A comparative study of system response time on program developer productivity

by G. N. Lambert

Skilled programmer time and computer time and resources are valuable. Earlier studies had shown that added computer resources can decrease system response time and increase programmer productivity significantly. A controlled study has been made to determine whether that finding is true for the particular conditions in another program development organization. That study is reported here. Programmer productivity increased sixty-two percent with subsecond system response time. A new finding is that individual group project offices lead to greater efficiency than large open rooms.

IBM Information Services, Limited (known by its initials ISL) is a company within the IBM Europe organization, whose mission is to develop software for internal use by the IBM countries. This software is to be run either centrally at Portsmouth or locally in the countries themselves. As a development organization, therefore, this company has many similarities to other IBM customers. It shares the same objectives for productivity and must justify its new equipment in the same way.

This paper reports a study we conducted to assess the benefits of fast response times as one way of improving our program developer productivity. The purpose of the study was to justify to higher management the case for improved data processing resources, based on savings in numbers of developer personnel and/or faster delivery of applications.

In 1979, Doherty and Kelisky[1] reported several valuable observations on application development productivity. One of the most significant of these, and the one we were concerned with in our study, is the following: Each second of system response degradation leads to a similar degradation added to the user's time for the following request.

These authors attribute this user delay to a short-term memory buffer. This buffer holds the sequence of actions that a terminal user has thought out in advance of his first action and which, if the response time is short enough, he can work through without having to rethink each transaction. If, however, the response is long (i.e., greater than one second), the user becomes distracted and has to think the planned sequence of actions each time. Thus a delay is incurred on every transaction. Doherty and Kelisky also highlight the disruptive effect of inconsistent response time, and show how inconsistency causes users to become upset, frustrated, and inefficient.

Doherty and Kelisky further show that, with the IBM Research Center program development load of nine million terminal interactions per month, a response time of two seconds gives rise to a user delay of another two seconds. Users would therefore lose four seconds on every transaction, for a total of 36 million seconds per month for all users. That is equivalent to ten thousand lost hours, or 60 persons, per month. IBM Research Center management thus justified and obtained facilities for ninety percent of their transactions to be processed within 0.5 second.

In another paper, Thadhani[2] shows the improvement in transactions per user per hour as response time decreases. That paper discusses a situation with a high percentage of transactions defined as trivial and human intensive, rather than those that are computer intensive, such as compilations or test activity. Thadhani's curve is now well known and confirms the finding of Doherty and Kelisky that productivity increases significantly when the response time is reduced to less than one second.

In our study, we reviewed the preceding work and classified it as to its relevance to our environment and consistency with our objectives to improve our development productivity. We noted, however, that Thadhani's results had been based on an increase in transactions and not necessarily in real work output. There appeared to be an obvious relationship between transaction rate and work produced. In order for us to justify new hardware, however, it was necessary to confirm that we were observing a real improvement and not simply a change in work pattern due to different user habits when confronted with instant response.

We conducted a study in ISL, using a development project that was given subsecond response time facilities. We monitored that project's performance against that of a control group doing similar work on the same machine with our usual response times. The project was the second stage of a two-stage development, the first stage having been completed prior to the start of the study. In addition to transaction measurement of the new project, the real work output of the development team (measured by function points per man month) would be compared with their performance in the first stage. *Function points* are a measure of the size and complexity of the work product of an application development project. They are based on the premise that work product size is related to the number of user inputs and outputs, logical master files, and interfaces to other systems. In addition, the function point count of a work product may reflect other factors inherent in the development process.

The ISL subsecond study

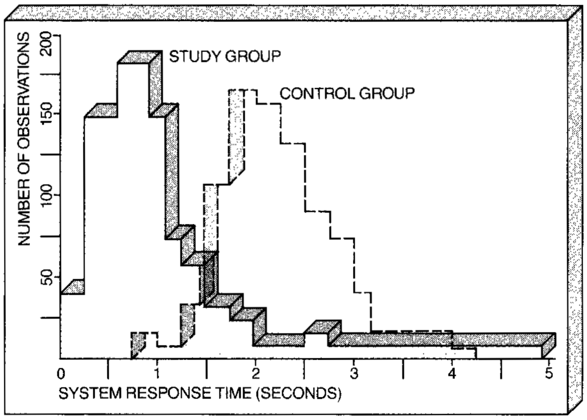
The software development project selected for our study is known as the Machine Analysis Table (MAT) enhancement project. MAT is used to check incoming order configurations against a table of valid feature combinations. The second stage of the MAT enhancement system was a hybrid development of basic PL/I modules and an internal screen generator, known as Screen Handler/2. The project as developed originally incorporated MAT code that used the IMS Application Development Facility (IMS/ADF) to allow fast specification of the screen rules for use within ISL. The main logic code and surrounding IMS interface were retained for MAT Stage 2, but several modules concerned with screen I/O changed from being IMS/ADF-based to the internal screen control generator, Screen Handler/2. (This was done to conform to standard ISL external on-line systems access methods.) In our study, Screen Handler/2 control statements were written and the ADF rules removed. Other functions were added or enhanced within the PL/I modules.

The study project team consisted of five developers with mixed skills, from expert to trainee.

The project size was estimated, by a method known as Managing the Application Development Process (MADP), to be 3697 hours for tasks and 514 hours for management—a total of 35 man-months. The estimates were validated using two other techniques. The project was scheduled from November 1981 to March 1982. The scope of the project was the detail design from Hierarchy and Input Process Output (HIPO) to pseudocode. This was followed by coding and unit testing, and then integration testing of the screen rules and modules under the Batch Terminal Simulator (BTS). Full system testing was carried out under a separate project.

Study team and environment. The study project team consisted of five developers with mixed skills, from expert to trainee. The provision of subsecond response required that local lines be used. These were provided by locating the study team in a project room close to the machine room. The project room had the added benefit of relative quiet. (The normal

Figure 1 Frequency distribution of System Response Time



working environment for both the study and the control groups was an open-plan office where only remote terminals were available.) One terminal was provided to each study team developer, and their TSO logons were assigned high-priority service parameters to ensure subsecond response. This meant that two TSO priority classes were involved, one for the study group and one for the control group. The control group terminal ratio was 1.8 people per terminal.

Two processors were available to the study group, with shared DASD and common TSO sign-on facilities. The machines were an IBM System/370 Model 168 multiprocessor and an IBM 3033 uniprocessor, both running MVS and both providing TSO/SPF for interactive application development. There was no difference, from the study point of view, which system the study team used. They switched their terminals from one system to the other, according to response time and machine availability. The control group used the System/370 Model 168 multiprocessor.

Steps were taken to reduce the effect of the measurement process on the study team's performance (the Hawthorne effect). We limited their visibility of the measurement and analysis process, and maintained only normal management and professional contact with them.

Measurement techniques. The Resource Management Facility (RMF) was used to record transaction volumes and response times, and other data processing operations reports recorded user data. Details obtained for the study team and the control group can be summarized as follows. The number of TSO transactions within each fifteen-minute segment throughout each day of the study period was recorded. This quantity was broken down by TSO period. (Period 1 consisted of trivial transactions normally associated with TSO/SPF edit or browse interactions and were the type most relevant to this study.) Also recorded was the average number of study and control users on the machine for each of the fifteen-minute segments and the average system response times for the study and control user TSO transactions.

Overall work productivity was measured in function points.

Data were collected on TSO session times, computer resource usage, and batch job submissions, in order to evaluate the effects of subsecond response time work on the development environment. Overall work productivity was measured in function points, which were calculated from the design documents for Stages 1 and 2 of the MAT Project. Because the mix of on-line and batch code differed between MAT 1 and MAT 2, adjustments were made on the function-point values to reflect the differing productivity rates of the on-line and batch aspects of system development.

Results processing. The daily transaction volumes and numbers of users were fed into a simple program to calculate User Productivity (UP). This was stored for each fifteen-minute segment of time. User Productivity is expressed as follows:

UP = Number of interactions per hour/average number of users.

The average System Response Times (SRT) were used as input for each of the TSO priority classes for each fifteen-minute period. We decided that, for the purpose of work output comparison, response time at the terminal was the important measurement and that line delays on remote terminals must be taken into account. The SRT was, therefore, weighted for control group users. The weighting was determined by observation and was set to 0.5 second for fewer than twenty users and 1.0 second for twenty or more users.

The times of day when there were not enough users on the system to be representative of a typical load were not considered. Therefore, readings for the hours from 0930 to 1130 and from 1400 to 1630 only were selected. There were also times when readings were corrupted by problems in the environment. In order to avoid these extremes, two standard deviations were taken at either side of the mean System Response Times to give a 92 percent sample of the total.

Combining both the study- and control-group data on the same graph and considering only the human-intensive trivial transactions, the calculation program produced coordinates for a least-squares fitted curve of the following polynomial form:

The net result was an improvement of 62 percent in terminal user productivity.

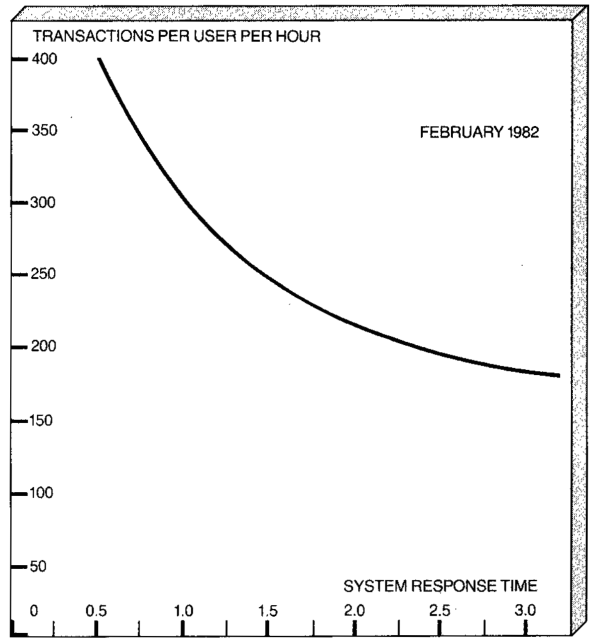
$$y = ax^{-3} + bx^{-2} + cx^{-1} + d$$

The resulting curve represents User Productivity versus System Response Time. The curve was plotted for each month of the study and for the bandwidths 1000–1100 hours and 1400–1500 hours for the whole period. The curves were plotted on scatter graphs showing the individual readings.

Results of the study

System response times. System Response Times (SRT) for the study and control groups were distributed as shown in Figure 1. Based on the 92 percent sample, the average System Response Time improved from 2.22 seconds to 0.84 second, and the terminal user productivity increased from 160 to 258 transactions per user per hour. The net result was an improvement of 62 percent in terminal user productivity.

Figure 2 A typical monthly plot of user productivity

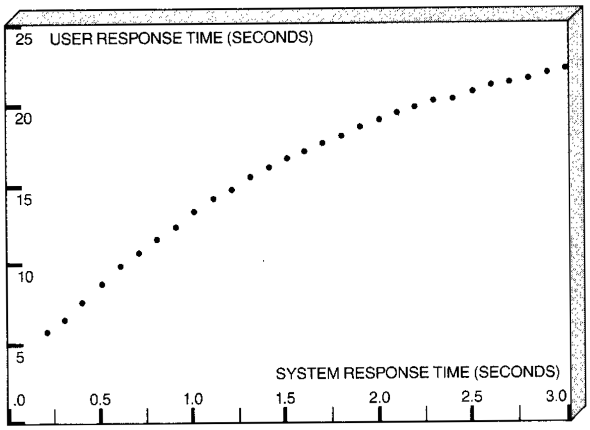


User productivity. The graph shown in Figure 2 is one of the monthly user productivity curves that we plotted, using the least-squares method, from data points for readings from the study group and the control group. Productivity increased rapidly as response time dropped below one second. Our average response time was 0.84 second. Clearly, we were not able to exploit the full potential of subsecond response time, as we were unable to average less than 0.84 second. One fact that did emerge from the study was that our high-performing team members made better use of the subsecond facilities than did the less-experienced members. All the top transaction rates were attained by the better-performing members of the team.

User response time. The improvement in productivity shown on the graph in Figure 2 is made up of the following two components: (1) improvement in the system itself, and (2) improvement in user response time as a result of improved system response time.

If the improved system time is subtracted from the total transaction time, the remainder is the time taken by the user to think about the reply and to enter it. Assuming that over a four-month period the typing component can be regarded as a constant, the decreased time per transaction represents shorter

Figure 3 User component of the total transaction elapsed time



thinking time by the user. The graph in Figure 3 shows the user component of the total transaction elapsed time. At a System Response Time of 2.22 seconds, for example, it took 20 seconds to reply to each transaction, whereas at 0.84 second it took 12 seconds to reply. This is an improvement of 8 seconds, or 40 percent of the original time. Our findings lend support to the validity of the Doherty-Kelisky short term memory buffer theory.

MAT schedules. The following is a comparison of the overall hours and man-months planned for the study team tasks and those actually worked.

- Planned for team tasks: 3697 hours; 30.8 man-months.
- Actually worked by team: 2252 hours; 18.7 man-months.
- Planned schedule: November 9, 1981 to March 22, 1982.
- Actual schedule: November 9, 1981 to February 22, 1982.

Thus there was a time saving of 1445 hours or 12 man-months, plus management time. The study team produced a 39 percent improvement over the time planned.

Productivity measurement by lines of code. As a check against our other measurements, a comparison was made of lines of new PL/I code developed for the two MAT projects. Under the conditions of our study, a line-of-code check is valid, and in this case, it confirmed the transaction rate productivity readings, as follows:

- MAT 1 had 10 730 lines of code in 1172 task hours, or 9.2 lines of code per hour.
- MAT 2 had 12 554 lines of code in 828 task hours, or 15.2 lines of code per hour.

This gave a productivity improvement of 65 percent for MAT 2 over MAT 1. MAT 1 was the control in which regular computing facilities were used for comparison with MAT 2. MAT 2 had subsecond response time available to the study project team.

Work output. The work output of the MAT 2 project, in terms of function points, was calculated to be 364 function points. The MAT 1 project was exactly half of this, or 182 function points. With a MAT 2 team task effort of 18.7 man-months and a MAT 1 task effort of 17.5 man-months, the following productivity resulted:

- MAT 1 productivity is 182 function points in 17.5 man-months, or 9.2 function points per man-month.
- MAT 2 productivity is 364 function points in 18.7 man-months, or 15.2 function points per man-month.

Purely on the basis of function-point count, MAT 2 productivity was 86 percent greater than that of MAT 1, which yielded a savings in effort of 46 percent.

Because function points can be developed twice as easily in on-line systems, for a true comparison of work productivity it is necessary to adjust the two function-point counts to allow for the differing proportions of on-line and batch work in the two stages of the MAT enhancement project.

We first converted both projects to the following equivalent batch function points:

- MAT 1 function was 160 function points.
- MAT 2 function was 269 function points.

This gave the following true work output comparison:

- MAT 1 productivity was 160 function points in 17.5 man-months, or 9.1 function points per man-month.
- MAT 2 productivity was 269 function points in 18.7 man-months, or 14.4 function points per man-month.

Thus there was an increase in work productivity of 58 percent, which gave a savings of 37 percent in effort.

Quality. One of the areas that could have suffered with faster data entry was that of quality. When the MAT 2 system was subsequently system-tested, however, it was found to have fewer errors than the MAT 1 project, even though MAT 2 was larger. The numbers of errors were, in fact, both small—eleven and eight trouble reports for MAT 1 and MAT 2 respectively. Errors found were thus not statistically significant, but the small number of errors indicates that quality had not suffered.

TSO resource usage. The TSO connect hours for the study team increased with their extension of terminal usage to cover on-line pseudocode and documentation, and with the availability of a terminal for each person. This extra terminal usage has meant that these previously manual tasks were done more quickly. Thus quality improved because it was easier to update an on-line document than to rewrite a handwritten one.

The amount of computer resource used also increased. The number of European Work Units (EWUS) used per minute of TSO session time increased by 42 percent, and the work output (in adjusted function points) increased by 58 percent. This supports the productivity findings. One expects to use more machine resource in doing a given amount of work in less time.

The team noticed that LOGON/LOGOFF time became a major part of each work session, for both the study team and the control, consuming time and computer resources. Measurements showed that the study team's use of their terminals increased to 72 percent of their task hours, which indicated more intensive development activity when a one-to-one terminal ratio was available.

Interpretation of results

When the study commenced, it was apparent that the project-office environment that had been created for the study team to provide subsecond response time was, in itself, contributing to their productivity. This was of no surprise to the team members, who share most developers' preference for project team offices rather than open-plan work areas. As the study progressed we concluded, however, that the two aids to productivity were complementing each other to an increasing degree and that the benefits of subsecond response time were being considerably enhanced by having an environment conducive to the concentrated effort that the machine was making possible.

In analyzing our results, we first considered trying to extract a factor for the environmental benefits. However, one of the main objectives of the study was to reproduce and confirm the results of References 1 and 2. Knowledge of those work environments led us to believe that their studies had been conducted with their users in small project rooms. Therefore, we retained the combined benefits of subsecond response time and project working environment.

A project office offering close team contact and low distraction plays a positive part in increasing developer productivity.

We believe that a project office offering close team contact and low distraction plays a positive part in increasing developer productivity. We know from previous work that the subsecond response time makes possible considerable productivity gains. We have proved that together a project office and subsecond response time constitute the ideal program developer working environment; productivity increases significantly when they are available together.

Having said that, we have to acknowledge that there were two variables in the comparison between our study team and the control group and that some of the benefit may be directly attributable to the improved environment. However, anyone who has observed a group of programmers working with subsecond response time realizes that no single environmental improvement can account entirely for the improved rate at which the programmers enter transactions. To confirm this observation, we have installed subsecond response time in ISL since the time of the experiment reported here. However, we have not yet implemented the environmental conditions used by the study group reported here. Our transaction rate per user is already regularly well above the rate achieved in the study. Therefore, although we have been unable to identify the two components that contribute to improved programmer performance individually, we have chosen to implement both the subsecond response time and the project office environment. Our present understanding of this situation is that any estimate of the benefit that is obtained with subsecond response time also depends on the characteristics of the current environment. Thus, a certain amount of indeterminacy is inherent in measuring programmer productivity that depends on the subjective estimate of the contribution of the environment to the total improvement.

The team used their terminals for longer periods and for extra tasks.

Study team comment. The study team members were invited to complete a questionnaire on their experiences when working with subsecond response time. The following is a summary of their replies. They were happy with the working environment provided for the study, especially the one-to-one terminal ratio. Team members came to regard the machine as a real aid to productivity and good work. The team used their terminals for longer periods and for extra tasks. They spent seventy to eighty percent of their task time at their terminals and enjoyed being able to complete whole tasks (such as a module design or encoding) in one session. They found that this continuous work helped concentration, which in itself aided productivity. However, they felt vulnerable if the machine was unavailable or if response deteriorated. Study team members were not aware of an improvement in the quality of their work (which was very high), but they were aware that it was much easier to achieve their normal quality. In addition, they were happier to go to the terminal and fix small defects in design documents or in coding. The overheads of accessing, correcting, and saving data no longer deterred them.

Concluding remarks

From the results of our study, given an average System Response Time of 2.23 seconds which was common at the time, and an average System Response Time of 0.84 second that was available to the study team, we draw the following conclusions:

- User productivity increased on average by 62 percent. This represents the increase in trivial transactions entered through the terminal in a fixed period of time. The gains come, in part, from the actual reduction in machine delay, but mainly from the associated decrease in User Response Time.
- User Response Time decreased on average by 40 percent. That is, the time taken for the user to respond to the terminal decreased as System Response Time increased. This supports the short term memory buffer theory.[1]
- Work output productivity, as measured in the MAT 1 and MAT 2 projects, increased by 58 percent. We can, therefore, see consistent gains across terminal productivity, user response time, work performed, and schedule completion.
- Although the study project used TSO resources at a 40 percent higher rate, it did not use greater resource per module developed.
- In addition to the productivity gains, the study project also gained by a reduction in overtime to almost zero and by early installation of the finished product.

Based on the results of this study and on the project statistics for the MAT Stage 2 development, the facilities of subsecond response time should be made available to as many developers as feasible. Higher productivity encourages developers to do more work at the terminal. Thus each developer should have a terminal as part of his work station, in the same way that a desk and telephone are provided. The provision of project team offices enhances the benefits of subsecond time and should be considered wherever possible. As developers spend more time at terminals, high reliability must be maintained, and perhaps fallback facilities should be provided by the computer center for day-to-day development work. The practice of using development facilities as fallback for production systems at short notice should be re-examined.

The study project therefore recommended that subsecond response time facilities be installed in ISL together with a project development environment.

Emphasis should be placed on getting System Response Time consistently as low as possible. (The average of the study project was just less than one second, although the response times were as low as 0.2 second from time to time.) The facilities provided should include a terminal for every developer, a project work room, and higher machine availability. The benefits accrued from subsecond facilities, with or without better work areas, should be reflected in our estimating methods in order to ensure that they are realized in project plans.

As a result of this study, IBM Information Services Limited increased its planned hardware upgrades to provide subsecond response time to its development community. A plan has also been implemented to install improvements to the project development environment and to provide a terminal for each developer. Improved programmer productivity and increases in delivered products have fully justified the costs.

The new hardware is now operational and the one-to-one terminal ratio has been achieved. Detailed measurements of the system and user performance are planned. First samples are showing that transaction rates in excess of 400 transactions per user per hour are being achieved when response times of around 0.3 second are provided at the terminals. The settling down of the system, the waning of the "novelty factor," and the staged refurbishing of the work areas will all affect these figures in the forthcoming months, but it is clear that transaction rates are already being obtained that substantiate the results of this study and confirm the theoretical work of Doherty and Kelisky and Thadhani.

Cited references

1. W. J. Doherty and R. P. Kelisky, "Managing VM/CMS systems for user effectiveness," *IBM Systems Journal* **18,** No. 1, 143–163 (1979).
2. A. J. Thadhani, "Interactive user productivity," *IBM Systems Journal* **20,** No. 4, 407–423 (1979).

Geoffrey N. Lambert *IBM Information Services Limited, P.O. Box 39, North Harbour, Portsmouth, P06 3AA, England.* Mr. Lambert is currently Development Services Program Manager in the end user services group of ISL, where he is responsible for the strategy and implementation of end user productivity tools. He joined IBM in 1968 and has spent the last ten years in the application development environment. In 1980, he was appointed manager of a development department in the group responsible for the IBM Total Supply Management System. The group's work with new application development methodologies and techniques led to the setting up of the study described in this paper. Since the publication of the original report on the study, Mr. Lambert has become increasingly involved in the developer productivity arena and has lectured in the U.S.A. and in Europe on system response time and program developer productivity. His present appointment reflects the increasing attention being given to the support of both development and business end users.

Reprint Order No. G321-5208.