Annotated CRF自動化に伴う業務効率化について

藤原由　宮川元志　楊敏　篠津和夫
(株式会社タクミインフォメーションテクノロジー)

要旨

2016年10月より国内の臨床試験の承認申請時にCDISC標準に準拠した電子データ提出が開始され、2020年4月の電子データ提出義務化に向け、製薬企業各社では対応が進められている。
そこで当社では、CDISC関連の電子データ提出物のうち、Annotated CRFに着目し、fdfファイル(pdfの注釈ファイル)作成の自動化を実現する3種類の方法について検討し、評価することとした。
define.xmlでのCRFページが記載される項目は、該当する変数名やValue Level MetadataでのCRFページの記載が考えられるためこちらの対応も検討事項とし、
3種類の方法の自動化プログラム作成とメンテナンス作業の難易度も考慮に入れて検討した。

1. SASでのAnnotated CRF自動化の方法検討

Annotated CRFのfdfファイル作成自動化にあたり、下記の方法で検討を行った。
① define.xmlよりPerl正規表現を利用し、変数ごとに自動で該当ページの注釈を作成する方法
② SDTM及びdefine.xmlの検証で利用されるPinnacle 21 Community版の「Create Spec」の機能を利用してAnnotated CRF作成用のExcelを作成する方法
③ XML Mapperを活用しdefineをSASデータセットに変更して変数ごとに自動で該当ページの注釈を作成する方法

2-1. 正規表現を利用した実装検討

Perl正規表現(prxchange関数)を利用して該当変数の実装のページ数を自動的に取得する方法を検討した。
実装方法としては以下の方法で行った。

Define.xml一部抜粋

```
<ItemDef OID="IT.DS.DSTERM" Name="DSTERM" DataType="text" Length="34"
  SASFieldName="DSTERM">
      <Description>
        <TranslatedText xml:lang="en">Reported Term for the Disposition Event</TranslatedText>
      </Description>
      <CodeListRef CodeListOID="CL.TREASFF"/>
      <def:Origin Type="CRF">
        <def:DocumentRef leafID="LF.blankcrf">
          <def:PDFPageRef PageRefs="6 16 18" Type="PhysicalRef"/>
        </def:DocumentRef>
      </def:Origin>
    </ItemDef>
```

上記を1オブザベーションとして格納する方法

```
data WORK.VAR ;
  set WORK.DEFINE ;
  retain FLG 0;
  length CODE $10000 ;

  * </ItemDef から始まる場合変数を初期化 *;
  retain VAR ;
  if ( kindex ( _CODE , "</ItemDef" ) ) then do ;
    VAR = "" ;
    FLG = 1 ;
  end;

  * </ItemDef>から次の</ItemDef>までの文字列を結合 *;
  if ( FLG ) then VAR = catt ( CODE , _CODE ) ;
  if FLG = 1 ;

  keep VAR ;
run ;
```

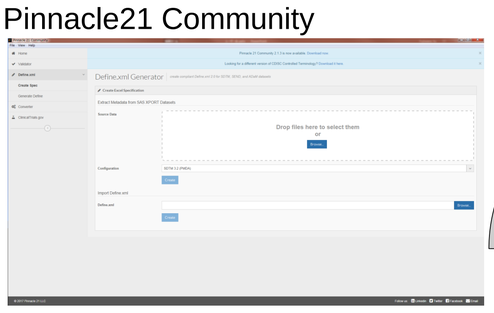
色字部分を取得する方法

```
VAR = prxchange ( 's/<ItemDef OID=¥"(.*?)¥".*/$1/i' , -1 , CODE ) ;
PAGE = prxchange ( ‘s/(.*?)PageRefs=¥”(.*?)¥”.*/$2/I’ , -1 , CODE ) ;
```

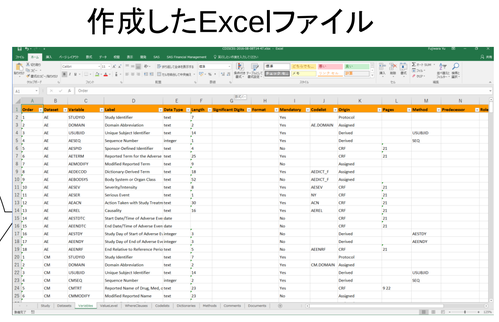
2-2. Pinnacle 21 Community版を利用した実装検討

Pinnacle21 Community版のCreate Spec機能を利用した実装方法を検討した。

Pinnacle21 Community



作成したExcelファイル



Variable取得部分の参考プログラム

```
data WORK.TMP1 ;
  set WORK.VARIABLE ;

  if ( ORIGIN = "CRF" ) ;

  length VAR $100 DOMAIN $100 ;
  VAR = trim  (  DATASET ) || "." || trim ( VARIABLE ) ;
  DOMAIN = DATASET ;
  A=  countw ( PAGES , " " ) ;

  do I = 1 to countw ( PAGES , " " ) ;
   PAGE = input  ( scan ( PAGES , I , " " ) , best. ) ;
   output ;
  end ;

  keep VAR DOMAIN PAGE ;
run ;
```

Excelファイル一部抜粋

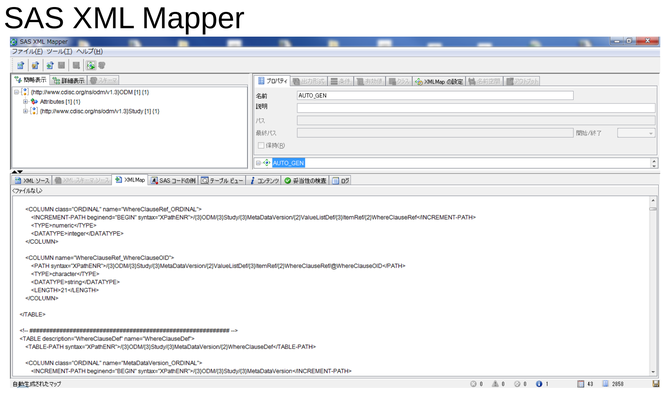
Pages	position
34	10,10
34	10,20
35	10,10
35	10,20
38	
33	
36	
36	
37	

fdfファイルに出力する注釈の位置についての指定箇所はrectで設定でき、CRFのページがA4の場合、“0,850,600,0,”で1ページのサイズとなる。(下、上、左、右の座標)
この情報を利用して右からの割合、上からの割合を指定することにより出力位置を指定できる機能も組み込んだ。

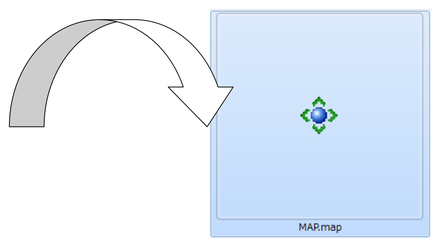
2-3. SAS XML Mapperを利用した実装検討

3つ目の実装方法としてSAS XML Mapperを利用した実装方法を検討した。

SAS XML Mapper



作成されたMAPファイル

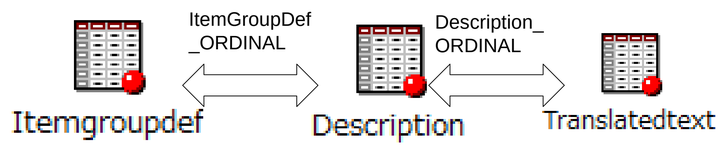



```
filename  define 'C:¥Users¥Temp¥define.xml';
filename  SXLEMAP ‘C:¥Users¥Temp¥MAP.map';
libname   define xmlv2 xmlmap=SXLEMAP access = READONLY ;
```

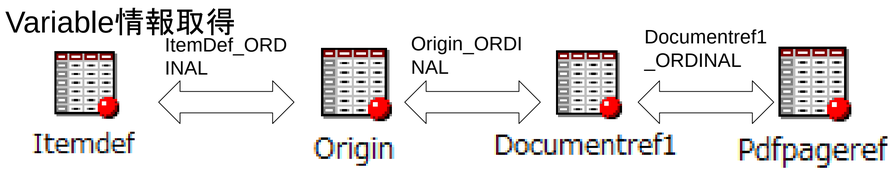
赤字部分：データセット変換を行うdefine.xmlを指定
青字部分：SAS XML Mapperより自動で作成されたMAPファイルを指定

作成されたデータセットを結合してCRFページを取得

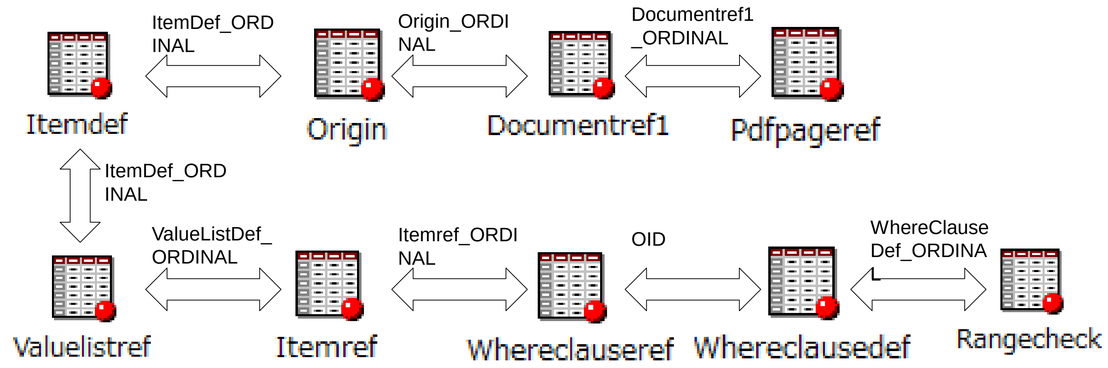
■Dataset情報取得



■ Variable情報取得



■ VLM情報取得



3. 比較結果

それぞれの実装にあたっての利点・欠点を下記の区分で表にまとめた。

実装方法	fdfファイル作成のシステム化	プログラムの可読性	Value Level Metadataの設定	注釈位置の設定	Not Submittedの対応
2-1	○	△	×	×	×
2-2	×	○	○	○	△
2-3	○	○	○	×	×

一つ目として、fdfファイルの作成に手作業なく自動作成ができるかという観点で比較すると、②の方法はdefine.xmlからExcelへの変換に手作業が必要なため、自動作成の実装には適さないと判断できる。
二つ目として、プログラムの可読性という観点で比較すると、①の方法で使用したprxchange関数はPerl正規表現の構文の理解が必要なため可読性が高いとは言えない。
三つ目として、Value Level Metadataの対応の観点で比較すると、
①に比べて、②、③の方がValue Level Metadataの対応が容易に行える。
四つ目として、注釈位置・Not Submittedの対応の観点で比較すると、②の方法が容易である。

以上のことから、③の実装方法が最も自動化に適していると判断できるが、fdfファイル作成過程での手作業を許容できるのであれば②の実装方法も手作業が入る以外の欠点はない。

4. 今後の検討事項について

今後の検討課題としては、以下を検討中
・Not Submittedの具体的な対応方法の検討
・The bookmarks act as Table of Contents (TOC)をSASで作成することを検討
・2次利用の検討 (コピー試験などの対応)