

When you need
to be sure

```
mirror_mod.use_z = False  
operation == "MIRROR_Z"  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True
```

```
#selection at the end -add
```

```
mirror_ob.select= 1
```

```
mirror_ob.select=1
```

```
context.scene.objects.active
```

```
("Selected" + str(modifier.ob
```

```
mirror_ob.select = 0
```

```
= bpy.context.selected_object
```

```
data.objects[one.name].select
```

```
print("please select exactly
```

```
-- OPERATOR CLASSES --
```

Pharma, Clinical research

SI04: What We Gained and Lost by Using Python in Clinical Data Management

Katarzyna Bzymek - Clinical Data Programmer

PHUSE EU CONNECT 2025 | Hamburg, Germany | 16-19 November 2025



Disclaimer and disclosures

The views and opinions expressed in this presentation are those of the author and do not necessarily reflect the opinions of PHUSE members or SGS.

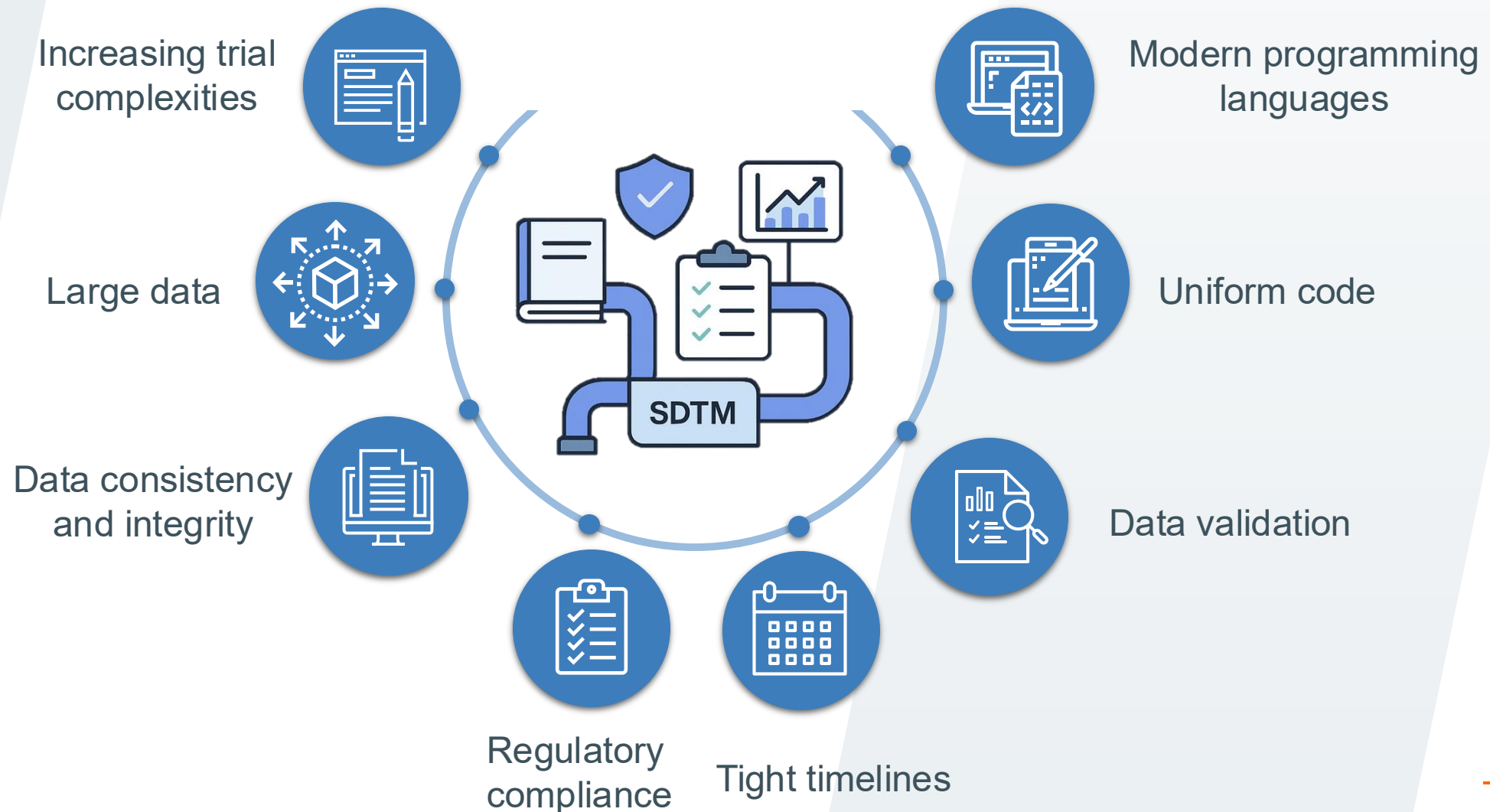
Agenda

- 
- 01 Introduction
 - 02 Embracing Python in SDTM Conversion
 - 03 What We Gained & What We Lost
 - 04 Future Directions and Conclusions

Agenda

- 
- 01 **Introduction**
 - 02 Embracing Python in SDTM Conversion
 - 03 What We Gained & What We Lost
 - 04 Future Directions and Conclusions

Why SDTM workflows need Smarter tools & Strategies



Past Programming Challenges

Programming



Long conversion run-times:

Impact on milestones & timelines

Spaghetti-Code:

Difficulty to maintain code over time

Difficult handover:

Challenging for other programmers to maintain or take over an ongoing trial

Learning



Long learning-curve:

Different tools & coding language

- Annotations
- Mapping
 - In-house tool
- Conversion
 - PL/SQL

Desirable solutions



Shorten conversion times



Feel at ease with taking over trials



More general-purpose language:
Easier to find new programmers



Lowering learning curve



Lowering setup time SDTM database



Agenda

- 
- 01 Introduction
 - 02 **Embracing Python in SDTM Conversion**
 - 03 What We Gained & What We Lost
 - 04 Future Directions and Conclusions



- ✓ High-level, object oriented
- ✓ Versatile (AI, web, data, automation, cloud)
- ✓ Simple & readable syntax
- ✓ Extensive support libraries
 - Pandas/Polars
 - Numpy
 - Parquet
- ✓ Open source
- ✓ Cloud Friendly: Massive global community

Visual Studio Code + Git



Code editor

- Jupyter Notebooks
- Language support
- PEP8 linting
- Git – version control



More uniform code
across different
programmers



SGS PyConv implementation

EDC data
+ convert
to .parquet



Pre-processing
phase

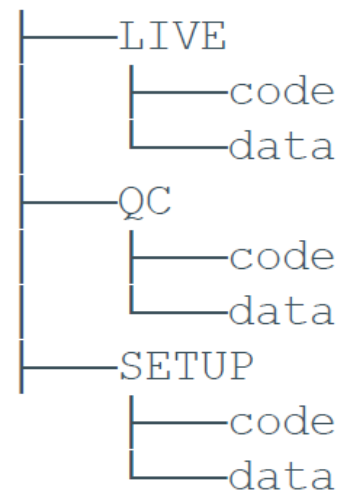
- Date of visit



Enrichment phase

- Formatting
- Datetime conversion
- Unpivot fields
- Find best matching trial

Domain
transformations
SDTM mapping



Datasets are
uploaded to Oracle

Derivations

- GRPID
- Death flag
- ARM/ARMCD

Post-processes

- RELREC
- EPOCH
- SE completion

SDTM Mapping Process

FA (Findings About Events or Interventions)

QS (Questionnaires)

		QSSCAT = OBSERVED AT CLINIC	
Project Name:		FACAT = COUGH MONITORING	
Form: VAS Monitoring of Cough		FAOBJ = COUGH	
		QSCAT = COUGH MONITORING	
Tick if the same as visit date or complete		[NOT SUBMITTED]	
Date of assessment (complete if different from visit date)		FADTC	QSDTC
Did a cough episode occur within 10 mins after the inhalation?		Yes ☐	No ☐
FAEVLINT = PT10M		FAORRES when FATESTCD = OCCUR	
VAS quantification within 10 mins after the inhalation		Fixed Unit: mm	
QSEVLINT = PT10M		QSORRES when QSTESTCD = VAS	
Was the cough episode clinically significant?		Yes ☐	No ☐
FACLSIG in SUPPFA			
Did a cough episode occur within 24h after the inhalation?		Yes ☐	No ☐
FAEVLINT = PT24H		FAORRES when FATESTCD = OCCUR	
VAS quantification within 24h after the inhalation		Fixed Unit: mm	
QSEVLINT = PT24H		QSORRES when QSTESTCD = VAS	
Was the cough episode clinically significant?		Yes ☐	No ☐
FACLSIG in SUPPFA			
<i>In case of any clinically significant cough episode(s), not due to a pre-existing condition, it should be recorded as an Adverse Event.</i>			

	SUBJID	RESULT	UNPIVOTED	VASDAT_ISO8601	VASDTSAME_RAW	DOV	VAS10_CLSIG_STD	VAS24_CLSIG_STD
0	012002	N	COUGH10MIN_STD	2024-11-10	1	2024-11-10	NaN	NaN
1	012002	Y	COUGH10MIN_STD	2024-11-11	1	2024-11-11	NaN	NaN
2	012002	Y	COUGH10MIN_STD	2024-11-12	1	2024-11-12	N	NaN
3	012002	N	COUGH10MIN_STD	2024-11-13	1	2024-11-13	NaN	NaN
4	012002	N	COUGH10MIN_STD	2024-11-14	0	2024-11-14	NaN	NaN
5	012002	Y	COUGH10MIN_STD	2024-11-01	1	2024-11-01	Y	N
6	012002	N	COUGH10MIN_STD	2024-11-02	1	2024-11-02	NaN	NaN
7	012002	Y	COUGH10MIN_STD	2024-11-03	1	2024-11-03	N	Y
8	012002	Y	COUGH10MIN_STD	2024-11-04	1	2024-11-04	N	NaN
9	012002	N	COUGH10MIN_STD	2024-11-05	1	2024-11-05	NaN	N
10	012002	N	COUGH10MIN_STD	2024-11-06	1	2024-11-06	NaN	NaN

FACAT : COUGH MONITORING ,

"FASTRESC": None,
 "FASTRESN": None,
 "FASTRESU": None,
 "FASTAT": None,
 "FAEASND": None,
 "FALOC": None,
 "FALAT": None,
 "FAEALRES": None

"FADTC": np.where(dfdict["X_VAS_PREP_FA"]["VASDTSAME_RAW"]=='1',
 dfdict["X_VAS_PREP_FA"]["DOV"],
 dfdict["X_VAS_PREP_FA"]["VASDAT_ISO8601"]),

"FADY": None,

"FATPT": np.where(dfdict["X_VAS_PREP_FA"]["UNPIVOTED"]=="COUGH10MIN_STD", "WITHIN 10 MIN POSTDOSE", "WITHIN 24 H POSTDOSE"),

"FATPTNUM": np.where(dfdict["X_VAS_PREP_FA"]["UNPIVOTED"]=="COUGH10MIN_STD", "9", "1436"),

"FATPTREF": "STUDY DRUG INTAKE " + dfdict["X_VAS_PREP_FA"]['VISIT'].str.strip(),

"FAEVLINT": np.where((dfdict["X_VAS_PREP_FA"]["UNPIVOTED"] == "COUGH10MIN_STD") & (dfdict["X_VAS_PREP_FA"]["RESULT"].notna()), "PT10M",
 np.where((dfdict["X_VAS_PREP_FA"]["UNPIVOTED"] == "COUGH24H_STD") & (dfdict["X_VAS_PREP_FA"]["RESULT"].notna()), "PT24H",
 None)),

"SQ_FACLSIG": np.where(dfdict["X_VAS_PREP_FA"]["UNPIVOTED"] == "COUGH10MIN_STD", "FASEQ|" + dfdict["X_VAS_PREP_FA"]["VAS10_CLSIG_STD"].str.upper(),
 np.where(dfdict["X_VAS_PREP_FA"]["UNPIVOTED"] == "COUGH24H_STD", "FASEQ|" + dfdict["X_VAS_PREP_FA"]["VAS24_CLSIG_STD"].str.upper(),
 None))

"SQ_FACLSIG": np.where(dfdict["X_VAS_PREP_FA"]["UNPIVOTED"] == "COUGH10MIN_STD", "FASEQ|" + dfdict["X_VAS_PREP_FA"]["VAS10_CLSIG_STD"].str.upper(),
 np.where(dfdict["X_VAS_PREP_FA"]["UNPIVOTED"] == "COUGH24H_STD", "FASEQ|" + dfdict["X_VAS_PREP_FA"]["VAS24_CLSIG_STD"].str.upper(),
 None))

Project Name:

Form: VAS Monitoring of Cough

FACAT = COUGH MONITORING

FAOBJ = COUGH

QSCAT = COUGH MONITORING

Tick if the same as visit date or complete

[NOT SUBMITTED]

Date of assessment (complete if different from visit date)

FADTC

QSDTC

Did a cough episode occur within 10 mins after the inhalation?

FAEVLINT = PT10M

FAORRES when FATESTCD = OCCUR

No

VAS quantification within 10 mins after the inhalation

Fixed Unit: mmHg

QSEVLINT = PT10M

QSORRES when QSTESTCD = VAS

Was the cough episode clinically significant?

FACLSIG in SUPPFA

Yes

No

Did a cough episode occur within 24h after the inhalation?

FAEVLINT = PT24H

FAORRES when FATESTCD = OCCUR

Yes

No

Early Data Preview in CDISC tables

- ✓ Immediate feedback
- ✓ Efficient mapping process
- ✓ No conversion needed
- ✓ Faster debug times

```
view(cdisc_domains["CDISC_FA"])
```

Project Name: FACAT = COUGH MONITORING
Form: VAS Monitoring of Cough FAOBJ = COUGH
QSCAT = COUGH MONITORING

Tick if the same as visit date or complete [NOT SUBMITTED]
Date of assessment (complete if different from visit date) FADTC QSDTC
Did a cough episode occur within 10 mins after the inhalation? Yes ☐
FAEVLINT = PT10M FAORRES when FATESTCD = OCCUR No ☐
VAS quantification within 10 mins after the inhalation Fixed Unit: mmc
QSEVLINT = PT10M QSORRES when QSTESTCD = VAS
Was the cough episode clinically significant? FACLSIG in SUPPFA Yes ☐
No ☐
Did a cough episode occur within 24h after the inhalation? Yes ☐
FAEVLINT = PT24H FAORRES when FATESTCD = OCCUR No ☐

DOMAIN	FATESTCD	FAOBJ	FACAT	FAORRES	VISITNUM	VISIT	VISITDY	EPOCH	FADTC	FATPT	FATPTNUM	FATPTREF	FAEVLINT	SQ_FACLSIG
FA	OCCUR	COUGH	COUGH MONITORING	Y	112	DAY 12	12		12/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 12	PT10M	FASEQ N
FA	OCCUR	COUGH	COUGH MONITORING	N	113	DAY 13	13		13/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 13	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING	N	114	DAY 14	14		14/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 14	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING	Y	101	DAY 1	1		01/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 1	PT10M	FASEQ Y
FA	OCCUR	COUGH	COUGH MONITORING	N	102	DAY 2	2		02/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 2	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING	Y	103	DAY 3	3		03/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 3	PT10M	FASEQ N
FA	OCCUR	COUGH	COUGH MONITORING	Y	104	DAY 4	4		04/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 4	PT10M	FASEQ N
FA	OCCUR	COUGH	COUGH MONITORING	N	105	DAY 5	5		05/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 5	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING	N	106	DAY 6	6		06/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 6	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING	Y	107	DAY 7	7		07/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 7	PT10M	FASEQ Y
FA	OCCUR	COUGH	COUGH MONITORING	Y	108	DAY 8	8		08/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 8	PT10M	FASEQ N
FA	OCCUR	COUGH	COUGH MONITORING	N	109	DAY 9	9		09/11/2024	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 9	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING		112	DAY 12	12			WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 12		
FA	OCCUR	COUGH	COUGH MONITORING		113	DAY 13	13			WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 13		
FA	OCCUR	COUGH	COUGH MONITORING	N	114	DAY 14	14		23/01/2025	WITHIN 10 MIN POSTDOSE	9	STUDY DRUG INTAKE DAY 14	PT10M	
FA	OCCUR	COUGH	COUGH MONITORING	N	110	DAY 10	10		10/11/2024	WITHIN 24 H POSTDOSE	1436	STUDY DRUG INTAKE DAY 10	PT24H	
FA	OCCUR	COUGH	COUGH MONITORING	N	111	DAY 11	11		11/11/2024	WITHIN 24 H POSTDOSE	1436	STUDY DRUG INTAKE DAY 11	PT24H	

Efficient Code Review



Allows the code to remain clear and consistent



Efficient conversion: everything in the correct cycle



Standardized way of working across teams



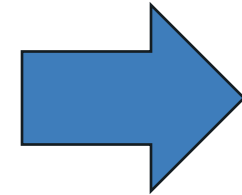
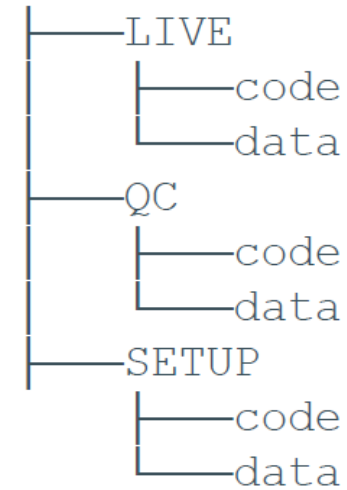
Take over trials with confidence



Prevent spaghetti code



Library of a good quality



Code review

- Track source and type of changes
- Streamlines personal code management
- Audit trails
- Maintain compliance with regulatory standards



Agenda

- 
- 01 Introduction
 - 02 Embracing Python in SDTM Conversion
 - 03 **What We Gained & What We Lost**
 - 04 Future Directions and Conclusions

Overall Improvements



Clear data conversion

- ✓ Divided into steps
- ✓ Easy to get started and understand the conversion processes



Every cycle can be run separately



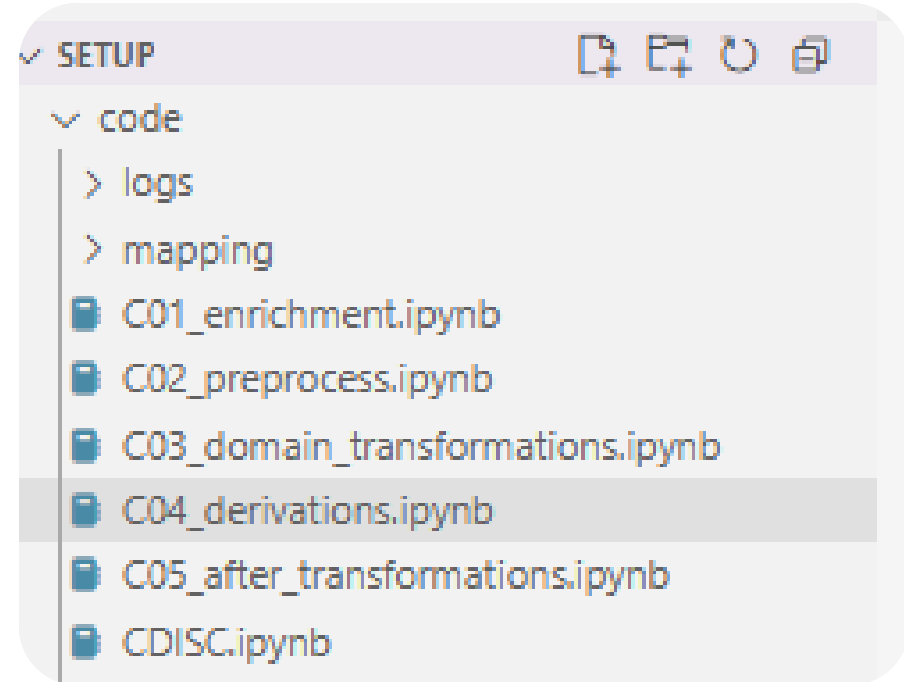
Simplified Code management and understanding



Faster Problem Solving

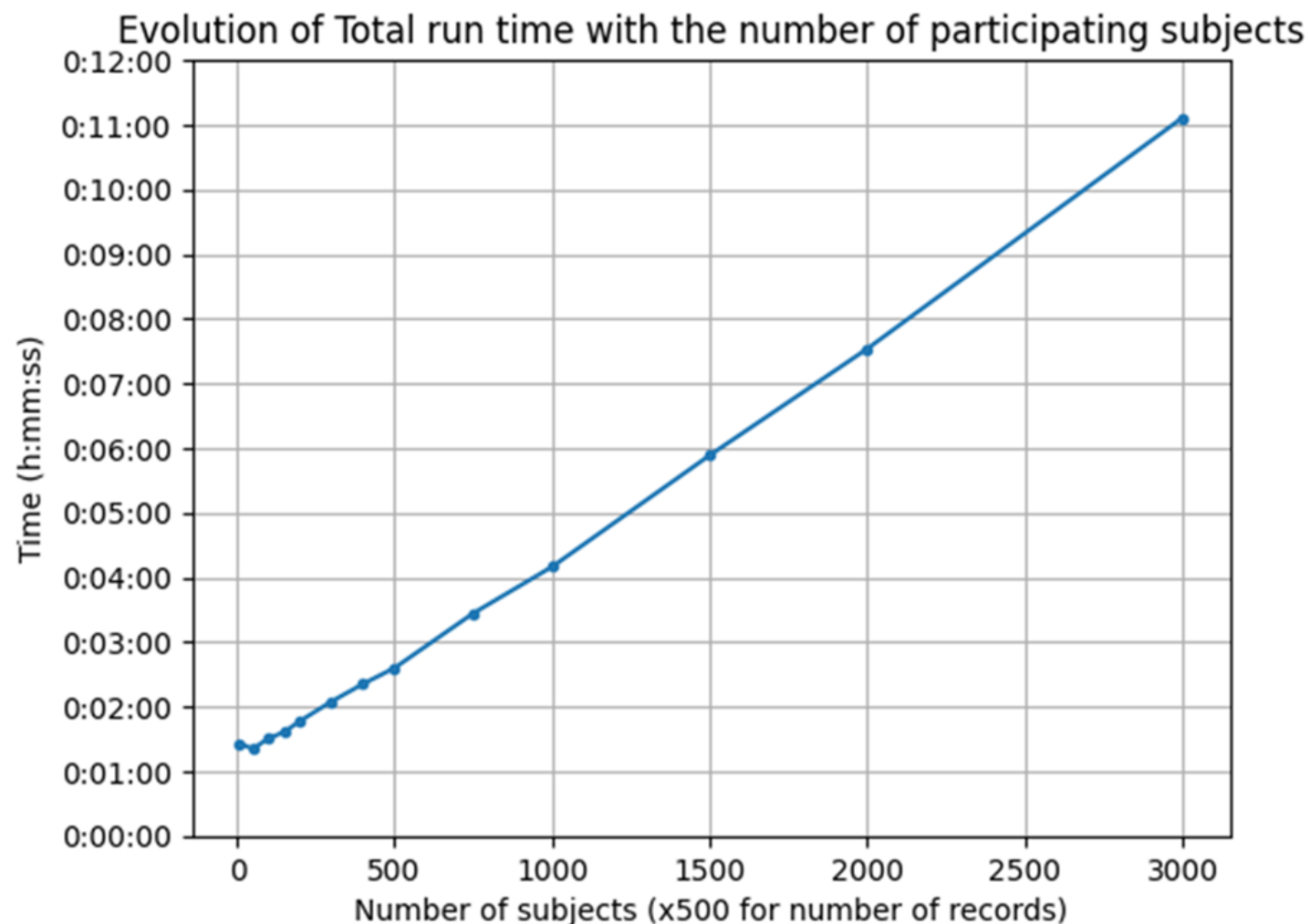


Faster debugging



Further improvements

- XPT Dataset Creation
 - Python can read and generate XPT files (the FDA-compliant file format)
 - Removes reliance on commercial third-party software
- Shorter conversion run times
- Reduced times for problem solving during QC stage
- More standardized approach across different sponsors



What we lost using Python



Requires more people

- Code review → guidelines
 - Issues caught early in the process
 - Cross-team alignment
 - Increased collaboration



Code review may slow down the process

- Need for enough programmers trained for code review
 - Only during urgent updates
 - Temporary issues: more code reviewers in the future



Long standing code is lost

- Outdated scripts refactored into cleaner maintainable Python code



Many updates to packages

- Decreases as the framework becomes more stable
- Reflects evolving ecosystem with continuously improved performance, security and functionality

Agenda

- 
- 01 Introduction
 - 02 Embracing Python in SDTM Conversion
 - 03 What We Gained & What We Lost
 - 04 **Future Directions and Conclusions**



Future directions

- Extending SGS Python packages
- Creating libraries for different sponsors
- More code reviewers
- Possible automation
 - Use of validated functions to reduce QC on conversion efforts
 - Automate code generation based on machine readable aCRF
- Metadata/define.xml completion

Conclusions

Transitioning to Python for SDTM conversion is more than a technical upgrade it's a strategic shift toward agility, scalability, and improved collaboration.



Benefits outweigh challenges



Standardized workflows

- Reduced variability
- Improved consistency in data processing



Improved Code Management

- Refactored legacy code into modern maintainable Python scripts
- Increasing adaptability to new study designs



Scalable and future-ready framework

- Faster conversion running times
- Capable of handling large datasets, wearable data, and evolving sponsor-specific requirements



Early Data Preview

- Allows immediate feedback
- Faster debugging & reduced QC effort
- Improving confidence in outputs



Questions?

Contact us

Katarzyna Bzymek - Clinical Data Programmer



clinicalresearch@sgs.com



www.sgs.com/pharma



sgs.com/linkedinpharma



Visit us at Booth 37

