PharmaSUG China 2025 - Paper AA-124

Automating TA.ARM Mapping and TV/TPT Specification Generation

Guangan Cao, Liming Xie, Qiuan Hu and Chunpeng Zhao, BeOne Medicines

ABSTRACT

In clinical trials, the accurate and consistent derivation of key variables such as ARM (planned treatment arm), VISIT, and time point identifiers is essential for producing high-quality SDTM datasets aligned with the protocol-defined study design. Traditionally, these derivations are performed manually, which can be time-consuming and inefficient, increasing the risk of inconsistency, particularly in large or complex clinical trials.

This paper will provide a metadata-driven approach, implemented using VBA and SAS, to streamline the derivation of ARM, visit, and time point variables, leveraging trial design metadata from the TA (Trial Arms) and TV (Trial Visits) domains, by integrating raw randomization or enrollment data sources. The ARM derivation is automated by mapping planned treatment assignments collected in EDC (Electronic Data Capture) and IRT (Interactive Response Technology) systems to TA domain metadata, with the relationships stored in a structured trial design matrix called TA_ARM.

Visit and time point derivations are automated through dynamic calculations based on protocol-defined schedules, enabling standardized generation of VISIT, VISITNUM, TPT, and TPTNUM variables using specifications from the TV and time point (TPT) metadata.

This integrated automation framework significantly improves programming efficiency, reduces manual rework, and ensures consistent, traceable alignment with protocol specifications. Notably, no programming is needed anymore for ARM mapping, further streamlining the process and reducing the potential for human error.

INTRODUCTION

In the development of clinical trial data, generating high-quality and protocol-aligned SDTM (Study Data Tabulation Model) datasets is a critical step toward regulatory submission and downstream analysis. Among the essential components of SDTM data are variables such as ARM (planned treatment arm), VISIT, and time point identifiers (e.g., TPT and TPTNUM), which directly reflect the study's planned design and visit schedule. Ensuring these variables are derived accurately and consistently is fundamental to data integrity and traceability.

Traditionally, the derivation of these variables has relied heavily on manual programming based on the interpretation of protocol documents and raw clinical data. While flexible, this manual approach is often labor-intensive, prone to inconsistency, and difficult to scale across multiple studies or complex protocols. In particular, mapping planned treatment arms (ARM) from IRT or EDC systems to the SDTM TA domain often requires study-specific coding, which can introduce variability and inefficiency.

To address these challenges, this paper presents a metadata-driven automation framework implemented using Visual Basic for Applications (VBA). By leveraging structured metadata from SDTM trial design domains—specifically TA (Trial Arms), TV (Trial Visits), and time point specifications (TPT Specs)—along with raw data from enrollment or randomization sources, the framework enables automated and standardized derivation of ARM, VISIT, and time point variables. A key innovation of this approach is the introduction of the TA_ARM matrix, which encodes the relationships between treatment assignments and trial arms, eliminating the need for custom programming in ARM mapping.

Furthermore, the framework supports the dynamic generation of TV and TPT specifications, based on protocol-defined visit schedules and assessment windows. These specifications serve as the foundation for automated derivation of VISIT, VISITNUM, TPT, and TPTNUM variables, enabling consistent, specification-driven output across studies.

Through this approach, the derivation process becomes not only faster and more scalable but also fully aligned with protocol specifications in a traceable and auditable manner. This paper will detail the architecture, implementation, and benefits of this framework, demonstrating how metadata can be effectively used to automate what were once manually intensive tasks in clinical programming.

TA_ARM MATRIX: A METADATA-DRIVEN ARM SOLUTION

Accurate derivation of ARM and ARMCD is critical for ensuring each subject is correctly mapped to their protocol-defined treatment group. In traditional workflows, this mapping often requires study-specific code to interpret raw data from either EDC systems or IRT systems and reconcile it with the protocol-defined arms listed in the SDTM TA domain. Such approaches can be inconsistent, error-prone, and inefficient, especially in large-scale or multi-arm trials.

To address these limitations, we introduce the TA_ARM Matrix—a flexible, structured metadata table embedded within the trial design framework. It standardizes the derivation of ARM and ARMCD by establishing configurable relationships between raw subject-level data and trial design metadata. The matrix supports both EDC-based enrollment data and IRT-based randomization systems, enabling automated, study-independent mapping without custom code.

The matrix defines planned arms (ARM) and arm codes (ARMCD) using a set of configurable metadata columns that map raw subject-level values to the corresponding TA domain values.

Columns	Required	Description
Dataset	Yes	Source dataset name. 1. Enrollment SDTM minus dataset name, e.g., DC_ for Study with Field Arm in Form Enrollment 2. IRT Dataset Name with library name, e.g., IRT.rand for Study without Field Arm in Form Enrollment.
ArmVariable	Yes	1. Arm corresponding CRF Field Names for study with Field Arm in Form Enrollment. CRF Field Names are separated by “|”, if there are multiple ones. e.g., phase|part|doselvl. 2. Arm corresponding IRT Variable Names for study without Field Arm in Form Enrollment. IRT Variable Names are separated by “|”, if there are multiple ones.
ArmVariableValue	Yes	1. Arm corresponding CRF Field Options for study with Field Arm in Form Enrollment. CRF Field Options are separated by “|”, if there are multiple fields. e.g., PHASE 1A|PART A|30 MG BID. 2. Arm corresponding IRT Variable Code list for study without Field Arm in Form Enrollment. IRT Variable Values are separated by “|”, if there are multiple variables.
ArmCDVariable	Optional	ARMCD corresponding IRT Variable Name.
ArmCDVariableValue	Optional	ARMCD corresponding IRT Variable Code list
ArmRatio	Yes	Arm Allocation Ratio. e.g., 1:1, 3:1, etc.

Table 1. TA_ARM: Key metadata columns for Trial Design Matrix

C	D	E	F	G	H	I	J
ARMCD	ARM	Dataset	ArmVariable	ArmVariableValue	ArmCDVariable	ArmCDVariableValue	ArmRatio
P1A-PA-30	PHASE 1A - PART A - 30 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|30 MG BID			1:1
P1A-PA-60	PHASE 1A - PART A - 60 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|60 MG BID			1:1
P1A-PA-120	PHASE 1A - PART A - 120 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|120 MG BID			1:1
P1A-PA-240	PHASE 1A - PART A - 240 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|240 MG BID			1:1
P1A-PA-450	PHASE 1A - PART A - 450 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|450 MG BID			1:1
P1A-PA-750	PHASE 1A - PART A - 750 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|750 MG BID			1:1
P1A-PA-900	PHASE 1A - PART A - 900 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART A|900 MG BID			1:1
P1A-PB1-30	PHASE 1A - PART B1 - 30 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|30 MG BID			1:1
P1A-PB1-60	PHASE 1A - PART B1 - 60 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|60 MG BID			1:1
P1A-PB1-120	PHASE 1A - PART B1 - 120 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|120 MG BID			1:1
P1A-PB1-240	PHASE 1A - PART B1 - 240 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|240 MG BID			1:1
P1A-PB1-450	PHASE 1A - PART B1 - 450 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|450 MG BID			1:1
P1A-PB1-750	PHASE 1A - PART B1 - 750 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|750 MG BID			1:1
P1A-PB1-900	PHASE 1A - PART B1 - 900 MG BID	DC_	phase|part|doselvl	PHASE 1A|PART B1|900 MG BID			1:1

Display 1. Example of TA_ARM for Automated Mapping of ARM and ARMCD

To operationalize this metadata-driven approach, we developed a companion SAS macro that reads the TA_ARM Matrix and applies the defined mappings to derive each subject's planned arm. During the creation of the SDTM DC domain, this macro is invoked to match raw subject-level values to the corresponding ARM and ARMCD definitions based on the matrix logic. By integrating metadata interpretation directly into the derivation process, the macro ensures that all subjects are assigned the correct planned treatment arm automatically and consistently, without requiring any study-specific programming.

OVERVIEW OF THE SAS PROGRAM

Step 1, Retrieve structured Arm metadata from TA_ARM Matrix

```
data _taarm;
    set TAARM.ta_arm end = lastofobs;
    where not missing(ArmVariable);
    retain arm_condition;
    length arm_condition $20000;
    array armvar(10) $200 _temporary_;
    array armvalue(10) $200 _temporary_;
    num_armvar = count(ArmVariable, '|') + 1;
    num_armvalue = count(ArmVariableValue, '|') + 1;

    if num_armvar ne num_armvalue then do;
       put 'USER WAR''NING: Variable and Value provided need to be check -
' ArmVariable = ArmVariableValue = ;
    end;
    else do;
    ***Get the condition to derive ARMCD ARM;
       do i = 1 to num_armvar;
          armvar(i) = scan(ArmVariable, i, '|');
          armvalue(i) = scan(ArmVariableValue, i, '|');
          if i = 1 then arm_condition = catx(' = ', cats('if cats(upcase(',
cats(armvar(i)), '))'), cats('"', cats(upcase(armvalue(i))), '"'));
          else arm_condition = catx(' and ', cats(arm_condition), catx(' =
', cats('cats(upcase(', cats(armvar(i)), '))'), cats('"',
cats(upcase(armvalue(i))), '"')));
```

```
                    if i = num_armvar then arm_condition = catx(' ',
arm_condition, ' then do; ', cats('armcd = ', '"'||cats(armcd)||'";'),
cats('arm = ', '"'||cats(arm)||'";'), ' end;');
       end;
        if index(Dataset, '.') then do;
           if libref(scan(Dataset, 1, '.')) = 0 and
lowcase(cats(scan(Dataset, 1, '.'))) ne 'work' then do;
              call symput('_irt_dataset', Dataset);
           end;
        end;
    end;
    call symput('var_list', '"'||upcase(cats(tranwrd(ArmVariable, '|', '"
"'))))||'"');
    call symput('var_num', cats(num_armvar));
run;
```

Step 2, Derive ARM and ARMCD value base on condition on step 1

```
data _null_;
    set _taarm end = lastofobs;
    if _n_ = 1 then call execute("data _dc_arm; length ARM $200 ARMCD $20;
set &DsIn.; ");
    call execute(arm_condition);
    if lastofobs then call execute("run;");
run;
```

AUTOMATING TV AND TPT SPECIFICATION GENERATION

In SDTM data preparation, accurate derivation of visit-related (VISIT, VISITNUM) and time point variables (TPT, TPTNUM, ELTM, TPTREF) is essential to ensure the data reflects the planned protocol schedule and dosing-relative events. Traditionally, these derivations rely on manual programming, referencing protocol documents and visit maps. However, such approaches are labor-intensive, study-specific, and error-prone.

To address these limitations, we developed an automated solution to generate TV (Trial Visits) and TPT (Time Point) specifications directly from structured protocol metadata using VBA. These specifications serve as the foundation for downstream derivation of subject-level visit and time point variables in a standardized and scalable manner.

TV SPECIFICATION AUTOMATION

The TV (Trial Visits) domain defines the protocol-planned visit structure, including VISITNUM, VISIT, and optional visit windows. In this framework, the TV domain is generated through a VBA macro that identifies forms associated with multiple folders based on the ALS file exported from the EDC Rave system.

The macro processes the MASTERDASHBOARD tab in the ALS to detect forms collected across multiple folders, extract folder names, and map them to their corresponding protocol-defined visits. It encodes the visit schedule, including visit names and numbers, and programmatically constructs the TV domain.

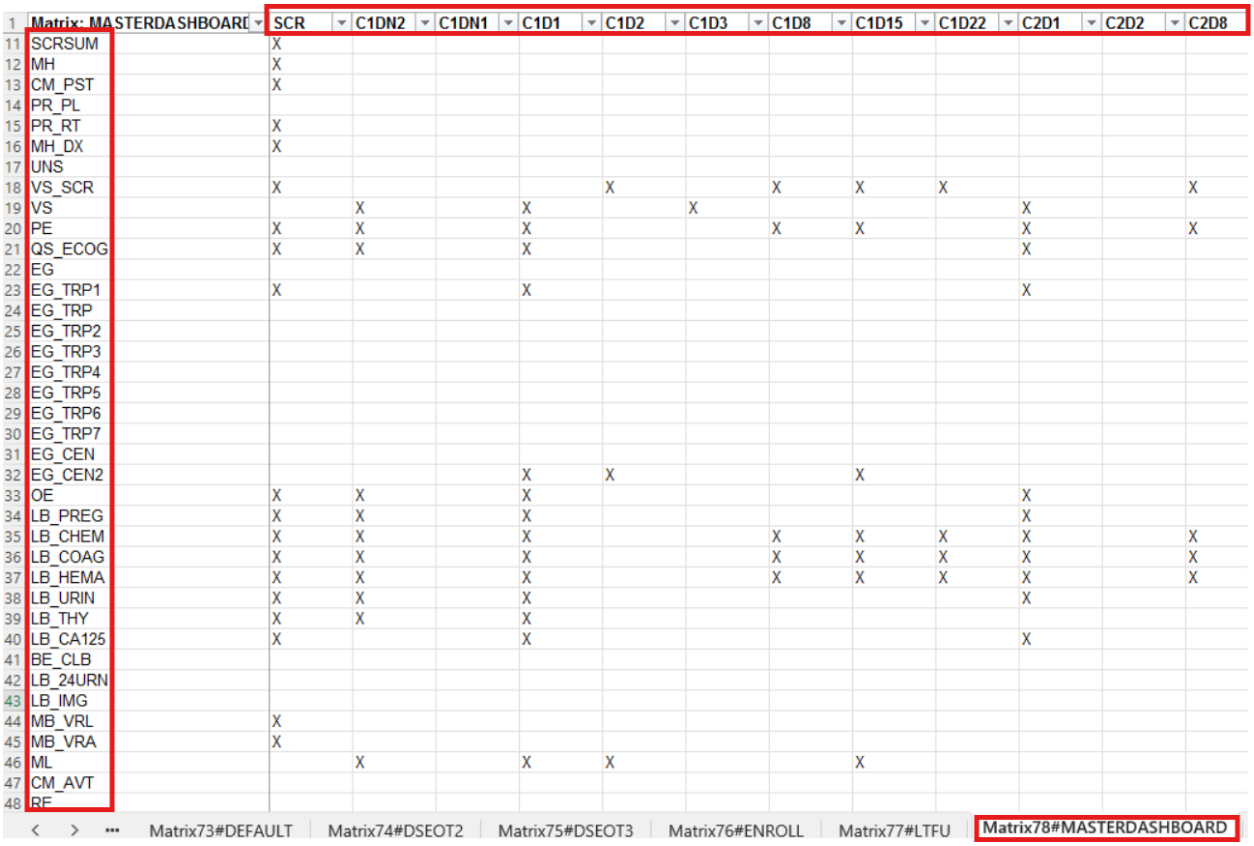
This folder-to-visit mapping is used to drive consistent derivation of VISIT and VISITNUM in downstream domains (e.g., LB, VS, EG), ensuring alignment between data collection structure in EDC Rave system and the protocol-specified visit plan.

This direct TV generation and integration with ALS folder logic brings several key benefits:

- Little manual transcription of visit schedules

- Automated handling of multi-folder form collections
- Standardized visit labels and numbers across SDTM domains
- Protocol-driven logic embedded within reusable macro code

By removing dependency on external visit spec files and directly interpreting protocol structure and ALS metadata, this method ensures traceable, efficient, and consistent visit handling in SDTM preparation.



Matrix: MASTERDASHBOARD	SCR	C1DN2	C1DN1	C1D1	C1D2	C1D3	C1D8	C1D15	C1D22	C2D1	C2D2	C2D8
SCRSUM	X											
MH	X											
CM_PST	X											
PR_PL												
PR_RT	X											
MH_DX	X											
UNS												
VS_SCR	X				X		X	X	X			X
VS		X		X		X				X		
PE	X	X		X			X	X		X		X
QS_ECOG	X	X		X						X		
EG												
EG_TRP1	X			X						X		
EG_TRP												
EG_TRP2												
EG_TRP3												
EG_TRP4												
EG_TRP5												
EG_TRP6												
EG_TRP7												
EG_CEN												
EG_CEN2				X	X			X				
OE	X	X		X						X		
LB_PREG	X	X		X						X		
LB_CHEM	X	X		X			X	X	X	X		X
LB_COAG	X	X		X			X	X	X	X		X
LB_HEMA	X	X		X			X	X	X	X		X
LB_URIN	X	X		X						X		
LB_THY	X	X		X								
LB_CA125	X			X						X		
BE_CLB												
LB_24URN												
LB_IMG												
MB_VRL	X											
MB_VRA	X											
ML		X		X	X			X				
CM_AVT												
RF												

Matrix73#DEFAULT | Matrix74#DSEOT2 | Matrix75#DSEOT3 | Matrix76#ENROLL | Matrix77#LTFU | Matrix78#MASTERDASHBOARD

Display 2. Example of the MASTERDASHBOARD tab within the ALS file

OVERVIEW OF THE VBA PROGRAM

Step 1, Extract folders for forms collected across multiple folders

```
Sub Create_TV()
Dim RegEx As Object, Strings, VisitList As String, rb, aimf As Workbook
Set RegEx = CreateObject("VBScript.RegExp")
RegEx.IgnoreCase = True
Set rb = ThisWorkbook
InPath_ALS = Trim(rb.sheets("TDM_AUTO").Cells(5, 2))
Set aimf = CreateObject(InPath_ALS)
sheetn = aimf.Sheets.Count
For j = 1 To sheetn
    If InStr(aimf.Sheets(j).Name, "MASTERDASHBOARD") Then
       Set dic_page = CreateObject("scripting.dictionary")
       allRows = aimf.Sheets(j).Range("a65536").End(xlUp).Row
       cols= aimf.Sheets(j).Range("IV1").End(xlToLeft).Column
```

```
         For h = 2 To allRows
            pagesum = 0
            formname = Trim(aimf.Sheets(j).Cells(h, 1))
            dic_page(formname) = "Exclude"
                For f = 2 To cols
                    Set cell = aimf.Sheets(j).Cells(h, f)
                    If cell.Value <> "" Then
                       pagesum = pagesum + 1
                       If pagesum > 1 Then
                          dic_page(formname) = "Include"
                          Exit For
                        End If

                     End If
                Next f
          Next h
          VisitList = ""
          For f = 2 To cols
             visitsum = 0
             For h = 2 To allRows
                Set cell = aimf.Sheets(j).Cells(h, f)
                cname=Trim(aimf.Sheets(j).Cells(1, f))
                If cell.Value <> "" And dic_page(formname) = "Include" Then
                   visitsum = visitsum + 1
                   If visitsum > 1 Then VisitList = VisitList & "#" & cname
                      Exit For
                   End If
                End If
              Next h
           Next f
    End If
Next j
```

Step 2, Mapping extracted folders to appropriate VISIT and VISITNUM

```
'Get TDM_AUTO last row
If UCase(Sheets("TDM_AUTO").Cells(7, 2)) = "NO" Then
   TDM_LastRow = 14
   Visit_Not_By_Arm = 1
 Else
   TDM_LastRow = rb.Worksheets("TDM_AUTO").Range("a65536").End(xlUp).Row
End If
'Loop reading Folders worksheet
For i = 14 To TDM_LastRow
  lastrow = aimf.Worksheets("Folders").Cells(Rows.Count, "C").End(xlUp).Row
  studyid = rb.Worksheets("TDM_AUTO").Range("B3")
  arm = rb.Worksheets("TDM_AUTO").Cells(i, 1)
  armcd = rb.Worksheets("TDM_AUTO").Cells(i, 1)
  For m = 2 To lastrow
      Abbstr = Trim(aimf.Worksheets("Folders").Cells(m, 1).Value)
      If InStr(VisitList, "#" & Abbstr) Then
         Strings = aimf.Worksheets("Folders").Cells(m, 3).Value
         VisNum = aimf.Worksheets("Folders").Cells(m, 2)
         temp = temp + 1
         rb.Worksheets("TV").Cells(temp, 1) = studyid
         rb.Worksheets("TV").Cells(temp, 2) = "TV"
```

```
            rb.Worksheets("TV").Cells(temp, 4) = upcase(Strings)
            rb.Worksheets("TV").Cells(temp, 3) = VisNum
            If Visit_Not_By_Arm <> 1 Then
               rb.Worksheets("TV").Cells(temp, 5) = arm
               rb.Worksheets("TV").Cells(temp, 6) = armcd
            End if
            'TVSTRL TVENRL
            rb.sheets("TV").Cells(temp, 7)="The Start of " & Strings &" Visit"
            rb.sheets("TV").Cells(temp, 8)="The End of " & Strings & " Visit"
            rb.Worksheets("TV").Cells(temp, 9) = Strings
            rb.Worksheets("TV").Cells(temp, 10) = "Non-Image"
            'Check if image visit
            RegEx.Pattern = "Response Assessment"
            Set Matches_RASS_VisType = RegEx.Execute(Strings)
            For Each Match_RA In Matches_RASS_VisType
                rb.Worksheets("TV").Cells(temp, 10) = "Image"
            Next
        End If
    Next
Next
End Sub
```

STUDYID	DOMAIN	VISITNUM	VISIT	ARMCD	ARM	TVSTRL	TVENRL	FolderName	Visit_Type
BGB-XXX-XX	TV	10000	SCREENING			The Start of Screening Visit	The End of Screening Visit	Screening	Non-Image
BGB-XXX-XX	TV	20101	CYCLE 1 DAY 1			The Start of Cycle 1 Day 1 Visit	The End of Cycle 1 Day 1 Visit	Cycle 1 Day 1	Non-Image
BGB-XXX-XX	TV	20122	CYCLE 1 DAY 22			The Start of Cycle 1 Day 22 Visit	The End of Cycle 1 Day 22 Visit	Cycle 1 Day 22	Non-Image
BGB-XXX-XX	TV	21001	CYCLE 10 DAY 1			The Start of Cycle 10 Day 1 Visit	The End of Cycle 10 Day 1 Visit	Cycle 10 Day 1	Non-Image
BGB-XXX-XX	TV	23699	RESPONSE ASSESSMENT WEEK 109			The Start of Response Assessment Week 109 Visit	The End of Response Assessment Week 109 Visit	Response Assessment Week 109	Image
BGB-XXX-XX	TV	60000	END OF TREATMENT			Within 7 Days of EOT Decision is Made		End of Treatment	Non-Image
BGB-XXX-XX	TV	70000	SAFETY FOLLOW-UP			At 30 Days (+/- 7 Days) after Last Dose of Study Drugs		Safety Follow-up	Non-Image
BGB-XXX-XX	TV	80001	SURVIVAL FOLLOW-UP 1			Every 3 Months (+/- 14 Days) after Last Safety Follow-up Visit		Survival Follow-up 1	Non-Image
BGB-XXX-XX	TV	90000	LONG-TERM FOLLOW-UP			Every 6 Months (+/- 14 Days) after Last Safety Follow-up Visit		Long-Term Follow-up	Non-Image

Display 3. Example of TV Specification for Automated Mapping of VISIT

TIME POINT SPECIFICATION AUTOMATION

The Time Point (TPT) specification defines how protocol-scheduled assessments are translated into standardized TPT and TPTNUM variables in SDTM domains such as PK, LB, or VS. Traditionally, these time point derivations rely on manual programming based on protocol tables, which can be inefficient and prone to inconsistency, especially for studies with complex sampling schedules.

In this framework, time point derivation is fully automated using a VBA macro that generates a TPT Specification Table. This table defines the protocol-planned time point structure needed to derive --TPT and --TPTNUM consistently within each SDTM domain and includes only records with non-missing time point values.

During SDTM preparation, the macro matches the collected data to the TPT Specification Table. It extracts the specific time point variable name and value from the in-house CRF2SDTM specification and assigns the corresponding --TPTNUM within each domain. This ensures each record is mapped accurately and consistently in line with the protocol-defined time point schedule.

Each record in the TPT Specification Table includes:

Columns	Required	Description
Domain	Yes	the SDTM domain where the time point applies.
VariableName	Yes	the specific TPT variable name (e.g., PKTPT).
VariableValue	Yes	the actual value for the variable (e.g., “Pre-dose”).
VariableValueSeq	Yes	The numeric sequence of the time points within each domain, serving as the --TPTNUM.

Table 2. TPT Specification: Key metadata columns

	A	B	C	D	E
1	Studyid	Domain	VariableName	VariableValue	VariableValueSeq
2	BGB-XXX-XX	LB	LBTPT	PREDOSE	1
3	BGB-XXX-XX	LB	LBTPT	4H POSTDOSE	2
4	BGB-XXX-XX	LB	LBTPT	8H POSTDOSE	3
5	BGB-XXX-XX	LB	LBTPT	12H POSTDOSE	4
6	BGB-XXX-XX	LB	LBTPT	24H POSTDOSE	5
7	BGB-XXX-XX	BE	BETPT	PREDOSE	1
8	BGB-XXX-XX	BE	BETPT	2 HOURS POSTDOSE	2
9	BGB-XXX-XX	BE	BETPT	4-6 HOURS POSTDOSE	3

Display 4. Example of the TPT Specification for Automated Mapping of TPTNUM

OVERVIEW OF THE VBA MACRO

```
Sub Create_TPT()
    Dim rb, aimb as Workbook, dict As Object
    Set rb = ThisWorkbook
    temp = 1
    'Get the CRF2SDTM spec
    InPath_TPT = Trim(rb.sheets("TDM_AUTO").Cells(7, 5))
    If Dir(InPath_TPT) <> "" Then
       TPT_LastRow = rb.sheets("Timepoint").Range("a65536").End(xlUp).Row
       rb.sheets("TPT").Rows("2:" & TPT_LastRow).Clear
       Set aimb = CreateObject(InPath_TPT)
       lastrow = aimb.Worksheets("TPTList").Range("c65536").End(xlUp).Row
       For m = 2 To lastrow
         temp_tpt= temp + m - 1
         With rb.Sheets("Timepoint")
             .Cells(temp_tpt, 1) = rb.Sheets("TDM_AUTO").Range("B3")
             .Cells(temp_tpt, 2) = aimb.sheets("TPTList").Cells(m, 7)
             .Cells(temp_tpt, 3) = aimb.Sheets("TPTList").Cells(m, 9)
             .Cells(temp_tpt, 4) = aimb.Sheets("TPTList").Cells(m, 11)
             .Cells(temp_tpt, 5) = aimb.Sheets("TPTList").Cells(m, 12)
         End with
       Next
       rb.sheets("Timepoint").Range("A2").Resize(lastrow,
5).RemoveDuplicates Columns:=Array(2, 3, 4), Header:=xlYes
       Set dict = CreateObject("Scripting.Dictionary")
       seqNum = 1
       lastrow = rb.sheets("Timepoint").Range("a65536").End(xlUp).Row
       For m = temp + 1 To lastrow
           tptvar = rb.sheets("Timepoint").Cells(m, 3)
           If Not dict.exists(tptvar) Then
              dict.Add tptvar, seqNum
           Else
              rb.sheets("Timepoint").Cells(m, 5) = dict(tptvar) + 1
               dict(tptvar) = rb.sheets("Timepoint").Cells(m, 5)
            End If
         Next
    End If
    aimb.Close SaveChanges:=False
    Sheets("Timepoint").Select
    Range("A2", "J1000").Sort Key1:=Range("E2"), Key2:=Range("C2"),
Header:=xlYes, Order1:=xlAscending, Order2:=xlAscending
    MsgBox "Nice! TPT Spec complete!"
End
```

LIMITATION & FUTURE EXPECTATION

- **Metadata Dependence:** The automation relies on the accuracy and completeness of source metadata (Protocol, ALS MASTERDASHBOARD, CRF2SDTM spec). Gaps may still require manual review.
- **AI Integration:** Future enhancements could leverage AI-enabled metadata extraction and processing to automatically extract ARM, visit, and time point details directly from protocols and ALS files, reducing manual metadata setup.
- **Broader Coverage:** Extending the framework to handle more SDTM variables and complex or adaptive study designs will further enhance its value.

CONCLUSION

This paper demonstrates how a metadata-driven framework, implemented using VBA and SAS, transforms manual, study-specific SDTM derivations into an efficient, scalable, and traceable process. The TA_ARM Matrix, combined with automated generation of TV and TPT specifications, reduces programming burden, improves consistency, and ensures alignment with protocol designs.

In summary, automating TA.ARM mapping, TV, and TPT specifications significantly improves data quality, consistency, and scalability—delivering a modern, robust solution for SDTM preparation in today's evolving clinical trial landscape.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Guangan Cao

BeOne Medicines

guangan.cao@beigene.com

Liming Xie

BeOne Medicines

liming.xie@beigene.com

Qiuan Hu

BeOne Medicines

qiuan.hu@beigene.com

Chunpeng Zhao

BeOne Medicines

chunpeng.zhao@beigene.com