How to make SAS® Drug Development more efficient
Xiaopeng Li, Chun Feng and Peng Chai
Celerion, 621 Rose Street, Lincoln, NE 68502

ABSTRACT
SAS® Drug Development (SDD) is a web-based SAS system on UNIX operating system. Internet status, limited abilities to run multiple processes at a time and some SDD unique features take more time to operate a program in SDD. To run SDD efficiently is important for accomplishing tasks for programmers. In this paper, several methods that can make SDD more efficient will be examined.

INTRODUCTION
SAS® Drug Development (SDD) is a web-based UNIX operating system with a user interface platform. SDD is developed by SAS for efficient execution and management of pharmaceuticals and analysis and reporting activities to support clinical research in drug development. It contains some programming-friendly features such as the parameter window and Scheduler. Several pharmaceutical and biotechnology companies, as well as clinical research organizations (CRO) use SDD to analyze and report clinical study data. Reading and storing information in SDD takes more time due to the web-based distance transmission. To help programmers complete tasks in less time, several methods for improving SDD efficiency are discussed.

METHODS
Calling in appropriate macros and data sets from the parameter window
In SDD, calling in macros and data sets are different from PC SAS®. In SDD, the parameter window can define and locate macros and data sets (Display 1). Since SDD is a web-based system, executing programs requires sending and receiving information to the server. The more data and codes are involved, the longer it will take to execute a program. When executing a program, eliminating useless data, macros and codes can improve SDD efficiency.

To define the location of SAS data sets in the SDD parameter window, the location should be set as a folder. In the SDD parameter window, there are three kinds of folders: “Input”, “Output”, and “Input and Output”. The “Input” folder defines the location of a folder for data used in a program. One or more data sets from the defined folder can be read into SDD when a program is executed. The “Input” folder is defined as “Get all files”, all the SAS data in the defined folder will be read into SDD. The “Input” folder is defined as “Get selected files”, only selected files in the defined folder will be read into SDD (Display 2). Display 2. Input folder with “Get all files” option (left) and Input folder with “Get selected files” option (right)

The data being used should be clearly defined to avoid missing data or reading in data multiple times. In addition, avoiding calling data that will not be used in a program helps SDD work more efficiently.

Macros in SDD can be called into a program two ways. First, similar to SAS data sets, macros can be called from a defined folder. It is more efficient to call only the macros that will be used in the program. Second, macros can be called as input files. As shown in Display 3, the name and location of an input file are referenced in the parameter window. The second way is more efficient and preferable.

Display 3. Output folder
Display 4. Input and Output folder

Macros in SDD can be called into a program two ways. First, similar to SAS data sets, macros can be called from a defined folder. It is more efficient to call only the macros that will be used in the program. Second, macros can be called as input files.

Display 5. Input file

Using Job Editor
Job Editor, a unique feature in SDD (Display 6), can execute all the programs together instead of opening all the programs one by one.

Display 6. Job Editor

Using Scheduler
Jobxml, a unique feature in SDD (Display 7), can run a set of programs in order. The Scheduler can define a time to execute one or more Job Editors. It can also define the frequency required to execute the Scheduler, such as once, daily, weekly, and monthly. It therefore makes SDD more efficient, especially for programs requiring long execution time.

Display 7. SDD Scheduler

While SDD is limited to execute one program at one time, using the Scheduler allows more programs to be handled simultaneously. It is preferable to use the Scheduler to execute programs that are ready, and require long execution times. An additional benefit is that another program can be modified and executed, at the same time.

CONCLUSION
In conclusion, calling in appropriate macros and data sets from the parameter window, using Job Editor, the Scheduler, and appropriate data sets, can make SDD more efficient. Programmers can select a combination of these methods to complete SDD tasks in less time.

REFERENCES

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the author at:
Xiaopeng Li
Celerion Inc.
621 Rose Street
Lincoln, NE 68502
402-437-6280
xiaopeng.l@celerion.com

Chun Feng
Celerion Inc.
621 Rose Street
Lincoln, NE 68502
402-437-6280
chun.feng@celerion.com

Nancy Wang
Celerion, Inc.
621 Rose Street
Lincoln, NE 68502
402-437-4850
nancy.wang@celerion.com

Available at http://www.cdisc.org