PhUSE 2013

Paper AD05

Organizing Deliverables for Clinical Trials – The Concept of Analyses and its Implementation in EXACT

Hansjörg Frenzel, PRA International, Mannheim, Germany

ABSTRACT

Clinical trials can have deliverables for multiple purposes. Some are described in the initial protocol (like interim analyses or final analyses), while others are decided upon during discussions with regulatory agencies (e.g., annual safety reports or data monitoring committee reports). In addition, ad-hoc requests can come up at any time during the trial. While clinical trials with multiple types of deliverables are fairly common, it can get hard to organize data, programs and reports. Teams try to solve this by introducing customized naming conventions or branching out the study folder structures. Both solutions make documentation of dependencies more challenging and harder to understand for new programmers. PRA deployed EXACT, a system that enables programmers to easily organize reporting deliverables in separate folder structures. The integration of data, programs, logs and reports is done in the application itself, and thus lets the programmers only see or execute what belongs to a specific analysis.

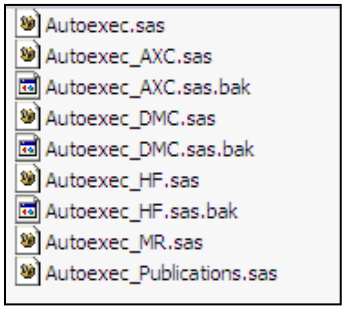
INTRODUCTION

Clinical studies – especially those with a long duration – can have deliverables of multiple types, some of them already announced in the study protocol, some of them being unplanned. Interim analysis results and final analysis results are examples of planned deliverables; deliverables for publication or medical review reports are examples of reports that are often requested on an ad-hoc base. Most companies have standards for folder structures, in which the various files of a clinical trial can be placed, and an accompanying set of rules about which files to store in which directories. Folder structures must work with a wide range of clinical trials and are general in nature. If there are no specific rules about how to handle multiple types of deliverables, teams must come up with solutions themselves. As those solutions are usually made under pressure, some of them might be sub-optimal and lacking proper documentation. Especially long-lasting clinical trials might face changes in the programmer teams over time, and with each new team lead upcoming new deliverable types might be stored in a different way. This makes it hard for every outsider to step in and understand the inner mechanics of the study programs.

THE PROBLEM: UNPREDICTABLE STUDY FOLDER STRUCTURE

The following study had one interim analysis and one final analysis described in the study protocol, and data monitoring committee deliverables described in the contract. Additionally data safety update reports (DSUR) were included in the contract, but these were not recognized by the programming team during the initial planning phase. During the case of the trial the client requested regular data monitoring reports, and after the first interim analysis also publications were planned. The SDTM database had to be sent on a regular schedule as well.

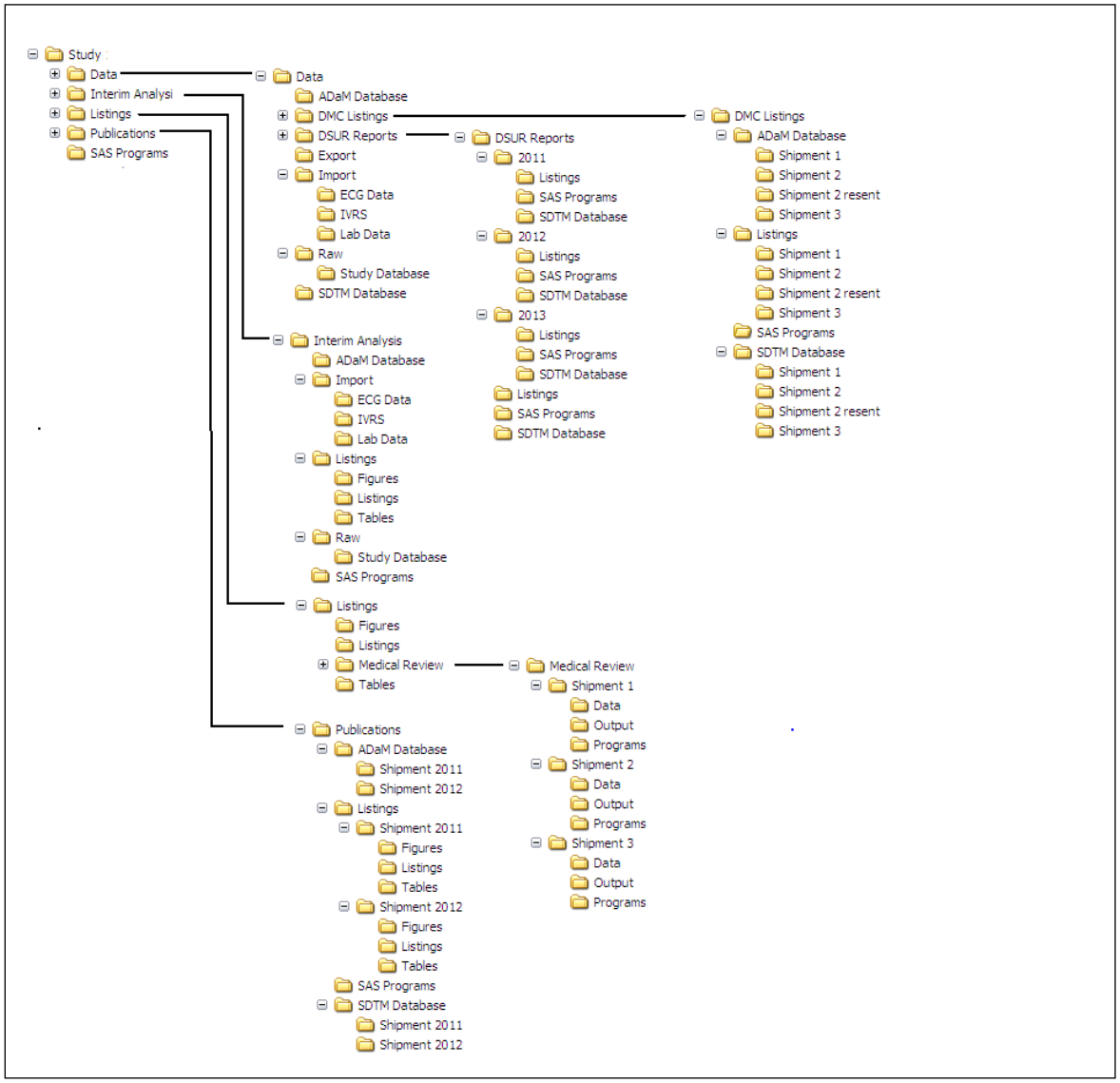
Initially the team planned to program the interim analysis database and tables, figures and listings, which were supposed to be a subset of the final analysis, and after completion head on towards the final analysis. But weeks after the deliverable of the interim results, the analysis files had to be resent because of some data updates in the interim database. The team solved the obvious problem of maintaining two sets of programs in parallel by copying the complete Interim Analysis folder structure into a subfolder. Because special databases needed to be developed for the data monitoring committee reports and data safety update reports, complete separate folder structures were stored in the data folder. The medical review reports on the other hand were placed in the listings folder, because of their deliverables mostly being listings, and also because the project was handed over to another programming team (the original lead programmer left the company, and the team had to work on other projects).



A new lead programmer will find the following set of AUTOEXEC files when being asked to take over the study. The standard name is obviously AUTOEXEC.SAS, but by their names there are some special purpose files identifiable, such as a specific autoexec file for publications (Autoexec_Publications.sas), as well as one for DMC reports (Autoexec_DMC.sas). Others seem to be copies made by different programmers (Autoexec_HF.sas and Autoexec_AXC.sas), and there is

even a specialized autoexec file for medical review listings (autoexec_MR.sas), but as there was a programmer on the team with these initials, it is not easily decidable if this is a personalized autoexec file or not. Other deliverables seem not to be covered by an own AUTOEXEC file.

The picture below shows a mapping of the resulting folder structure.



Even if it is comprehensible how the folder structure for this study originated, there are some problems associated with it:

1. Different types of deliverables are not consistently incorporated into the study folder structure, which results in
    a. Sub-folder structures for DMC and DSUR reports can be found in the data folder. The subfolder structures themselves follow different organization patterns and the naming of the subfolders is inconsistent.
    b. Subfolder structures for medical review listings are stored in the listings folder.
    c. Publications folders and Interim Analysis folders are created in the top folder layer of the study
    d. Although there are separate sets of folders for every other type of deliverables, there is no separate structure for the final deliverable.
    e. As mentioned before, folder structures are without reason inconsistent across the different types of deliverables.
2. The differences in folder structures require multiple AUTOEXEC files in order to keep the files simple.

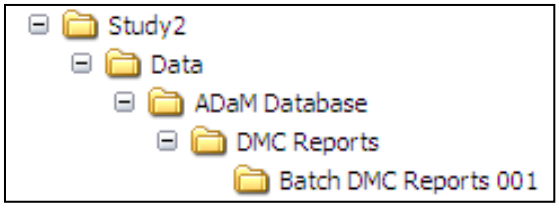
3. The naming of the AUTOEXEC files is not consistent, thus their purpose remains unclear. Also, some autoexec files are set up to work for more than one deliverable type, but this is not reflected in their names.

THE SOLUTION: IMPLEMENTING MULTIPLE DELIVERABLES SYSTEMATICALLY BY USING ANALYSES AND BATCHES

How can this be addressed in a systematic way? What we need is a way to organize the different types of shipments, and a mechanism to deal with periodic updates:

Analysis covers the different types of deliverables. There can be a list of different deliverable types pre-specified. In the case of the previous studies this could be: "Final Analysis", "Interim Analysis", "DMC Reports", "Medical Review Reports", "DSUR Reports", "Publication" and "Client Requests".

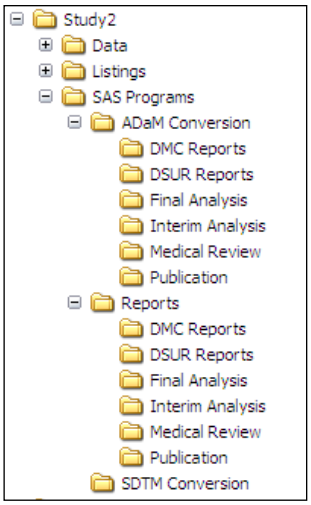
Batch covers the periodical updates. Batches are different sets of output files of a specific deliverable over time. They can be simply named using a consecutive number; in our examples batches are named using the following pattern: "Batch <Analysis> <consecutive number in z3. style>" (e.g. "Batch Interim Analysis 001" or "Batch DMC Reports 002").

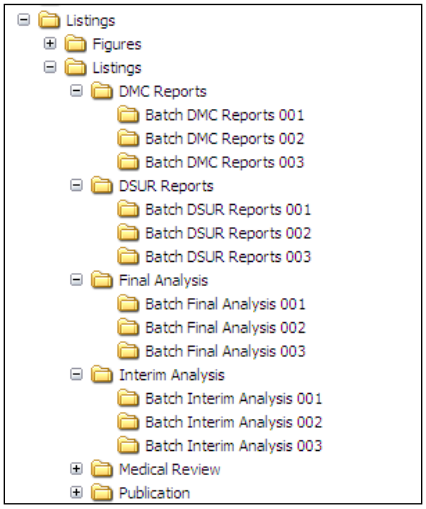


Different types of deliverables are implemented into the standard folder structure by creating subfolders in every standard folder with the name of the respective deliverable. Each batch that is produced for every deliverable is stored as a sub folder in the folder for the respective deliverable type.

SAS® Programs not necessarily change with each update of study data and reports. Thus, they not necessarily need to be stored independently for each new batch. Instead, programs can be maintained using a versioning tool.

For the SDTM conversion programs, we assume that this database will support all types of deliverables and will not change. Thus there are no deliverable specific versions of the SDTM conversion programs, and thus no subfolders in the "SDTM Conversion" folder.





For reports, this is how the folder structure would look like for three shipments of every type of deliverable. In a real study this would of course look differently, as shipment cycles are rarely synchronized across multiple output types. Please note that the folder structure looks similarly for medical review and publication deliverables; these folders were collapsed simply to save space.

Also, the folder structure looks similar for ADaM datasets, as there might be a different database structure required for every type of deliverable. For the SDTM database we assume that it stays the same for all deliverables, thus only different batches of the database need to be stored for each shipment.

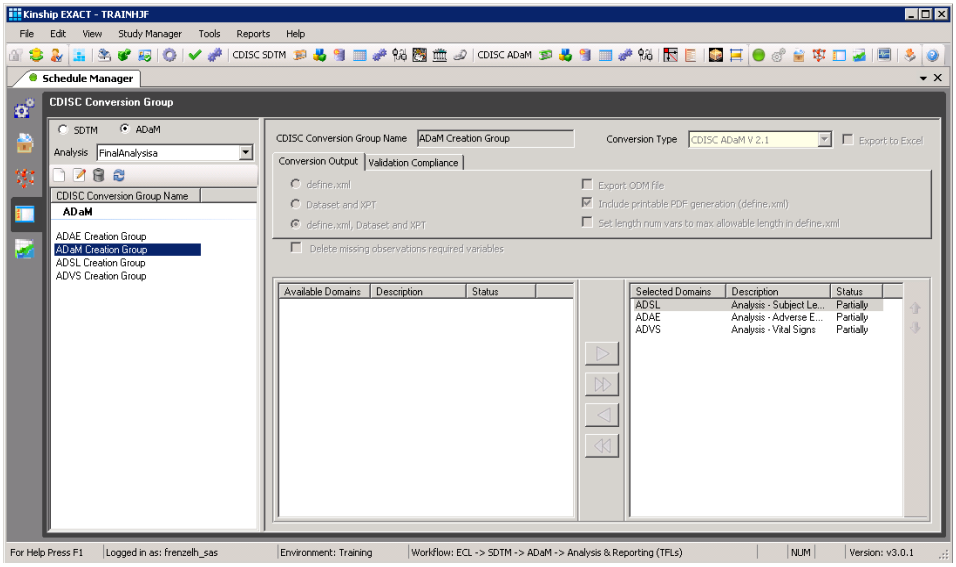
As this structure is consistent across multiple types of deliverables, it also simplifies the AUTOEXEC file and makes multiple versions of this file for multiple types of deliverables unnecessary. The following picture shows a simple AUTOEXEC file that covers all different deliverable types that are mentioned in the study above:

```
/************************************************************************
- - - - - - - - - - - - - DO NOT EDIT THE NEXT 4 LINES  - - - - - - - - - - - - -
PROGRAM NAME: $RCSfile: autoexec.sas,v $
REV/REV AUTH: $Revision: 1.02 $ / $Author: FrenzelH_SAS $
REV DATE    : $Date 2013/10/13 08:12:55 $   GMT
- - - - - - - - - - - - - DO NOT EDIT THE 4 LINES ABOVE - - - - - - - - - - - - -
CLIENT/PROJ : www.Phuse.eu / Presentation
PURPOSE     : AUTOEXEC - Provide global settings and access to file sytem
INPUT FILES : -
OUTPUT FILES: -
MACROS USED : -
NOTES       :
            :
Copyright (C) 2013  PRA International   All Rights Reserved.
************************************************************************/

*** study location on the file servers *** ;
%global Servier Client Project StudyRoot;
%let Server = FILESERVER ;
%let client = PhUSE Brussels 2013 ;
%let study  = Study 2 ;
%let StudyRoot = \\&Server.\&Client.\&Study. ;

*** Analysis and Batch *** ;
%global Analysis Batch BatchID ;
%let Analysis = Interim Analysis ;
%let Batch = 001 ;
%let BatchID = Batch &Analysis &Batch ;

*** SAS Libraries *** ;
libname raw      "&StudyRoot.\Data\Raw\Study Database" access=readonly ;
libname ECG      "&StudyRoot.\Data\Import\ECG Data" access=readonly ;
libname IVRS     "&StudyRoot.\Data\Import\IVRS" access=readonly ;
libname SDTM_DB  "&StudyRoot.\Data\SDTM Database" ;
libname ADaM_DB  "&StudyRoot.\Data\ADaM Database\&Analysis\&BatchID" ;

*** Output locations *** ;
%let figures  = &StudyRoot.\Listings\Figures\&analysis.\&BatchID ;
%let listings = &StudyRoot.\Listings\Listings\&analysis.\&BatchID ;
%let tables   = &StudyRoot.\Listings\Tables\&analysis.\&BatchID ;

*** global settings ***;
filename macros "&StudyRoot.\SAS Programs\Macros" ;

libname smacros "&StudyRoot.\SAS Programs\Macros\StdCat" ;
libname dmacros "&StudyRoot.\SAS Programs\Macros\DBExt" ;

libname allmacro (smacros dmacros) ;

options merror
        mautosource
        mstored
        mprint
        notes
        symbolgen
        nobyline
        sasmstore = allmacro
        sasautos  = (macros, sasautos)
        fmtsearch=(work raw SDTM_DB ADaM_DB) ;
```

The resulting folder structure can become quite complex, but it still is easy to understand because it is consistent for all deliverables of a study. It might be worth developing tools to automate the creation of sub-folders, or to archive the contents of a shipment from all the different sub-folders into one zip-file automatically. These tools are generic in a way that they can be used for almost every study that is set up with this structure, so it might be worth putting some effort into this.

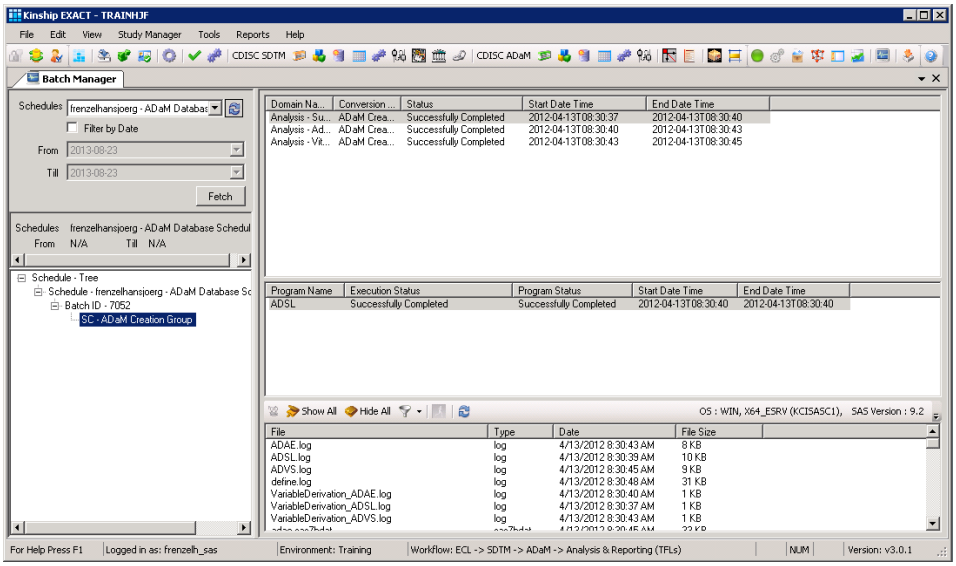
HOW EXACT USES ANALYSES AND BATCHES TO MANAGE MULTIPLE DELIVERABLES

EXACT is the tool used at PRA International to create SDTM and ADaM databases and reports to support different aspects of clinical trials. For each study within EXACT one SDTM database created. Depending on the kind of deliverables required for the study, one or multiple analyses are set up, and under each of these analyses a deliverable specific ADaM database as well as specific tables, figures, and listings can be created.



Once conversions from SDTM to ADaM have been configured, all programs that make up the analysis database can be combined in one CDISC Conversion Schedule. It can also be controlled whether a define.xml and an openCDISC validator report is created for the datasets in this schedule. In addition to all the conversion configurations also the schedule is stored under the same analysis it is created for. Schedules similar to this can be created for all tables, figures, and of a certain deliverables type.

For any production run, this schedule is then executed via the Scheduler, a tool to submit production runs to PRA's SAS connect servers. The Scheduler takes care of tracking the batch ID (a unique number for each submitted batch), of creating the necessary sub folders to store the result files and of executing all the programs and reports that are configured in the schedule.



The results from the schedule can be viewed via the Batch Manager. Without knowing the underlying folder structure, programmers can see all outputs, corresponding log files, reports, the define.xml as well as all programs that were created or executed for that schedule. They will not by accident review the wrong output, or the wrong log files. In addition they can view a report that parses the log files for unwanted messages, and openCDISC Validator reports for the databases that were created. Finally, they can store all components of a schedule in a zip file which is ready to be sent to the client.

CONCLUSION

Clinical Studies often have multiple types of deliverables, some of which are planned and documented in the clinical protocol, others which are unplanned and come up at a time. Multiple types of deliverables can lead to difficult to understand folder structures, especially if multiple shipments go hand in hand with them. Organizing deliverables in Analyses and Batches allows maintaining a complex but easy understandable folder structure independently of the deliverables that need to be created, a folder structure that serves fine for most of the programming endeavors in clinical programming. PRA uses EXACT to organize all SAS related programming activities around clinical trials. Using analyses and batches, EXACT provides interfaces to view all parts of a deliverable in one place, without the need to access the underlying folder structure, thus making it even easier for programmers to handle the different types of deliverables for a clinical study.

CONTACT INFORMATION

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

Hansjörg Frenzel
PRA International
Gottlieb-Daimler-Str. 10
Mannheim / 68165
Work Phone: +49 621 878 2328
Fax:
Email: frenzelhansjoerg@praintl.com
Web: praintl.com