ＥＤＩ用汎用トランスレータの実装

杉山敬三　小花貞夫　鈴木健二

ＫＤＤ研究所

種々の利用形態やデータフォーマットに柔軟に対応するEDI(電子データ交換)用トランスレータを体系的に開発する手法について論じる．まず，データフォーマット変換をセマンティクスの対応付けとシンタックス変換に分離し，シンタックス変換のモジュールを交換可能とする汎用データフォーマット変換方式を提案し，データ構造，ファイル，処理等の設計について述べる．また，本方式を適用したCII(産業情報化推進センター)用トランスレータの実装概要について報告する．

1. はじめに

近年の商取引の活発化に伴い，受発注処理等の企業間におけるデータ交換の効率化が望まれている．そこで，これまで事務書類等の物理的手段によって交換されてきた異企業間の取引活動に関する情報を，標準的な規約を用いて電子化し，ネットワークを介して情報通信システム間で直接交換するEDI(電子データ交換)が注目されている．筆者らもこれまでに，パソコンをベースとし，ローカルプロトコルによる企業内EDIネットワークと，EDI標準による企業間EDIネットワークを相互に接続可能とするパソコンEDIシステムを開発している[1],[2].

このように扱うデータフォーマットが異なる場合には，データフォーマット間の変換を行うトランレータが必要となる．このトランスレータは，単に企業内のローカルフォーマットと標準フォーマット間の変換を行うだけでなく，標準フォーマットどうしの変換や，端末で直接標準フォーマットを扱う場合など，複数の利用形態が存在する．また，EDIのデータフォーマットの標準自体も，国際標準に限らず，国内標準や業界標準など，現状では様々なものが存在している．したがって，トランスレータのソフトウェアを効率的に開発するためには，これらの利用形態や種々のデータフォーマットに柔軟に対応できる体系的な開発手法が重要となる．

本稿では，まず種々の利用形態やデータフォーマットへの対応を容易とする汎用データフォーマット変換方式を提案する．次いで，その方式に従ったトランスレータの基本設計として，データ構造，ファイル並びに処理について論じる．最後に，基本設計に基づいた汎用トランスレータの実装例として，CII（産業情報化推進センター）用トランスレータの実装概要について報告する．

2. EDIにおける標準の概要

EDIのデータフォーマットの標準は，一般にシンタックスルール，標準メッセージ，データエレメントディクショナリの３つの要素で構成される[3]．シンタックスルールは，日付や注文番号など

Implementation of Multiple Data Format Translator for Electronic Data Interchange
Keizo SUGIYAMA, Sadao OBANA and Kenji SUZUKI
KDD R&D Laboratories

のデータエレメントからシステム間で交換するデータを組み立てるための文法規則である．通常図１に示すように階層的に規定され，階層毎にヘッダ(UNA, UNG etc.)とトレーラ(UNZ, UNE etc.)が付く場合が多い．標準メッセージは，注文書やインボイスなどの帳票が含むことのできるデータエレメントの集合を定義する．データエレメントディクショナリは，全ての標準メッセージで使用されるデータエレメントの集合である．

現在，国際標準としてISOのEDIFACT[4]，米国の標準としてANSIのX.12[5]，また国内の標準として業界標準であるEIAJ（日本電子機械工業会）標準[6]をベースにしたCII[7]，さらには銀行やチェーンストアなどの業界標準など，種々のEDI標準が存在している．

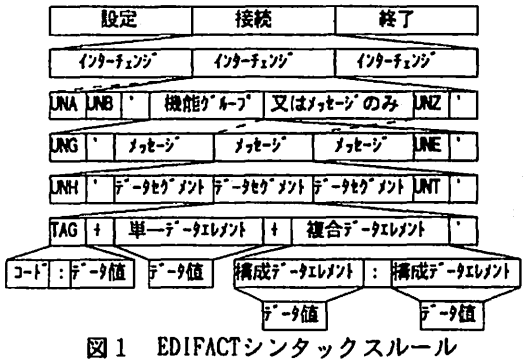

図１　EDIFACTシンタックスルール

EDIFACTのシンタックスルールを図１に示す．ここでは，インターチェンジがシステム間で交換する単位であり，機能グループは同種の帳票のグループに，メッセージは帳票に，データセグメントは一般のデータ処理におけるレコードに相当する．

3. トランスレータの３つの利用形態と従来のトランスレータの問題点

3.1 トランスレータの３つの利用形態

EDI用トランスレータには，以下の３つの利用形態が考えられる．

形態１：ローカルフォーマットと標準フォーマットの変換

各企業のローカルなデータフォーマットと標準のデータフォーマットとの変換を行う．従来のトランスレータ[8]は，この形態に属する．

形態２：標準フォーマットどうしの変換

X.12, EDIFACT, EIAJといったように，異なるEDI標準間の変換を行う．現にEIAJでは，EIAJの既存の伝票に対応するEDIFACTの標準メッセージを定義しており，その場合はEDIFACTとEIAJの変換が必要になる．

形態３：端末での直接変換

上記の２つの利用形態は異なるデータフォーマット間の変換であるが，これは端末で直接標準フォーマットの作成・解析を行う場合である．

3.2 従来のトランスレータの問題点

従来のトランスレータの利用形態はローカルフォーマットと標準フォーマット間の変換しか対象としておらず，データフォーマットに関してもEIAJ用トランスレータのように特定の標準フォーマットに固有となっている．したがって，前述した利用形態や種々のデータフォーマットに対応するには，個別に変換プログラムを開発する必要があった．

このため，種々の利用形態やデータフォーマットに柔軟に対応する汎用トランスレータを開発するには，利用形態やデータフォーマットが異なっても，共通に適用できる体系的な開発手法が必要となる．

4. 汎用データフォーマット変換方式の提案

EDIのデータフォーマットは，以下のように，転送される情報の意味内容(セマンティクス)と表現形式(シンタックス)の２つの側面があると考えられる．

●セマンティクスの側面

システム間で交換する情報の内容．例えば，発注伝票では，企業Ａから企業Ｂに対して商品ＣをＤ個注文するといった情報が，これに該当する．

●シンタックスの側面

上記のセマンティクスを特定のシンタックスルールに従ってビット列として表現したデータ．例えば，発注日という情報をYYMMDDという文字列で表現し，ヘッダやトレーラを付与したデータがこれに該当する．

このようにセマンティクスとシンタックスを分離するという概念は，OSI（開放型システム間相互接続）にも存在している．OSIでは，転送される情報のセマンティクスとシンタックスをそれぞれ抽象構文と転送構文と呼び，セマンティクスを記述するためのASN.1（抽象構文記法1）を規定している．しかしながら，EDIでは各標準が各々独自の方法でデータフォーマットを規定しており，しかもセマンティクスとシンタックスを明確に分離していないため，既存のトランスレータは特定のEDI標準に固有の変換プログラムとなっている[2]．

そこで，トランスレータのプログラムでは，シンタックスとセマンティクスを分離して扱うことが必要である．これは，トランスレータのプログラム内部にセマンティクスに対応するデータ構造を定義し，各データフォーマットのシンタックスはそのデータ構造からデータフォーマットのビット列に変換する符号化規則として捉えることで実現できる．

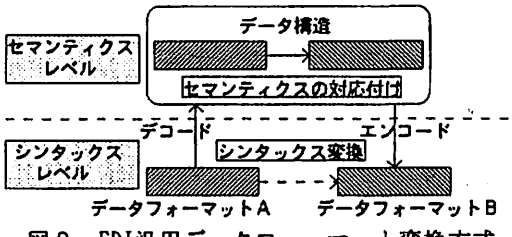

図2　EDI汎用データフォーマット変換方式

データフォーマット変換は，図2に示すように，まずデータフォーマットをプログラム内部のデータ構造としてデコード（シンタックス変換）して，シンタックスレベルからセマンティクスレベルへの変換を行う．その後，セマンティクスを保存して項目の型（整数，文字列など）や長さの変換（セマンティクスの対応付け）を行った後に，異なるデータフォーマットへエンコード（シンタックス変換）することで実現できる．

これにより，シンタックス変換を行うエンコーダ／デコーダモジュールを交換することで，種々のデータフォーマットや利用形態に対応できる（図3）．

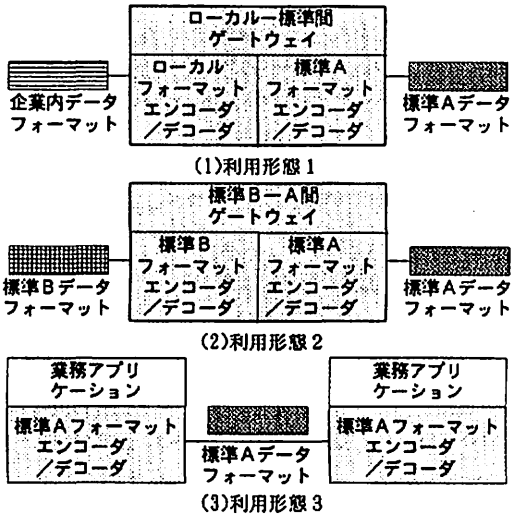

図3　3つの利用形態への対応

つまり，図3で，利用形態2は，単に扱うデータフォーマットが利用形態1と異なるだけとみなすことができ，ローカルフォーマットのエンコーダ／デコーダを標準フォーマットのエンコーダ／デコーダに交換するだけで実現できる．また，利用形態3については，業務アプリケーションから直接エンコーダ／デコーダのモジュールを呼び出すことで実現できる．このエンコーダ／デコーダのモジュールは，利用形態1と2で使用するものと同じモジュールを流用できる．

5. EDI用汎用トランスレータの基本設計

5.1 設計方針

(1) 4章の汎用データフォーマット変換方式に従い，種々のデータフォーマットや利用形態へ容易に対応可能とする．

(2)EDIでは，運用が進むにつれ新たな帳票の定義や既存の帳票へのデータ項目の追加等の帳票の変更が行われる場合があるため，トランスレータのプログラムを変更せずにこれらに対応できるようする．

5.2 ソフトウェア構成

汎用トランスレータは，図４のように，シンタックス変換を行うエンコーダ／デコーダ，セマンティクスの対応付けを行うゲートウェイの各モジュールと，辞書ファイル，データ項目対応ファイル，環境設定ファイルから構成する．

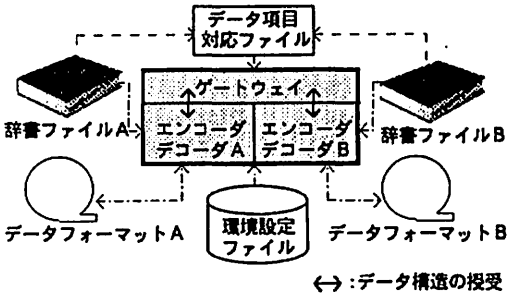

図４　トランスレータのソフトウェア構成

以下では，エンコーダ／デコーダとゲートウェイ間で授受されるデータ構造，各ファイルの内容，並びに各モジュールの処理について検討する．

5.3 データ構造

(1)種々のデータフォーマットや利用形態への対応

種々のデータフォーマットに対応するには，異なるデータフォーマットに対しても同一のデータ構造を使用できるように定義する必要がある．現在複数のEDI標準が存在しているが，そのシンタックスルールは類似しており，基本的には図１に示すような階層構造をとっている．その中で，EDIFACTは，表１に示すように，他の標準のシンタックスルールを包含していると考えられる．そこで，EDIFACTをベースとしたデータ構造を規定することとする．

また，3.1節の３つの利用形態に対応するには，エンコーダ／デコーダのモジュールが容易に交換できなければならない．これは，エンコーダ／デコーダが１回の変換をシンタックスルールのどの階層で行うかに依存する．階層が低いほど特定の標準に依存した部分を抽出するのが困難となるため，最上位の階層であるインターチェンジ単位でデータ構造を規定することとする．

表１　シンタックスルールの対応

EDIFACT	X.12	CII
インターチェンジ	インターチェンジ	ファイル
機能グループ	機能グループ	メッセージグループ
メッセージ	トランザクションセット	メッセージ
データセグメント	データセグメント	―
複合データエレメント	―	―
単一データエレメント／構成データエレメント	データエレメント	TFD (Transfer Form Data)

注：-は対応する階層がないことを示す．

(2)帳票の変更に対する対応

プログラムを変更せずに帳票やデータ項目の変更等に対応するためには，データ構造では特定の帳票のデータエレメントに固有の変数名等を定義しないようにする必要がある．そこで，データエレメントのデータ構造は，データエレメントの識別子，型を示す識別子並びにその値から構成し，それをリストで連結することで，メッセージを表わすこととする．また，機能グループなど他の階層も同様のリスト構造とし，それらを上位の階層から順に連結することでインターチェンジのデータ構造とする．

図５に，Ｃ言語の構造体として，インターチェンジのデータ構造を定義した例を示す．ここでは，リスト構造で連結する部分は自己参照のポインタ(*Next)とする．また，単一データエレメントと複合データエレメントのように，選択可能な要素には共用体を用い，選択された要素を示すための変数を定義する．

5.4 変換に必要なファイル

汎用トランスレータでは，以下のファイルが必要となる．

```
typedef struct {
      ATRB_FLAG   AtrbFlag;
      union {EDI_STRING       Str;        /* 文字列 */
             EDI_INT          IntVal;     /* 整数 */
             EDI_REAL         RealVal;    /* 実数 */
      } AtrbVal;
} SINGLE_DATA_ELEM_VAL;        /* 単一データエレメント */
typedef struct ComponentDataElem {
      struct ComponentDataElem      *Next;
      SINGLE_DATA_ELEM_VAL ComponentVal;
} COMPONENT_DATA_ELEM;         /* 構成データエレメント */
typedef struct CompositDataElem {
      USHORT ElemNo;
      COMPONENT_DATA_ELEM ComponentDataElemList;
} COMPOSIT_DATA_ELEM;          /* 複合データエレメント */
typedef struct DataElem {
      struct DataElem   *Next;
      BOOL        Simple;
      union {     SINGLE_DATA_ELEM_VAL SingleVal;
                  COMPOSIT_DATA_ELEM CompositVal;
      } ElemVal;
} DATA_ELEM;       /* データエレメントの選択(単純/複合) */
typedef struct DataSeg {
      struct DataSeg    *Next;
      SEG_TAG           SegTag;     /* セグメントタグ */
      USHORT            ElemNo;
      DATA_ELEM         *DataElemList;
} DATA_SEG;                    /* データセグメント */
typedef struct Msg {
      struct Msg        *Next;
      USHORT            KindOfMsg;
      USHORT            ElemNo;
      DATA_SEG          *DataSegList;
} MSG;                                /* メッセージ */
typedef struct FuncGroup {
      struct FuncGroup  *Next;
      USHORT            KindOfFuncGroup;
      USHORT            ElemNo;
      MSG               *MsgList;
} FUNC_GROUP;                         /* 機能グループ */
typedef struct Interchange {
      BOOL        Group;
      USHORT      ElemNo;
      union {
            FUNC_GROUP  *FuncGroupList;
            MSG         *MsgList;
      } ElemVal;
} INTERCHANGE;                /* インターチェンジ */
```

図5　モジュール間で授受するデータ構造

(1)辞書ファイル

以下に示す3種類の辞書ファイルを設ける.

●標準項目辞書定義ファイル

データエレメントの値を精査するため，個々のデータエレメントの属性を格納する．このファイルはデータフォーマット毎に作成する.

●ＩＤ選択項目定義ファイル

データエレメントの値がコード化されている場合に，コードの値を精査するためのコードの一覧表である．このファイルはデータフォーマット毎に作成する.

●帳票別項目一覧辞書ファイル

帳票を構成するデータエレメントが正しく含まれているかどうか確認するため，標準メッセージに含まれるデータエレメントの集合を定義する．このファイルは標準メッセージ毎に作成する.

(2)データ項目対応ファイル

ゲートウェイモジュールでは，セマンティクスの対応付けを行うために，２つのデータフォーマットのデータエレメント間の属性の対応表であるデータ項目対応ファイルが必要となる．これは，変換を行う両方の辞書ファイルに含まれるデータエレメントの識別子の対応表を設けることで実現できる．このファイルは，標準メッセージ毎に作成する.

(3)環境設定ファイル

トランスレータの動作を決定するために，各種オプション，例えばCIIの場合，可変長レコードが取り扱えない通信回線のための分割モード，通信制御文字と同一のオクテットが使用できない通信回線のため非透過モード，EDIFACTと並行使用するためのTYPE-Eモード，を指定したり，発信者コードや文字コードなどメッセージグループヘッダに固定的に設定する情報を格納するファイルである．これはデータフォーマット毎に作成する.

5.5 各モジュールの処理

以下では，エンコーダ／デコーダ／ゲートウェイにおける処理の概要について述べる．また，図6に，エンコード／デコード処理の概念を示す.

(1)エンコーダ

エンコーダでは，初期化処理として環境設定ファイルを読み込み動作モードを決定した後，シンタックスルールに従って，上位から順に一つ下の階層の処理を呼び出す.

インターチェンジ処理では，エンコード結果を格納するファイルをオープ

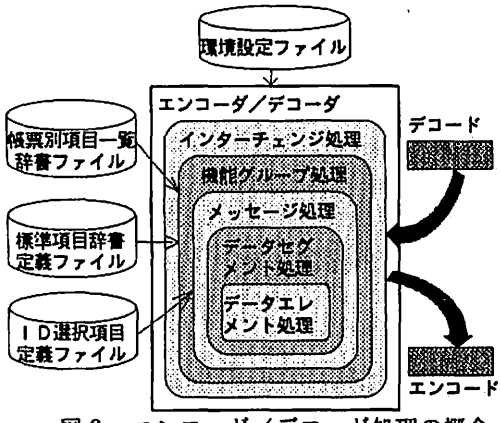

図６　エンコード／デコード処理の概念

ンし，構造体が存在する数だけ機能グループ処理を呼び出す．

機能グループ処理では，使用する帳票別項目一覧辞書や標準項目辞書を決定し，機能グループヘッダを書き出す．その後メッセージ構造体が存在する数だけメッセージ処理を呼び出し，最後に機能グループトレーラを書き出す．

メッセージ処理では，メッセージヘッダを書き出し，構造体の存在する数だけデータセグメント処理を呼び出しながら，エンコードされた結果をファイルに書き出し，最後にメッセージトレーラを書く．

データセグメント処理やデータエレメント処理では，標準項目辞書やID選択項目定義を参照して，正しい値が構造体に設定されているかどうか確認し，値をエンコードする．

(2)デコーダ

デコーダもエンコーダと同様に，シンタックスルールの階層に従って，処理を行うが，下位の階層の処理に移る前に予め下位層の構造体の領域，例えば機能グループ処理ならメッセージの構造体を割り当てる．

また，デコード時には，受信したインターチェンジがどのモードでエンコードされているかわからないため，機能グループヘッダ等を解析して解析に必要となる情報を取得し，TYPE-Eモードなど対応するモードで動作するようにする．

(3)ゲートウェイ

ゲートウェイでは，変換するメッセージの種類のデータ項目対応ファイルを参照して型と長さの属性を比較し，差異がなければ必要ならばデータ項目の識別子を変換し，差異があれば以下のような属性の変換を行う．

変換元のデータ値の長さが変換先の最小の桁数に足りない場合には，例えば数字列の場合は０を埋めるなど省略時値を埋め込み，多い場合には後ろから切捨てるようにする．また，変換元にしかデータエレメントがない場合にはその項目を削除し，変換先にしかない場合には，デフォルトの値を設定するようにする．さらに，表１のように，変換先に対応する階層がない場合には，下位の階層の構造体へ展開する．

6. 汎用トランスレータの実装

以上の基本設計に基づき，汎用トランスレータをMS-DOSパソコン上にＣ言語を用いて実装した．データエレメントが固定長のローカルフォーマットとCII標準間のトランスレータを対象とし，帳票としてEIAJの注文伝票と注文請け伝票を扱った．以下では，エンコーダ／デコーダインタフェース仕様，シンタックスルールのデータ構造への対応付け，処理及びファイルの概要について記述する．

6.1 エンコーダ／デコーダのインタフェース仕様

データフォーマットを授受する方法として，ファイル渡しとバッファへのポインタで渡す方法が考えられる．処理速度の点ではポインタ渡しが，メモリ量の点ではファイル渡しが有利である．今回の実装はパソコン上であり，メモリ量の観点からファイル渡しにした．また，環境設定ファイルにはトランスレータのオプション情報をデフォルトの値として設定しているが，関数の呼び出し時にオプ

ションを変更する場合を考慮し，オプション情報も入力パラメタとした．

図７に，エンコーダ／デコーダの各モジュールのインタフェース仕様を示す．どちらの関数にもエラー情報を出力パラメタとした．関数名の中でXXXになっている部分には，データフォーマットを示す文字列（CII/FIX）が入る．

```
USHORT encode_XXX( file, len, Inter, envinfo, err )
CHAR    *file;          /* エンコード結果のファイル名 */
ULONG   *len;           /* エンコード結果のファイル長 */
INTERCHANGE *Inter;     /* インターチェンジ構造体 */
ENV_INFO *envinfo;      /* オプション情報 */
USHORT  *err;           /* エラーコード */
```

(1)エンコーダ

```
Interchange *decode_XXX( file, envinfo, err )
CHAR    *file;          /* デコードするファイル名 */
ENV_INFO **envinfo;     /* オプション情報 */
USHORT  *err;           /* エラーコード */
```

(2)デコーダ

図７　インタフェース仕様

6.2 シンタックスルールとデータ構造の対応付け

(1)シンタックスルールの対応

CIIにはデータセグメントの概念はないため，CIIの最下位の階層であるTFD(Transfer Form Data)のデータタグはデータセグメント構造体のセグメントタグに，データ値は単一データエレメント構造体に対応付け，データセグメント構造体には一つのデータエレメント構造体しか含まないようにした．

固定長の場合は，ほぼCIIの対応付けと同様であるが，機能グループの階層を設定せず，インターチェンジから直接メッセージの構造体を指すようにした．

(2)TFD制御子の扱い

TFDには，通常のデータ値が設定されるデータエレメントと，表形式のデータを表わすためのマルチ明細ヘッダやEIAJシンタックスルールとの互換性を示すための拡張モード指示子などのTFD制御子の２種類が存在する．このTFD制御子をユーザが明示的に指定できるようにするため，セグメントタグにTFD制御子の値を設定させるようにした．ただし，拡張モードに関しては，エンコーダが自動的に判断するようにした．

6.3 ファイルの概要

標準項目辞書定義ファイルは，各項目が固定長でTFDのデータタグの昇順とした．項目毎に項目名，型，最小長，最大長，整数部桁数，小数部桁数及び値が登録されているかを示すフラグを格納し，値がコード化されているかを示す特別の型を設けた．今回はEIAJの帳票を対象としており，データタグの範囲はEIAJで定義される１（データ処理No.）から158（指定メーカー名）までとした．ＩＤ選択項目定義ファイルも項目No.の昇順であるが，項目数が不定であるため可変長とした．また，帳票別項目一覧辞書ファイルには，注文情報と注文請け情報で使用される項目No.を列挙した．

データ項目対応ファイルは，データタグが１対１に対応するようにした．

環境設定ファイルには，TYPE-EなどのCIIの各種オプションや，メッセージグループヘッダに固定的に含まれる発信者コードなどを設定した．

6.4 各モジュールの処理

基本的な処理については5.5節で示した通りであり，以下ではCII標準特有の処理について述べる．

(1)エンコーダ

CIIのメッセージにはヘッダにメッセージ長のフィールドが含まれており，メッセージを組み立てるまではメッセージ長が決定できない．そこで，メモリ上でメッセージを組み立てた後でファイルに書き出すようにした．また，メッセージの組み立て処理をどのモードでも共通に使えるようにするため，分割モードや非透過モードの処理はメッセージの組み立て時ではなく，ファイルへの書き出し時にフィルタ処理として行うようにした．

(2)デコーダ

デコードするファイルが，CIIでEDIFACTをサポートするためのTYPE-Eモードかどうかを判断するため，メッセージグループヘッダの先頭を1バイト先読みし，この値によりモードの処理を振り分けるようにした．

(3)ゲートウェイ

固定長にはメッセージグループの階層を設定しなかったため，CIIから固定長に変換する場合には，機能グループ構造体に連結されているメッセージ構造体のリストをインターチェンジ構造体につなぎ変え，固定長からCIIへの変換の場合はその逆の処理を行うようにした．

7. 評価と考察

(1)トランスレータの汎用性について

ゲートウェイモジュールは，データフォーマット毎にプログラムを作成しなければならない場合がある．1対1にデータエレメントが対応する場合には属性の変換だけで済む．一方，データエレメントが1対1に対応しない場合（例えばX.12の発注日はEDIFACTの日付と発注に対応）には，プログラム内に変換ロジックを組み込む必要がある．

しかしながら，本方式は，データフォーマット毎のエンコーダ／デコーダをライブラリとして用意し，ゲートウェイモジュールと組み合わせさえすれば，任意の利用形態やデータフォーマットに対応でき，EDIシステムを効率的に開発する有効な手法であるといえる．

(2)大きなインターチェンジへの対応について

今回採用したデータ構造はインターチェンジを単位としているため，必要なメモリ量が大きいという欠点が存在する．このような場合，大きいインターチェンジを扱うには，処理速度は遅くなるがインターチェンジ構造体をファイル化して授受する方法や，メモリ量の大きいマシンで一旦デコードされた一つのインターチェンジ構造体を複数のインターチェンジ構造体に分割してエンコードし，複数のインターチェンジに分割することで対処できると考えられる．

8. おわりに

本稿では，標準フォーマットどうしでの変換や端末での直接変換といった利用形態並びに種々のデータフォーマットに柔軟に対応するEDI用トランスレータを体系的に開発する手法について論じた．まず，トランスレータの機能をセマンティクスの対応付けとシンタックス変換に分離し，シンタックス変換のモジュールを交換することで種々の標準への対応を容易とする汎用データフォーマット変換方式を提案し，その方式に基づいたデータ構造，ファイル並びに処理等の設計を行った．また，これらの設計に従ってCII標準対応のトランスレータを実装し，本方式の有効性を実証できた．

今後とも複数標準が並行して使用される状況においては，複数のEDI標準への対応が容易な本方式はEDIシステム開発の有効な手法になると考えられる．最後に，日頃ご指導頂くＫＤＤ研究所小野所長，浦野次長及びご討論頂いた浅見通信網支援ソフトウェアグループリーダに感謝します．

参考文献

[1]:Keizo SUGIYAMA et al. "New System Architecture for PC based Electronic Data Interchange (EDI)", ICCC'92

[2]:「パソコンEDIシステムの設計」電子情報通信学会，情報ネットワーク研究会，IN91-4

[3]:「最新EDI事情」通商産業省編，工業調査会

[4]:ISO9735,"Electronic Data Interchange for administration, commerce and transport (EDIFACT) - Application level syntax rules"

[5]:ANSI X.12, "American National Standard for electronic business data interchange"

[6]:「EIAJ取引情報化対応標準１Ｃ」日本電子機械工業会，EDI推進センター

[7]:「CIIシンタックスルール1.10」産業情報化推進センター

[8]:「商品化が進むEDIパソコンソフト」コミュニケーションテクノロジ，1991.2