315　情報モデルのEXPRESS言語表現から計算機言語表現への変換方法

北海道大学工学部　○高橋竜哉　田中文基　岸浪建史

要旨

本報では、アプリケーション構築のために、Expressにより記述された情報モデルを計算機言語表現に変換するツールの必要性を明らかにし、変換ツールFed-X及びFed-Xを用いたアプリケーションの基礎部分SCLについて考察を行い、報告する。

1.はじめに

現在、製品モデルデータ交換標準としてSTEP (STandard for the Exchange of Product Model Data)が提案されている。異なるCAD間のデータ交換とは、製品モデルデータ及び製品モデルデータの定義を与える情報モデルの交換を意味しており、STEPにおいては、情報モデルを、曖昧さを排除するために記述言語Express[1]により記述する。

Express言語で記述された情報モデルを各種の計算機言語へ自動変換するツールの存在は、CADで情報モデルを利用するために必要である。現在入手可能な変換ツールとしてはFed-X (NIST (Federal) Express Translator)がある。

そこで本報では、Express言語表現から計算機言語表現に変換するツールの必要性を明らかにし、変換ツールFed-X及び、Fed-Xを用いたアプリケーションの基礎部分SCLについて考察を行い、報告する。

2. 変換ツールの必要性

STEPでは、製品モデルデータを記述するための情報モデルを定義している。そして、情報モデルはExpress言語で記述される。しかし、Express言語は製品モデルを正確に記述することを目的とした記述言語であり、計算機で実行可能な言語ではない。

異なるCAD間のデータ交換のための中間ファイル生成モジュール、および情報モデルに基づいたCADデータベースを構築するためには、Express言語で記述された情報モデルを計算機言語表現に変換し、それを利用してシステムを構築することは有用である。

すなわち、Express言語表現から計算機言語表現への自動変換を行うツールが必要である。

このツールの持つべき機能は、次の様なものである。

(1)Express表現のモデルを各種の計算機言語表現に自動変換出来ること。

(2) Express言語の仕様の変更による変換ツールの入出力機構の変更を最小限とすること。

(3)Express表現の情報モデルの内容が変換によって損なわれていないこと。

3.変換ツールFed-X[2] [3]

NISTで開発されたFed-Xは、Express言語表現のファイルを計算機言語表現に自動変換するツールである。

Fed-Xの構造を図1に示す。Fed-Xは、まずExpress表現された情報モデルを読み込み、情報モデル内の要素定義、要素間の関係等モデルの構造を表現したExpress Working Form (EWF)を作成する。このEWFから、各種の計算機言語の出力を生成する。EWFにおけるExpressの各要素の構造は、図2で示される。現在、Fed-XはSQL、C++、及びSmalltalk-80表現への出力が可能である。

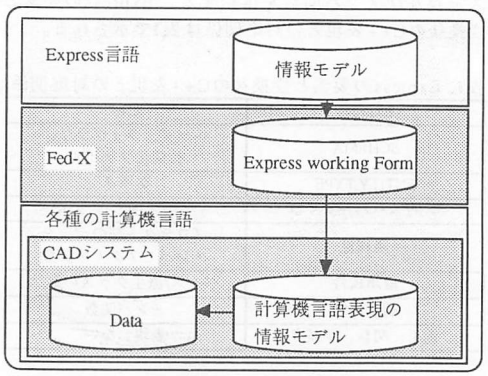

図1. Fed-Xの構造

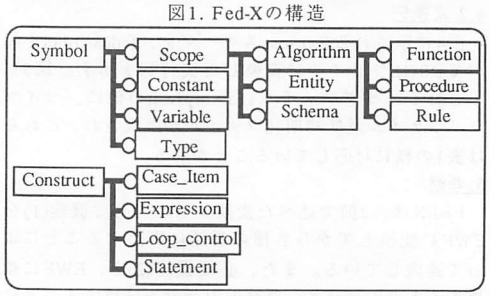

図2. EWFにおけるExpressの各要素

4.SCL[4]

NISTで開発されたSCL(STEP Class Library)は、STEPで定義された情報モデルを使用するアプリケーションシステム構築のための基礎となる部分を供給する、C++クラスライブラリである。

SCLの構造を図3に示す。SCLは、Fed-XにC++言

語への出力モジュールを加えたプログラムfedex_plusを用いてC++言語表現のファイルSSCLを生成する。SCLは、この他にSSCLのベースクラスSCCL、及びSSCLからエディタ用のオブジェクトを探して取り出すSDPCLからなる。これらのライブラリは、STEPデータを創成、検索及び操作するSDAIS (STEP Data Access Interface Specification)ライブラリの基礎を形成する。この内、情報モデルがC++言語で記述されているSSCLの詳細を説明する。

4.1 SSCL(STEP Schema Class Library)

Expressの各々のスキーマは、fedex_plusにより3個のファイル、すなわちクラス定義のヘッダファイル、クラス関数のライブラリファイル、及び初期化ファイルに変換される。これらのファイルをSSCLと呼び、SSCLファイル内のC++コードは、アプリケーションプログラムで必要となるSTEPエンティティーのクラス定義及びメンバ関数を供給する。Expressの要素と変換後のC++表現との対応関係は表1で示される。

表1. Expressの要素と変換後のC++表現との対応関係

Express	C++
SCHEMA	ヘッダ、ライブラリ、初期化ファイル
ENTITY,TYPE	クラス
サブ/スーパータイプ	ベース/派生クラス
陽属性	プロテクト部のデータメンバとアクセス関数
継承属性	ベース/派生クラスの継承
誘導属性	メンバ関数
拘束、関数、手続き	表現しない

4.2.変換例

図4に示されるExpressスキーマを、fedex_plusを用いてSSCLファイルに変換した実行例を示す。図5に示されるヘッダファイルsExample.hの他に、ライブラリファイル及び初期化ファイルが生成され、これらは表1の様に対応していることが解る。

5.考察

Fed-Xは、2節で述べた変換ツールの必要機能(1)を、EWFに変換してから各種の言語に出力することによって達成している。また、必要機能(2)を、EWFに変換するまでの部分と、最終出力部分を分離することにより達成している。

SCLは、必要機能(3)について、拘束、関数、手続き等のいくつかのExpress表現が失われるという問題点がある。

6.おわりに

本報では、Expressにより記述されたモデルを計算機言語表現に変換するツールの必要性を明らかにし、変換ツールFed-X及び、Fed-Xを用いたアプリケーションの基礎部分SCLについて考察を行い、報告した。

参考文献

[1]Spibyほか：ISO 10303-21 "The EXPRESS Language Reference Manual",(1991)
[2]Clarkほか："NIST Express Working Form Programmer's Reference",NISTIR 4814(1992)
[3]Clarkほか："Fed-X:The NIST Express Translator",NISTIR 4822,(1992)
[4]Mclayほか："The NIST STEP Class Library", NISTIR 4411,(1991)

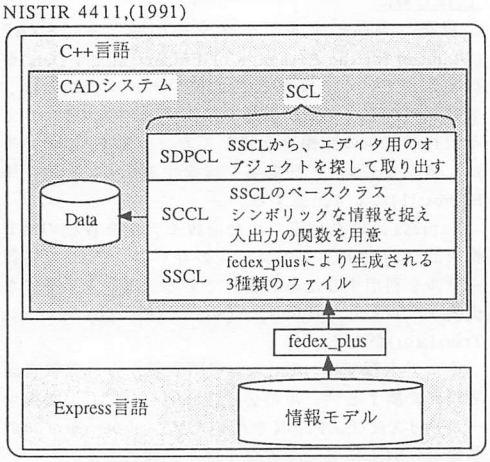

図3. SCLの構造

```
SCHEMA example;
ENTITY point;
  x : REAL;
  y : REAL;
  Z : OPTIONAL REAL;
DERIVE
distance_from_origin:REAL:=dist(x,y,z);
END_ENTITY;
FUNCTION dist(x,y,z:REAL):REAL;
RETURN (SQRT(x**2+y**2+z**2));
END_FUNCTION;
END_SCHEMA;
```

図4. Expressスキーマの例

```
#ifndef SEXAMPLE_H
#define SEXAMPLE_H
#include <STEPattribute.h>
#include <STEPentity.h>
#include <schema.h>
extern void sExampleInit (Bintree * t);
//    *******ENTITIES
class sPoint  :  public STEPentity  {
  protected:
     real _x ;
     real _y ;
     real _z ;  // OPTIONAL
  public:
     sPoint ( );
     ~sPoint ();
     char *Name () { return "Point"; }
     int opcode ()  {return 0 ; }
     real  x()
           { return (real ) _x; }
     void x (real x)
           { _x = x; }
     int opcode ()  {return 0 ; }
....
};
inline sPoint *
create_sPoint ()  {  return new sPoint ; }
#endif
```

図5. ヘッダファイルsExample.h