PhUSE 2011

Paper PD03

Programming for Early and Late Phase – really that different?

Katherine Macey, Roche Products Ltd, Welwyn Garden City, UK

ABSTRACT

This presentation will compare and contrast the statistical programming needs of phase I/II studies (early development) versus phase II/III studies (late development). I will start by comparing typical features of early and late phase studies and the resulting technical and 'soft' skills required by programmers working on those studies. For example, juggling multiple exploratory healthy volunteer studies with a fast turn-around vs. long-term planning and programming of large confirmatory phase III studies, may require a different mind- and skill-set. I will describe some examples of typical clinical trials as illustrations.

I will then discuss the advantages and disadvantages of various ways of organising your programming teams; for example, having separate teams supporting early and late phase studies, having a single project team supporting all phases. I will illustrate with examples from Roche of how programming teams are structured.

1. INTRODUCTION

As a drug project progresses through the phases from 'entry into human' to regulatory submission, the requirements from a statistical programming point of view may change. This presentation will compare and contrast the programming needs across the phases, compare the technical and 'soft' skills required from programmers, and finally discuss possible options for organising teams across the phases.

In this presentation I refer to 'early' and 'late' phase clinical trials. These definitions may differ across different pharmaceutical companies and contract research organisations (CROs). In this context my definitions are as follows:

- Early phase trials: Phase I and Phase IIA up to decision to proceed to full development
- Late phase trials: Phase IIB and Phase III up to submission/registration

2. EARLY AND LATE PHASE – TYPICAL STUDIES

In order to discuss the programming needs across the spectrum of studies from entry into human to submission, it is useful to re-cap the 'features' of typical types of studies conducted in early and late phase and illustrate with some examples.

EARLY PHASE

Phase I studies are exploratory in nature. They are usually conducted in small numbers of healthy male volunteers (or patients for some indications such as oncology). Studies are designed to characterize the pharmacokinetics (PK), pharmacodynamics (PD), safety, tolerability and toxicity of the compound and determine a safe dosage range and may also include collection of exploratory endpoints (e.g. biomarker data). These studies typically require a quick turnaround and short time from study start-up to reporting. Studies may see frequent adjustments to study designs e.g. adding or removing cohorts, amendments to the objectives of the study.

Example: SAD (Single Ascending Dose) phase I study in healthy volunteers, 64 subjects in 8 cohorts, single centre, single-blind, randomized (6:2 per cohort), study length ~11 months, safety, tolerability, PK, PD, exploratory endpoints. Includes objectives to evaluate effect of food, activated charcoal, formulation, i.v. microdose of drug.

Phase IIA studies build on what has been learned in phase I, give the first indication of efficacy in selected

populations of around 100-300 patients who have the condition to be treated, including 'proof of concept' (POC) and 'proof of mechanism' (POM) studies. Studies look to further evaluate safety, enable dose ranging and look for a dose response. They are often multi-centre, and may include elements of exploratory study design such as adaptive designs, interim analyses and exploratory analyses. Phase IIA studies will have certain parallels to phase III trials in terms of programming activities (examples: creation of efficacy analysis datasets, multi-centre, randomized, controlled trials) but on a much smaller scale and with less emphasis on planning for future regulatory submissions.

Example: Phase IIA study in patients, proof of concept, ~400 patients with depression, randomized (drug at 3 dose levels or placebo as adjuvant therapy), Bayesian interim futility analysis to drop ineffective arms, DMC (data monitoring committee), double-blind, multi-centre, study length ~19 months, efficacy, safety, PK, PD, exploratory endpoints.

LATE PHASE

Phase IIB studies are well-controlled trials to focus on further evaluation of efficacy in patient populations at the doses selected from phase IIA studies. These trials usually represent a rigorous demonstration of a medicine's efficacy and can be pivotal trials.

Example: Phase IIB study in patients with RRMS (relapsing remitting multiple sclerosis), ~200 patients, dose ranging, randomized (drug at two dose levels or placebo added to usual care), partially-blind, multi-centre, study length ~> 3 years, efficacy including MRI scans, safety.

Phase III (pivotal) studies are confirmatory in nature; studies are designed to confirm efficacy and safety in a large sample of patients (hundreds or thousands), to compare to other frequently used treatments, are likely to be longer in duration and be global multi-centre trials. There should be less scope for frequent changes to study designs during the study conduct and the focus should be less on exploring and more on confirming knowledge gained from earlier trials. One difference to early phase is that PK data collection is usually confined to sparse sampling for purposes of population PK; for example: confirmation of exposure in the target patient population, confirmation of the exposure vs. response relationship. Intensive PK sampling is not routinely done.

Example: Phase III pivotal study in patients with acute coronary syndrome, ~300 patients, randomized (drug at one dose level or placebo added to usual care), double-blind, multi-centre, study length ~10 months, efficacy, safety.

This description of the progression through the phases of clinical trials is somewhat simplistic. The paradigm is changing with increasing use of adaptive designs to combine across phases (e.g. combining phase I and IIA or phase II and III), phase I studies in patients rather than healthy volunteers where appropriate, combining of objectives into single phase I studies; example: studies evaluating SAD, MAD (multiple ascending dose), PET or MRI (positron emission tomography, magnetic resonance imaging), formulation, food effect in a single trial.

There are no official definitions on what constitutes a phase IIA vs. phase IIB trial[1]. The 'FDA Guidance for industry on End-of-Phase 2A Meetings'[2] defines end of phase IIA as: "For the purposes of this guidance, end of phase 2A occurs after the completion of phase 1 trials and the first set of exposure-response trials in patients, and before beginning phase 2B (i.e., patient dose-ranging trial) and phase 3 clinical efficacy-safety trials."

3. EARLY AND LATE PHASE - SKILLS

Having described the features of studies conducted across the phases, it is now useful to translate those into the technical skills (e.g. programming expertise) and 'soft' skills (e.g. interpersonal skills such communication, team working) needed by statistical programmers working in each phase and contrast where the differences might lie.

WHAT TECHNICAL SKILLS DO PROGRAMMERS NEED TO WORK ON EARLY PHASE TRIALS?

- Ability to use/develop a standardised reporting tool or macros for safety and/or PK data will be important to ensure quick turnaround of studies - data displays may be required within hours, particularly for dose escalation studies (SAD/MAD),
- Good knowledge of phase I study designs: SAD, MAD, DDI (Drug:drug interaction), PET, MRI, mass balance using radio-labeled compounds, PK studies; and the data collected – imaging data, PK, PD data such as cognitive tests, questionnaire data, PROs (patient reported outcomes) and also the processes around data delivery (particularly for tight timelines),
- Ability to produce simple, clear code perhaps without too much concern for efficiency of programs (if dealing

with small numbers of subjects). Also, programmers may not need to be too concerned with project consistency if the studies and reporting tool are very standard anyway and would not be pooled. However, for proof of concept studies in patients, this may become more of a concern e.g. plans for standardized handling of efficacy and PD data with a view to future phase IIB/III studies and consistency across projects within a therapeutic area,

- Ability to produce innovative graphical displays to aid decision making and work interactively with customers,
- Ability to handle PK and other data i.e. loading PK data to a separate system (example: Pharsight Knowledgebase Server) to enable analysis of data by the pharmacologist,
- A flexible approach to programming in order to deal with adaptive designs etc.; for example, designing programming set-up to anticipate possible changes to the study design,
- Able to strike the right balance of formal QC (quality control) and writing of detailed programming specifications vs. exploratory approach required ('quick and dirty'). The level of rigorous validation will likely be less; for example 'double-programming' by a second programmer may only be required in a small number of cases. Ultimately, many projects in early phase will never make it to late phase. Therefore programmers need to get the right balance to enable the team to make quick decisions (which may be to terminate the project) based on quality outputs.

WHAT SOFT SKILLS DO PROGRAMMERS NEED TO WORK ON EARLY PHASE TRIALS?

- Programmers will need to be able to work closely with 'suppliers' (e.g. data management) and 'customers' (e.g. statisticians / pharmacologists) on a day to day basis, requiring good communication skills,
- Ability to plan and juggle own work tasks, work on multiple studies simultaneously. Changes to studies means planning of the resources required by managers may be a challenge,
- Flexibility to respond to changes to plans and required data displays, examples: a biomarker which was thought to be important at the start of the study may not be by the end, so plans for the reporting of those data may change, cohorts may be added or removed from studies, objectives amended,
- Self-reliant, often one programmer per study so working "alone" with rest of the multifunctional team (albeit with good general support from rest of early programming team),
- Able to prioritise, able to negotiate timelines to meet own and teams needs, able to be assertive when required.

WHAT TECHNICAL SKILLS DO PROGRAMMERS NEED TO WORK ON LATE PHASE TRIALS?

- To be able to think strategically about programming strategy; for example, ensuring project consistency in handling and reporting of efficacy data, designing analysis datasets with a view of future pooling across studies, production of standard project code or macros to harmonise certain programming tasks common to all studies, preparing for electronic submissions,
- Ability to develop specialisms in handling certain types of data e.g. they may be an expert on creating derived efficacy analysis datasets in that disease area requiring excellent data manipulation skills, or doing all project-specific adverse event reporting on a project. This needs to be balanced with the need to be able to program standard safety and efficacy analysis datasets and outputs,
- Programmers need to be able to assess the most appropriate form of QC technique to be used (not everything needs to be double programmed), based on the complexity of the report object, but always bearing in mind the highly regulated environment we work in,
- Once submissions have been made, teams will be asked to respond to fast turnaround regulatory requests, preparation for advisory committees, publications, other ad-hoc requests, so planning and anticipating these will be key,
- Graphics play an important role in helping regulatory authorities to understand the data better. Programmers need to have the ability to use the latest graphical developments to produce better graphics.
- The size of the source data will be greater than for early development (larger numbers of patients and longer studies) and so efficiencies will be needed to reduce the run time for processing the data. This requires the programmer to really think how best to write a program with this in mind.
- The number of outputs required is greater than for early development, especially when considering the Summaries of Clinical Efficacy and Safety (including pooling across studies). Many of these could be due to subgroup analysis resulting in repeats of the same template. The programmer needs to employ logical and innovative thinking to create a template that can be easily re-used via the use of macros.

WHAT SOFT SKILLS DO PROGRAMMERS NEED TO WORK ON LATE PHASE TRIALS?

- Late phase study programmers will need good long term planning of programming activities for pivotal studies, and an ability to be proactive and forward-thinking,
- Programming specifications are more important for late phase programming. Specifications enable programming rules used to create analysis datasets and outputs to be transparent to regulatory authorities

(feeds into the electronic submission documentation), allows other team members to understand what has been done and is essential for independent validation programming,
- Programmers are more likely to need to become specialists in one disease area,
- The project manager and programming leads will need to coordinate a large team of programmers across multiple studies and reporting events,
- Ability to be flexible when responding to changes in filing strategy, changes in programming requirements in response to stakeholders including internal partners such as Clinical Science, Medical Writing, publication authors, external data review boards, regulatory authority questions etc.
- If a programmer is not a lead on a study, they may have fewer direct interactions with Scientists and work mainly with their programming lead and as part of a larger programming team to deliver a study. They will require good communication skills within the team, and other related functions such as Biostatistics.

This section has highlighted differences in the technical and 'soft' skills required for programmers. To sum up, early phase reporting requires flexibility, is fast-moving, with an ultimate aim to deliver results to enable fast decision-making in an exploratory setting. Late phase reporting requires a more considered programming strategy with a view to eventual pooling and submission for confirmatory studies requiring higher levels of documentation and QC. Both roles require good communication, team working and high levels of programming skill, albeit different types of programming. With training and support, all programmers should be able to cover work from studies in any phase.

4. EARLY AND LATE PHASE - ORGANISING TEAMS

Having described the types of studies and skills needed by statistical programmers working in early and late phase, I will now discuss different options for organizing programming teams and the pros and cons of each approach.

Many factors will influence how programming teams can be organized; a few examples:
- The number of people in the statistical programming department and overall size of the company
- Historical or internal company 'political' reasons
- Distribution of programmers across multiple sites across the world
- Distribution of key partners across the sites (i.e. Data Management, Statistics, Pharmacology)
- Tasks covered by other closely related functions e.g. does your department also cover programming of the database and data extraction activities or only statistical programming?
- Are statisticians and statistical programmers two separate functions or together?

In this section I will describe various options as to how programming departments could be organized, based on examples I've seen at Pharmaceutical companies and also at CROs. I then go on to discuss the pros and cons of each model:
- 'Separate early / late phase programming'. Dedicated teams to exclusively cover early phase and late phase studies only. A variation on this is to also have a group which covers all programming activities concerned with PK data across all studies.
- 'Everyone does everything'. Programmers regularly work on a range of studies across all phases.
- 'Something in between': All studies for a project are handled within a single programming team but with some variations:
  - Safety programming group: to cover standard safety reporting across all phases
  - Healthy volunteer dedicated group: Separate groups to cover studies in healthy volunteers and studies in patients. Taking it even further, if a company has its own phase I clinic for healthy volunteer studies, have dedicated programmers and statisticians on site (this is a model at some specialized early phase CROs)

PROS AND CONS OF 'SEPARATE EARLY / LATE PHASE PROGRAMMING':

PROS
- Gain efficiencies from having specialists who are trained on tasks specific to the study phase, such as handling of PK data, common phase I study designs, adaptive designs etc.,
- If you are working with different operations groups for early and late phase development (i.e. the internal organization who conducts the trials), you may want your programming departments to 'mirror' this
- Gives opportunities to junior programmers to be able to lead and take responsibility for entire studies,
- From an organizational point of view, it may be easier to organize a team of programmers who are all working on fairly short-term assignments, rather than a mixture of short and long studies,
- Provides more varied career development path for each group, people may have a preference for early or late phase.

CONS

- Once a project passes to late phase, need to work harder to ensure communication between the early and late phase teams is maintained; example: non-critical path phase I studies which are ongoing during phase III can get forgotten, but these may be important as part of the electronic submission planning,
- Harder to maintain consistent programming approach between phase IIA and phase IIB/III studies,
- May not make sense on therapeutic areas such as oncology where all studies are conducted in patients and have an element of efficacy data collection even in phase I,
- Potential for diverging processes or practices across the whole department,
- Given there may not be a definitive line between early and late phase development, any separation of groups may be somewhat arbitrary.

PROS AND CONS OF 'EVERYONE DOES EVERYTHING':

PROS

- Enables more consistent programming approach, particularly in areas such as oncology where phase I studies are often in patients
- Possibility for a programmer to follow a drug project from entry into human through to submission (albeit over a fairly long period of time)
- Enables better communication within projects, particularly when phase I studies are conducted in parallel to phase III
- Increased flexibility of resources to be able to work on any study

CONS

- Not efficient use of resources to train everyone on every task, process, trial design and specialism like PK data handling,
- Ultimately the time taken for a project to move from early to late phase will mean programmers will spend years working only on phase I studies anyway and so by default will become specialists,
- Likelihood of having a programmer work on the same project from phase I to III is small giving the timescales involved plus the likelihood of a molecule in phase I to reach phase III is low.

PROS AND CONS OF 'SOMETHING IN BETWEEN':

PROS

- Safety programming group: recognizes that the data collection and reporting of most safety data will be standard across all phases, these programmers could also become experts in the standard reporting tool if one is used,
- Healthy volunteer dedicated group: this group can be experts in these types of studies and enable very quick ('real time') decision making and turnaround.

CONS

- Safety programming group: Having a safety team will result in having two programmers even for very small studies which may not be efficient; also will standard safety reporting alone be fulfilling as a long-term role?
- Healthy volunteer dedicated group: Fairly narrow focus to only report studies from a single clinic, harder to maintain consistent safety and PK reporting across clinics and with patient studies.

EXAMPLE FROM ROCHE:

Roche has two organizations conducting early development studies ('gRED' – Genentech Research and Early Development and 'pRED' - Pharma Research and Early Development) and another for late development studies ('PD' - Product Development). Projects pass from early phase development (pRED and gRED) to full late stage development (PD) via a full development decision point 'milestone'. At this milestone (at some point during phase II), project teams need to demonstrate to senior management that the molecule has met a series of criteria to enable the project to proceed to late development. The criteria are agreed at the start of phase II.

Statistical programming supports all three organizations and 'mirrors' this set up with dedicated programming groups for gRED, pRED and PD studies. The gRED and pRED programming groups also provide PK data handling support to PD studies where required. From a process point of view, following the integration of Roche and GNE, processes

such as data models (SDTM), data delivery processes from data management and statistical programming environments are in the process of being aligned across all organisations.

5. CONCLUSION

In this presentation I have compared the statistical programming needs for early and late phase studies. Broadly, as we move through the phases, trials are moving from exploratory to confirmatory; small numbers of subjects to large numbers of patients; shorter decision-making trials to assess PK and safety to longer trials to confirm efficacy and long-term safety.

I also highlighted differences in the technical and 'soft' skills required for programmers. Early phase reporting requires flexibility, is fast-moving, with an ultimate aim to deliver results to enable fast decision-making in an exploratory setting. Late phase reporting requires a more considered programming strategy with a view to eventual pooling and submission for confirmatory studies requiring higher levels of documentation and QC. Both roles require good communication, team working and high levels of programming skill, albeit different types of programming.

Finally, I compared some different options for organizing programming teams across the phases and the pros and cons of each. All will have their place depending on size of the organization and the types of studies conducted. As an example, I described how programming teams are structured in Roche.

My overall conclusion is that with training and mentoring, all programmers should be able to work across all phases and in all situations and indeed should during their careers. Efficiencies can be gained by careful consideration of the programming needs of studies across the phases and organizing teams accordingly.

REFERENCES

1: 'FDA Guidance for Industry - End-of-Phase 2A Meetings'
http://www.fda.gov/downloads/Drugs/GuidanceComplianceRegulatoryInformation/Guidances/ucm079690.pdf

2: 'The two-headed beast' (from 'Signals' online magazine of analysis for biotechnology executives)
http://pharmalicensing.com/public/articles/view/1074610862_400d42aebbc76

ACKNOWLEDGMENTS

Thanks to Ben Szilagyi for reviewing my initial drafts / ideas. Thanks to Ben, Vijay Reddi, Karen Rowe and Christine Leigh for their review of the final paper and Darren Bentley for advice on PK data collection in phase III trials. Also thanks to colleagues for supplying examples of studies across the phases.

CONTACT INFORMATION

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

Author Name:	Katherine Macey
Company:	Roche Products Ltd.
Address:	PDBP – SPA, 6 Falcon Way, Shire Park,
City / Postcode:	Welwyn Garden City, AL7 1TW, United Kingdom
Work Phone:	+44 (0)1707 365850
Email:	katherine.macey@roche.com

Brand and product names are trademarks of their respective companies.