ODS markupを使ったADaM define-xmlの作成

坂上　拓　矢嶋　友也　西本　優美
株式会社 中外臨床研究センター
バイトメトリクス部 データサイエンスグループ

Creation of ADaM define-xml using SAS ODS markup

Taku Sakaue Tomoya Yajima Yumi Nishimoto
Biometrics Dept. Data Science Group, Chugai Clinical Research Center., LTD

要旨：

承認申請時のCDISC標準に準拠した臨床電子データの提出義務化を控え、これら臨床電子データ定義書としての位置づけとなるdefine-xmlの作成は，解析業務プロセスの一つに組み込まれることが予想される。
ADaMや解析帳票を作成する解析プログラミング業務のプロセスを考えた場合、define-xmlを作成するためのデータソースとして、プログラム開発者向けのプログラム仕様書を用いると、ADaMや解析帳票のプログラム仕様とdefine-xml間の整合性を保持する面で大きなメリットがある。しかしながら、define-xmlの要素や属性を考慮すると、プログラム仕様書には解析とは直接関係のない多大な情報を記入する必要があり、これらを埋めるための時間を割くことは、解析を主として行う担当者にはストレスとなる。
本発表は、当社で検討中のMicrosoft Excelで作成されたプログラム仕様書をデータソースとした、SAS ODS markupを使ったdefine-xml作成方法と、define-xmlの要素や属性に関する記入箇所を極力減らしたプログラム仕様書を紹介する。

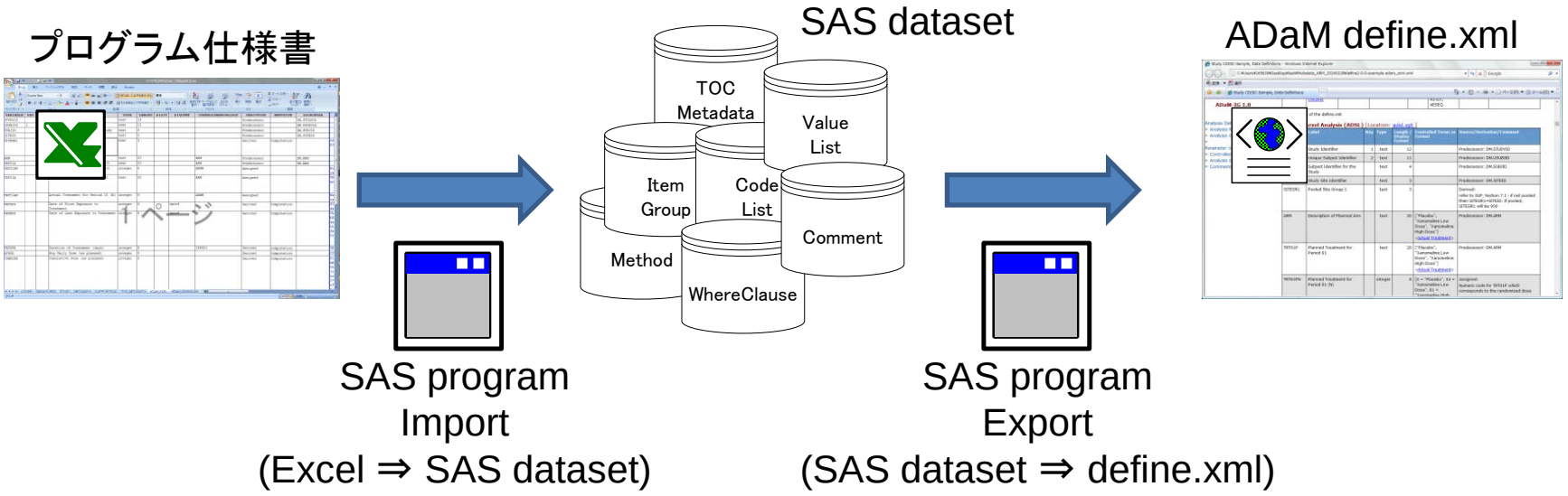
キーワード：ADaM define-xml、プログラム仕様書、SAS ODS markup

ADaM define-xml

- ADaM define-xmlの特徴
  基本的にデータセットに関してはSDTMのdefine-xmlと同じスキーマだが、解析帳票の仕様を定義したAnalysis Result Metadataが新たに必要となる。また、ADaMでは導出変数がほとんどで、DerivationやCommentが手厚く記載される傾向にある。

ADaM define-xml

- ADaM define.xml作成プロセス



プログラム仕様書

- Contents
  - ADaMや解析帳票のプログラム仕様と共に、define.xmlのデータソースとなるため、define.xmlを作成するために必要な情報も記載される。

ADSL Variable Metadata (sheet名：ADaM_ADSL)

VARIABLE	KEY	LIST	REQ	LABEL	TYPE	LENGTH	DIGIT	DISPFMT	CNTRLTERM	DERIVTYPE	METHTYPE	SOURCEVAR
STUDYID	1			Study Identifier	text	18				Predecessor		DM.STUDYID
USUBJID	2			Unique Subject Identifier	text	11				Predecessor		DM.USUBJID
SUBJID				Subject Identifier for the Study	text	4				Predecessor		DM.SUBJID
SITEID				Study Site Identifier	text	3				Predecessor		DM.SITEID
SITEGR1				Pooled Site Group 1	text	3				Derived	Computation	
ARM				Description of Planned Arm	text	20			ARM	Predecessor		DM.ARM
TRT01P				Planned Treatment for Period 01	text	20			ARM	Predecessor		DM.ARM
TRT01PN				Planned Treatment for Period 01 (N)	integer	8			ARMN	Assigned		
TRT01A				Actual Treatment for Period 01	text	20			ARM	Assigned		

DERIVATION
Numeric code for TRT01P which corresponds to the randomized dose
TRT01A=TRT01P, i.e., no difference between actual and randomized treatment in this study.

プログラム仕様書

- プログラム仕様書の定義情報
  - 定数値として定義ができる要素や、プログラム仕様書で定義されている情報から自動的に生成できる要素、プログラム仕様書の作成時に運用レベルで解決できるような要素は、プログラム仕様書には入力箇所を設けていない。

プログラム仕様書

- 定数値として定義ができる要素

```
<ODM xmlns="http://www.cdisc.org/ns/odm/v1.3"
     xmlns:def="http://www.cdisc.org/ns/def/v2.0"
     xmlns:xlink="http://www.w3.org/1999/xlink"
     xmlns:arm="http://www.cdisc.org/ns/arm/v1.0"
     ODMVersion="1.3.2"
     FileOID="Example"
     FileType="Snapshot"
     CreationDateTime="YYYY-MM-DDThh:mm:ss"
     Originator="CCRC">

<MetaDataVersion OID="MDV.EXAMPLE.ADaMIG.1.0.ADaM.2.1"
      Name="Example"
      Description="Example for SUGJ2014"
      def:DefineVersion="2.0.0"
      def:StandardName="ADaM-IG"
      def:StandardVersion="1.0">
```

プログラム仕様書

- プログラム仕様書で定義されている情報から自動的に生成できる要素

ADaM Metadata (sheet:TOC_METADATA)

NAME	REPEATING	DESCRIPTION	STRUCTURE	CLASS	PATH	DOCUMEMTATION
ADSL	No	Subject-Level Analysis	One record per subject	ADSL	./ADSL.xpt	Screen Failures are excluded since they are not needed for this study analysis

```
<ItemGroupDef OID="IG.ADSL"
              Name="ADSL"
              SASDatasetName="ADSL"
              Repeating="No"
              IsReferenceData="No“
              Purpose="Analysis"
              def:Structure=“One record per subject"
              def:Class=“ADSL"
              def:CommentOID='COM.ADSL"
              def:ArchiveLocationID="LF.ADSL">
```

プログラム仕様書

- 運用レベルで解決できるような要素

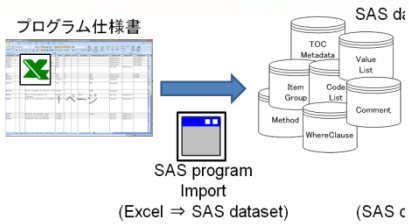
ADSL Variable Metadata (sheet名：ADaM_ADSL)

VARIABLE	KEY	LIST	REQ	LABEL	TYPE	LENGTH
STUDYID	1			Study Identifier	text	18
USUBJID	2			Unique Subject Identifier	text	11
SUBJID				Subject Identifier for the Study	text	4
SITEID				Study Site Identifier	text	3
SITEGR1				Pooled Site Group 1	text	3
ARM				Description of Planned Arm	text	20
TRT01P				Planned Treatment for Period 01	text	20
TRT01PN				Planned Treatment for Period 01 (N)	integer	8

ItemRef (OrderNumber)順に変数を定義する

```
<ItemRef ItemOID="IT.ADSL.STUDYID"   OrderNumber="1“ />
<ItemRef ItemOID="IT.ADSL.USUBJID"   OrderNumber="2“ />
<ItemRef ItemOID="IT.ADSL.SUBJID"    OrderNumber="3“ />
<ItemRef ItemOID="IT.ADSL.SITEID"    OrderNumber="4“ />
<ItemRef ItemOID="IT.ADSL.SITEGR1"   OrderNumber="5“ />
<ItemRef ItemOID="IT.ADSL.ARM“       OrderNumber="6“ />
<ItemRef ItemOID="IT.ADSL.TRT01P"    OrderNumber="7“ />
<ItemRef ItemOID="IT.ADSL.TRT01PN"   OrderNumber="8“ />
```

プログラム仕様書のSAS dataset化



Value Level Metadata (sheet名：VALUELIST)

DATASET	VARIABLE	CHECKVAR	COMPARATOR	CHECKVALUE	DERIVATION	REFDOC
ADQSADAS	AVAL	PARAMCD	IN	ACITM01, ACITM02, ACITM03, ACITM04, ACITM05, ACITM06, ACITM07, ACITM08, ACITM09, ACITM10, ACITM11, ACITM12, ACITM13, ACITM14	QS.QSSTRESN where QSTESTCD=PARAMCD	
ADQSADAS	AVAL	PARAMCD	EQ	ACTOT	Sum of ADAS scores for items 1, 2, 4, 5, 6, 7, 8, 11, 12, 13, and 14, see Analysis Data Reviewers Guide (Page 3) for details on adjusting for missing values.	ADRG / PhysicalRef / 3

ValueListDef/ItemRef/def:WhereClauseRef

Value Level Metadata

MethodDef/Description/TranslatedText
MethodDef/def:DocumentRef/def:PDFPageRef

Method

def:WhereclauseDef/RangeCheck/CheckValue

Where Clause

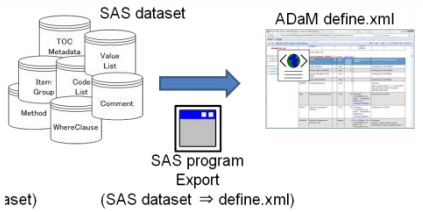
define.xmlの作成

- SAS ODS markup
  - SAS ODS markupは、XMLやHTMLのようにタグに囲われた文書の入出力に特化した機能を持っている。

```
ODS markup tagset=sample.definexml_v2 file=”C¥:xmlout¥define.xml”;

  proc print data=WORK.odm;
  run;

  proc print data=WORK.itemgroup;
  run;

ODS markup close;
```

ODS markup実行方法

define.xmlの作成



- Template作成
  - SAS XML engineのevent modelを使用して作成する

```
proc template ;
  define tagset <tagset name> ;

    define event <event name >;
      start :
        <processing>;
      finish :
        <processing>;
    end;

  end ;
run;
```

カスタマイズするtemplateの定義領域

イベント定義領域

- XML engine由来のイベント
- ユーザー定義イベント

templateの基本構造

define.xmlの作成

- イベントで実施可能な処理

```
define event <event name >;
  start :
    <processing>;
  finish :
    <processing>;
end;
```

変数の定義

```
set $adamigver '1.0' ;
set $constant['adamigver'] '1.0' ;
set $varlist[1] 'A' ;
set $varlist[2] 'B' ;

unset $adamigver ;

eval $index $index+1;
```

scalar変数
dictionary変数
list変数
変数の初期化
変数の上書き
計算値の代入

出力

```
ndent ;
xdent ;
```

define tagset statement option indentで定義した分の
インデント発生(ndent)、インデント解除(xdent)

```
put '<?xml version="1.0" encoding="UTF-8" ?>' NL;
putl '<?xml-stylesheet type="text/xsl" href="../stylesheets/define2-0-0.xsl"?>' ;
```

ファイルへ出力

define.xmlの作成

- イベントで実施可能な処理

条件分岐

```
do / if cmp(upcase($dataname), 'ODM');
  trigger ODMmeta;
else
  break
done;

break / if not exists($dataname);

trigger CLOSETAG / if $dataname any ('ODM', 'ITEMGROUPDEF', 'ITEMREF');

trigger CLOSETAG / if $dataname contains('ADAM');
```

一般的な if / else式
cmpは完全一致

変数 or 値の存在確認 (notは否定)

いずれかの値の完全一致

部分一致

define.xmlの作成

- eventの挙動

```
define event <event name >;
    start :
      <processing>;
    finish :
      <processing>;
end;
```

イベント発生時の処理

イベント終了時の処理

```
define event IGmeta;
 start  :
   ndent;ndent;ndent;
   put '<ItemGroupDef';
 finish :
   put '</ItemGroupDef>' NL;
end;

define event XMLversion;
 put  '<?xml version="1.0" encoding="UTF-8" ?>' NL;
end;
```

start / finishを指定しないパターンも可
(目的に応じて使い分ける)

define.xmlの作成

- eventの種類
  - XML engine由来のイベント
    - イベント名が既に決まっている
    - 決まったタイミングで実行される
  - ユーザー定義のイベント
    - 起動タイミングはユーザーが決める。

```
define event doc;
   start  :
      trigger hello_world;
   finish :
      trigger hello_world;
end;
```

XML engine由来イベント

実効命令

```
define event hello_world;
   start  :
      put 'HELLO' NL;
   finish :
      put 'WORLD';
end;
```

ユーザ定義イベント

define.xmlの作成

- XML engine由来イベント実行タイミング
  - 文書単位のイベント

initialize (start , finish)

doc (start)
doc_head (start, finish)
doc_body (start)

XML file

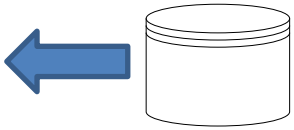
doc_body (finish)
doc (finish)

- initialize
  文書全体で使用する定数値の宣言
- doc_head
  XMLコメントの出力等
- doc / doc_body
  文書の冒頭と末尾で指定したいelementを出力（XMLコメント，ODM element , Study Element ）

define.xmlの作成

- XML engine由来イベント実行タイミング

proc (start)
leaf (start)
output (start)
table (start)
colspec_entry (start, finish)
table_head (start, finish)
table_body (start)
row (start)
data (start, finish)
row (finish)
table_body (finish)
table (finish)
output (finish)
leaf (finish)
proc (finish)



- leaf
  input データセット名をVALUEから取得

- colspec_entry
  input datasetの変数名を1つずつ順番に参照
  (NAMEから取得)

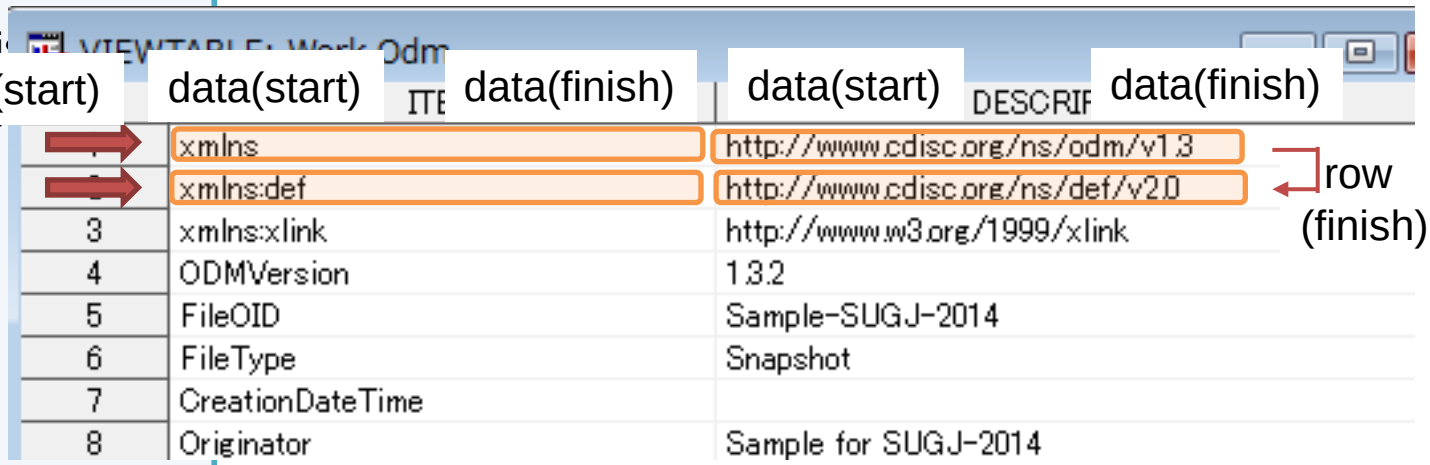
VIEWTABLE: Work.O

	ITEM	DESCRIPTION
1	xmlns	http://www.cdisc.org/ns/odm/v1.3
2	xmlns:def	http://www.cdisc.org/ns/def/v2.0
3	xmlns:xlink	http://www.w3.org/1999/xlink
4	ODMVersion	1.3.2
5	FileOID	Sample-SUGJ-2014
6	FileType	Snapshot
7	CreationDateTime	
8	Originator	Sample for SUGJ-2014

define.xmlの作成

- XML engine由来イベント実行タイミング

proc (start)
leaf (start)
output (start)
table (start)
colspec_entry (start, finish)
table_head (start, finish)
table_body (start)
row (start)
data (start, finish)
row (finish)
table_body (finish)
table (finish)
output (finish)
leaf (finish)
proc (finish)



define.xmlの作成

- XML engine由来イベント実行タイミング
  - データセットの情報を取得するための一連の処理

```
define event colspec_entry;
  set $col_names[] upcase(NAME);
end;
```

```
define event row;
  start  :
      eval  $idx 1;
      unset $col_values;
  finish :
      trigger <ユーザー定義イベント>;
end;
```

```
define event data;
  start  :
      unset $vname;
      set $vname $col_names[$idx];
      set $col_values[$vname] VALUE;
  finish :
      eval $idx $idx+1;
end;
```

VIEWTABLE: Work.Odm

	ITEM	DESCRIPTION
1	xmlns	http://www.cdisc.org/ns/odm/v1.3
2	xmlns:def	http://www.cdisc.org/ns/def/v2.0
3	xmlns:xlink	http://www.w3.org/1999/xlink
4	ODMVersion	1.3.2
5	FileOID	Sample-SUGJ-2014
6	FileType	Snapshot
7	CreationDateTime	
8	Originator	Sample for SUGJ-2014

VARIABLE	KEY	ORDER	REQ	LABEL	TYPE	LENGTH	DERIVTYPE	METHTYPE
STUDYID	1	1	No	Study Identifier	text	18	Predecessor	
USUBJID	2	2	No	Unique Subject Identifier	text	11	Predecessor	
SUBJID	.	3	No	Subject Identifier for the Study	text	4	Predecessor	
SITEID	.	4	No	Study Site Identifier	text	3	Predecessor	
SITEGR1	.	5	No	Pooled Site Group 1	text	3	Derived	Computation

```
define event IRattr;
  set $var $col_values["VARIABLE"];
  do / if $var;
    put '<ItemRef ItemOID="IT.ADSL.' $var '"';
  else ;
    break;
  done;
  set $val $col_values["ORDER"];
  put ' OrderNumber=' quote($val);
  set $val $col_values["REQ"];
  put ' Mandatory=' quote($val);
  set $val $col_values["KEY"];
  do / if strip($val);
    put ' KeySequence=' quote($val);
  done;
  set $val $col_values["METHTYPE"];
  do / if cmp(upcase($val), 'COMPUTATION');
    put ' MethodOID="MT.ADSL.' $var '"';
  done;
  put '/>' NL;
end;
```

< ItemRef ItemOID=“IT.ADSL.STUDY” **OrderNumber=“**1” **Mandatory=“**No” **KeySequence=“**1”**/>**
< ItemRef ItemOID=“IT.ADSL.USUBJID” **OrderNumber**=“2” **Mandatory=**“No” **KeySequence=**“2” **/>**
< ItemRef ItemOID=“IT.ADSL.SUBJID” **OrderNumber**=“3” **Mandatory=**“No” **/>**
< ItemRef ItemOID=“IT.ADSL.SITEID” **OrderNumber**=“4” **Mandatory=**“No” **/>**
< ItemRef ItemOID=“IT.ADSL.SITEGR1” **OrderNumber**=“5” **Mandatory=**“No”
MethodOID="MT.ADSL.SITEGR1” **/>**

define.xmlの作成

```
define event doc;
 start  :
    trigger Study;
 finish :
    trigger Study;
end;
```

```
define Study;
 start :
    putl '<Study OID="SUGJ2014">';
 finish :
    putl '</Study>';
    trigger OMDclose;
end;

define ODMclose;
   put '</ODM>';
end;
```

doc (start)

doc (finish)

define.xml

```
<?xml version= …….. ?>
<?xml-stylesheet ……..?>
 < ODM …….. >
<Study OID="SUGJ2014">

</ Study>
</ ODM>
```

まとめ

- プログラム仕様書をdefine.xmlのデータソースとして使用する場合、以下の情報は省略が可能
  - 定数値として定義できる要素・属性情報
    - metadataとして管理すべき
  - 変数名、データセット名といった、基本的な情報の組み合わせで定義できるような情報
  - 運用レベルで解決できるような情報
- ODS markupとtemplateを使うことで、SAS datasetからdefine.xmlの作成を容易にする
  - 運用にあわせて、templateをカスタマイズして運用することが出来る。
  - ユーザ定義イベントとして、define-xml要素単位での管理ができるため、保守性に優れている。