Paper AD05

Further Adventures in SCE Alchemy

Eleanor Sparling, GSK, London, UK
Steve Griffiths, GSK, London, UK

ABSTRACT

It's sometimes said that a year is a long time in politics, but it's definitely a long time in the development of a new SCE. In 2022 in Belfast, we presented on the development of METEOR (Multilingual Environment to Transform, Enable, and Optimize Reporting); GSKs new language agnostic SCE solution powered by Domino Data Science Platform with GitHub Enterprise integration, leading up to the initial Minimal Viable Product (MVP) release. With an ambitious target of complete adoption by the end of 2023, we will share our continuing story of study team onboarding, training and end-user support and migration of legacy studies whilst continuing to work using Agile methodologies to improve the user experience.

INTRODUCTION

In early 2020, a three-year mission was formally launched by GSK Pharma Biostats in partnership with R&D Tech to modernise clinical trial reporting. The aim was to leverage both modern technologies and methodologies to create a new Statistical Computing Environment which would:

- Deliver Business Efficiency
- Provide Enhanced Capability
- Enable Faster Analysis & Reporting
- Simplify Ways of Working

Very early on in this mission, GSK announced a plan to integrate the Rx and Vx R&D units, and the new SCE, now known as SPACE (Statistics & Programming Analytics Computing Environment) was seen as the vehicle to deliver a single, multi-lingual, unified data analytics platform. Almost simultaneously there was a move from GSK IT to develop into a 'product-orientated DevOps organisation', structured around the Scaled Agile Framework, which for the majority of the team meant a whole new way of working.

BRIEF MISSION RECAP

DOMINO

Before considering a custom solution, it's necessary to first see what's out there in the market. When SPACE was first initiated, the SCE market was scoured for commercially available 'commercial off the shelf' (COTS), and 'software as a solution' (SaaS) SCE products. These were evaluated first before considering a build project.

Experts within GSK evaluated and compared these against the in-house standards – HARP (Rx) or LSAF (Vx), and whilst these were generally good for statistics, programming and data management, the scope of SPACE was broader than those functions and the conclusion was no one product was as good as or better than HARP / LSAF, and that a custom solution would provide both better value and transform the business.

Domino hadn't originally been marketed as a traditional SCE, but the team liked the way it allowed people to collaborate on a project in any programming language of their choice, in addition to providing flexibility with multiple workflows, and so there was a compelling reason to initiate a Proof of Concept using Domino. This consisted of a short list of business and IT criteria, most of which were met 'out of the box'. Those few which didn't needed either a bespoke solution or the team were able to demonstrate an alternative 'out of the box' approach in order to fulfil the criteria.

Features we liked included (but not limited to):

- GxP installation of R,
- SAS Studio is easy to use within Domino,
- Easy to set up projects,
- Multiple programming languages with consistent look & feel,
- Versioning and snapshots,
- Granting access and collaboration with others is easy,

Domino was selected to POWER our SCE. It's not the entire solution, it provides (interactive development

environments) IDEs, workspaces, and batch execution elements of the SCE, which was renamed METEOR (Multilingual Environment to Transform, Enable, and Optimize Reporting).

STUDY CODE MANAGEMENT

Whilst evaluating Domino, work was progressing on the code management system, which was required to promote collaboration, code re-use and permit version control. GitHub was chosen over GitLab as it was already being used in the "Statistical Data Sciences" group and therefore was familiar to some users. A branching model specific for our study needs was developed, where each reporting artifact would have its own branch during development, before getting pushed to the main project branch following initial QC.

CONTROLLED EXECUTION

One of the fundamentals of a secure SCE is that the final 'production' run of study artifact production is controlled, and so the concept of 'controlled execution' was developed. For every planned analysis, two Domino projects would be created: one for 'developmental' work, and the other for the final production run or CX run. The two would be set-up with the same project / imported code repositories and mounted SDTM data, yet access would be limited in the CX project to submitting batch jobs. These jobs would run a per-defined configuration file listing program interdependencies (for example all the ADaM programs are reliant on ADSL running first) and programming language. Upon completion of the batch job (or multijob), the system would automatically create a snapshot of the planned analysis artifacts which would then be 'pushed' to an area from where the other SPACE components PULSAR (publishing tool) and RAPIDO Data Viewer (data visualization tool) could access them.

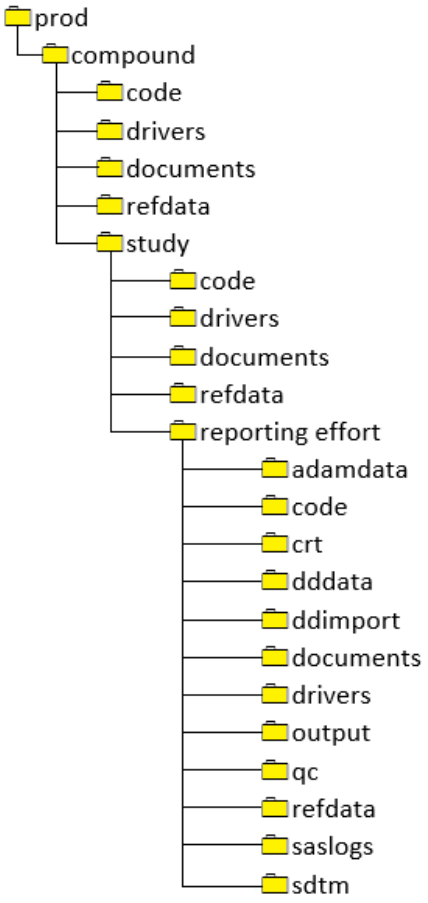
EARLY ADOPTERS

Subject Matter Experts were responsible for developing and reviewing new training modules for Domino, GitHub, METEOR and the 'spacification' of Pharma (Rx) and Vaccines (Vx) legacy code, and a user support model was created. Two small studies (one from Rx, the other from Vx) were carefully identified based upon their likelihood of success and invited to be early adopters. Additional hyper-care support was provided to these teams, and as expected (as per Agile), there were continuous feedback loops between these teams, the METEOR business team and tech team, with the aim of improving everything prior to mass scale up.

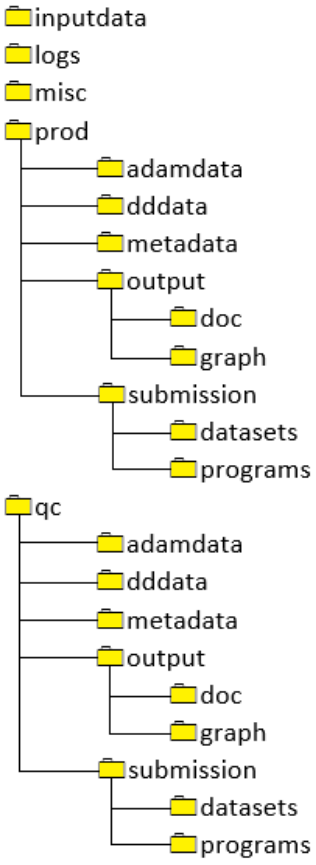
ADOPTION AND MIGRATION

Moving from a Minimal Viable Product (MVP) to Business as Usual (BAU) is no small task, especially when you incorporate accelerated timelines. Whilst feature refinements and enhancements were still underway, adoption and migration were also being scaled up. In order to smooth the transition period different milestones were put in place for adoption and migration. Targets were initially set for all new studies to begin working in METEOR rather than the legacy systems, this was then expanded to any new study analysis. Adopting METEOR at the start of a new study analysis makes sense, it is a clean starting point from a data and code perspective and project creation follows the designed METEOR workflow. Migration, on the other hand, has proved to be much more complex, requiring an entirely separate migration tool to be built.

The migration team faced several challenges, one of the most significant of these was the format and organization of legacy code and data. Unfortunately, there isn't a simple 1-to-1 mapping of the legacy directories to the new METEOR counterpart. The figure below demonstrates the differences between HARP and METEOR; there are also differences between LSAF and METEOR.



Simplified HARP directory tree

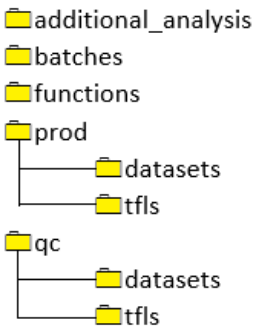


METEOR Planned Analysis directory tree

Following consultation with experts on each legacy system, it was possible to build a generic set of rules which would account for most legacy artifacts; ADaM datasets for a given 'reporting effort' would migrate to 'prod/adamdata' within the new planned analysis, however any ADaM dataset specification files would map into '/inputdata' along with any other non-programmatically created inputs (for example spreadsheets with AEs of Special Interest – AESI).

Mapping two legacy systems into one also meant that sometimes Rx / Vx artifacts had to treated differently. So, Rx text outputs (.L10) would be mapped directly to 'prod/output', but Vx outputs which tended to be in PDF / RTF format would go into '/prod/output/doc'. All log files would be placed at the top level '/logs' directory with the assumption that study teams had followed the good programming practice concept of a program naming convention highlighting production / QC programs (i.e., QC programs and outputs are pre-fixed with 'qc_').

Code (table, figure, listing or macro) were migrated into project code repositories within GitHub, which had its own directory structure (as shown below)



GitHub directory tree

The legacy systems both had a set of standard macros, which were migrated (and adapted) in METEOR and placed in a different code repository, available to each planned analysis as an imported repo. This made dealing with macro

code slightly easier, as it meant that only study (or reporting effort for Rx) specific macro code needed migration to the 'functions' directory. A standard naming convention made it (relatively) easy to differentiate between ADaM dataset and TFL code. Following a lot of discussion, and rule refinement it was agreed that anything still not fitting the rules we had created would not be migrated, and study teams would be supplied with a migration status report stating what had / hadn't been migrated.

Another aspect of migration is dataset and TFL code would need modification to work within METEOR. Libname statements would need updating to reflect the new directory structure, and for SAS® programs the SASAUTOS option would need changing to point to the new macro locations. A Python script was developed to automatically make the necessary amendments to these programs following migration.

Dummy studies based on real-world counterparts were created in a development environmental to enable the developers to create and test migration scripts. At the same time, study teams were asked to get their projects migration ready; archiving all completed work and making sure all required files were available in the production environment (and not the working / developmental directories). The migration team would then work closely with study teams providing proposed migration dates, and making sure they didn't clash with important study milestones like Statistical Analysis Complete (SAC). Following migration, study teams are advised to work through a checklist to ensure everything has been migrated successfully.

Variety itself, was a common theme in migration challenges. As the migration team had full exposure to all studies in every disease area and wider analysis groups, many additional bespoke requirements were unearthed that had not been previously considered during the MVP stage. To achieve a true BAU the METEOR team needed to re-assess and re-prioritize feature developments in addition to all new process workflows and trainings. Whilst it is not a technical or implementation challenge of migration, the differences in training experience for a user migrating half-way through a study analysis compared with that of a user onboarding onto a brand-new study must be recognized. It is currently too early for us to reflect on the differences in these experiences, but it is important to monitor and react in an agile approach to enable a successful migration at scale.

SUPPORT MODEL

Our support model for first adopters worked very well; they were provided with live interactive training, regular drop-in sessions with Subject Matter Experts (SMEs) and on demand escalation for technical issues. Scaling up from two initial MVP studies to BAU required a different strategy as it was not possible to provide the same level of individual care to an entire department with the resources we had available.

A combination of different support mechanisms were put in place in an attempt to give users the experience of a hands on support model whilst not being as resource intensive as the MVP model required. Hypercare calls, hosted by the SME team, were put in place on a weekly basis at which any METEOR user could attend to ask specific, more technical questions. These calls provided the face-to-face support that was so successful in MVP and feedback continues to show users value this time as a key support pillar. As the number of adopters increased, the breadth in experience and skill also grew. It was not efficient to keep repeating basic introductions to METEOR within the hypercare calls, therefore a targeted delivery approach was implemented. High level concept sessions were repeated monthly to a wider audience much earlier in their adoption timeline, allowing users to get familiar with concepts and terminology before any mandatory training and system onboarding took place.

A large increase in user base also required training material to reflect the variety of learning types. There became a greater emphasis on self-service training in which a user has access to instruction guides, video demos, sandbox environments and FAQs. This self-service model was intended to reach wider audiences without increasing the overhead for the METEOR team, however, periodical reviews and updates due to feature enhancements brings back these limited capacity issues. To meet these METEOR team capacity issues, community led initiatives were rolled out. Power users within each disease area are used a first point of contact for issue guidance and updates on new releases. Additionally, more emphasis is being placed on using our dedicated Teams channel to ask for help as well as provide help. This METEOR channel has been extremely successful with the large increase in user base, particularly in building a sense of community in an increasingly virtual environment.

GITHUB

One of our main goals in developing METEOR was to become more efficient and future ready. Whilst our new computing platform is an environment users must learn how to use, there is also the big transition to new processes and ways of working. A key example of this is using GitHub for study code version control. It is said that you should never build a process to fit the tool, however, what is the compromise when you are striving towards future unknowns?

Version control systems like GitHub allow you to access an incredible level of granularity in your metadata, allowing you to pinpoint bugs, track project progress and rollback to historic releases. However, our previous code development process would not leverage any of these benefits without considerably rethinking our ways of working. The decision was made to adapt our current process slightly now, in order to fully utilize more capabilities in the future. Branching models were designed to build upon sequential analyses whilst separating any concurrent analyses. Tags have been leveraged to access historic versions without the need for duplicated storage. At face value there appear to be many selling points to use of GitHub, however you must also take into account the effort and time required to upskill an entire department into a very new way of working. Feedback from our early adopters shows that moving to GitHub is a steep learning curve with complications expected, however once you have built up the new muscle memory it provides a much-improved version control environment. Migration also posed a key challenge for adopting version control. It is very difficult to move halfway through an analysis containing code versions separated by folders into a system which relies on connections.

USER ACCEPTANCE / REGRESSION TESTING

Working towards and within a state of 'business as usual', there is almost a continual state of UAT work. Each new METEOR feature developed during each PI requires UAT (following system testing). In addition, GSK works closely with the Domino development team in identifying new requirements (for example, improved audit capabilities, or the ability to copy Domino projects). Therefore, additional regression UAT is required for each new deployment of Domino, testing not just what GSK have requested, but all the other upgrades (including releases GSK skip) which may impact METEOR: have permissions, roles and access changed? Has the code used to set up the SAS environment changed behavior? Regression testing doesn't mean testing everything though. There is no need to regression test our Data Access Manager if Domino has been updated, however there may be a need to test aspects of the authorization tool if it 'works' with Domino.

Two of the biggest questions regarding testing are 'when should testing occur?', and 'who is available to perform testing?'.

The need for UAT tends to sneak up and before you know it, needs to be completed within a very tight time frame. Scripts are written (following a quick demo), reviewed and approved, and formal UAT commences. Often script errors are encountered, or the tester fails to produce an adequate recording. These in turn lead to additional testing cycles, and increased pressure. Findings lead to a more complex summary report, which takes longer to review.

Ideally a feature is available in the development environment and demo-ed. UAT script development starts following this demo, and informal testing can take place almost immediately following finalization of the scripts. This allows for fine tuning of the test steps and expected results (reducing the risk of additional UAT cycles), and occasionally identifies unexpected behavior. These behavior finds (which can include bugs) are a double-edged sword though; it's highly important to address such findings prior to formal UAT (and the push to production) as the impact at this point is minimal, however the UAT delivery timelines seldom change enough to take the additional development work required into account, and therefore the pressure on UAT completing in time increases. This can impact upon the 'who?' question.

Traditionally, volunteers from within the business are sought to assist with performing UAT. The lure of being the first to see and use the new functionality is too great for some, whilst others like the learning opportunity. Becoming a tester does involve time commitment away from 'normal' pipeline delivery work; there are numerous training courses to complete before UAT can start, and often there are additional testing pre-requisites around METEOR training. The initial cycle of any feature normally works well in parallel with the testers' usual tasks, but clashes start to occur when further cycles are required.

IMPROVEMENTS TO TESTING

Moving forward there is a drive to move to use more METEOR business users for UAT. Everyone has been trained, and performing the UAT helps improve their understanding of the features, thus allowing them to assist users more efficiently following the production release of said feature. Additionally, features will made available a lot earlier within a testing environment which will facilitate the earlier creation of UAT scripts and training materials.

CONCLUSION

Developing a new SCE is not a feat for the faint hearted, especially when you must also consider accelerated timelines; department wide migration and unification between Rx and Vx functions. That being said, the process of developing a new SCE has been invaluable as an opportunity to re-evaluate historic ways of working and modernize technical capabilities. Numerous challenges have been met but also overcome, providing lessons that can be sought upon for future feature releases. Working in an agile way, collaborating between business and tech teams, has proved fundamental in our continual progress.

REFERENCES

AS01: SPACE – Launching GSK Biostatistics Towards New Horizons SPACE - Launching GSK Biostatistics towards new horizons (phuse.s3.eu-central-1.amazonaws.com)

CONTACT INFORMATION

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

Eleanor Sparling
Email: eleanor.x.sparling@gsk.com

Steve Griffiths
Email: Stephen.j.griffiths@gsk.com