Implementing CDISC CORE in a cloud native, regulated environment supporting Rare Diseases submissions

Sandeep Juneja, Clinical Solutions Lead, argenx
Emmy Pahmer, Senior Clinical Programmer, OCS Life Sciences

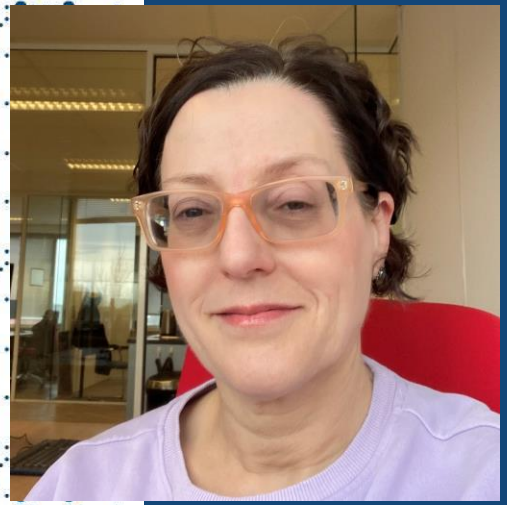
Meet the Speakers

Sandeep Juneja

Title: Clinical Solutions Lead

Organization: argenx

With over 20 years of expertise in Healthcare & Life Sciences applications on Cloud systems, Sandeep is a seasoned Clinical Solutions Lead. Sandeep's proficiency covers the entire clinical trials process, showcasing leadership in AI/ML teams, successful deployment of analytical solutions, and adherence to industry regulations.



Emmy Pahmer

Title: Senior Consultant, Clinical Programming

Organization: OCS Life Sciences

Emmy has been working in clinical research for over 20 years, primarily in SAS programming roles. She has experience with Data Management processes, PK/PD reconciliation and reporting, generating SDTM data as well as supporting other teams: Biostatistics, Medical Monitoring, Sample Management, Pharmacovigilance and Operations.

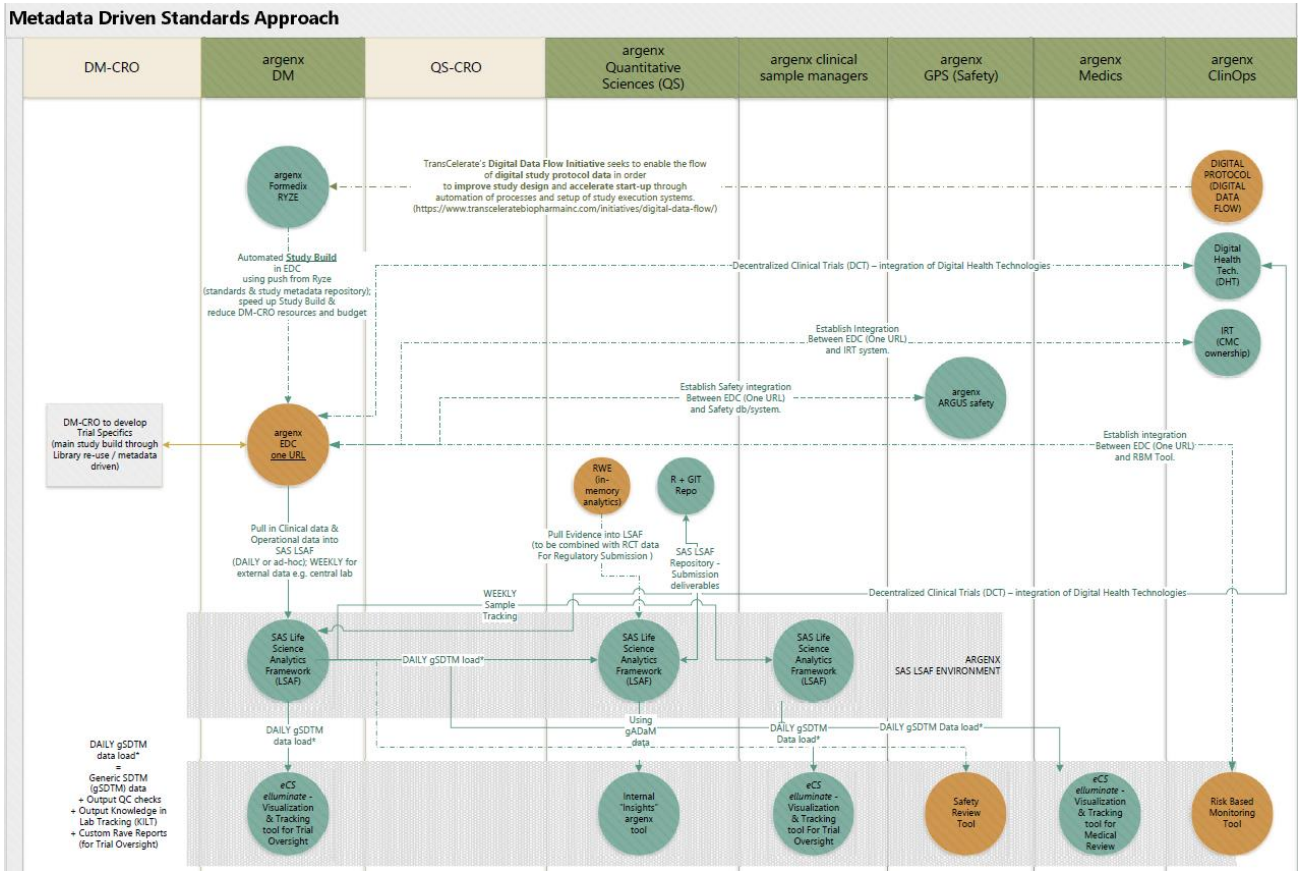
Agenda

1. Introduction
2. Implementation Journey
3. Technical Implementation & Execution
4. Evaluation - CORE findings after running it on argenx ongoing study (one pilot)
5. Next Steps

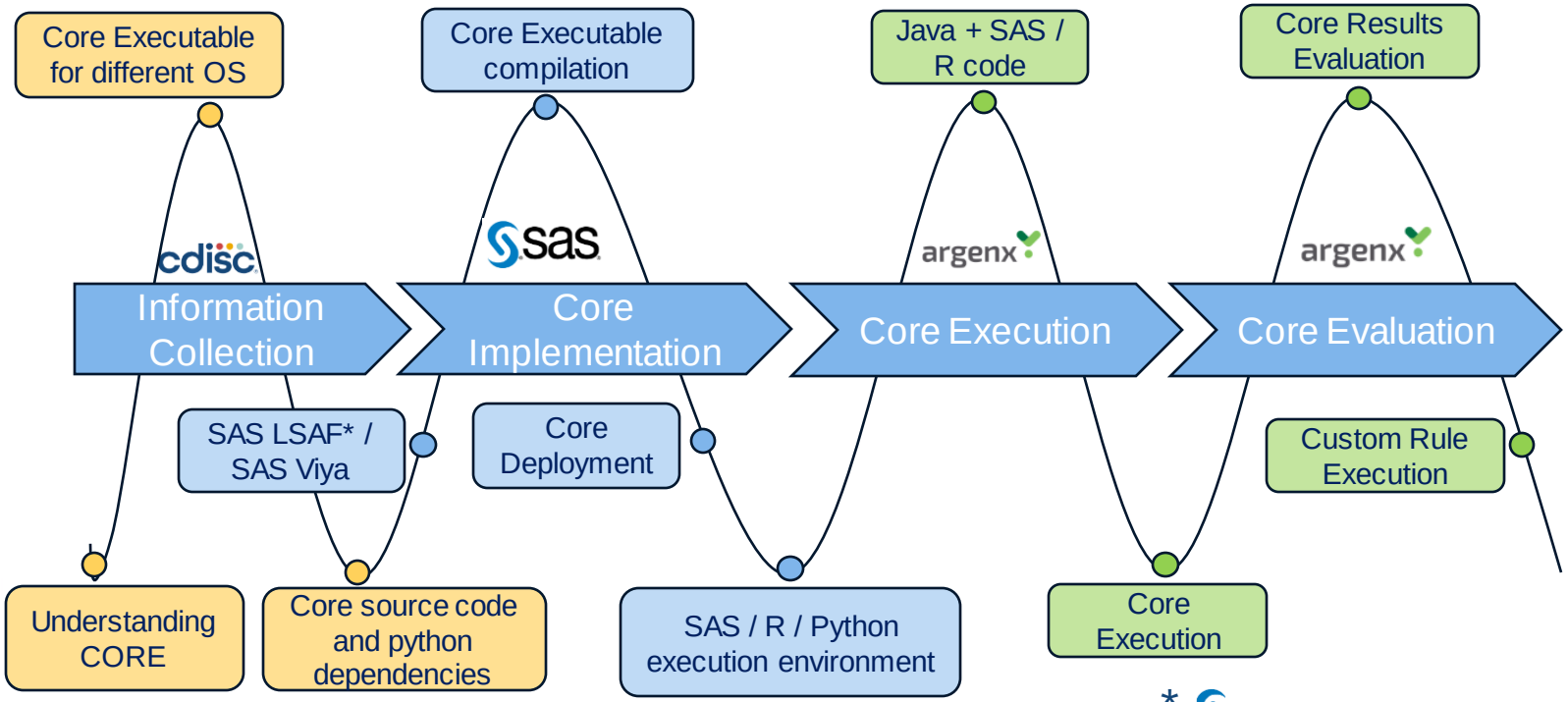
CDISC CORE intro

- Overall goal of the CDISC Open Rules Engine (CORE) Project is to deliver a governed set of unambiguous and executable Conformance Rules for each Foundational Standard
- **Conformance Rules** - over 1900 rules covering multiple versions of implementation guides for various Foundational Standards and Regulatory Rules
- **CORE Engine** – Open-source framework, available in github and free to all CDISC community. Release the open-source engine under the CDISC Open-Source Alliance (COSA)
- **CORE Rule Editor** – Web-based application to develop CORE rules in YAML with real-time syntax checking.
- CORE Github - https://github.com/cdisc-org/cdisc-rules-engine

argenx Environment



Implementation Journey



* SAS Life Science Analytics Framework

Execution - SAS / Java

- Java method
  - Pass parameter list ProcessBuilder object

```
public void Execute(String cdisc_core_root_dir, String core_cmd, String core_standard, String core_standard_version, String resource_template, String study_xpts, String result_file) {

    List<String> params = Arrays.asList(cdisc_core_root_dir, core_cmd, "-s", core_standard, "-v", core_standard_version, "-rt",resource_template,"-d",study_xpts,"-o",result_file);
    ProcessBuilder pb = new ProcessBuilder(params);
    pb.redirectErrorStream(true);

    try {
      Process p = pb.start();
      BufferedReader reader=new BufferedReader( new InputStreamReader(p.getInputStream()));
      String s;
      while ((s = reader.readLine()) != null){
            System.out.println(s);
      }

    } catch (IOException e) {e.printStackTrace();}
}
```

- SAS macro
  - Call Java method using JavaObj

```
%macro run_cdisc_core_checks(core_cmd=,core_standard=,core_standard_version=, study_xpts=, study_report_path=, study_report_name=%str());
%let resource_template=%str(&cdisc_core_root_dir./resources/templates/report-template.xlsx);
%put resource_template = &resource_template;

%if %length(&study_report_name.)<=1 %then %let study_report_name = CDISC_Core_Results_%SYSFUNC(translate(%SYSFUNC(datetime(), E8601DT.),'-',':')).xlsx;
%let result_file = %str(&study_report_path./&study_report_name.);
%put result_file = &result_file;

* Initialize the Classpath;
%init_classpath_update;
* Add the jar file to the Classpath;
%add_to_classpath(&base_path./jar/run_cdisc_core.jar);

data cdisc_core_cli;
   dcl javaobj u("lsaf_cdisc_core/Run_CDISC_Core");
   u.callVoidMethod("Execute3",kstrip("&cdisc_core_root_dir/core"),kstrip("&core_cmd"), kstrip("&core_standard"),kstrip("&core_standard_version"),
                    kstrip("&resource_template"),kstrip("&study_xpts"), kstrip("&result_file"));
run;
*Reset to Original ClassPath;
%reset_classpath;
%mend run_cdisc_core_checks;
%run_cdisc_core_checks(core_cmd=&core_cmd.,core_standard=&core_standard.,core_standard_version=&core_standard_version.,
                study_xpts=&study_xpts., study_report_path=&study_report_path., study_report_name=&study_report_name.);
```

- LSAF UI (Jobs)

Name: test_run_cdisc_core.job
Documentation:

Label	Value
CORE Command	validate
CORE Standard	sdtmig
CORE Standard Version	3-4
Study Data (xpt files) location	/general/share_general/utili
Study Report path	/general/share_general/utili
Study Report Name	argx_test_study

Execution R (within SAS LSAF)

- R Code

execute_core3.R* ×

Code | Log

```
# check if its transient path, switch to workspace path and if its workspace, use workspace path
ws_path <- ifelse(grepl(".transient",LSAF_WS), paste("/lsafshared/SASWorkspaces/",LSAF_USR,sep=""), LSAF_WS)

cdisc_core_utility <- ifelse(exists("cdisc_core_utility"), cdisc_core_utility, paste(ws_path,"/general/share_general/utilities/cdisc_core/executable",sep=""))
# cdisc_core_utility <- paste(LSAF_WS,"/Customer Preview/CDISC_Core/core",sep="")
# print(cdisc_core_utility)

core_script <- paste("'",cdisc_core_utility,"/core","'",sep="")
# print(core_script)

core_cmd <- ifelse(exists("core_cmd"), core_cmd, "validate")

core_standard <- ifelse(exists("core_standard"), core_standard, "sdtmig")
core_std_cmd <- paste("-s", core_standard)

core_standard_version <- ifelse(exists("core_standard_version"), core_standard_version, "3-4")
core_stdv_cmd <- paste ("-v", core_standard_version)

core_report_template <- ifelse(exists("core_report_template"), core_report_template, paste(cdisc_core_utility,"/resources/templates/report-template.xlsx",sep=""))
core_rptt_cmd <- paste("-rt '", core_report_template,"'", sep="")

study_xpts <- ifelse(exists("study_xpts"), study_xpts, paste(LSAF_WS,"/general/share_general/utilities/cdisc_core/source_code/test_data",sep=""))
study_data <- paste("-d '", study_xpts,"'", sep="")

study_report_path <- ifelse(exists("study_report_path"), study_report_path, paste(LSAF_WS,"/general/share_general/utilities/cdisc_core/source_code/reports",sep=""))
study_report_name <- ifelse(exists("study_report_name"), study_report_name, "test")
study_report <- paste("-o '", study_report_path,"/",study_report_name, "'", sep="")

core_exec_cmd <- paste(core_script,core_cmd,core_std_cmd, core_stdv_cmd, core_rptt_cmd, study_data, study_report, sep=" ")
core_exec_cmd <- gsub("[\r\n]", "", core_exec_cmd)
print(core_exec_cmd)

system(core_exec_cmd)

rm(cdisc_core_utility, core_script, core_cmd, core_standard, core_standard_version, core_report_template, study_xpts, study_report_path, study_report_name)
```

- LSAF UI (Jobs)

Name: execute_core3.job
Documentation:

Label	Value
CORE Command	validate
CORE Standard	sdtmig
CORE Standard Version	3-4
Study Data (xpt files) location	/general/share_general/utili
Study Report path	/general/share_general/utili
Study Report Name	test

Implementation Challenges

- **Different Operating System**
  - CDISC executable compiled for windows and Linux (ubuntu) whereas SAS LSAF uses CentOS Linux.
  - CDISC source code needed to be compiled into executable for SAS LSAF CentOS by downloading all dependencies.
- **Execution Permission**
  - CDISC Core executable required execution permission which required working with SAS team.
  - SAS LSAF does not allow command line execution using Base/SAS
- **Programming Language(s) support**
  - Additional source code required to be written in Java, compiled into executable jar and wrapped into SAS macro to support Core execution via SAS
  - Source code written in R programming language too, to execute Core in SAS LSAF.
- **Shared Workspace**
  - Multiple users can access same executable

Evaluation - Report

- Core executed and output reviewed for one trial.
- Only reviewed the rules where issues were reported

Conformance Details	
Report Generation	2024-03-04T10:50:10
Total Runtime	37.06 seconds
CORE Engine Version	0.6.2
Standards Details	
Standard	SDTMIG
Version	V3.3
CT Version	
Define-XML Version	
UNII Version	Not configured
Med-RT Version	Not configured
MedDRA Version	Not configured
WHODRUG Version	Not configured
SNOMED Version	Not configured

Conformance Details - Standards

	A	B	C	D	E	F
1	Dataset	Label	Location	Modified Time Stamp	Size (kb)	Number of Records
2	mh.xpt	Medical History	D:_EP\cdisc_core\data\xpt	2024-02-29T08:35:22	174.8	254
3	suppec.xpt	Supplemental Qualifiers for EC	D:_EP\cdisc_core\data\xpt	2024-02-29T08:30:55	328.4	2091
4	suppsv.xpt	Supplemental Qualifiers for SV	D:_EP\cdisc_core\data\xpt	2024-02-29T08:40:55	2.64	4
5	suppda.xpt	Supplemental Qualifiers for DA	D:_EP\cdisc_core\data\xpt	2024-02-29T08:40:05	8.08	68
6	qs.xpt	Questionnaires	D:_EP\cdisc_core\data\xpt	2024-02-29T08:36:55	6425.28	20783
7	suppfa.xpt	Supplemental Qualifiers for FA	D:_EP\cdisc_core\data\xpt	2024-02-29T08:38:19	3.68	8
8	ft.xpt	Functional Tests	D:_EP\cdisc_core\data\xpt	2024-02-29T08:37:53	4696.32	16351
9	suppds.xpt	Supplemental Qualifiers for DS	D:_EP\cdisc_core\data\xpt	2024-02-29T08:30:27	9.76	82
10	lb.xpt	Laboratory Test Results	D:_EP\cdisc_core\data\xpt	2024-02-29T08:34:43	13419.28	19439
11	eg.xpt	ECG Test Results	D:_EP\cdisc_core\data\xpt	2024-02-29T08:31:26	933.44	2347

Dataset details – datasets used, size and number of records

A	B	C	D	
Dataset	CORE-ID	Message	Issues	Explanation
DM	CORE-000047	ARM value in DM dataset is not among the values of ARM variable in TA dataset. This is allowed only in a multistage study with incomplete ARM assignment. Please confirm if your study is a multistage assignment study	17	
DM	CORE-000209	ACTARM value in DM dataset is not among the values of ARM variable in the TA dataset. This is allowed only in a multistage study with incomplete ARM assignment. Please confirm if your study is a multistage assignment study.	17	
MH	CORE-000272	MHCAT is equal to DOMAIN.	213	
PC	CORE-000200	PCORRES cannot be null when PCSTAT is null or PCDRVFL not equal to 'Y'	338	
PC	CORE-000225	PCSTAT should be "NOT DONE" when PCREASND is provided.	55	

Issue Summary - Shows the number of cases found for each issue. Explanation column to capture follow-up comments

Evaluation - Report

CORE-ID	Message	Executability	Dataset	USUBJID	Record	Seque	Variable(s)	Value(s)
CORE-000200	when PCSTAT is null or PCDRVFL not equal to 'Y'	fully executable	PC	0010013001	21	21	PCDRVFL, PCORRES, PCSTAT	Not in dataset, ,
CORE-000200	when PCSTAT is null or PCDRVFL not equal to 'Y'	fully executable	PC	0010013001	22	22	PCDRVFL, PCORRES, PCSTAT	Not in dataset, ,
CORE-000200	when PCSTAT is null or PCDRVFL not equal to 'Y'	fully executable	PC	0010013001	23	23	PCDRVFL, PCORRES, PCSTAT	Not in dataset, ,
CORE-000200	when PCSTAT is null or PCDRVFL not equal to 'Y'	fully executable	PC	0010013001	24	24	PCDRVFL, PCORRES, PCSTAT	Not in dataset, ,

Issue Details – Detailed list of issues, for each case encountered

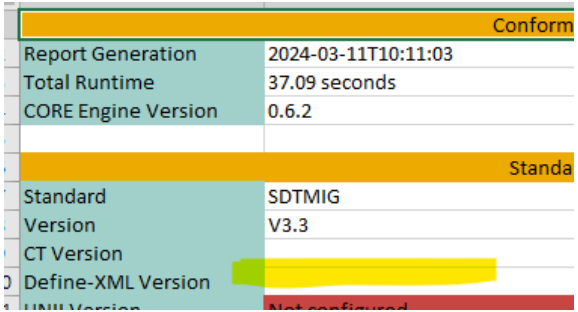
CORE-ID	Version	CDISC RuleID	FDA RuleID	PMDA RuleID	Message	Status
CORE-000028	1	CG0008			--TPTREF is empty and --ELTM is not empty	SKIPPED
CORE-000237	1	CG0027			--SCAT is equal to --CAT	SUCCESS
CORE-000201	1	CG0029			USUBJID is not found in DM.USUBJID	SUCCESS
CORE-000168	1	CG0034			VISITNUM is not among VISITNUM in SV domain.	SUCCESS
CORE-000012	1	CG0040			AEOCCUR is present in AE dataset.	SUCCESS
CORE-000022	1	CG0041			At least one of the Seriousness criteria (AESCAN, AESCONG, AESDISAB, AESDTH, AESHOSP, AESLIFE, AESOD or AESMIE) = 'Y', but AESER = 'N' or empty.	SUCCESS

Rules Report – Rules executed Successfully vs Skipped. Link between CORE-ID & CDISC rule

Evaluation - Challenges

Define.xml

- Core rules are executed with or without Define.xml file
- If non-well-formed Define.xml was present, CORE didn't execute the rules and the log provided a general message about the Define.xml



	Conform
Report Generation	2024-03-11T10:11:03
Total Runtime	37.09 seconds
CORE Engine Version	0.6.2
	Standa
Standard	SDTMIG
Version	V3.3
CT Version	
Define-XML Version	

```
[ERROR 2024-03-11 12:03:12,654 - excel_report.py:110] - Missing required keyword argument _content in TranslatedText
Traceback (most recent call last):
  File "cdisc_rules_engine\services\define_xml\base_define_xml_reader.py", line 80, in read
  File "odmlib\loader.py", line 28, in MetaDataVersion
  File "odmlib\define_loader.py", line 68, in load_metadataversion
  File "odmlib\define_loader.py", line 37, in load_document
  File "odmlib\define_loader.py", line 37, in load_document
  File "odmlib\define_loader.py", line 32, in load_document
  File "odmlib\define_loader.py", line 37, in load_document
  File "odmlib\define_loader.py", line 25, in load_document
  File "<string>", line 1, in <module>
  File "odmlib\odm_element.py", line 85, in __init__
ValueError: Missing required keyword argument _content in TranslatedText
```

Evaluation - Challenges

CORE Log

- Processing is not sequential (one check completing then the next starting) resulting in difficulty to read and process log information for checks

```
[INFO 2024-03-17 21:52:47,759 - console_logger.py:41] - is_suitable_for_validation. rule id=CORE-000001, domain=SUPPAE, result=False
[INFO 2024-03-17 21:52:47,759 - console_logger.py:41] - is_suitable_for_validation. rule id=CORE-000002, domain=AE, result=False
[INFO 2024-03-17 21:52:47,759 - console_logger.py:41] - Skipped domain SUPPAE.
[INFO 2024-03-17 21:52:47,759 - console_logger.py:41] - Skipped domain AE.
```

- No information on why rules are skipped in log

CORE-000001	1	CG0176			IECAT equals "INCLUSION" and IEORRES is not equal to "N".	SKIPPED
CORE-000002	1	CG0208			SESTDTC is required.	SUCCESS

Evaluation - Challenges

Rules Logic evaluation

- A few checks didn't work as expected, e.g.

Dataset	Check #	Message/Rule	Comment
FA	CORE-000281	FASTDTC is after FAENDTC	FASTDTC=2023-07-13T11:57 FAENDTC=2023-07-13 Compares partial dates, includes when they are equal.
RS	CORE-000200	RSORRES cannot be null when RSSTAT is null or RSDRVFL not equal to 'Y'	There are cases where ORRES is null because it's derived. Looks like the check is only looking to see if ORRES is null, not other criteria. For RS & VS.
TS	CORE-000107	An appropriate subject identifier is not present. In addition to STUDYID, DOMAIN, and TSSEQ being required for all domains based on one of the 3 general observation classes, one of USUBJID, APID, SPDEVID, or POOLID must also be present.	APID=Not in dataset POOLID=Not in dataset SPDEVID=Not in dataset USUBJID=Not in dataset It's not a general observation class.

- Issues were discussed with the CORE team
- Useful for them to get feedback on running the checks with real data

Next Steps

- Qualification of CDISC CORE
  - LSAF is qualified environment. To use CDISC Core in production environment, it requires to pass through argenx qualification process
  - Qualification of SAS / R utilities to execute CDISC Core

  Note: Qualification of Core rules engine is out of scope
- Working with CDISC team on automating the CI/CD for LSAF OS
- Involvement in development of CORE engine and rules
  - Technical & Functional
- Working with SAS team to support CORE execution in LSAF
- Collaboration with CRO partners
- **Community effort, industry needs to provide feedback to CORE**

Thank You!

Emmy Pahmer - epahmer@argenx.com / emmy.pahmer@ocs-consulting.com

Sandeep Juneja – sjuneja@argenx.com