MWSUG 2018 – Paper HS132

Toward Adoption of Agile Software Development in Clinical Trials

Troy Martin Hughes

ABSTRACT

Agile methodologies for software development, including the Scrum framework, have grown in use and popularity since the 2001 Manifesto for Agile Software Development. More than having obtained ubiquity, Agile has demonstrably defined software development in the 21st century with its core foci in collaboration, value-added software, and flexibility gained through incremental design, development, and delivery. Although Agile principles can easily be extrapolated to other disciplines, Agile-related nomenclature, literature, application, and employment descriptions often remain focused on software development alone. The clinical trials/pharmaceutical industry often embodies *data analytic development*—software development in which the ultimate business value is derived not from software but from data products and data-driven decision making. Because these outcomes are more highly valued than their underlying software, the pharmaceutical and clinical trials industry is more likely to focus on research and analytics than the software that underpins these endeavors, and more likely to invest effort in implementing analytic best practices than in software development best practices. Because of this shift in focus, clinical trials organizations—despite their need to build high-quality, reliable, enduring software—are less likely to implement Agile principles, methods, and artifacts than organizations building similar software in more software-focused industries. This text introduces Agile software development for use in clinical trials organizations. Moreover, the paucity of reference to Agile or any software development life cycle (SDLC) or development methodology within the pharmaceutical industry is demonstrated through examination of SAS® user-published white papers, SAS Institute books, and more than 15,000 pharmaceutical employment postings.

INTRODUCTION

The Manifesto for Agile Software Development (aka, the Agile Manifesto) was cultivated by a group of seventeen software development gurus who met in Snowbird, Utah, in 2001 to elicit and define a body of knowledge that prescribes best practices for software development. The Agile Manifesto states:

> We are uncovering better ways of developing software by doing it and helping others do it.
>
> Through this work we have come to value:
>
> - Individuals and Interactions over processes and tools
> - Working software over comprehensive documentation
> - Customer Collaboration over contract negotiation
> - Responding to change over following a plan
>
> That is, while there is value in the items on the right, we value the items on the left more. [i]

In addition, twelve Principles of Agile Software[ii] augment the Agile Manifesto and are espoused by methodologies that embrace Agile, including Scrum, Lean, Extreme Programming (XP), Crystal, Scaled Agile Framework (SAFe), Kanban, and others. Agile has been instrumental in accelerating and transforming software development teams and has largely supplanted the classic Waterfall methodology in the software development arena. Major corporations such as Amazon[iii], Yahoo[iv], Marriott[v], and even SAS[vi] have demonstrated the revolutionary impact of Agile on information technology (IT) innovation and implementation. Within the federal government, the Government Accountability Office (GAO) in a multi-agency study identified "modular development" of Agile as a best practice[vii], while agencies including the Department of Defense (DoD)[viii], Federal Bureau of Investigation (FBI)[ix], Department of Veterans Affairs[x], Department of Homeland Security (DHS)[xi], and others have validated the role of Agile in positively transforming fledgling or failing IT projects.

Although the benefits of Agile are clear and convincing, its application and adoption across the software development community has not been without bias. Agile-related nomenclature, training, literature, and employment postings often depict an idealized "developer" archetype who codes incessantly in one or more object-oriented programming (OOP) languages, whose value is measured by the software he creates, and who may or may not have an unhealthy addiction to zombies, Zork, and popping zits that have never seen the light of day. Stepping away from stereotypes, *actual* software developers represent a much broader segment of society and include many professionals who might not even embrace the "developer" community or title. Biostatisticians, statistical programmers, data analysts, financial analysts, clinical researchers, data scientists, physicians, psychologists, and an abundance of other professionals instead often view software development not as endpoint, but rather as a means to that end—a tool to support the empirical quest

for truth. Thus, these professionals—despite constituting both primary and ancillary developer roles—may be less likely than so-called "traditional" developers to gravitate toward the software-centric, unnecessarily narrow interpretations of Agile that unfortunately persist.

Because SAS practitioners embody such a diverse array of roles in equally diverse organizations and industries, they often derive business value not from code they create but rather from derivative data products—including data sets, analyses, analytic reports, and data-driven decisions. And, because they possess not only this technical expertise but also the business savvy and domain expertise to further transform data into information and knowledge, they can more flexibly structure and adapt software to fulfill ultimate business intent. This end-user model, in which SAS practitioners are both the author and consumer of their code, can benefit a team or organization because of the power and flexibility it provides. However, a potential vulnerability of end-user development is a tendency for practitioners to focus on growing their *domain* knowledge and skills to the detriment of their *technical* knowledge and skills. Thus, a clinical researcher whose responsibilities have evolved into building an enduring SAS infrastructure while administering his department's SAS server should be commensurately growing his technical knowledge toward established systems and software engineering best practices—including software development methodologies (like Agile) that maximize his team's development environment and potential.

Thus, software development best practices include not only *technical* solutions—such as the efficient use of hash joins, modular code, or software that can be extensibly repurposed with ease—but also *process* solutions that describe software development methods, principles, and environments that most engender success. Software developers whose education, training, and experience lie in software engineering, systems engineering, computer science, or similar disciplines are more likely to be aware of, already embracing, and benefitting from Agile development methodologies and principles. Notwithstanding this inherent advantage, cross-functional developers who occupy diverse roles throughout pharmaceutical, clinical trials, and research organizations should also explore Agile methods to understand the benefits they can provide.

The fear, however, is that they do not. This text demonstrates an alarming trend that SAS professionals are less likely to espouse Agile, as demonstrated by clinical trials job postings, SAS user-published white papers, and SAS, Inc. books that reference clinical trials and healthcare. While no further introduction to Agile, its nomenclature, artifacts, or implementation is provided in this text, a separate text by the author introduces its more basic tenants, processes, and benefits to data analytic-focused developers and organizations: *When Software Development is Your Means Not Your End: Abstracting Agile Methodologies for End-User Development and Analytic Application*.[xii] Moreover, for a more traditional and software development-centric introduction to Agile, a bevy of excellent publications, trainings, and open source material exists to guide organizations unabashedly toward success.

CLINICAL TRIALS SAS JOB DESCRIPTIONS

A breadth of positions falls under the vast clinical trials/pharmaceutical umbrella, comprising all levels of technical expertise. Studies require planning, design, administration, certification, oversight, and auditing—to name a few activities—and often persist for years. At the heart of studies are enrollment data, biographical data, treatment data, outcomes data, and other data that must be collected, cleaned, transformed, validated, analyzed, interpreted, visualized, documented, and archived. This is to say that even among the subset of clinical trials professionals that utilize SAS software, great discrepancies in technical skill and usage will always exist. At one extreme lie the hardcore SAS software developers—they may have neither interest in nor domain knowledge of clinical trials (despite working in the industry), yet know the ins and outs of SQL, SAS macros, and may even accidentally include gratuitous semicolons in text messages.

At the other end of the spectrum lie researchers, clinicians, and other users of SAS who may be comfortable clicking their way through SAS Enterprise Guide menu selections, but who would never consider themselves to be “programmers” or “developers.” And across this wide spectrum are clinical trials coordinators, clinical research associates, biostatisticians, auditors, software developers, system architects, physicians, pharmacologists, nurses, and others, all of whom may be required to have varying knowledge of SAS. Thus, although it would be unfair to expect all SAS users within the clinical trials industry to adopt or even benefit from Agile, a large percent of practitioners could benefit from its methods and principles, including their application to software development as well as broader extrapolation to other (i.e., non-software) project development initiatives and activities.

A 2014 study by the author examined a five-month cohort of more than 80,000 employment postings from the popular job site Indeed.com that demonstrated significant differences between positions mentioning data analytic languages (SAS, SPSS, or R) and OOP third generation languages (3GLs) Java or Python.[xiii] Indeed.com was sampled due to its vastness, open interface, and direct access to data via Python scripts. Data from the ten most populous cities in the US (New York City, Los Angeles, Chicago, Houston, Philadelphia, Phoenix, San Antonio, San Diego, Dallas, and San Jose) were included in the study. In sum, although approximately 25 percent of Java and Python job postings referenced Agile, less than 4 percent of SAS positions did. These results were unfortunate given the tremendous

advantage that Agile can yield to developers within the SAS community, but not altogether unexpected given the focus within the Agile community on OOP languages. *Note that this 2014 study did not isolate the clinical trials industry.*

A second study collected data over a two-month period from February 1 through March 31, 2015. All job descriptions posted in the US that included the word "SAS" were extracted, comprising 15,702 unique positions (by Indeed.com job ID as well as SHA-256 hash checksum). An additional 1,884 jobs had been eliminated due to duplicate hash values, which often appeared when companies would repetitively post an identical job, and 399 jobs had been removed because they did not reference SAS software but rather other instances of "SAS." For example, some of these postings referenced constructs such as "school administration/supervision", "second avenue subway", or "space and airborne systems."

Of the 15,702 positions that were verified to reference SAS software, 22.3 percent (N=3,495) had either "clinical" or "pharmaceutical" (or derivatives) in the job description, with the most common job titles including variations of:

- clinical data manager
- clinical programmer
- clinical data associate
- clinical data analyst
- clinical informatics manager
- lead clinical data manager
- clinical SAS programmer
- clinical project manager
- senior clinical data manager
- contract clinical SAS programmer

Although a job title is never a perfect proxy for a verbose employment description (just as a job posting rarely perfectly represents the *actual* employment responsibilities, requirements, or environment), these job titles and others demonstrated construct validity with positions found in the clinical trials industry. Thus, all subsequent analyses were performed on this data subset. As a caveat, this subset may include positions that only glancingly mention SAS. Future steps to refine this model, thus, should incorporate a measure of "SASiness" sufficient to demonstrate the degree to which SAS is central to the position. For example, a "SASier" position might reference "SAS" in the job title, or otherwise include multiple references to SAS, specific SAS modules, or SAS certifications that would be required for employment.

Of the 3,495 positions, only 2.3 percent (N=79) mentioned Agile, a lower percent than was found in the 2014 study, but otherwise consistent with the proclivity that SAS-related positions do not mention the software development life cycle (SDLC). Because so few clinical trials SAS positions did reference Agile, no further conclusions about this subpopulation were made, such as whether geography, job title, or other responsibilities or requirements influenced the presence or absence of Agile. This underscores the necessity to draw from a larger sample in the future to obtain this granularity with confidence and while minimizing sample bias. For example, given the tendency for Agile nomenclature, literature, and training to present the traditional "developer" archetype mentioned previously, one would expect SAS positions in clinical trials whose job titles most mimicked this archetype (e.g., developer, software engineer, programmer, systems architect) to espouse Agile more than other cross-functional, traditionally less technical roles.

Only Agile-related phrases that unambiguously referred to the software development methodology (i.e., big-A Agile) were included. Thus, four positions that ambiguously mentioned "agile" (i.e., little-a agile) were excluded because they more generically referenced "flexibility." Thus, although "agile teams" or "agile environment" were excluded if no other mentions of Agile were present, most references to Agile (N=79) were included. For example, some “Agile” text from various postings includes:

- good software development practices (SDLC, Agile ETC), peer reviews, unit testing, and other software quality practices
- experience with Agile development methodologies a plus
- participate in daily SCRUM meeting and support the AGILE methodology
- strong understanding of software development methodologies (RUP, Agile, or other SDLC)
- development efforts using Agile development frameworks such as Scrum or Kanban
- familiar with agile software development lifecycle and quality management

An additional 3.8 percent (N=132) of positions that did not reference Agile did encouragingly reference the SDLC or software development methodology. Thus, approximately 6 percent of the clinical trials positions referenced the SDLC

in some form. Although this does not denote that the remaining 94 percent of positions develop software in a vacuum or devoid of process methodology, one must wonder what methodologies or principles are espoused in those remaining positions and their respective organizations, and why no mention is warranted in job postings.

To that end, the inclusion of information about a particular life cycle, software development methodology, or project management philosophy espoused by an organization or required by a position can tremendously benefit the hiring process. Moreover, in more technical billets that do require some level of software development, a description of development activities is warranted and should include quality assurance aspects of the SDLC, such as code reviews, software integration, software testing, and verification and validation against stated software requirements. Including this pertinent information demonstrates a commitment to software quality, software industry best practices, and the expectation and intent to hire stellar candidates. And, regardless of job title, role, or responsibilities but commensurate with degree of software development focus and activities, developers, end-user developers, and SAS practitioners of all flavors should understand and embrace SDLC and software development best practices.

AGILE IN SAS USER-PUBLISHED LITERATURE

For decades, SAS software technical advancements and best practices have been promulgated by creative, forward-leaning practitioners who author and present white papers at regional and national conferences. These publications have tended to focus on a mix of SAS technical best practices and domain knowledge—in the case of PharmaSUG, specific to all things clinical or pharmaceutical.

Despite the breadth of topics in these publications, an often-overlooked niche is the clinical trials software development environment itself, including the SDLC and software development methodologies or principles that are or should be upheld. Thus, although Agile has been chronicled in SAS white papers for more than ten years, the publications are sparse, the mentions are few, and the recommendations lack the same vigor, tenacity, and shameless conviction that Agile enthusiasts espouse when writing about OOP languages. Nevertheless, they represent a consistent trend that represents hope for the industry.

The following white papers, in descending order of publication date, reference Agile principles as applied within the clinical trials community:

- ***SAS in clinical trials – A relook at project management, tools and software engineering*** (2014)[xiv]. Nandigama notes that as SAS moves from "standalone application" instances to an integrated environment, the project management methodologies and tools must commensurately increase in rigor. Software development methodologies are described, including the Waterfall, Agile, and Scrum. In concert with these competing theories, Nandigama depicts the use of JIRA software for issue, change, and risk management tracking and modeling. The woes of not espousing a standardized approach to enterprise SAS development are also described, including technical debt that can accumulate when low priority tasks are delayed.
- ***Kaizen*** (2014)[xv]. Kumar describes the process and production methodology Kaizen (translated *change for better*) that espouses continuous quality improvement. Although not mentioned in the text, Agile methodologies often espouse both Kaizen and Lean principles of maximizing productivity through minimization of waste. The "Five-Why" method to elicit root cause analysis in risk management and resolution also is discussed.
- ***Statistical Computing Environment Implementation – An Agile Approach*** (2013)[xvi]. Cozzolino describes the plight of pharmaceutical companies shifting toward enterprise solutions managed in statistical computing environments (SCEs). The paper examines these advancements from the business process perspective but considers "process and technology together," thus demonstrating benefits that iterative methodologies such as Agile can provide.
- ***SAS® Data Management Techniques: Cleaning and transforming data for delivery of analytic datasets*** (2013).[xvii] Schacherer explores definitions of and approaches to data quality. He describes practices that increase the robustness of SAS code, for example, by dynamically terminating a process and emailing an administrator if a prerequisite process has failed. Automation is also discussed with an example utilizing the Windows scheduler presented. In closing thoughts, he states his ambivalence toward specific software development methodologies—listing Agile and the Waterfall—and instead states that clear communication is paramount to success, a tenet central to Agile.
- ***Building a Controlled Statistical Programming Environment*** (2013)[xviii]. Woo only mentions Agile in a couple paragraphs and never takes a stance for or against Agile or the Waterfall. More importantly, however, he champions standardized development processes that uphold principles of Agile. He describes the SDLC and the advantages that prescribed processes can afford. He mentions that "Formalizing the programming process requires integrating the SDLC." This is critical because a development environment that lacks consistency will introduce variation, the antithesis of reliability. To that end, he describes configuration management, version management, formalized software testing, and integration within a production environment.
- ***Good Programming Practices in Healthcare: Creating Robust Programs*** (2012[xix]). Nelson and Zhou include 17 best practices amassed from their experiences. Themes such as modularity, robustness, and reliability persist

through this text. In addition to software best practices, they also enumerate software development methodologies and best practices, including test-driven development (TDD), Agile, Extreme Programming, Six Sigma, and defensive programming. Thus, they focus not only on the quality of the code, but moreover on the quality of the coding environment. It provides a great introduction to some of these schools of thought to SAS practitioners not otherwise exposed to these methodologies or methods.

- ***Unit Testing as a Cornerstone of SAS® Application Development*** (2011)[xx]. Di Tommaso and Hoffmann-La Roche describe the importance of unit testing and automated testing, stating that while ad hoc or manual tests suffice for end-user development, this is "inappropriate for programs and results intended for others." They present a standardized framework for unit testing that provides a generalizable, reusable capability with consistently formatted developer feedback.
- ***Navigating a Major Regulatory Submission Project to Success*** (2007)[xxi]. Meier describes the undertaking involved in assembling a team and charting a path toward regulatory submission success. While the majority of the paper details logistics involved in team setup and organization, a single paragraph references that daily standup meetings—borrowed from the Scrum framework—were implemented and were highly successful in helping the team meet competing priorities and rapidly approaching deadlines for technical implementation.
- ***Drawkcab Gnimmargorp: Test-Driven Development with FUTS*** (2006)[xxii]. While Wright never mentions Agile, test-driven development (TDD) lies at the heart of this paper and is demonstrated through automatic testing paradigms. Thus, rather than building code first and testing later, tests are written first—often in SAS macro code—and actual code only subsequently is written. The paper also contrasts data analytic development, thus distinguishing SAS from many other languages, stating "SAS programs typically crunch through large amounts of data, perhaps in complex and arcane structures, and perhaps creating more data as output."
- ***Using Extreme Programming Processes in a SAS Environment*** (2005)[xxiii]. Mitchell introduces Extreme Programming concepts, nomenclature, and benefits in support of an FDA-focused SAS initiative. Despite being a succinct overview, the paper introduces concepts such as compliance testing against customer requirements, refactoring, coding standards, and continuous integration of new code into the enterprise code base, thus incrementally delivering business value.
- ***SASUnit: Automated Testing for SAS*** (2004)[xxiv]. Nelson introduces the SASUnit system that monitors and validates clinical research code, thus providing "testability, robustness, and manageability" to the software development process. The framework follows test-first design principles espoused in some Agile methodologies, such as Extreme Programming. More than introducing SASUnit, however, the paper emphasizes the importance of fault-tolerant code, event and error detection through exception handling, and the importance of automated and other testing paradigms.

AGILE IN SAS INSTITUTE LITERATURE

A private software company such as SAS, Inc. admittedly might be hesitant to espouse Agile in its publications as this could potentially bias readership, especially among readers who were either unfamiliar with or opposed to Agile, or among readers who were interested only in its software. In contrast, open source communities such as Python and Java readily and consistently support Agile methodologies in published work. Although the majority of SAS books are published by the SAS Institute, technical publications on open source software enjoy a freer market, and authors readily interweave SDLC processes and best practices throughout their writing to the benefit of developers.

The glaring irony, however, is that SAS, Inc. itself uses Agile methodologies to build the software we all love so much! In Tim Arthur's *Agile Adoption: Measuring its Worth*, he describes the successful transition to Scrum that SAS underwent in 2007 which ultimately led to the development of the SAS Agile Framework, the SAS in-house implementation of Agile. Thus, with recognition and demonstration of the tremendous success of Agile within SAS, Inc., it's a mystery why so few SAS publications reference Agile or even the SDLC. A review of SAS Institute books covering clinical trials follows, with only two publications referencing Agile or the SDLC.

- ***SAS Programming with Medicare Administrative Data, 2ed*** (2014)[xxv]. Readers may be deterred by the somewhat restrictive title referencing Medicare, but Gillingham presents cogent arguments toward the pursuit of quality code and a quality software development environment. Despite only mentioning Agile once to reflect that Agile "facilitates changes during the development of code," he devotes sections toward planning, designing, and operationalizing a comprehensive SDLC approach. This includes aspects such as requirements generation, the role of quality assurance, and unit testing. While not the focus of the book, his depiction of and dedication to quality is a refreshing addition.
- ***SAS Programming in the Pharmaceutical Industry, 2ed*** (2014)[xxvi]. This publication as well only mentions Agile once. However, Shostak proclaims to readers, "I recommend that you follow a more traditional software development life-cycle model (SDLC) when developing a comprehensive, SAS macro-based reporting system. Unlike much of the one-time-only SAS programming that occurs for a clinical trial, you need to ensure that a general-purpose SAS macro system is robust enough to handle any problem that it encounters." He adds, "If you get into the development of software systems, then learning about software development methodologies would be worth your time." Kudos for raising the bar!

The remainder of clinical trials texts that were examined, although providing excellent technical and domain information and best practices, make no mention of the software development environment or software development methodologies such as Agile.

- ***Risk-Based Monitoring and Fraud Detection in Clinical Trials Using JMP and SAS*** (2014)[xxvii]
- ***Implementing CDISC Using SAS: An End-to-End Guide*** (2012)[xxviii]
- ***Common Statistical Methods for Clinical Research with SAS® Examples, 3ed*** (2010)[xxix]
- ***Analysis of Observational Health Care Data Using SAS*** (2010)[xxx]
- ***Pharmaceutical Statistics Using SAS®: A Practical Guide*** (2007)[xxxi]
- ***Analysis of Clinical Trials Using SAS®: A Practical Guide*** (2005)[xxxii]

CONCLUSION

That Agile is lauded as a best practice in software development is beyond refute and is demonstrated through hundreds of books and thousands of publications that span the last decade. And, despite the reliance of the clinical trials/pharmaceutical industry on SAS software development to drive and enable empirical research, a chasm seems to exist between Agile and its implementation within teams and organizations that develop SAS software. This disparity has been demonstrated through the analysis of clinical trials job descriptions, SAS user-authored white papers, and SAS Institute books, in the hopes of facilitating a bridge across this gap toward the union of SAS development and Agile methodologies.

REFERENCES

[i] The Manifesto for Agile Software Development. The Agile Alliance. Retrieved from http://www.agilealliance.org/the-alliance/the-agile-manifesto/.

[ii] The Twelve Principles of Agile Software. The Agile Alliance. Retrieved from http://www.agilealliance.org/the-alliance/the-agile-manifesto/the-twelve-principles-of-agile-software/.

[iii] Atlas, A. Accidental Adoption: The Story of Scrum at Amazon.com. *2009 Agile Conference*. IEEE. August 2009, pp. 135-140.

[iv] Chung, M. Agile at Yahoo! From the Trenches. *2009 Agile Conference*. IEEE. August 2009, pp. 113-118.

[v] Fewell, J. Marriott's Agile Turnaround. *2009 Agile Conference*. IEEE. August 2009, pp. 219-222.

[vi] Arthur, T. Agile Adoption: Measuring its Worth. *SAS Institute*. 2013. Retrieved from http://support.sas.com/rnd/papers/2013/AgileAdoptionPaper.pdf

[vii] US Government Accountability Office. Report to the Subcommittee on Federal Financial Management, Government Information, Federal Services, and International Security, Committee on Homeland Security and Governmental Affairs United States Senate. Software Development: Effective Practices and Federal Challenges in Applying Agile Methods. GAO-12-681. July, 2012. Retrieved from http://www.gao.gov/products/GAO-12-681

[viii] Broadus, W. The Challenge of Being Agile in DoD. *Defense Acquisition, Technology, and Logistics.* January-February 2013.

[ix] Fulgham, C., Johnson, J., Crandall, M., Jackson, L., Burrows, N. The FBI Gets Agile. Federal Bureau of Investigation*. IT Pro*, September/October 2011.

[x] Thibodeau, P. Budget Woes Force Feds to Embrace Agile. *Computer World*. February 11, 2013.

[xi] Thibodeau, P. DHS Shifts to Cloud, Agile Development. *Computer World*. April 8, 2013

[xii] Hughes, Troy Martin. 2014. When Software Development is Your Means Not Your End: Abstracting Agile Methodologies for End-User Development and Analytic Application. *Western Users of SAS Software (WUSS).*

[xiii] Hughes, Troy Martin. 2014. Differentiating SAS Development from Third Generation Language (3GL) Object Oriented Programming (OOP): Lessons Learned from Indeed.com Employment Postings. *South Central SAS Users Group (SCSUG).*

[xiv] Nandigama, Sameera. 2014. SAS in clinical Trials – A relook at project management, tools and software engineering. 2014. *Pharmaceutical Users Software Exchange (PhUSE).*

[xv] Kumar, Usha. 2014. Kaizen. *PharmaSUG.*

[xvi] Cozzolino, Gary. 2013. Statistical Computing Environment Implementation – An Agile Approach. *PharmaSUG.*

[xvii] Schacherer, Chris. 2013. SAS® Data Management Techniques: Cleaning and transforming data for delivery of analytic datasets. *SAS Global Forum.*

[xviii] Woo, Wayne. Building a Controlled Statistical Programming Environment. 2013. *Western Users of SAS Software (WUSS).*

[xix] Nelson, Gregory S. and Zhou, Jay. 2012. Good Programming Practices in Healthcare: Creating Robust Programs. *SAS Global Forum.*

[xx] Di Tommaso, Dante and Hoffmann-La Roche, F. 2011. Unit Testing as a Cornerstone of SAS® Application Development. *Pharmaceutical Users Software Exchange (PhUSE).*

[xxi] Meier, Alan. 2007. Navigating a Major Regulatory Submission Project to Success. *SAS Global Forum.*

[xxii] Wright, Jeff. 2006. Drawkcab Gnimmargorp: Test-Driven Development with FUTS. *SAS Users Group International (SUGI) 31*.

[xxiii] Mitchell, Brian. 2005. Using Extreme Programming Processes in a SAS Environment. *SAS Users Group International (SUGI) 30*.

[xxiv] Nelson, Greg Barnes. 2004. SASUnit: Automated Testing for SAS. *PharmaSUG.*

[xxv] Gillingham, Matthew. 2014. *SAS Programming with Medicare Administrative Data, 2ed*. Cary, NC. SAS Institute.

[xxvi] Shostak, Jack. 2014. *SAS Programming in the Pharmaceutical Industry, 2ed.* Cary, NC. SAS Institute.

[xxvii] Zink, Richard C. 2014. *Risk-Based Monitoring and Fraud Detection in Clinical Trials Using JMP and SAS.* Cary, NC. SAS Institute.

[xxviii] Holland, Chris and Shostak, Jack. 2012. *Implementing CDISC Using SAS: An End-to-End Guide*. Cary, NC. SAS Institute.

[xxix] Walker, Glenn and Shostak, Jack. 2010. *Common Statistical Methods for Clinical Research with SAS® Examples, 3ed*. Cary, NC. SAS Institute.

[xxx] Faries, Douglas E; Obenchain, Robert; Haro, Josep Maria; Leon, Andrew C. 2010. *Analysis of Observational Health Care Data Using SAS.* Cary, NC. SAS Institute.

[xxxi] Dmitrienko, Alex; Chuang-Stein, Christy; D'Agostino, Ralph. 2007. *Pharmaceutical Statistics Using SAS®: A Practical Guide*. Cary, NC. SAS Institute.

[xxxii] Dmitrienko, Alex; Molenberghs, Geert; Chuang-Stein, Christy; Offen, Walter. 2005. *Analysis of Clinical Trials Using SAS®: A Practical Guide.* Cary, NC. SAS Institute.

CONTACT INFORMATION

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

Name: Troy Martin Hughes
E-mail: troymartinhughes@gmail.com