Measuring the Effects of Laboratory Automation: The power of empirically derived models

By Gerry Fitzgerald[1] and Jim Swanson

Abstract

Laboratory data systems are automated for a variety of scientific and management reasons. A key part to maintaining these systems is to regularly assess the impact that automation has had on the laboratory and the organization as a whole. SmithKline Beecham R&D is using a number of different types of measurement as well as a number of different tools for measuring how automated laboratory systems are affecting the workflow and information flow in the laboratory. This targeted program of metrics has increased management confidence in laboratory automation efforts, helped anticipate data processing bottlenecks, and highlighted end-user support needs.

Introduction

SmithKline Beecham operates a complex ensemble of laboratory automation applications. These applications have been designed and implemented through a variety of technological eras and for a wide variety of scientific and business objectives. We must constantly determine if the current systems are the optimal methods for meeting the needs of R&D, or if new systems are required. We also understand that the automation of a laboratory fundamentally changes the operation of the laboratory, often in unpredictable ways. Allocation of our limited laboratory automation resources also demands that we proactively identify opportunities for improved automation.

One advantage in our environment is that our users are quite technically sophisticated. However, their area of expertise is biology, biochemistry, or other areas of biotechnology. Only a few have (or need) a sophisticated understanding of how their computer systems work. They may not notice error messages, or may blame poor system performance on general conditions, rather than on faults that have crept into their particular applications. Alternately, some users may be "pushing" systems beyond their designs out of need for a new system, and inadvertently introduce problems that were not anticipated. By monitoring applications in a standard way, we can isolate application problems from system problems.

Few people argue about the need to monitor applications, measure performance, and report key indicators to management. We hope to go beyond those "motherhood and apple pie" pronouncements and propose a detailed strategy for an automated laboratory. The key questions asked by most laboratory managers asked to introduce metrics are

1. What parameters to measure?
2. How to measure those parameters?
3. How to report the information gathered?

Our starting point is always "Why do you need metrics?"

Experimental

Our quest for a targeted, effective system of laboratory automation metrics began with surveys and interviews of senior management. They were to be the primary audience for the information from the system, so it was only natural to ask what information they would be seeking. After several interations of "blue sky" and "hard realities," we obtained a short list of six key objectives that the metrics system should work toward. These were:

1) Capacity Planning -- when to purchase new capital equipment (instruments, computers & peripherals), and in what amounts, to meet anticipated needs

[1] SmithKline Beecham R&D. 709 Swedeland Road, M/S L-331, King of Prussia, PA 19406-0939.

2) Cost Justification -- proving the cost / benefit justifications for the original systems to be accurate (or even conservative)
3) Allocation of Resources -- which lab automation systems had the most potential for improvement?
4) Performance Improvement -- what could be done to make the performance (uptime / response time) of existing systems more predictable?
5) Job Satisfaction -- what could be done to show the people who implemented the system (often with considerable effort) that their work had paid off?
6) Process Improvement -- what opportunities still existed for improving the workflow in the laboratory, and what opportunities existed in improving the service level provided to users of the system?

The next step was to determine how to generate measurements that would meet these objectives. We were also limited by rather severe resource constraints for this project, as it was designated as "overhead to overhead." Our measurements had to be non-intrusive on the existing systems, and overall all of our metrics were to use no more than 2% of the available CPU. We also had to present our information in simple terms, avoiding computer jargon. We were also determined to automate the measurement and analysis process, as manually producing such information would not improve our job satisfaction. Two advantages that we had turned out to be key. One was that we did not have to be overly precise in our measurements, as resources were allocated and purchased in rather large chunks in relation to the resource usage of the laboratory automation systems. The other advantage was that we could focus in on just a few issues rather than trying to measure everything that could be measured.

The first set of metrics was aimed at measuring captial resource use. We needed to know for each application, what the total use of CPU, Disk, and Memory was, so changes could be translated into budget estimates. Operating system utilties existed to aid in gathering this information. The capital resource use was related to the volume of work (samples / transactions) processed during the same time period. This would, over time, generate curves showing how resource use varied with volume of work, a key measurement for capacity planning.

We added to this allocations to each application for maintenance costs for the software licenses and hardware. This became the financial cost of ownership. The support effort attributed to the application also had to be measured, and its cost added to the financial cost to obtain a total cost of ownership.

The benefits of the application would be measured in terms of the volume of data, samples, or transactions processed through the system. These benefits would be related to the performance of the system. The slower the system became (as data accumulated or as the workload of the lab increased), the less incremental benefit the system would have. Part of that performance rating was also to be based on the effectiveness of support. Penalties for downtime, errors, and long response time to support questions would decrease the benefit of the system.

Of course, the true financial impact of these things would be very hard to determine. However, by presenting the information in units clear to the laboratory (samples processed, hours connected, dollars of maintenance paid, and maintenance hours required), managers could use their own estimates of the value of these things to arrive at their decisions.

We also began this project with a set of prototype applications. Rather than measure every application, we chose the most important applications in each category, and used those as proxies for the whole task. We looked at RS-232 data collection, Chromatography, Data Analysis Tools, and general utilities. Our proxies were RIGEL (an in-house RS-232 data collection systems), PE Nelson's ACCESS*CHROM (the larger of our two main chromatography systems), RS/1 (the most used analysis tool in the laboratories), and VAX Mail (the most-used office automation tool). We then gradually expanded coverage as we developed the data collection and reporting tools.

Results and Discussion

The prototype applications provided excellent illustrations of the methods for acquiring and reporting metrics information. Data were collected both continuously and in descrete chunks. All of the data were aggregated and reported on a monthly basis. The goal set was to report the data no later than the third working day following the close of the month. In some cases minor modifications had to be made to applications in order to automatically count the number of samples, users or transactions. In most cases, simple once per month procedures could gather the required information, process it, and print & mail the reports.

For example, the RIGEL application required a variety of measurement techniques. First, the application was modified to log usage by each user and by each instrument collecting data. The application would also log the number of samples in each run. A separate class of problems was created within the help desk problem tracking system to record RIGEL specific calls. [Previously all lab automation calls had been lumped together.]

The chromatography data had to be gathered without modifying the application, as this was a purchased, rather than a developed package. The number of RAW data files produced by the system was used a a proxy for the number of injections on the system. A method of scanning the RAW data files to track the instruments used was devised and discarded, as it took too long to run. Total disk usage of the RAW, processed, and method files was measured by looking at the creation dates of the files. Performance and CPU load were measured by re-integrating a standard run of 100 raw files. The load of data collection was assumed to be negligible. Best results were obtained after we copied the files before re-integrating, to mimic the current level of disk fragmentation.

RS/1 usage was trickier to measure. We used a batch procedure to scan the users logged into the system. The batch procedure used a DECUS utilty, WHOZON, that would display the image name that the user was running. [The procedure must be installed with privileges.] The number of users running the RS/1 image was logged to a file every 15 minutes by the batch procedure. Performance was measured by running a short standard job that generated a standard curve and plotted a set of data every 60 minutes during prime time and recording the system resources used. Similar procedures were used for the mail utility. In fact, the batch procedure has been expanded to simultaneously monitor over 50 utilities to date.

The reports generated from this information had to be carefully crafted to present the information in a suitably summarized fashion. We finally settled on a flexible set of reports. Managers could request any subset of the reports, depending on the applications of interest to their groups and their responsibilites for those applications (throughput, capital improvement, support). Every month, we produced a two paragraph summary of the metrics, modelled on a weather report. This summary is included in the departmental monthly report circulated to senior management.

Graphical trend reports have been the best received. To smooth out normal, short term variations in workload, moving averages are employed. These trends allow senior managers to predict when capital equipment will be needed. Often, they have been quite suprised at the volume of samples, transactions, data, etc., that flows through the laboratory. We try to highlight milestones for the managers, such as "50% increase in sample throughput since the system was installed," so that they may convey congratulations to the laboratory. Unanticipated decreases in system use or throughput usually bring a visit from an analyst to find out if the workload is down or the system is not meeting users' needs.

One particular report on the utility software has saved us quite a bit of money. By analysing the number of simultaneous users for a particular utility, we are able to negotiate improved, and less expensive licensing agreements with most of the vendors. These license agreements also give us increased flexibility in distributing the application to multiple platforms. The monitoring on RS/1, for example, shows a continued decline in use. This corresponds to a major campaign of putting PCs into the laboratories. Anecdotal evidence points to the replacement of the package by PC based spreadsheets and graphics packages.

Another very useful set of reports have been the support reports. These reports show the number and types of calls handled by the support analysts, and the amount of time these analysts spent responding to those calls. Each application has between five and a dozen call types, which are general types of questions or problems with that application. For example, in a data acquisition application, the problem type can be a communications problem, a parsing problem, a documentation problem, or even a user training problem. These problem types can be accumulated over all data acquisition applications to give us an overall view of our communications reliability, documentation quality, and user training needs. Our conclusions are submitted to senior management along with the costs (in person-hours) that could be avoided.

Successes

The models that are used for the various applications we are tracking have provided us with a variety of successes. These successes have manifested themselves as tools, guidelines, and benefits not only for the applications we are currently tracking, but also affecting areas such as resources, finances, and job satisfaction.

The information gathered from a particular application gives a good overview about usage, performance, and support requirements. Usage statistics yield information concerning processor constraints such as CPU and I/O load, license requirements for multi user based products such as RS/1, and capital needs by estimating future usage from current trends. Expectations on performance from the user community has been more proactively met due to anticipating resouce requirements such as computing power, people, and finances. Support of these applications has also been enhanced in areas such as error tracking, improved response time, and coordinated support.

The benfits of these models has allowed for guidelines to be established for future applications that are either developed or purchased. These guidelines include the ability to report on usage and performance statistics and information about the application to perform proper problem tracking. As these guidelines are implemented the burden of support, resource allocation, and financial requirements becomes less of a best guess approach and more of an educated decision.

Successes due to these models has also been directly seen in financial savings and overall job satisfaction. As resources and licences are more accurately allocated and throughput increased due to error reduction, financial and analyst resources have been able to be better utilized. Thus allowing for more projects to be financed, better use of avaliable man power to complete more projects, and overall greater job satisfaction due to increased productivity.

Futures

The metrics that are implemented are applied to a small subset of the total number of applications that are currently in place. Since our role as developers and support personnel exists beyond the abbreviated list of currently tracked applications and systems it is essential that we broaden our scope. The benefits that we have achievied thus far can be exteneded to other areas such as various computer systems such as PCs, all existing applications, incorporating metrics in the design phase of future applications, and integrating metrics into support mechanisms.

To broaden our scope to the extent that would cover all applications and systems it is essential to educate senior management on the benefits of metrics. This is required not only because of the effort that would be expended to maintain the information retrieved from existing applications and systems, and the amount of effort involved to incorporate metrics into a new application, but also to get their support that this does show a benefit and that it is effort well spent. We feel the time spent developing the metrics and tracking the applications has been time well spent. The benefits have greatly outweighed the effort involved. That is the beauty of empirically derived models, it is based on experience and simply, what makes sense.