Paper 306

Using SPAM for project estimation in SAS®Application Development

Sarah Ragland, SAS Institute Inc., Cary, NC

ABSTRACT

There is a humorous expression that says managing technical people is like herding cats. Can't you picture it - SAS developers are typically quite technical and creative people who each have a unique way of using software to solve the same problem. They don't need to be managed; they need to be free to do problem solving without confining boundaries. After all, isn't that the beauty of SAS software? Yes, of course it is. Then how can one suggest using project management principles, which strive for definition and planning, in the creative SAS software development environment?

Applying project management in a SAS software development environment is unique. It requires a different breed of project management that considers large project teams, a fast pace, a creative culture, and an ever-changing development environment. This new breed of project management is what I affectionately call SPAM (Sarah's Project Anti Management). Don't be alarmed by the word 'anti'; it does not mean that traditional project management is not appropriate. You just have to approach project management in a SAS application development environment with a bit of flexibility.

The following discussion will provide some specific examples of Sarah's Project Anti Management 'tricks of the trade' that can be used specifically in project estimation for SAS application development.

THE CHALLENGE OF ESTIMATION FOR SAS APPLICATION DEVELOPMENT

Project estimation in any environment is challenging but estimation in the SAS application development environment has its own set of issues. Consider the following:

Premise 1	SAS software is unique in that the breadth of tools is so diverse.
Premise 2	You could develop the same SAS application three times and never repeat an approach.
Premise 3	Each approach can have a vastly different level of effort.
Premise 4	Experience with one approach does not equal experience with others.
Premise 5	Good estimation depends on solid historical performance data.
Premise 6	Premises 2, 3, and 4 make building relevant historical performance data difficult.

Without relevant historical performance data, quality project estimation and planning can be quite impaired.

A logical and traditional answer is to gather all requirements first. That is a great concept, but this doesn't always solve the problem. There are immediately two problems: 1) a client doesn't like to pay for (or spend time with) requirements before they understand a time and financial commitment, and 2) requirements don't always reveal the appropriate development approach. Complete requirements aren't typically completely determined until the design is complete. (For example, a requirement may be that a user screen offers data driven graph generation. Is the appropriate product SAS/AF, SAS/EIS, SAS/Assist, or SAS/Graph?)

What do you do? Here is what Sarah's Project Anti Management experience recommendations:

- Use the Delphi technique to arrive at an estimate.
- Develop a project plan where the estimates are revisited after each phase until the design is complete.
- Avoid committing to a fixed-price model (Time and Materials is more successful with SAS application development).
- Work towards team and technology synchronization.
- Strive for solid historical performance data by warehousing data by technology rather than by project.

Use the Delphi technique to arrive at an estimate.

Norman Dalkey and Olaf Helmer at Rand Corporation developed the Delphi technique in the 1950s. Its original purpose was to provide 'a method of combining the judgements of knowledgeable individuals. It is relevant when there is no determinate answer (e.g., hard data or well established theory) available, but some persons (often called experts) have relevant information about the topic of concern.' Dalkey calls this the 'n heads are better than one' approach. This method has three features: anonymity, controlled feedback and iteration, and formal group judgement.

Colin Murray, author and instructor of **Object Oriented Project Management**, wrote the following description of this technique in action in an application development environment.

- *The Delphi technique can be used for estimating various aspects of a project such as duration, resource requirements, risks, cost and benefits.*
- *It is necessary to have a monitor and a group of at least 10 people. Members of the group should have an informed opinion on the question being considered.*
- *The Delphi method is conducted in four rounds. The procedure for each round is simple, but highly structured.*

There are many variations on the procedure to be followed in conducting a Delphi experiment. Following is a description of one procedure that is relatively simple. It is oriented toward a specific problem, namely, the estimation of parameters in an I.S. project.

For example, it can be used to answer such questions as: How long will this project take if three people are assigned to it? What is the overall cost of this project? What is the worth of the information that will be supplied by the proposed system?

The questions posed to the group should require a quantitative answer – a duration, a cost, or some other number. Most questions involving project estimates lend themselves to quantitative answers.

With a little practice, those few that don't can be revised to meet this requirement. For example, a question might be posed so as to require rating importance of a topic on a scale from 1 to 10. Or a question might require assessing the probability of an event. Or a question might involve several factors that are ranked in order of importance.

The experiment is conducted in four rounds. Each round calls for a written response from each group member. Each response contains an answer to the question, and it might also contain supporting reasons. The answer to the question is a number, which will be referred to as an estimate, because of the nature of the question being asked. The responses, however, are anonymous, as participants do not identify themselves on their written response. At the end of each round, a monitor provides the group with certain information.

Round 1. The monitor poses a question that requires a numerical answer. Participants respond to the question in writing. The responses are gathered by the monitor, tabulated, and analyzed. The monitor then gives the group the following information:

The minimum response.
The maximum response.
The first, second, and third quartiles (the value of each boundary).
The interquartile range (the middle 50% of the responses).
The mode.

Round 2. The monitor asks each participant to make a new response (estimate) to the question. This new estimate can be the same as the one the participant made in Round 1 or it can be different. After making the new estimate, the participant is to compare it with the interquartile range presented to the group at the end of Round 1. If the new estimate is outside the interquartile range, the participant is to write a brief statement of his reason for his estimate. (The new estimate is outside the interquartile range if it is below the first quartile and above the third quartile.)

The monitor tabulates the new estimates exactly as in Round 1 and presents the group the same kind of summary information as Round 1, namely the minimum, maximum, first, second, and third quartiles, interquartile range, and the mode.

Before presenting the reasons for extreme views to the group, the monitor should edit them to make certain they don't reveal the identity of the writer. This requires careful judgements on the part of the monitor to avoid interjecting bias. After editing the reasons, the monitor presents them to the group.

Round 3. This resembles Round 2 except that reasons are now given for an estimate inside the interquartile range.

The monitor asks each participant to make a new estimate. Then the participant is to compare the new estimate with the interquartile range given at the end of Round 2. If the new estimate is inside the range, the participant should state why the arguments for the extreme views given at the end of Round 2 did not convince him.

The monitor collects all responses, tabulates the new estimates, and gives the group the same type of summary information as was presented in Rounds 1 and 2. After editing the written reasons, the monitor presents them to the group.

Round 4. The monitor asks for a new estimate; no reasons need be given. The new estimates are tabulated and summary statistics are presented to the group in the same manner as was done in the three previous rounds.

The estimates of Round 4 are the final results of the experiment. The median or second quartile from this last round is taken as the central group view on the question. The interquartile range gives some idea of the range in which the actual answer might be found. In addition, the spread of the range gives some indication as to the degree of consensus among the group on the question. The greater the spread, the less the consensus and the greater the risk associated with the median estimate.

Colin Murray also points out that it is important to note assumptions before starting the exercise and to estimate more than one factor to arrive at a final estimate.

This technique is quite advantageous when you consider the myriad of possibilities with SAS software. You can begin to explore the cumulative experience of the group with comparable technologies, styles, or coding disciplines. In addition, you can use this technique several times for the same project after the assessment, requirements gathering, and design phases.

Develop a project plan where the estimates are revisited after each phase until the design is complete.

You might think that this approach would be a hard sell. My experience is that it is not hard. There are typically two major phases that precede design in a classical application development project plan: assessment and requirements gathering. The assessment activities are exploratory in nature and there are just as many times as not that the clients engage in an assessment to get a handle on the most basic of project objectives themselves. It is completely logical then (work with me) that the client support the notion that what exploration can uncover is quite unpredictable. In addition, requirements as we have discussed, don't automatically translate into the technology or approach that is most appropriate for development. All of these issues can dramatically affect the level of effort, therefore, the accuracy of your estimate. At the time that you present a proposal or set the client expectation for the project dynamics, provide a sample summary project plan that clearly indicates a review of the project estimates at the end of the assessment, requirements gathering, and design phases. The final estimate review, after design, is the most probable and responsible estimate you can provide.

Avoid committing to a fixed-price model (Time and Materials is more successful with SAS application development).

Software development is a matter of probability. There are no standards to refer to when asked how long it will take to write a page of documentation, for example. Immediately we have questions about the subject, author, audience, format, expertise, and even medium. Even with answers to these questions, we cannot refer to standards charts to get an industry answer. In the business of application development, we estimate based on a thorough (as possible) understanding of the task and past experience. Even so, we are humans and being human is a whole science in itself. Additionally, add in the interesting dynamic of developing with SAS software, which serves to enhance creative problem solving in unlimited ways, and you can see how the word probability comes up. Probability and fixed-price project models are not mutually exclusive. It can work. It has been my experience, however, that even with strong estimation techniques, a time and materials project model works best.

When the situation is unclear, be creative. One SPAM approach is to let original estimates be thought of as budgetary and plan for the revised estimates to become fixed after the design phase. Or leapfrog the phases so that the completion of one phase sets a fixed price for the following phase. This works when you allow the client the

option to intercede with a 'go/no-go' decision between each phase.

Work towards team and technology synchronization.

Any one who participates in application development can attest that no matter how well you plan the project, new or unexpected things always come up. Within the SAS application development environment, this can be especially true for the same reasons the use of SAS is such a great choice. The options are limitless, which means that possible choices for a solution to unexpected problems can also be limitless. Without some common technology bounds, the team could use any of the SAS tools at their discretion for problem solving. This can easily and quickly blow an estimate. A common mistake for SAS application development environments is to not choose or commit to a specific technology or suite of products until development begins. The team does not begin with a common technology boundary. Even within the context of the Delphi estimation technique, developers can still be thinking about very different SAS solutions to solve the same problem. If you want estimation to work within the SAS application development environment, the team and technology must be synchronized from the beginning. Sarah's Project Anti Management experience says to:

Use a technical lead. All projects should have a senior technical member whose role is to ensure that all technology boundaries are respected, from estimation to delivery. This can be supported with activities such as code walk-thrus, mentoring, and consistent team communication and leadership. This person should participate in the estimation activities to promote 'buy in' to the technical solution proposed from the beginning. Additionally, they must be technically knowledgeable and respected to ensure that the project team complies with their decisions for technology and its application.

Assess limitations to options early. The development process always has a client. Their environment or business needs typically sets some limits to the technology boundaries. A process of elimination, if nothing else, can begin to paint a picture of the technology choices. The release of SAS software, the operating system, or the exploitation requirements are a good place to start.

Assumptions must be made. Well-documented assumptions are the project manager's best friend. An estimate cannot be anything more than a budgetary guesses if there are not technology assumptions. Shifts in the technology can then be addressed with change management. More importantly, the team understands from the start what technologies they can draw from for both the development process and problem solving.

Assemble the team based on skills that compliment the technology. The breadth of the SAS System usually requires developers to specialize. It is impossible to be an expert in all of what SAS software has to offer. If a business need can be solved in a variety of ways, developers will solve it with what they know and have expertise in. That is not to say that they cannot be flexible or that a creative process should be stifled. Just don't expect an AF developer to solve problems the same way that a Web developer might.

Strive for solid historical performance data by warehousing data by technology rather than by project.

Project estimation is better when you have experience in the same application development project. Past experience exposes the unexpected and provides a baseline from which an estimate can be given with confidence.

But SAS application development, with all its choices and options, makes finding an exact (or even close) match in the historical data rather difficult. A past data warehouse project can have the same number of data sources, on the exact same operating systems, with similar data content, and still vary so dramatically in the exploitation that the historical performance data quickly becomes to abstract to use.

Storing historical performance data by project has little use in an environment where technology solutions have limitless application. Level of effort in one project cannot be a predictor in another if the technology implementation varies. Exploiting data through the Web versus an 'out-of-the-box' solution with EIS, for example, does not translate into comparable level of effort, no matter how closely the data warehousing projects mimic each other in every other way.

Solid historical performance data can be much more relevant and easy to assess if it is warehoused by technology rather than by project. Pieces of past projects that match more closely to the current technology estimate challenge can be 'mixed and matched' to give a clearer picture to the appropriate level of effort.

CONTACT INFORMATION

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

Sarah Ragland
SAS Institute Inc.
SAS Campus Dr.
Cary, NC 27513
Work Phone: (919) 677-8000 x4843
Fax: (919) 677-4444
Email: sasssd@unx.sas.com