Article from

The Modeling Platform

November 2019

Issue 10

Confessions of an Efficiency Junkie

By Jeff Samu

Throughout my career, I've attended many all-employee meetings. And in almost all of those meetings, I heard about how important efficiency is:

- "We must get more efficient!"
- "Our goal is continuous improvement!"
- "Process improvement is a major key to our future success!"

The orders were always clear: In order to succeed, everyone would need to become more efficient. Armed with this directive, I would set out to see where I could improve efficiency. In most cases, I would look at a specific process that I was responsible for and think about where the bottlenecks were. I made a habit of rebuilding my processes, introducing macros to eliminate manual steps and make them run faster. I developed strong coding skills to generalize the macros and allow them to be applicable for broader uses. I started playing around with Office libraries and application programming interfaces (APIs) to enable my programs to talk to others without my having to pass the information back and forth. I expanded my reach beyond my own processes and developed tools that were helping others to automate their work. Every day, I was making the company more and more efficient.

Or so I thought.

WHAT IS EFFICIENCY?

I had never really questioned what it meant for something to be more efficient. In my mind, an efficient process was one that ran fast when I clicked the button. As long as I was improving runtime, I was making things more efficient. I could take a process that would take a week of manual data preparation and condense it down to 15 minutes of machine time. I could click a button, go to lunch and return to find my custom-built digital servant ready with my results.

Then, one day, I ran into an issue. One of my beautiful time-saving processes had crashed. At first, I was confused. What

could have possibly happened? It had worked well the previous time I had run it. I combed through the code and found nothing that would raise an alarm. I checked the inputs and those looked good as well. I couldn't understand what went wrong.

Eventually, I discovered the source of the issue. One of our Excel source files had a field that had contained projection dates. In previous quarters, these were entered as text (e.g., 01/01/2019). In the current quarter, the field looked identical, but Excel was actually storing the date as a number (43466). I changed the code for the process so that it interpreted the date correctly, and the process ran without issue.

Except, the next quarter, the format changed back. Since I had updated the code the previous quarter, the process was crashing again. I updated my code again to include a check to see how the date was being presented, and to process it correctly regardless of the presentation.

Meanwhile, I had another problem. I had submitted my results from a separate process to management and was receiving a lot of questions about the numbers. Again, I was confused. My efficient, automated process shouldn't have made mistakes. I reviewed the inputs and noticed a row had been inserted in one of the source files. My process was structured to take a number from a certain cell of the workbook, but that number no longer meant what it had in the past, and the number I was now pulling was different by a factor of 10. I coded some additional logic to allow for more flexibility in the inputs—to look for data labels rather than blindly pulling a value from a particular cell in the workbook.

Then the phone rang. One of my business partners in a less-technical area had a question about a process I had helped him with. It was working fine, but he wanted to make an enhancement and didn't know how to go about doing it. He could follow the code to some extent but wasn't sure where to begin to make the changes he wanted. I realized that each time he wanted an enhancement, he would come to me and I would need to make every update because my ultra-efficient code, which ran as fast as lightning, was hard for anyone else to understand and modify.

These experiences showed me that my initial view of process efficiency had been rather rudimentary. I had a localized view of efficiency: I wanted my components of a process to run fast and structured them accordingly. I realized I had made a number of faulty assumptions in my process design. I had assumed that:

1. My inputs and outputs would be static.

2. Automation was always the best approach.

At its core, efficiency is about trying to get the greatest value from limited resources.

3. The controls in place were the only ones needed.

4. Any solution I built would always be the best one.

5. Others had the same technical skills I did, and they would be able to maintain the processes.

6. A single program used for different areas and needs would be more efficient than individual programs.

7. Any incremental gains to any process were worth pursuing.

Ultimately, I had assumed a definition of efficiency that I was starting to question.

So then, what *is* efficiency? When management says they want something to be more efficient, what are they really asking for? At its core, efficiency is about trying to get the greatest value from limited resources. It's about building solutions that can answer specific business questions so actions can be taken, without requiring an onerous amount of work. Fundamentally, it's about trying to achieve the best balance of quality, resources and time.

QUALITY

The first element of efficiency, and the most important one, is quality. If a process doesn't help me answer the fundamental question that I'm asking, it cannot be efficient. If my valuation software can't calculate a reserve properly, or if my projection model can't account for a benefit that 90 percent of the in-force policies have, or if I'm developing a hot new product feature I can't price, I need to search for (or build) a solution that can.

That is not to say the quality needs to be perfect. Valuing an exotic product feature that is no longer available and which only 10 clients have may not be necessary. There are many good reasons why model compression or simplifications may be used, and as long as these don't materially affect the results, solutions that use them can still be considered high quality and perhaps even more efficient than ones which do attempt to model everything.

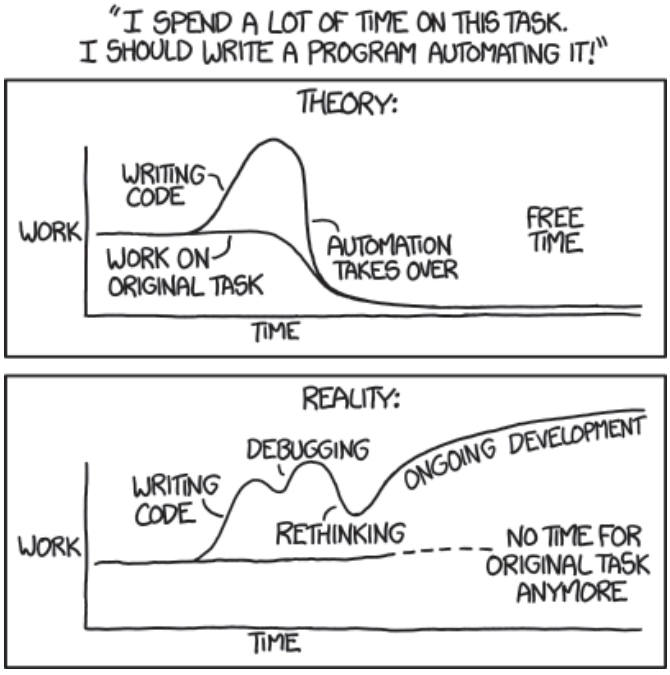
Controls are an integral part of assuring quality. It is critical that effective controls be in place to ensure the results are reliable. These can include validation of the input data, logs of the input files, logs of errors and reports with key values throughout the process. Without these controls, the end result can be suspect, which can lower the ultimate efficiency, as more time needs to be spent reviewing the results.

Quality should be considered on an end-to-end basis, from the initial inputs to the final reports. If my modeling software has some limitations, but I have a post-valuation process (PVP) that can account for these limitations, then the quality of the process as a whole can still be good. A caveat to this is that handoffs between different components in a process can introduce additional risk. In this case, data may be interpreted differently between the components and issues may be more difficult to trace, which can render the overall process less efficient unless appropriate controls are in place.

RESOURCES

When most managers think of resources, they usually think of the number of people involved, and for good reason. Personnel costs are generally the greatest component of a department's budget, and it is generally much harder and more expensive to add or shift people than it is to add technology resources.

When considering efficiency, though, it's important to consider not just how many people are involved, but also what their skills are and what value they are bringing to the process. If a process requires a very specialized skill to build, it will be more difficult to find someone to build it, more expensive to implement it and more challenging to maintain it. There may be only a few hundred actuaries who are intimately familiar with a given modeling platform, but there are thousands of actuaries who can understand Excel, and hundreds of thousands of programmers who are fluent in C++, Java, SQL or VB.net.



Source: Munroe, Randall. Automation. Jan. 20, 2014. *https://xkcd.com/1319/.*

This variation in skills becomes more critical when the users and developers are different. If users can't understand the process, they won't be able to maintain it, and the developers' specialized knowledge will be needed for updates. Over time, this can result in the developers solely supporting existing applications and not being able to build new solutions, constricting their ability to address bigger issues and reducing their overall value.

Resource availability and opportunity cost are often overlooked. When one of my customers asked for a particular fix to a process I was supporting, I remarked it would take an hour to fix, but it would take me two months to find that hour. Every hour I spent on that process was an hour I couldn't spend on something else, and since I was the key technical resource for a number of critical processes, my availability had to be prioritized. As a result, while each individual process may have been quite efficient on its own, the overall efficiency of the portfolio suffered.

TIME

When making a process more efficient, saving time is often the goal. When I would redesign a process, my goal was often to reduce runtime by a factor of 10 or more. If it took a day to do before, I wanted to get it down to less than an hour.

But runtime should not be the only time considered. The measurement of time should be all-inclusive, incorporating runtime of a calculation engine with time spent in design, development, debugging, testing, enhancements, review and reporting. Suppose I have a monthly process, and I reduce the runtime by an hour each month, but it took 20 hours to code the new process, five hours to test and debug it, and an extra 10 minutes to review the inputs and results each time to make sure the process didn't miss anything. Investing that time may not give the desired payoff and may even reduce the overall efficiency.

Furthermore, the amount of time isn't as important as the value of that time. The value of time is not the same for each person and can change over time. The value of time for an entry-level actuary is quite different from that of a chief actuary. A

Source: Munroe, Randall. Data Pipeline. Oct. 3, 2018. *https://xkcd.com/2054/*.

valuation actuary would likely spend 10 hours in the middle of May to develop something that saves one hour in January. Opportunity cost needs to be considered in these valuations as well, as the same time cannot be spent on multiple projects, so the relative value of the projects will influence the value of time spent on them.

CONCLUSION

When I had considered efficiency in the past, I always thought of it as an exercise in improving runtime through automation. Today, I realize it's not nearly that simple. Efficiency is about balancing quality, resources and time in a way that makes sense for that particular application. There is no universal approach to efficiency that works for all situations. But by considering the requirements of a process and the conditions of the resources, you can make intelligent steps toward making your process more efficient. ■

Editor's note: For a deep dive into the process assumptions, see the expanded version of this article on The Modeling Platform *digital edition at* https://sections.soa.org/view/society-of-actuaries/the-modeling-platform.

Jeff Samu, FSA, MAAA, is a vice president and actuary in the Business Architect group at Prudential Financial. He can be reached at *jeffsamu@hotmail.com.*