Open Source Statistical Software (OS^3) in Pharma Development: A case study with R

Anthony Rossini, David A. James

Modeling & Simulation

Novartis Pharmaceuticals

Outline

- Application context: Clinical development, Statistics, Modeling and Simulation (M&S)
- Regulatory Statistical Computing: GxP, 21CFR Part 11, others
- Open Source Statistical Software: Fears and benefits (real and imagined)
- Case Study: R
- Discussion

Disclaimer

- This talk represents opinions not necessarily shared by others at Novartis, and in specific, does not necessarily represent Novartis Clinical Quality, Corporate IT, Statistics, Development IT, or Legal opinions.
- It does provide a good-faith interpretation of comments from those groups.

Regulated Stat Computing (1/3)

"Good practices are good practices, but explicit descriptions which cover every reasonable approach are hard to write."

Most regulatory requirements concern data, audit trails, "explicit documentation that you know what you did and are doing".

Documenting work processes by:

- "Tamper-proof" logging
- Enforcement of SOPs and WPs

Regulated Stat Computing (2/3)

Regulations follow from a data-centric (computer-science-based) model view: the data changes, but the analysis is fixed and stays the same.

Confusing to statisticians who follow an analysis-centric model view: the data stays the same, but the analysis varies to address specific questions.

Pharma Development has a low tolerance for errors (they can have an extremely high cost; e.g. Cox-2 inhibitors).

Regulated Stat Computing (3/3)

GxP and 21CFR Part 11 are frameworks for “best practices”.

In a nutshell: It’s about documenting what you know you did, how you should have done it, and any discrepancies between them that occurred and why, for the reviewing health authorities.

Qualification and Validation of Software Systems (1/2)

- These are dependent on **institutional** (corporate) SOPs (external qualification and validation is an oxymoron).
- A system is qualified if it is well developed and supported (vendor audit), and passes IQ/PQ/OQ tests (see next slide). Definitions characterized externally, but specified internally.
- Validation is the above, plus completing tests to ensure required functionality.

Qualification and Validation of Software Systems (2/2)

A validation **plan** includes:

- User requirements (specifies OQ)
- Design/Functional specifications (specifies PQ)
- Testing:
  - Installation Qualification (IQ).
  - Performance Qualification (PQ).
  - Operational Qualification (OQ).

A validation **report** summarizes these results.

Open Source Software (1/2)

1. Generally **not** Public Domain (which is a legal term denoting unrestricted reuse, not necessarily disclaiming liability).
2. Released under a commonly understood open license (www.opensource.org, GPL, BSD, etc.) which describes terms of use of the software (applications, modifications, redistribution, …)

Note: Including the source does not make it open (e.g. NONMEM) and just because it’s free it doesn’t make it open either (WinBUGS).

Open Source Software (2/2)

In many ways, we are “back to the future”, as this approach is reminiscent of past practices in the 60s and 70s (vendors provided source code to clients, but under a strict license, i.e. current NONMEM practice)

The license is the **only** part that distinguishes Closed and Open Source Software. (not cost, not quality, not…).

Red Herrings & FUD* everywhere

The following are **not** unique to Open Source:

- Who is liable? (no one, for nearly all off-the-shelf/internet software)
- Who will fix problems? (not clear; who gets to define “problem”?)
- Quality Assessment/Management? (this is strictly in-house…)

FUD*: **Fear, uncertainty, and doubt** is a sales or marketing strategy of disseminating negative (and vague) information on a competitor's product. (Wikipedia, June 1st, 2007.)

Red Herrings & FUD everywhere

- Developer qualifications? (do you know who wrote the code? Do they know what they are doing? Are they aligned with your interests?
- Continued development or support? (companies disappear, professors disappear)
- If I have the source code, my associates could modify the program (maybe…)

These are common software problems.

Common Real Fears (1/2)

For most OSS projects the following can hold:

- Unclear release cycle rationalization. Why a release? Time-oriented or goal-oriented?
- Documentation. Software development methodology, QA/QC, release management, version control, design (pre/post). *“You have the code”,* but what if it’s unreadable?

Common Real Fears (2/2)

- Augment development/release cycle using in-house support to meet required in-house IT requirements. In-house competency? Competent out-source vendor?

There are OSS groups whose development procedures match the best practices for design and development.

There are commercial groups whose practices have been audited and found severely lacking.

Quality (1/2)

The ultimate issue is to ensure quality:

- Quality as in quality management (QM), not as in absolute quality
- QM relies on documented knowledge and behaviors, as well as enforcing these behaviors and principles.
- Quality from a business perspective includes interchangeability of humans, not just machines; documentation is intended to minimize loss of knowledge.

Quality (2/2)

- Quality is related to understanding and managing risks.
- If it was absolute quality, we could pick software independently of the organization.
- Quality needs to take into account the organizational context (people, processes, structure)

OS^3 Examples

- R (www.r-project.org)
- BioConductor project (www.bioconductor.org)
- Octave, OSS of Matlab (www.octave.org)
- Ggobi, interactive visualization of high-dimensional data (www.ggobi.org)
- Augsburg University visualization tools (rosuda.org/~unwin).
- PhysioNet www.PhysioNet.org
- Python (BioInformatics, Numeric Python, etc.)

R (www.r-project.org) (1/2)

1. Transparent development cycle, bug tracking, versioned development.
2. No reason, at face value, why a company can’t validate/qualify this for in-house use.
3. Large number of supporting data analysis methods, best-in-breed visualization.
4. A high-quality open-source statistical programming language.

R (www.r-project.org) (2/2)

5. Excellent support for data analysis through simple and compound data structures.
6. Modern programming language characteristics.
7. A vibrant and supporting user community.

Value proposition based on #4, #5, #6 and #7 (NOT COST!)

What are the real issues and problems?

Issues with R in Pharma Development (1/3)

All are serious but solvable either internally or externally.

- Documentation of design, design process, lifecycle planning.
- Who owns R’s future? Directions are driven by a mix of research and applications, but not necessarily yours (hence adaptation must recognize this risk and decide).

Issues with R in Pharma Development (2/3)

- Internal IT Implementation:
  R’s release cycle vs. in-house IT software release capability.
- QA/QC of critical 3rd party libraries (add value, even when of suspect quality).
- Licensing, patents, and ownership of useful, important 3rd party libraries (89% are GPL or GPL-variants, others PD or non-commercial).

Issues with R in Pharma Development (3/3)

- Support of R when “R-patches” can not be immediately tracked.
- Verify that what you download is what you think was downloaded.
- Commercially supported R (“redhat”) vs open-source or consulting support (“debian/ubuntu” model) are both options – we are considering the more challenging latter proposition in this talk, not the former)

Open Source Statistical Software (OS^3) Pros/Cons in Pharma Dev (1/3)

Problems not necessarily due to licenses but are more IP, IT, and quality related:

- IT integration of rapid release cycles with slower internal cycles (cultural change).
- The development team might not cover risk management for all components (R packages; additional work in-house.)

OS^3 Pros/Cons in Pharma Dev (2/3)

- Additional arrangements for internal or external 3rd party support.
- Assessment of ownership and rights (e.g., patents, unlawful release, but this needs to be done anyway).
- Quality assessment (in-house or external).

OS^3 Pros/Cons in Pharma Dev 3/3)

OS^3 adds value, but not only because of cost:

- Quality and transparent software development.
- Rapid provisioning of implementations of modern, cutting edge statistical procedures (visualizations, analyses).
- A vibrant and supporting online community.

Summary

Provided a bird’s eye view of OS^3 in pharma dev:

- OS^3 provides significant benefits, not only because of cost.
- OS^3 presents real risks, but manageable through established validation and qualification procedures. This does require considerable resources.
- R in particular provides a very attractive value proposition.