソフトなどへの適用を行なう。さらに、yacc/Ryaccの統合の可能性を検討していきたい。

参考文献

[1] Margaret A.Ellis and Bjarne Stroustrup. *The Annotated C++ Reference Manual*, 1990, Addison-Wesley