How predictive analytics can boost product development

Complex product-development projects are plagued by schedule slips and cost overruns. The up-front application of advanced and predictive analytics helps companies build plans they can stick to.

Arjun Balaji, Raghavan Janardhanan, Shannon Johnston, and Noshir Kaka
R&D projects are inherently unpredictable. When embarking on efforts to design complex things, companies often have little idea how long a project will take, what it will cost, or what they'll finally be able to deliver to the end customer. Their initial project plans are sometimes no more than educated guesswork.

For example, in an analysis of more than 1,800 completed software projects, we found that only 30 percent of them met their original delivery deadline and one in five of these did so by removing or deferring feature content. The average overrun is around 25 percent of the originally planned schedule. The performance of a sample of over 1,600 integrated-circuit-design projects was even more telling. Over 80 percent of those projects were late, and the average overrun was nearly 30 percent. Moreover, those projects were almost as likely to suffer an 80 percent overrun as they were to finish on time.

Delays, and the extra resources needed to counter them, mean higher costs too. The average budget overrun experienced by a group of factory-automation-software projects we studied was more than 10 percent. A fifth of those projects cost over 50 percent more than originally expected. Then there are the indirect costs. Delayed launches mean lost sales, opportunities for competitors to get ahead, and potentially damaged reputations.

The underlying causes of overruns
We’ve spent more than a decade investigating the root causes of R&D scheduling and budget challenges. In that time, we’ve interviewed hundreds of project stakeholders, including executive managers, technical leaders, and program and project managers. They highlight many issues that boil down to two primary root causes.

The first root cause is underestimating the complexity of the project. Managers and engineering teams are often surprised by the combined impact of all the features and performance targets and the cost of integration into a finished product. Engineering intuition tends to be linear, while the cumulative effect of increasing performance, features, and quality is highly nonlinear.

The second root cause is overestimating the productivity of the development team. Planners tend to assume that the issues that befell their previous project would be cured and that no new issues would crop up. They assume that specifications will not change and that resources will be available when needed. In practice, of course, such problems do affect almost every project.

How predictive analytics can help
Until recently, even companies that understood and sought to address these issues didn’t have effective tools for doing so. Conventional complexity metrics, like counting lines of code, story points, or function points (FPs) in software development, are difficult to estimate before the start of a project, especially one that requires many sprints from many teams to complete. Story points, by their nature, are qualitative and team specific, making estimation difficult when multiple teams are working on the same release. By their very nature, FPs focus only on function and not the actual effort drivers associated with implementation and validation, thereby leading to inaccuracies of greater than 60 percent in more than 50 percent of projects that use FP-based estimates. And traditional methods often fail to account for other external factors, like the programming and development styles adopted by the development team, multisite development, and the impact of challenges the team is facing for the first time.

Today, some companies are adopting a new approach, one that uses powerful data analysis and modeling techniques to bring new clarity to the estimation of project-resource requirements. At its heart, the new approach relies on the fact that, while every
development project is unique, the underlying complexity drivers across projects are similar and can be quantified. If companies understand the complexity involved in a new project, they can estimate the effort and resources required to complete it (Exhibit 1).

Doing that is harder than it sounds. Companies must collect a significant amount of data to determine what factors really impact project effort. But for practical reasons, the only useful factors are ones easily measured, consistently gathered, and known early enough to drive budget and planning decisions.

**Exhibit 1**  
A comprehensive complexity-unit measure solves for the limitations of effort-estimation methods.

- **Story points**
  - Qualitative, not quantitative; therefore, benchmarking not possible
  - Even measurements of same team require many iterations

- **Function points**
  - Effort intensive; often requires dedicated resource to create and track; therefore, hard to scale throughout organization
  - Difficult to do early external benchmarking (prior to architectural design), and generally cumbersome for benchmarking due to variations in function-point methods (eg, COSMIC, IFPUG, adjusted function point, Proprietary, Nesma, Mark-II)
  - By itself, not accurate enough for estimation because the method ignores factors that drive implementation effort and other factors (eg, team skill, tools used, hardware linkages, interfaces)

- **Use-case points**
  - Use cases are primarily a method of specifying customer requirements (vs classic Word document, business-requirements document, etc)
  - **Use-case points** are qualitative weights based on personal experience; therefore, no benchmarking/standardization
  - Use cases by themselves are insufficiently accurate for planning; some are big, others are small, etc

- **Numetrics complexity-unit measure**
  - Accepts stories, use cases, customer requirements, and function points as primary inputs, and automatically sizes them by also considering ≥20 measures (some optional) to improve accuracy
  - Useful for benchmarking and planning because of standardization of inputs, common predictive models, and a large industry database of projects
  - More holistic consideration of parameters that drive effort, leading to higher estimation accuracy (typically ±10% or lower) vs other methods such as story points/function points (±30% or higher)
  - Integrates seamlessly with current methods and tools that serve as inputs to the complexity calculation
In addition, analytical models provide a powerful new way to deal with constraints. A company can model the resource requirements of multiple projects scheduled to run concurrently, for example, to see if there are any points where those projects will demand more staff than it has available for a specific role. With warning of such resource bottlenecks, it can take appropriate action—adjusting the schedules to separate the peaks in demand, bringing in contractors, or outsourcing part of the work. Similarly, the models will show if an aggressive budget or timeline can be made achievable by adding more resources. And if it can’t, the company can run what-if analyses to evaluate the impact of dropping certain features or simplifying performance requirements. That allows a much more thoughtful, fact-based discussion, far preferable to missed deadlines or being forced to drop features at the last minute because they weren’t finished in time for launch.

Predictive analytics have already have transformed the outcomes of some high-value projects (Exhibit 2). As an example, at one company, a project to create a derivative of a newly released product was originally expected to take just 300 person-weeks of effort. The project’s planners arrived at this estimate on the basis that 90 percent of the new design would be carried over from its predecessor. When they reevaluated the plan using analytic models, they found that the project would actually take three or four times as much effort. The difference arose because while the amount of truly new work was small, it was widely distributed and affected nearly every part of the architecture. That meant significant extra testing and integration work, which the analytical models identified. Once the company understood the work involved, it changed its plans, keeping the team that developed the original product together to work on the derivative, and ultimately delivering it on time.

In another example, a company had a tight deadline to complete a new release for a big customer, with...
The use of function points in the industry, ISBSG, October 2016, isbsg.org. Right number of the right people on their projects at the right time, they also enjoy R&D-productivity improvements of 20 to 40 percent. For companies, that means lower costs and lower risks—a powerful combination of benefits to have in a highly competitive environment.

Organizations that apply analytics and predictive tools to their product-development and project-planning processes see a dramatic reduction in schedule slippage. And because they can put the right number of the right people on their projects at the right time, they also enjoy R&D-productivity improvements of 20 to 40 percent. For companies, that means lower costs and lower risks—a powerful combination of benefits to have in a highly competitive environment.

1 The use of function points in the industry, ISBSG, October 2016, isbsg.org.

Arjun Balaji is a partner in McKinsey’s Bengaluru office, Raghavan Janardhanan is a partner in the Chennai office, Shannon Johnston is a specialist in the Toronto office, and Noshir Kaka is a senior partner in the Mumbai office.

Designed by Global Editorial Services. 
Copyright © 2018 McKinsey & Company. All rights reserved.