Code Generation Using R Template Programs with Agent Skills

Masahito Katayama

Statistical Programming and Analytical Reporting, Clinical Data AI-Processing

Clinical Sciences & Operations, Research & Development, Sanofi K.K.

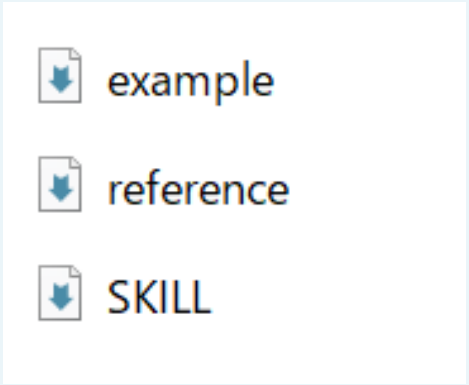
PharmaSUG Japan 2026 | Paper AI-027

Disclaimer

- The views and opinions expressed in this presentation are those of the presenter and do not represent the views, positions, or policies of Sanofi K.K. or any other organization.

What is an Agent Skill?

- A folder of Markdown files — not a program, not a prompt
- Instructions, procedures, reference material
- Loaded when the agent sees a task it matches
- **Persistence** — files, version-controlled, reviewable
- **Progressive disclosure** — small entry point always loaded
  - larger detail files opened only on demand




```
---
name: your-skill-name
description: Brief description of what this Skill does and when to use it
---

# Your Skill Name

## Instructions
[Clear, step-by-step guidance for Claude to follow]

## Examples
[Concrete examples of using this Skill]
```

Why TLF programming is a good fit

- One output needs the SAP, the shell, ADaM metadata, and an in-house template program — all at once
- TLF layout is **company-specific**
- A programmer joining a new company relearns the house style
- **AI Agents should too** — AI Agents knows R and CDISC, not our conventions



A Skill is the design specification for turning a general-purpose agent into *our company's* statistical programmer.

Our in-house R template programs

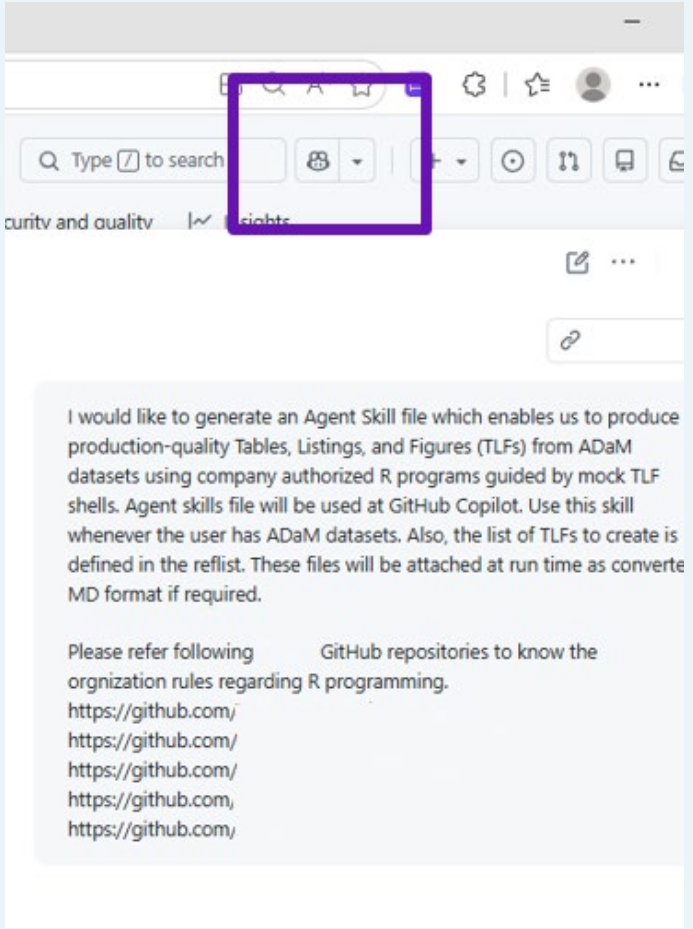
- Moving from SAS to R; template programs maintained on GitHub
- “Adapt this named template” is our control on the output; “write a program” is not



Templates cover the first two steps and return a table object. The catalog holds tables and listings only — a figure is built with {ggplot2} directly.

Designing the Skill

- Drafted the initial `SKILL.md` with **GitHub Copilot**, reading directly from the repositories where our template programs are maintained



Preparing the inputs

Input document	Source	As Markdown
ADaM metadata workbook	.xlsm 1.16 MB	12.04 MB
TLF shells	.docx 12.03 MB	1.08 MB
Statistical Analysis Plan	.pdf 0.88 MB	0.15 MB
Standard output templates	.pdf 2.74 MB	0.43 MB



Word ≈ 11× smaller, PDF ≈ 6× smaller — but Excel ≈ **10× larger**

Markdown conversion can cut token cost, but the effect **depends on the file format**

No Analysis Results Metadata — variables had to be **inferred**

Anatomy of SKILL.md

Section	Purpose
When to Activate	Inputs that should trigger the Skill
Your Role	Agent's role as an in-house statistical programmer, in four steps
Critical Rules	19 sections of strict rules, e.g. template selection, variable validation
Workflow	Ten steps, from loading study context to reporting what was validated
Pipeline Order	Fixed order for applying post-processing steps
Coding Rules	Nine project conventions — paths, diagnostics, a fix-history log
Key Inputs	What ADaM/SDTM metadata, the reference list, and mock shells contain
Maintaining These Files	Keep the entry point near its length budget

One file the agent **always** reads — the rest opens only on demand

Progressive disclosure

too short →
the agent invents details

SKILL.md
always loaded · 150–250 lines

too long →
weaker instruction adherence

on demand

reference.md
44 numbered code patterns
template catalog, authorized packages
Common Pitfalls Index

on demand

example.md
18 complete worked examples

first, if present

study-specific.md
rules true of one study only
one copy per study

The entry point's length budget

- Anthropic recommends keeping SKILL.md under **500 lines**
- We settled on **150 to 250**
- Too short → the agent invents details
- Too long → it follows the instructions less closely
- The rule lives **inside the Skill**, so it holds when the agent updates itself
- Specific, executable detail belongs in the **sub-files**

Image of Generated Code

```r
# Property of SANOFI
#
# Program Name    :              .R
# Description     : Demographic characteristics - All treated population
#                     (                    )
# Data            :          /adsl.sas7bdat
# Programmer      : Generated via tlf-generator skill
# Starting Date   : 18-AUG-2026
#

library(dplyr)
library(haven)
library(gtsummary)
library(lyra)
library(rtfmaker)
```

The development loop



each pass fixes ONE output

fixes the CLASS

the next output starts from an improved Skill

universal rule →
SKILL.md / reference.md

study-specific rule →
study-specific.md

Internal validation

- **Bottom-up** across model tiers is the recommended strategy
  - Claude Haiku, Sonnet and Opus
- We used **one tier only** — Claude Code Sonnet 5 — so no cross-tier evidence
  - Because TLF generation requires **complex reasoning**
- 3 studies, about 10 outputs each, 3 therapeutic areas
- 3 programmers reviewed **their own** output — not independent

Results

	Study A	Study B	Study C
Outputs targeted	10	8	7
Ultimately passed	**10**	**8**	**7**
Rounds to pass: 0	4	7	4
Rounds to pass: 1	1	1	1
Rounds to pass: 2	4	0	2
Rounds to pass: 3 or 4	0	0	0
Rounds to pass: 5	1	0	0

All **25** outputs **ultimately** passed

Most at **zero or one** round

The five-round output was a figure — appearance, not data

Zero rounds means the first generated program passed without revision.
Revisions were also done through Clade Code

What went wrong became the rules

Symptom	Rule now in the Skill
A pooled treatment column had the wrong denominator	Record the arm-to-column mapping in study-specific.md; never guess; check N against a known value
object not found, on a column just joined in	Rename incoming columns before the join; verify with names() afterwards
Rows silently vanished from a filtered population	Use %in% "Y", not == "Y", for a flag that can be NA

Common Pitfalls Index: **47 rows → 109**

Every row is a failure that actually happened — not a hypothetical one

The agent documents its own assumptions

```
# Template used for structure only - it requires an external reference
# dataset that does not exist for this study.

library(survival)    # not in the authorized package list. The only way to
                     # compute Kaplan-Meier. Flag for reviewer approval.

# OPEN ITEMS - confirm with study team before production use:
#   1. "Other" = records matching no named class variable. An engineering
#      inference from the shell, not a documented rule.
```

Nobody asked it to do this

“Review everything equally” is not realistic for hundreds of lines of R

Human-in-the-Loop stays mandatory — the Skill makes it **easier to target**

Lessons learned

- **Designing the Skill**
- Find the study-specific rules first, and move them out
- Keep the entry point small (150-250 lines); detail goes into the sub-files
- Standardization now has a second audience: **machines**

- **Operating the Skill**
- Human-in-the-Loop is not optional
- Design reproducibility in: LLM does not return identical output, adapt a **named template**
- Models change over time, and eventually disappear
- **Tokens are a real cost**
- The loop that adds rules can add **wrong** ones
- A Skill needs **governance**

Conclusion

- An Agent Skill is a **design specification**, not a program, not a prompt
- Keep the always-loaded entry point **small**
- Keep study-specific knowledge in a **dedicated file**
- Close the loop: fix the program **and** update the Skill
- Future improvements:
  - Held-out studies, exposure to more user
  - Using Agent Skill to SDTM and ADaM initiatives
  - Machine-readable ARM

Questions

Masahito Katayama

Sanofi K.K.

masahito.katayama@sanofi.com

Any brand and product names are trademarks of their respective companies.

Backup: References

- [1] Anthropic. “Agent Skills.” Claude Docs
- [2] Anthropic. “Skill authoring best practices.” Claude Docs
- [3] Zhang, Lazuka, Murag. 2025. “Equipping agents for the real world with Agent Skills.” Anthropic Engineering
- [4] Bowsher, P. 2026. “Agentic R in Clinical Trials.” PharmaSUG 2026, AI-164
- [5] Sjoberg et al. 2021. “Reproducible Summary Tables with the gtsummary Package.” The R Journal 13(1):570-580
- [6] CDISC. ADaM Implementation Guide v1.3. 2021
- [7] CDISC. Analysis Results Metadata (ARM) v1.0. 2015
- [8] Lu, H., and Z. Yang. 2026. “Introduction to {lyra}.” PHUSE APAC SDE