PharmaSUG 2023 - Paper MM-118

Masters of the Table Universe: Creating Table Shells Consistently and Efficiently Across All Your Studies

Michael Hagendoorn, Ran Li, Mimi Vigil, and Shan Yang, Seagen Inc.

ABSTRACT

Creating specifications for tables, listings, and figures (TLFs) is traditionally not for the faint of heart. Over the course of weeks or more, our brave author valiantly translates the statistical analysis plan into sometimes hundreds of pages of detailed Microsoft Word-formatted specifications for their study. Meanwhile, two offices over, another author is going through the same painful process of creating a separate Word shell for their own study… and so on.

While capturing TLF shells for each study in this manner is the norm, we developed an alternate approach by establishing a single set of shells at the compound level that provides the metadata for all CSR and integration TLFs across the entire product. Such a master shell setup yields many advantages:

- Faster shell development for new studies and less maintenance for existing ones
- Instant visibility into where studies differ on any detail
- Higher consistency across studies, which elevates quality in TLF specs and programming
- Increased programming efficiency through expanded macro coverage and less custom code
- Enhanced departmental standards adoption to benefit compliance and review
- Annotation to ADaM unlocks pathways for future submission documentation and metadata repository
- Potential for powerful metadata-driven output generation and other innovations

We will share the design, management, and governance model of our master shell implementation. We'll also discuss programming and biostatistics perspectives on the benefits and challenges we observed, along with our solutions – so you can immediately leverage this setup and unleash the power of compound-level specifications!

1. INTRODUCTION

Tables, listings, and figures (TLFs) are a key component in the presentation of clinical trial outcomes to the medical and regulatory community. These are created by statistical programmers based on table shells, which are specifications developed by study statisticians with input from lead programmers as an extension of the statistical analysis plan (SAP).

As shown in the example in Figure 1, a typical shell for an individual TLF presents a mock-up that displays the desired layout and content including titles and footnotes, headers, and content of the ultimate deliverable to be generated. These displays can be expanded with various enhancements, such as annotations showing from which analysis data set (ADaM) and variables each TLF section should be derived; instructions below the footnotes on data processing for specific sections; and guidance on repeating the same table layout for different subgroups in the actual output.

<Table x.x> Summary of Prior Therapies
Full Analysis Set

	Cohort A (N=xx)	Cohort B (N=xx)	Total (N=xx)
Number of subjects with medication used in the adjuvant setting, n (%)	xx (xx.x)	xx(xx.x)	xx(xx.x)
Anti-EGFR antibody [a]	xx (xx.x)	xx (xx.x)	xx (xx.x)
Anti-drug1 agent	xx (xx.x)	xx (xx.x)	xx (xx.x)
Hormone blocker	xx (xx.x)	xx (xx.x)	xx (xx.x)
Other	xx (xx.x)	xx (xx.x)	xx (xx.x)
Number of cycles of OldDrug received in the induction phase [b]			
n	xx	xx	xx
Mean (STD)	xx (xx.x)	xx (xx.x)	xx (xx.x)
Median	xx	xx	xx
Min, Max	xx, xx	xx, xx	xx, xx
Subjects who received <6 cycles of OldDrug due to toxicity	xx	xx	xx

Page 1 of 1

a. The first footnote
b. Another footnote

<Data Snapshot: ddMMMyyyy; Data Cutoff Date: ddMMMyyyy>
<t-myprogram.sas Output: t-myoutput.rtf (ddMONYY: hh:mm) Data:adsl, adbase, adcm>

Programming notes:
1. Only show medications with non-zero counts, and use N as denominator when calculating the percentages
2. To count the number of cycles: [instructions here]

Programming annotation:
Repeat adjuvant setting block for neoadjuvant setting; repeat section for cycles of OldDrug in induction phase for EvenOlderDrug in induction phase
Subjects who received <6 cycles of OldDrug due to toxicity: adbase.ptx6cyc
Number of subjects with medication used in the adjuvant setting: subjects where adctx.cmcat = prior therapy and adctx.cmsystx = adjuvant

Repeat for:

<*Table x.x*> Summary of Very Powerful Therapies Before MyDrug
<*Table x.x*> Summary of Very Powerful Therapies Before YourDrug

Figure 1. Traditional Table Shell Example: Prior Therapy Table

All shells for a single trial or analysis, such as a primary clinical study report (CSR) or safety monitoring committee (SMC), are often bundled together into a large MS Word document, along with a table of contents and a few general instructions for things like the presentation of decimal places or inclusion of unscheduled visits in by-visit tables. The statistician usually maintains this massive document from there on out. It will further be reviewed by other functions such as medical science or drug safety, and ultimately serves as the specification for the actual TLFs produced for this study or analysis.

For just one or two studies this traditional shell setup is perfectly feasible. But if a company runs many concurrent studies, managing shells in this manner quickly poses several challenges:

- Shells that are common across studies or analyses are developed and maintained in multiple individual documents, placing a heavy and redundant burden on statisticians and programmers
- Storing individual shell documents in many study-specific locations makes it difficult to implement consistency in similar shells from one study or analysis to the next, which can harm efficiency and quality of TLF production and slow down cross-functional and regulatory review
- Decreased consistency in turn limits opportunities to reduce custom study-level programming in favor of much more efficient higher-quality departmental macro coverage
- Word shells scattered across many locations provide no visibility into the degree to which studies align to departmental TLF standards, if any

- Word is not machine-readable, thus limiting potential pathways for direct shell-to-output automation as well as submission documentation in the form of analysis results metadata

To eliminate these hurdles, we pioneered an approach similar to our master data set specifications, where a single set of metadata is maintained at the product (i.e., drug or compound) level and all studies on the product subscribe to metadata in there, rather than in isolation within each individual study folder. Such a master shell setup not only remediates all challenges identified above, but also tees us up to realize our longer-term vision towards the elusive concept of a metadata repository (see Section 5). In addition, teams noted this innovative setup makes it easier to eliminate unnecessary variations due to different preferences by different study teams and can facilitate earlier cross-functional study team discussion aligning data collection with TLF analysis needs. Master shells deliver the following benefits:

- Faster shell development for new studies or analyses and less maintenance for existing ones
- Instant visibility into where studies or analyses differ on any detail
- Higher consistency across projects, which elevates quality in TLF specs and programming
- Increased programming efficiency through expanded macro coverage and less custom code
- Enhanced departmental standards adoption which benefits compliance and review
- Annotation capabilities unlock pathways for future filing documentation and metadata repository
- Opens up the potential for powerful metadata-driven output generation and other innovations

While commercial solutions are available to help realize this vision, we opted for a simpler approach that uses existing tools like Microsoft Excel backed by its built-in Visual Basic for Applications (VBA) and SAS®. This not only ensures our solution works exactly the way we'd like, but given the expertise among statistical programmers with these types of applications we can also quickly change or add desired functionality along the way. A setup similar to ours is easily implemented in any company, so let's take a closer look at how we got it done!

2. WHAT DO MASTER SHELLS LOOK LIKE?

To implement master shells, we decided to change the format from Word to Excel, because the latter is accessible to SAS and not limited in available space on a page from left to right - even more relevant for a master design, since information for each study will be captured in its own column to the right of each shell rather than in rows beneath it.

Each product has one master shell Excel workbook that contains the specifications across all studies for CSRs, SMCs, data monitoring committees (DMCs), regulatory integrations (ISS and ISE), and large interim analyses or publications that are reported across the product. This workbook contains sheets for each of the following, which collectively mimic the sections in a traditional Word shell document:

- One sheet for the table of contents (ToC)
- One sheet for general instructions and column headers for each study or analysis
- One sheet per individual TLF shell, such as the demographics table, AE listing, or efficacy waterfall plot

Figure 2 shows a partial example of this setup, corresponding to the table of contents shown in Figure 3.

Figure 2. Example Tabs (Worksheets) in a Master Shell Workbook

2.1 TABLE OF CONTENTS SHEET

The ToC lists all shells in the workbook, divided into sections (e.g., disposition, demographics, exposure) and sorted by the order in which TLFs would appear in a CSR appendix. Each row hyperlinks directly to the shell in question. This sheet essentially mimics a traditional Word shell document ToC. Figure 3 shows a partial screenshot of an example master shell ToC worksheet.

	A	B	C	D	E	F	G	H
1		Seq ID	Shell ID	Title (Double-click here for shell management console)	s_study1 CSR	s_study2 DMC	s_study2 CSR	s_study3 Interim
2								
3	**Section 0. General**							
4			Headers	Column header structures for each study	x	x	x	x
5								
6	**Section 1. Disposition and Protocol Deviations**							
7		1.01	t-ds	Summary of Disposition	x	x	x	x
8		1.02	t-ds-sf	Summary of Screening	x		x	x
9		1.03	t-ds-cty-site	Randomization by Country and Site	x		x	x
10		1.04	t-ds-disc	Summary of Reasons for Discontinuation from Study Treatments	x	x	x	x
11		1.05	t-ds-enr	Summary of Enrollment and Analysis Sets	x		x	x
12		1.06	t-ds-ipd	Summary of Important Protocol Deviations	x		x	x
13		1.07	t-ds-strat	Stratification Factors at Randomization				x
14								
15	**Section 2. Demographics and Baseline Disease Characteristics**							
16		2.01	t-dm	Summary of Demographics and Baseline Subject Characteristics	x	x	x	x
17		2.02	t-bl-dc	Summary of Baseline Disease Characteristics	x	x	x	x
18		2.03	t-prior-sys	Summary of Prior Therapies	x		x	

Figure 3. Example ToC Sheet

The following columns are presented:

- **Seq ID** is a unique sequential number for ease of reference in meetings or informal communications. As other shells are added or removed, this number will automatically update through VBA automation when the file is saved. For instance, if a new shell were added between rows 11 and 12, then 1.06 and 1.07 would automatically renumber to 1.07 and 1.08, and the new shell would receive 1.06. For this reason, seq ID should never be used as a reference in permanent documentation such as issue trackers or meeting minutes.
- **Shell ID** uniquely identifies each shell in a persistent manner: it will not change when other shells are added or removed. It corresponds to the name of the worksheet containing this shell, and while shell management staff manually provides this name, it is hyperlinked automatically by VBA to the corresponding worksheet when the file is saved so users can quickly navigate to the specific shell. In contrast to Seq ID, it is permanently associated with the shell and can be referenced in more formal documents such as minutes and trackers.
- The **title** corresponds to the first title line in the shell itself (minus the table number if included in the first title; see Figure 5 for an example). In this master shell setting, numbers can differ from one study to the next and thus are not all that informative here.
- **Study and analysis columns** (E – H) identify which studies and analyses use which shell. They are prefixed by "s_" for ease of identification in VBA or SAS processing. In our example, study 1 uses the

master shells for a CSR analysis; study 2 for their DMCs and for a CSR analysis; and study 3 for an interim analysis. If a shell is used, it will show "x" in the corresponding analysis column; otherwise this column remains blank. Using study 2 as an example, shell 1.02 (screening summary) is used in its CSR analysis but not in DMCs.

Excel autofilters (small gray squares with a down-arrow at the bottom-right corner of a cell) are set on the first row of this and most other sheets in the workbook. Viewers can use these to quickly filter content to only their study or analysis. Here and in all other sheets, content in study columns (s_*) is managed by individual study teams while the columns to the left are centrally managed at the product level by a Shell Management Team, as explained in further detail in Section 4.

2.2 HEADERS SHEET

The second worksheet in our file provides general instructions that apply across all or multiple shells in a given study or analysis, along with a visual representation of table header structures to be used in the study. As this is free-form Excel, the statistician can easily format study headers in any way desired. Figure 4 shows a screenshot of headers for our sample studies shown earlier in the ToC discussion.

Each section begins with an s_* label that corresponds to the study or analysis in the ToC. Mismatches will be auto-flagged by VBA. If only the study ID is provided (e.g., *s_study1*) instead of the more specific analysis from the ToC (e.g., *s_study2 CSR*) then the information that follows will apply to all analyses on that study and be included in shell exports for any of its analysis listed in the ToC.

Tables may use different headers within a given analysis. For example, safety tables may show different columns than those for efficacy. We use simple header IDs within each study section to uniquely identify each configuration along with any programming instructions. In Figure 4, an example is under study 2 where configuration DMC-A identifies the columns to show in tables for its DMC closed session, while DMC-B shows columns for open-session DMCs tables, which will not present arm-specific summaries.

Note that simple Excel grouping and outlining is used to dynamically expand and collapse study-specific sections on this sheet. Clicking the "2" at top left will show all rows on the sheet, while clicking the "1" will hide all rows except the orange study labels. Selecting the "-" along the left (e.g., for study 2 DMC) will hide the content for that section, while clicking the "+" as shown for study 1 and 3 will expand that section.

VBA plays no role on this sheet except for cross-checking to the ToC and auto-hiding all grouped rows on the sheet when the file is opened or saved, so it shows up cleanly for the next person to view it.

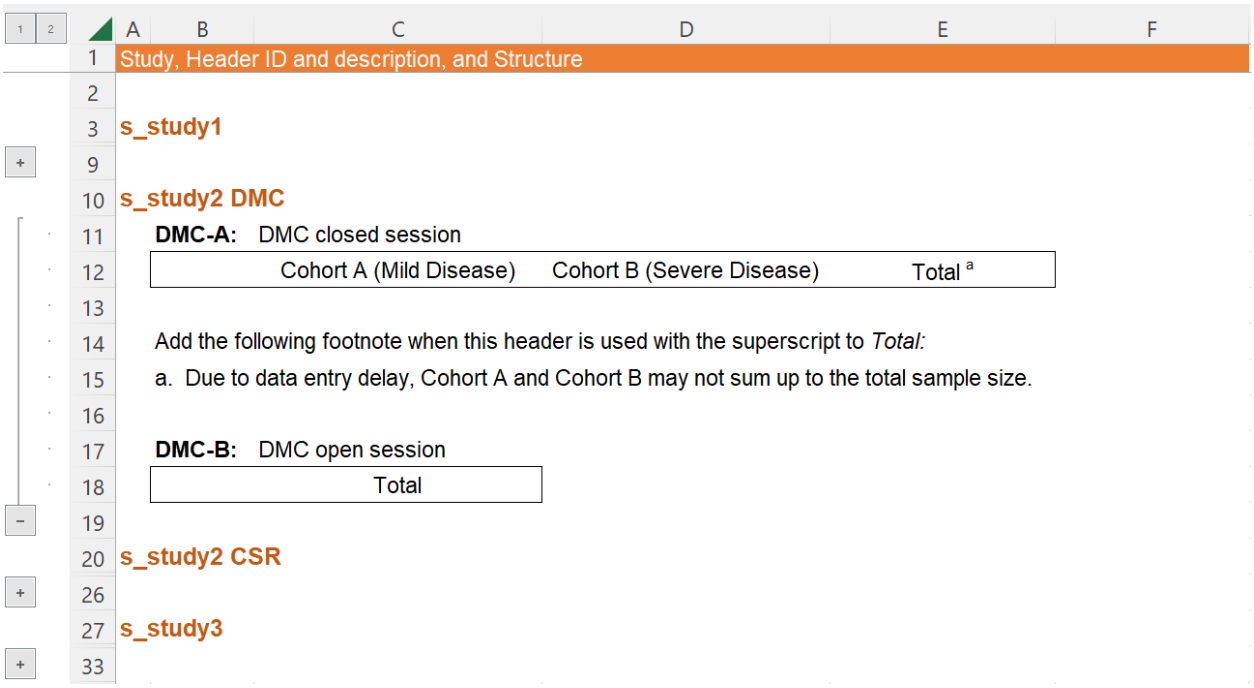

Figure 4. Example Headers Sheet

2.3 INDIVIDUAL SHELL SHEETS

The meat and bones of the master file are in the TLF shell sheets, where each provides the layout and analytical instructions for the structure to be used for specific TLFs. While it is technically possible to make a shell so generic that its structure can be used for hundreds of tables in an analysis (think: a simple empty square box), that kind of design would rely heavily on countless rows of programming instructions below the shell to clarify what exactly should be displayed for the different TLFs in an analysis. Making shells too general also forces programmers and reviewers to wildly use their imagination on how the actual TLFs might look, which is not only mentally exhausting but also busts the door wide open to incorrect interpretations and assumptions.

A better approach is to design each shell structure to be close to how the actual TLF will ultimately appear, which inevitably means developing a sizable number of shells. Some will provide specifications for just one TLF while others provide guidance for several variations and repeats that all use a similar structure. Exactly how granular to get is a judgment call by the shell authors. An average study CSR may reasonably reference 50 to 100 different shells to collectively specify all its TLFs.

In a traditional study-level Word shell document, it's relatively easy to get each shell to appear very close to the actual output to be produced on that study. However, master shells are designed to apply across all studies on a product, which means some parts in each shell may differ a little or a lot from one study to the next. Part of the power of master shells is to clearly expose such differences and provide a platform to harmonize them wherever feasible, and we'll look at how that's done in the sections that follow. We'll use the example TLF shell sheet in Figure 5 for that:

	A	B	C	D	E	F	G
1		Double-click here for shell filtering and export	Annotation	[Back to TOC]	s_study1	s_study2	s_study3
2	T	**Table 14.x.y Summary of Prior Therapies**					
3	T	**~Full~ Analysis Set**					
4	H			~Headers~			
5	H			(N=x)			
6	B	Number of subjects with medication used in the ~adjuvant~ setting, n (%)		x (x.d)	[r] neoadjuvant [r] adjuvant	[r] unresectable [r] metastatic	
7	B	Anti-EGFR antibody [a]		x (x.d)		x	
8	B	… (based on CRF)		x (x.d)	[r] Anti-drug1 agent [r] Hormone blocker	[r] Anti-drug2 agent [r] Enzyme inhibitor	
9	B	Other		x (x.d)	x	x	
10	B				x	x	
11	B	Number of cycles of ~drug a~ received in the induction phase [b]			[r] OldDrug [r] EvenOlderDrug		
12	B	n		x	x		
13	B	Mean (STD)		x.d (x.d)	x		
14	B	Median		x.d	x		
15	B	Min, Max		x, x	x		
16	B				x		
17	B	Subjects who received ~<6 cycles of OldDrug due to toxicity~	adbase.ptx6cyc	x (x.d)	x		
18	B				x	x	
19	B				x	x	
20	F	a. The first footnote					
21	F	b. Another footnote			x	x	
22	F				x	x	
23		Programming Notes					
24	1	Only show medications with non-zero counts			x	x	
25	2	To count the number of cycles: [instructions here]			x	x	
26	3	Use N as the denominator when calculating percentages			x	x	
27	4	For number of subjects with med use in a given setting, adctx.cmcat is "prior therapy" and adctx.cmsystx = "adjuvant" (or other value based on the study)			x	x	
28					x	x	
29		Repeat for					
30		Summary of Powerful Therapies				x	
31		Summary of Very Powerful Therapies before ~MyDrug~			[r] MyDrug [r] YourDrug		

Figure 5. Example of a Single Table Shell Sheet

The traditional Word shell shown in Figure 1 is reproduced in columns B and D on rows 2-21 of this example. Rows 23-27 present further instructions to the programmers and reviewers, while rows 29-31 show additional tables for which this same structure can be used. To the right, columns E through G show how each shell section is (or is not) to be used for table generation on each study. Note that study 3 does not use this shell per the ToC in Figure 3, so no items are marked in its column.

2.3.1 Center of the Universe: The Shells Per Se

2.3.1.1 Titles, headers, body, and footnotes

Looking top to bottom in columns B and D, we first see the table title and subtitle in rows 2 and 3. The table number is added only for a more table-like look and feel as it will be included in the actual output; it's not fully numbered since each study will number their output differently. We also see the orange text ~Full~. Such text enclosed in tildes is flexible, meaning it can be replaced with different content in exported shells and actual TLFs. Study columns and programming notes are used to describe such replacements, and VBA or SAS will handle such replacements in study exports (see Section 3.3).

Next up, rows 4 and 5 show the table header section, where ~Headers~ can either be replaced with a column header structure from the *Headers* sheet identified in the study column, or simply exported as is where reviewers and programmers then rely on programming notes or the study SAP to determine which headers should be applied to this shell.

The body of the shell is presented in rows 6 through 18. As in earlier sections, tilde-enclosed fragments in column B can be replaced with other content in study exports and actual TLFs. Row 17 shows why we opted to use tildes instead of the more commonly seen < and > to enclose such fragments; had we used

the latter, the flex text would show as <<6 cycles …> rather than ~<6 cycles …~, and the signifier preceding "6 cycles" might easily be overlooked or mistaken for a typo.

Analysis results will be displayed in the format shown in column D, where "x" represents any whole number and "d" a decimal place. For example, if x (x.dd) is indicated here, then an n of 17 that represents 14.9271% of the big N should be displayed as "17 (14.93)" in the actual TLF. Because x represents any whole number before the decimal point, only one is needed, which keeps things more legible than "xx.d" or "xxx.d" as often seen in traditional shells.

Coming towards the end of the shell section, rows 20 and 21 show footnotes that should be displayed in the actual TLFs. Note that in this example, study 1 only uses footnote b but not footnote a, based on the checkmarks in the study columns. It's worth noting that using VBA and SAS, in the export file for this study we automatically replace [b] with [a] in the body of the table and the corresponding footnote.

2.3.1.2 Annotations

What about columns A and C? Notice that column A shows the letters T, H, B, and F in very light font. These are auto-populated by VBA keyed off basic TLF structure logic to identify the title, header, body, and footer rows in each shell; this provides a roadmap to SAS when it generates a study export from each Excel sheet so it can easily apply specific formatting to each section per company style guidelines.

Column C provides the option to annotate ADaM source variables for the endpoint being summarized. This is feasible for simple univariate summaries straight from standardized ADaM variables such as the example shown on row 17. On the other hand, this column does not provide enough room for more complex annotations, which are still placed below the table shell. Row 27 provides an example based on the traditional shell in Figure 1. As with other shell content, studies indicate which guidance applies to them by marking their study column or adding new rows if needed.

2.3.2 Studiously Studying Study Columns

Study columns (E, F, and G in Figure 6) are at the heart of master shell action. Here, study lead statisticians indicate which elements of a given shell are used on their study or analysis. In its simplest form, an "x" indicates that row applies to the study or analysis while a blank cell indicates it is not. Title and header rows are always marked with "x" by VBA in all study columns. This is done in white font to not distract users, who should focus more on selecting content in the body and footer sections. VBA will reset individual cells to black font if something other than "x" is provided in them.

Note that while ToC columns often specify both the study and analysis, similar to the logic in the Headers sheet (Section 2.2), shell columns have the option to specify only the study. Doing so means that if multiple analyses in one study use a particular shell, then the shell will look the same for each analysis. Alternatively, to show different content from one analysis to the next on the same study, add a separate column for each analysis on that trial. This means more maintenance but on the other hand allows more granularity if desired.

	A	B	C	D	E	F	G
1		Double-click here for shell filtering and export	Annotation	[Back to TOC]	s_study1	s_study2	s_study3
2	T	**Table 14.x.y Summary of Prior Therapies**					
3	T	**~Full~ Analysis Set**					
4	H			~Headers~			
5	H			(N=x)			
6	B	Number of subjects with medication used in the ~adjuvant~ setting, n (%)		x (x.d)	[r] neoadjuvant [r] adjuvant	[r] unresectable [r] metastatic	
7	B	Anti-EGFR antibody [a]		x (x.d)		x	
8	B	… (based on CRF)		x (x.d)	[r] Anti-drug1 agent [r] Hormone blocker	[r] Anti-drug2 agent [r] Enzyme inhibitor	
9	B	Other		x (x.d)	x	x	

Figure 6. Zooming in on Study Columns

In some cases, sections in the body of a shell may need to be repeated for different endpoints, and these repeating endpoints may further differ from one study to the next. Figure 6 presents a made-up example on row 6.

If a study column would simply check "x" for row 6, this section would describe prevalence of medication use in the adjuvant setting. However, study 1 indicates via the *[r]* tag in column E to replace the flexible ~adjuvant~ fragment with the text "neoadjuvant," and then a second time with "adjuvant." Because the rows that follow are indented, it is clear that in the actual output produced by programmers, rows 6-9 should first be displayed for the neoadjuvant setting, followed by a blank row, then repeated for meds in the adjuvant setting.

A similar process occurs for study 2 per column F, which instructs to show rows 6-9 for meds in the unresectable setting, insert a blank row, then repeat this block for meds in the metastatic setting.

Row 8 presents a variation on this theme. Here, no fragments are indicated as ~flexible~, yet the *[r]* tag is used in the columns for study 1 and 2. In such cases, in the programmed output the entire text in column B will be replaced with the text following each tag. Since this is a single-line item rather than the header for an entire block like row 6, multiple *[r]* tags will only repeat the row itself. For instance, for study 1, "… (based on CRF)" will be replaced by 2 rows in the actual table: one row for Anti-drug1 agent, and another row for Hormone blocker.

Putting it all together, here's what the export function (Section 3.3) "really sees" when creating a single-study export for study 1 as an example. Anti-EGFR antibody isn't marked for study 1 so it won't show up:

6	B	Number of subjects with medication used in the neoadjuvant setting, n (%)		x (x.d)
7	B	Anti-drug1 agent		x (x.d)
8	B	Hormone blocker		x (x.d)
9	B	Other		x (x.d)
10	B			
11	B	Number of subjects with medication used in the adjuvant setting, n (%)		x (x.d)
12	B	Anti-drug1 agent		x (x.d)
13	B	Hormone blocker		x (x.d)
14	B	Other		x (x.d)
15	B			

Figure 7. Expanded Shell Based on Instructions in Column E

This simple but immensely flexible interplay between study columns and table content allows new studies or analyses to quickly assemble tailored shells using pre-existing master shell content.

2.3.3 Taking Note of Programming Notes

As highlighted in Figure 8, programming notes appear after the last footnote in the shell. While simple replacements of fragments or repeats of table body sections can be handled in the study column as shown in Section 2.3.2, programming notes can be added to clarify derivations or additional explanations on how shell content should be applied on a given study or analysis. Study leads simply check the box in their column if a study note applies or leave it blank if it doesn't. In an export, all applicable notes thus marked will be included underneath the footnotes of each exported shell.

23		Programming Notes			
24	1	Only show medications with non-zero counts	x	x	
25	2	To count the number of cycles: [instructions here]	x	x	
26	3	Use N as the denominator when calculating percentages	x	x	
27	4	For number of subjects with med use in a given setting, adctx.cmcat is "prior therapy" and adctx.cmsystx = "adjuvant" (or other value based on the study)	x	x	

Figure 8. Zooming in on Programming Notes

2.3.4 Wash, Rinse, Repeat: the Repeats Section

If a given shell is to be used for multiple TLFs of the same type, leads can add the titles for such repeats underneath the last programming note as highlighted in Figure 9. Based on row 31, our shell for the summary of prior therapies is repeated twice on study 1: once for a summary of "very powerful therapies before MyDrug," and again for a summary of "very powerful therapies before YourDrug." Notice how the *[r]* tag is used even here to quickly indicate two repeats are needed and how their titles differ. For study 2, row 30 simply indicates to repeat this shell for a summary of "powerful therapies."

Figure 9. Zooming in on TLF Repeats

3. THE WIZARD BEHIND THE CURTAIN: THE SHELL CONSOLE AND VBA

3.1 VISUAL BASIC FOR APPLICATIONS (VBA)

The master shell setup described in Section 2 will not go far without automation. For example, making shell content for a single study available to reviewers would mean manually using the Excel autofilter in each individual worksheet on the appropriate study column or even hiding entire shells not used on the study. In addition, users are notoriously creative in changing Excel content structures, content, and autofilters in unexpected and inconsistent twists over time, so we'll need a way to keep the shells clean, consistent, and well-functioning with all expected elements in place.

While several different technologies could be used to implement such functional enhancements, we opted for VBA. This comes as part of Excel and is not too challenging to learn, especially when tapping into its code-generating recording functionality. It truly is a little gem that unlocks a brave new world of functionality any programmer can add to their arsenal with a little bit of elbow grease!

Using this technology, we were able to build functionality directly into the master shells without users needing to open external applications, taking care of pretty much everything we need to make master shells as user-friendly and feasible as possible. It can handle automated behind-the-scenes housekeeping like consistent sheet formatting and ToC-to-shell hyperlinking, as well as user-facing functionality like filtering content on demand to a specific study or analysis; adding, renaming, and removing studies; and exporting analysis-specific content to a PDF or Word file for team review.

While the technical details of our VBA setup are far beyond the scope of this paper, we did want to mention how we overcame a major obstacle associated with VBA code: it is always contained directly in an Excel file. Say we have 10 different shell docs for 10 different products. The same VBA code would then essentially be repeated 10 times: once in each file. This problem multiplies further if some products split their shell collection into smaller, section-specific docs, quickly turning any VBA approach into a maintenance nightmare. We needed to find a solution similar to SAS macros, where any master doc could run code from a single centralized VBA code library. We solved this by maintaining all central VBA code inside a blank departmental Excel file, which is automatically opened and immediately hidden from view whenever users open any master shell document. With both files open at the same time, the VBA code from the departmental file is automatically available to the master shell doc. Works like a charm!

3.2 THE SHELL CONSOLE

We used VBA to create a simple interactive shell management console that quickly and seamlessly facilitates many user-facing operations across the entire workbook. To access this console from any sheet, users double-click the cell on row 1 labelled "Double-click here for shell filtering and export" (see Figure 3 and Figure 5 for examples) and VBA will display the interactive console seen in Figure 10.

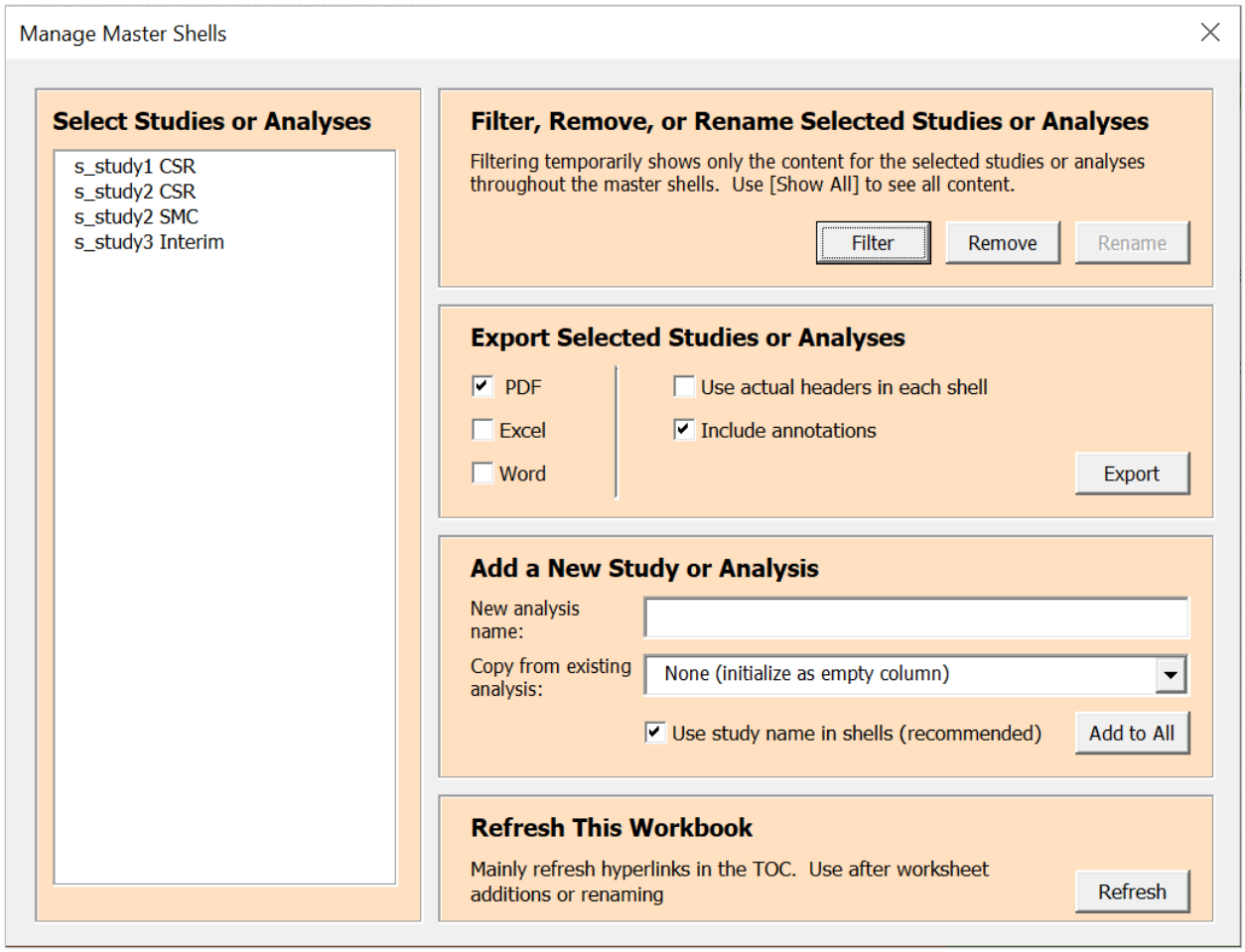

Figure 10. Shell Management Console

3.2.1 Selection Listbox

When the console opens, VBA dynamically populates this listbox by reading out row 1 in the ToC. Users select one or more studies or analyses, and applicable actions on the right will operate on that selection.

3.2.2 Filter Single or Multiple Studies or Analyses

Filtering provides a temporary focus on a specific study or analysis for review. Using VBA on the back end, this action will display only selected shells in the ToC, applicable header content, shells that are indicated in the ToC, and all applicable rows and notes within each such shell. All other content will be hidden. Opening the console again after filtering will change the “Filter” button to a “Show all” button which allows users to un-filter the entire workbook. VBA will also reset all filters when the file is opened.

3.2.3 Remove Single Study or Analysis

This will delete information for the selected study or analysis from the ToC, headers sheet, and all shells. This button is unavailable if multiple studies or analyses are selected.

3.2.4 Rename Single Study or Analysis

A pop-up window will ask users to type the new name for the selected study or analysis, then VBA will update column names accordingly in the ToC and each shell. This button is unavailable if multiple studies or analyses are selected.

3.2.5 Export Single Study or Analysis

The ability to export master shell content for a single study or analysis to its own document is key to successful operation of any master shell design. Cross-functional reviewers trying to understand what will be presented for their specific study are in no mood to slog through tons of information relating to other studies. Additionally, the Excel format doesn't lend itself very well for such reviews in general, and printing the entire workbook to PDF will squeeze font sizes to microscopic levels. Lastly, there may be cases where shells are best maintained outside the master, as explained in Section 4.3.

The export option resolves these challenges by essentially re-assembling a traditional single-study shell document based on info in the master shells. It will filter content throughout the ToC, headers, and shells to only the selected study or analysis, then save a copy of that filtered content to a single new hyperlinked and bookmarked PDF and/or Word file. Note that Excel is currently also an option, but we're considering removing that as it's redundant with filtering done directly in the master.

An optional checkbox allows to export the annotations column in each table shell. If left unchecked, that column is suppressed in the export for a more "true-to-output" look-and-feel. A final checkbox to replace the generic ~Headers~ section in each shell with the actual structures identified in the Headers sheet is an enhancement planned for the near future.

Section 3.3 provides more details and examples of this critical component of master shell design.

3.2.6 Add a New Study or Analysis

Entering a new study (or study + analysis) name will add this as a new study column to the ToC and all shell sheets. By default, this column appears after the last existing study column and no content will have been marked in it.

If a new study or analysis is similar to another study or analysis, you can select that predecessor in the *Copy from existing analysis* combobox. This will place the new study or analysis column to the immediate right of that predecessor column and initialize its values with the same selections as the source column.

3.2.7 Refresh This Workbook

This will invoke VBA to auto-refresh hyperlinks in the ToC after new shell sheets are renamed or added and perform several other chores on formatting and shell section identification in each shell.

3.3 THE INS AND OUTS OF EXPORTING A SINGLE-STUDY TRADITIONAL SHELL DOC

3.3.1 High-level Technical Export Implementation

To rebuild a traditional-looking shell document for a single study or analysis from the master shell, we looked towards a combination of VBA and Windows SAS Server. Once someone selects "Export," a few steps happen in succession:

- VBA writes a small SAS program to our server that contains a call to a department-level export macro. This calling program identifies the location of the master shell document to be processed as

well as the study or analysis being exported, along with parameters for PDF and Word format yes/no plus annotation yes/no and header replacements yes/no.

- VBA then batch-submits this small program to SAS on the server, which kicks off the departmental export macro to perform the following:
    - Assign the entire master shell workbook to a libref using the XLSX library engine, which exposes each worksheet as a data set to SAS
    - Process each data set (i.e., sheet) to filter on the indicated study or analysis and replace or repeat shell content per instructions in the study column (see Section 2.3.2); also replace the ~Header~ section with actual column headers if so indicated via the console
    - Print each processed shell data set to a single, concatenated Word file using the ODS Word destination that leverages the `contents="on"` and `toc_data="on"` options to add a table of contents in the beginning of the document. Here we also rely on the section identifiers (T, H, B, and F) in column A of each sheet to identify different fonts and styles to apply to each shell component (see Section 2.3.1)

3.3.2 Export Considerations

A few items we did not cover above:

- Do you want to overwrite shell exports each time, or permanently retain all exported copies? The latter can be done by simply adding the export date and time to the stapled file name, but this will generate tons of files over time, while the former keeps things clean but requires manual intervention to retain comments if any were made in previously exported copies. We implemented the former.
- Exporting figure shells: we keep figures as a separate export file printed to PDF directly from the master using VBA, as SAS cannot read image files from Excel via the XLSX engine. A more sophisticated way to integrate figure exports directly into the combined file, and frankly to create the entire combined Word file to begin with, is to leverage Word VBA in a departmental template, where Excel VBA copies the template to the export location, copies over all filtered study data from each shell sheet (including figures) to the Word file, and adds a hyperlinked ToC when all is said and done. At the time this paper was written, we were investigating that route to improve our export design.

4. USAGE AND GOVERNANCE

4.1 WELL HELLO, SHELL MANAGEMENT TEAM

As we operate master shells on products with up to dozens of programmers and statisticians, we advise teams nominate one or two programmers and statisticians on each product to form an informal “shell management team” (SMT) that maintains the product-level master. Their responsibilities include:

- Add new shells upon request and decide on shell numbering and naming
- Revise content in shell titles, body, footnotes, and general shell notes
- Seek input from study and product leads as well as TLF standards SMEs as needed
- Review study-specific pivotal analysis shells for general consistency with departmental standards and other studies on the product before database lock, upon study team request

4.2 AND HELLO TO YOU TOO, STUDY TEAM

Study statisticians, with input from lead programmers, maintain content in their study- and analysis-specific areas throughout the master:

- Use or cautiously replace existing SMT-owned content to fit their study design or analysis needs
- Communicate with the cross-functional study team on the need to add any study-specific shells. If the team requests a deviation from master shell content, critically assess whether that's truly science-driven or more of a personal preference. Updates based mainly on personal preference should be avoided; consult the SMT when in doubt
- Export master shells to study or analysis PDF and Word shells for team review and for reference in programming and QC
- Bring updates (such as those made via comments in any exported shell docs) back into the master shells whenever feasible to benefit future analyses on the study. If changes to SMT-managed central content are required, petition the SMT to make those changes; in our case, this communication flows through a product-level Excel issue log so all requests are centrally tracked and addressed. Each company can of course handle this communication in a manner that best fits their operations.

4.3 TO GO LOCAL OR NOT TO GO LOCAL…

Studies using master shells generally no longer maintain any separately managed study or analysis-level shell documents in Word or Excel format. There may still be occasions where teams decide to switch back to a single stand-alone study or analysis-level shell document, such as but not limited to:

- Once an analysis is close to delivery and ready to be disconnected from the master to avoid unwanted interference from product-level maintenance efforts at delivery time
- During quick-turnaround efforts such as regulatory requests, which do require shells but cannot afford to wait for product-level maintenance if they differ from the master
- Many of the shells in an analysis simply do not fit into the product-level master shell structure (e.g., for legacy reasons or to align with partner style guidelines)

While teams will try to operate within the master to reap the benefits described in Section 1, if they do by exception decide to work in a study-specific document separate from the master, teams can:

- Make a local copy of the master shell document in their study area, delete all other study and analysis columns, and directly adjust each shell as needed in this localized newborn "mini master"; or,
- Export a reasonably close study or analysis from the master to Word, move that file locally, and maintain this exported file as a traditional single-study Word shell document; or
- Start from scratch with a brand-new Word document altogether.

Whenever a team "disconnects" from the master in any of these ways, they assume responsibility for owning and maintaining them. Over time, such local versions may deviate more and more from the master; efficiency will suffer because departmental standards and macros developed to support those and the master cannot be used as easily anymore, while at the same time the risk of errors in specs, programming, and QC will increase. Thus, going local is advised only with an abundance of caution!

5. THE YELLOW BRICK ROAD TOWARDS THE FUTURE

In our department, master shell adoption is in an early stage. Once in place for all active analyses across the department, the door opens wide for a veritable buffet of delicious possibilities:

- The Excel format begs the exploration of adding metadata within each shell beyond mere variable annotation, which could ultimately allow autogeneration of actual TLFs directly from the shells.
- Our departmental TLF standards are migrating to a similar Excel format. Having standards and study shells in the same machine-readable format unlocks many ways to assess standards compliance and identify areas where either the standard or product shells should re-align for optimal quality.
- Our data set specifications are already in a master Excel format. Storing all data set and shell metadata across products in a departmental relational database will let us link it all together in an end-to-end manner. This would organically form the beginnings of a metadata repository (MDR). We would be able to see at a glance how SDTM structures flow into ADaM and from there into TLFs; how changes in metadata impact downstream structures; the degree to which teams are consistent from one to the next; and how each specification aligns to departmental and even Industry standards at any level – analysis, study, product, study type, disease area, and so on. New products moving into clinical trials could spawn a set of "starter" metadata directly from such a repository in full alignment with the latest standards, tailored to their disease area or design type. The list goes on and on!

6. CONCLUSION

Implementing master shells is a valuable step up from the classic single-study shell design. Our paper showed how this can be implemented with simple, readily available tools to yield tremendous benefits in heightened efficiency, consistency, and quality, while at the same time significantly reducing set-up and maintenance efforts that would otherwise be expended by authors of each single-study classic Word shell document.

7. ACKNOWLEDGMENTS

The authors would like to thank Jun Wei and Shawn Hopkins for their innovative contributions towards the implementation summarized in this paper.

8. CONTACT INFORMATION

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

Michael Hagendoorn
Seagen Inc.
mhagendoorn@seagen.com

Ran Li
Seagen Inc.
rli@seagen.com

Mimi Vigil
Seagen Inc.
mvigil@seagen.com

Shan Yang
Seagen Inc.
syang@seagen.com