EDP ANALYZER

JANUARY, 1980
VOL. 18, NO. 1

MANAGING THE COMPUTER WORKLOAD

Yes, distributed systems, interactive systems, new network services, and so on, are getting most of the attention in the computer field these days. But many organizations, in developing their data processing plans for the next five years or so, have found that the bulk of their computer workloads will continue to be 'plain old batch processing.' The interactive workload is growing rapidly, but batch seems to be growing at least as fast. Some (and often a lot) of this batch workload will be entered via remote batch terminals. There are tools and techniques available to aid in managing this workload—for determining needed capacity, for scheduling daily production, and for handling perturbations in the workload. Here are some user experiences.

The Standard Oil Company of California (Chevron), with headquarters in San Francisco, is a major energy producer and distributor serving world markets. Annual sales are about $30 billion; *Fortune* magazine lists the company as the sixth largest U.S. industrial firm. Chevron employs over 38,000 people.

Chevron spent much of the 1970 decade in consolidating its data processing workload into two large computer centers in the San Francisco Bay area. In these two centers, there are a total of seven large IBM computers (models 3033, 168, and 158), plus two Amdahl V6s and one V7. Most of the input for these centers comes from remote batch terminals located throughout the United States and Canada.

The main incentive for this consolidation was, as is so often the case, the desire to greatly reduce the incompatibilities of programs and data files that developed during the 1960s. That consolidation has been completed. Now the people in Chevron data processing are detecting a growing interest among the users in distributed systems. But they want to move cautiously in this direction, to avoid having incompatible systems develop all over again. So centralized batch processing will continue to be a prime workload within Chevron for some years to come, though distributed

ISSN 0012-7523.

systems will be installed in the next few years for some applications.

The workload on the IBM machines at the two centers consists of the following two components.

Scheduled daily production. This is the predictable part of the workload, made up of application systems that are run on a scheduled basis (daily, weekly, etc.). Chevron runs about 16,000 jobs a month of this category; these jobs, in turn, make up about one-third of the batch computer workload.

Most of this scheduled work is run on the evening and night shifts, local time.

Unscheduled work. This part of the workload is made up of jobs that users enter as their needs dictate. It consists of (1) production jobs run as needed, (2) program development, test, and execution of 'one-time' jobs (such as engineering calculations), (3) the development and maintenance of application systems, (4) database queries and updates (about 90,00 per day), and (5) a relatively small amount of time-sharing (TSO) for system software programmers. (The bulk of the company's time sharing service is provided via National CSS software run on the Amdahl computers.)

Let us look at how Chevron manages the scheduled portion of this workload.

Scheduling daily production. Chevron has been using the OS/VS scheduling system from Value Computing Inc. since 1978. Previously, the company had used another scheduling system, but encountered trouble in making it work and in getting needed support. They then investigated other offerings on the market and picked the VCI system. "Since then, the Value system has done the job we expected it to," we were told.

When a new application is first set up and is going into production, a written agreement is made between computer operations and the user department. This agreement defines the job characteristics, such as the general amount of the input, when that input can be supplied, when the results are needed, and so on. Using the VCI scheduler, this new workload is fitted into the production schedule. At this point, computer operations can tell the user just when the input *must* be supplied, if the results are to be delivered on time.

At the two Chevron centers, the computer operations people are *not* job submitters. The users are responsible for submitting jobs, according to the schedules they have been given. The users must also submit any necessary date cards, JCL input, and so on. The two centers do have the position of 'batch job monitor'; the person filling this position can see (via a terminal) what jobs are in the input queue at any time and can compare this list with the printed schedule for the day. But the users have the main responsibility for seeing that their work gets into the input queue on time.

Of course, not all of the scheduled work runs as planned. Schedule changes can and do occur. These come about generally from user requests or from job characteristic changes. Let us consider those changes.

User requested changes. Some of the user requests involve *permanent* changes. Examples include application systems that have been enhanced or modified, at user request, that lead to a change in their production schedule. In other instances, the application systems are satisfactory but the users want to change the times that they submit input or receive output. Other user requests involve *temporary* changes—input may be late in arriving on a certain day, or an extra edit run is needed, or such. Generally, the user department contacts computer operations when things like this occur, to see when their work can be fitted into the workload. Because of the large amount of capacity for handling unscheduled work, computer operations generally can fit such work into the workload fairly promptly.

Job characteristic changes. Most of the scheduled jobs have fairly stable charac-

teristics, including volume of input. But some jobs do vary significantly in the amount of computer resources they consume. The VCI scheduling system keeps track of the actual job times and compares them with the expected job times—and adjusts the expected times as necessary. Based on the new expected times, the scheduling system may determine that input must be provided, say, earlier than previously, if the output schedule is to be met. Messages to this effect are automatically sent to the user departments.

Planning computer capacity. The question here is: how much computer capacity will be required each year, for the next two to five years, in order to handle our workload?

Because two-thirds of Chevron's workload is unscheduled, this portion of the workload dominates the capacity planning question. For this part of the workload, the planners look at usage trends over the past several years, and extrapolate those trends a few years ahead to get the basic growth. Any large new types of unscheduled usage are also considered.

For the scheduled workload, the planners look at the usage trends plus any new applications that are expected to be added.

Based on these calculations, the amount of additional capacity and the time when it will be needed are determined.

Chevron has not made much use of the VCI scheduling system for estimating future capacity requirements, we were told. If the scheduled workload made up, say, three-fourths of the total workload, then such use of the system might be helpful by using it to simulate conditions at future points in time.

There have been some cases, though, where Chevron has made use of this simulation feature. In one instance, a remote data center was being phased out and the work transferred to one of the central sites. This workload had a tight daily schedule and involved a good deal of magnetic tape processing. A VCI workload control file was set up for this work, and was then tested against the workload control files of the two main centers, to see which center could best handle the new work.

Standard Oil Company of California is handling a large and growing computer workload. A fair amount of this workload is of a predictable nature, and the company has found the VCI OS/VS scheduling system provides an effective way to manage the daily production of this workload.

The workload structure

Before discussing computer workload management, it would be useful to identify the major types of work that data processing systems will be asked to perform in the future. We see four main components of the total workload: (1) system overhead, (2) batch workload, (3) interactive workload, and (4) computer message systems.

System overhead. It may seem incongruous to start out this discussion with the overhead item. But this item tends to be overlooked in discussions of workload. As computer systems take on more complex functions, the overhead goes up. Central system overhead may well be one of the prime factors that encourages users to get their own (distributed) systems.

System overhead includes the resources used by the operating system, database management system, communications executive, scheduler, job accounting system, checkpointing, rollback and recovery, reruns, makeups, and so on. The goal is to minimize the system resources consumed by these things.

Batch workload. This workload includes both locally and remotely entered work that is to run in a batch environment. Jobs are put in an input queue and are worked on (generally) in turn. There are several general types of work that make up the batch workload. Here are some.

Streams of jobs, with precedence relationships—the characteristic of most batch data processing applications. 'Payroll,' for instance, involves a series of jobs, starting with a variety of inputs and ending with many outputs, one of which is paychecks. The jobs must be done in a logical manner, such as validating input before files are updated.

'Spooling' of input/output. Since input/output devices are much slower than the computers (usually), the input/output operations are batched and performed separately from the updating runs.

Program development and test that is performed in a batch environment, with batch compilations and executions.

'Short cycle' batch processing, to approximate interactive service but with less expense. Transactions are accumulated for a short period of time (say, a few minutes) and then processed in a normal batch manner.

Report preparation, where a report requires the computer to examine some significant portion of a whole file.

These types of work make up most of the 'plain old batch workload.'

Interactive workload. This workload is both entered and the work initiated by the terminal user. Computer resources may be shared among a number of users simultaneously, where each is given the resources for very short periods of time, until their jobs are completed. We see this interactive workload as consisting of seven components.

On-line updates. Input transactions are validated as the user enters them and then the affected file records are updated immediately. Much of the use of micro-computers in data processing is of this variety.

Retrievals for queries and prepared reports. This will be a large component of future workloads, in terms of the number of requests processed. It involves the retrieval and display of designated data records or report pages.

Graphics output to CRT terminals. More and more use will be made of graphics for the presentation of report data. We believe that most such graphic outputs will commence to be prepared immediately after the user request has been entered.

Report preparation. As we have been discussing in recent issues, computer support for management will provide for on-line preparation of reports. The examination of a file will commence when the user requests it, not when a batch scheduler says it will commence.

Time sharing calculations. Conventional uses of time sharing—including the use of pre-programmed routines and models—will continue to grow.

On-line program development. More and more program development will be transferred from a batch to an on-line environment, as we discussed in our October and November reports.

Text editing, used for the preparation of reports, memos, letters, and so on, will certainly grow in use. One main use will be in conjunction with computer message systems.

Computer message systems. These systems represent the fourth major workload category for future computer use. We discussed these systems in our April 1977, September and October 1978, and May and September 1979 reports. This amount of discussion indicates the importance we attach to these new 'electronic mail' systems. They will become significant users of computer resources.

With all of this expected growth in the interactive and computer message workloads, why does batch processing continue to be important? Why isn't it gradually being displaced by these 'more modern' methods?

Why batch?

Some of our friends who work for large companies have pointed out an interesting fact to us. In the planning for the use of programming talents for new applications

over the next several years, most of the programmer time will be going for batch applications, not interactive. This is not the maintenance of existing batch applications that they are talking about; rather, it is the development of new applications that users have requested.

Why this continued demand for batch? We see three main reasons.

Processing that is time-related. Much of the data processing that organizations need is time-triggered. Cycle billing, payroll, month-end reports, financial period closings, and bank check processing are examples. People have become accustomed to operating this way, so that time-triggered processing will continue.

It is true, of course, that this type of batch processing can be combined with interactive processing. Transaction data can be entered interactively and either posted to the files or or accumulated. Then, when the proper time arrives, the transactions are posted, if needbe, and the file is processed to produce the needed outputs.

Processing that is event-triggered. An example of what is meant by this is the large file that is used only infrequently to produce outputs, although inputs flow in regularly. Data entry might be performed on the inputs, after which they are accumulated until a request for output is received (the triggering event). Then the file is updated and the output produced.

Economy. Interactive use is nice, but it does use more computer resources (usually) than batch. As mentioned earlier, short cycle batch might be used instead of an interactive system, if a short time delay can be tolerated. Where such economies are significant, batch processing will continue to be used.

So, all in all, batch will continue to constitute a large portion of the total computer workload as far ahead as we are able to see.

Workload management

The problem of computer workload management is concerned, to a great extent, with the question of how best to use shared resources. Most of today's computer-using organizations have some form of centralized data processing which serves a number of user departments. This centralized data processing will not be replaced in short order by distributed processing wherein each user department has its own computer. The management of shared resources will continue to be a challenging problem area.

A decreasing percentage of today's computers, we believe, provide *only* batch processing. More and more of them are also providing some types of interactive service, such as a query service and/or a time-sharing service. The batch work also may consist of both scheduled and unscheduled work.

For the scheduled batch component of the workload, automated aids are available for helping management make efficient use of the resources.

For the unscheduled work—both batch and interactive—management must provide sufficient capacity for handling some arbitrary peak load conditions. A charging mechanism also is usually needed, with increased charges during peak load periods, as a means for load levelling.

So the main elements of workload management include:

- Capacity management
- Charging for services
- Daily production management

This whole subject area is complex, and covers more material than is appropriate for one issue. In this report, we will consider the first two of these elements—capacity management and charging methods—only briefly, and will concentrate on the management of daily production.

Capacity management

The two major questions facing data processing management in connection with computer capacity, are: How much capacity do we currently have? How much capacity will we need, and when, to handle our anticipated workload?

These questions are very difficult to answer with much precision. It is hard to know the 'capacity' (the total amount of work that can be performed) of an existing configuration. It is even more challenging to try to figure out the capacity of a hypothetical configuration.

Why is this the case? The reason is, capacity is a function of many variables, and it is often not easy to know the values of those variables in an existing configuration or to know how that capacity will change if some rather minor changes are made in the variables. Another block of memory, or an additional input-output channel, might greatly increase the capacity of a particular configuration, while changing to higher speed tape units might do little.

If it is difficult to assess the real capacity of an existing hardware/software configuration, think how much more complex it is to try to estimate the capacity of some future configuration. There are three situations, each harder to estimate accurately than the previous. The first situation is where the user is considering adding, or changing to, a known hardware/software complex—a configuration that is in field use. In this case, the hardware and software parameters are more likely to be known. In the second case, the user is considering installing an announced but not yet delivered configuration that will be used to replace or augment the existing configuration. Here the user has only the supplier's sales claims to use instead of measured parameters. In the third situation, the user is considering an unfamiliar hardware/software complex—as in the case of installing a new distributed system. Here the user is in almost completely unfamiliar territory, so that errors of estimate are to be expected.

The other side of the coin is the future workload—the demand for capacity. It, too, is hard to predict accurately. For instance, some of today's batch systems will be converted to interactive, or to a combination of interactive and batch. Completely new uses, such as computer message systems, may be requested on a rather short lead time basis. Mergers and acquisitions can increase transaction volume; the sale of divisions of a company can reduce them. And so on.

The customary solution for determining the computer capacity that will be needed in the future is to assume that capacity has been used efficiently in the past. Capacity trends can then be determined for the past several years and projected for several more years into the future. This approach can lead to over-capacity, especially if current capacity is not used efficiently.

Another solution has been to develop a 'benchmark' workload that can be run and measured on an existing configuration. Using this as a starting point, the relative capacity of other configurations can be determined by running the same benchmark workload on those configurations. This approach has the advantage of bringing some unsuspected bottlenecks to light. And it does provide hard figures for estimating.

But there are some problems with the benchmark approach. For one thing, the benchmark workload must be representative of the total workload, and therefore has to be developed. It is not easy to make it truly representative, and it can be costly to develop. It probably must be based on the existing workload and may not consider important future applications. And it requires that the new hardware/software be available, for running the benchmark workload.

There are two other approaches to capacity management that we have come across—the BGS Systems, Inc. BEST/1 analytic model approach, and the Institute of

Software Engineering software physics approach.

Before beginning a brief overview of these two approaches, we should mention that we have had some discussions with users of both. The users have seemed well satisfied. But we have not yet had a chance to study the use of these methods in detail. We hope to do so for a not-distant future issue. In the meantime, we will confine ourselves to the following overviews.

Analytic model approach. BEST/1, which uses the analytic model approach, is a proprietary software system developed by BGS Systems, Inc., of Lincoln, Massachusetts, for evaluating and predicting computer system performance. For more information on it, see Reference 2.

BEST/1 uses a queueing theory model for predicting the processing time for streams of jobs; it is *not* a simulation model. Because of its analytic nature, it requires far less computer time to perform an analysis than does a simulation model that addresses the same level of detail.

To use BEST/1, the first step normally is to define the current computer workload. The different types of transactions are identified, as well as their flow through the system. The resources that each transaction type uses are identified, using accounting packages, hardware and software monitors, etc. The expected growth in volume of the current workload, over the time period under study, is estimated. The expected new workload is analyzed in the same way. Similarly, the hardware and software parameters, for both current and future models, is determined.

This data is then input to BEST/1. BEST/1 determines the probable service time for each part of the workload. The model is tested by calculating the time required to process the current workload, and then comparing this estimate with actual times. After any needed revisions are made, the model is used to calculate the times to process the future workloads. Also, sensitivity analyses can be made by varying the parameters, to see the effects on run time.

Software physics approach. This approach was developed by Kenneth Kolence, at the Institute for Software Engineering, Palo Alto, California. We discussed the approach in our March 1978 issue. For more information on it, see Reference 3.

The approach is based on the concept of 'software work.' An approximate definition of software work is: "A processor does one unit of software work on a storage device when it transfers one byte to that storage device."

Processing capacity is measured in terms of 'software power,' which is the ability of a processor to do software work per unit of time. Work is invarient, but power varies with the configuration.

In using this approach, one first determines the power of the hardware/software configuration under study—the CPU, the channels, the tape and disk drives, and so on. Initially, these power figures had to be determined using hardware and software monitors. But users have developed (and have exchanged among themselves) power tables for commonly-used hardware and system software. So determining the capacity (power) of a configuration is not all that complex, for configurations covered by power tables, users have told us.

The next step is to determine the amount of software work involved in the current workload and in the expected future workload. Again, these figures are expressed in units of software work. These figures represent the demand for capacity.

Programs are available for matching the demand-versus-supply of capacity, during typical daily operating hours. Results can be expressed as curves which show the amount of capacity needed to produce at least 95%, say, of the work on schedule.

Thus, BEST/1 and software physics represent objective, quantitative ways for estimating required future computer capacity.

As we have said, capacity management is a complex subject. We plan to return to it, along with a description of some user experiences, in the not-distant future.

But data processing management probably can never provide (within budget) all of the capacity that users might desire. This is particularly true when interactive systems enter the picture. As workload begins to approach the capacity of a configuration, response times get longer. This situation tends to happen during peak load periods, and users get frustrated as response times increase. Means are needed to shift workload away from the peak load periods and into the off-peak hours. One such means is the charging mechanism.

Charging for services

We discussed the subject of charge-back methods for computer services in our November 1973, July 1974, and August 1975 reports. Since that time, we have not come across any information that indicates that significantly improved methods have been introduced in the interim. So we believe that the discussion in those reports is still pertinent. Hence we will give only a very brief summary here.

Most charge-back systems use average pricing, we gather from our discussions. That is, the cost of the overall service (hardware, software, staff, etc.) is divided by the total hours of usage in, say, one month. Users are then charged at that average price per hour. All costs are thus recovered and (ideally) the service ends up with no profit and no loss.

Does this sound familiar? Does it seem practical? As we see it, this approach is familiar to many people because it does seem 'practical.' But actually, it is a very poor method for charging for services. When usage is low, rates are high—discouraging more use. When usage does build up, rates drop—encouraging an acceleration of use. As usage approaches capacity, rates reach their lowest point, encouraging more use. More capacity is brought in, and the cycle repeats.

Instead of an across-the-board average pricing method, it would be much better to use a pricing method that encourages desired user behavior, we believe. When usage is low, set low prices to encourage more use; in such instances, the service probably will end up the accounting period with a deficit. As usage increases, change the price structure to encourage off-peak use. For interactive use during peak periods, not only would response times be longer but the resource usage charges per unit of time also should be higher.

One advertisement we saw recently put the matter succinctly. The gist of the message was, "How much CPU time do you suppose is being spent in your shop by programmers playing 'star wars' games via TSO during peak load hours?" An effective charge-back system can help discourge such usage.

If computer capacity has been planned well, and if there is an effective charge-back system in place that encourages desired user behavior, the problem of managing daily production still remains. Let us low at that now.

Daily production management

The scheduling and management of daily production in the medium size and larger data processing centers has been a complex job for a good number of years. And the problem is getting worse as the workload increases. Depending upon the organization size, there can be dozens, hundreds, or thousands of jobs daily. The sheer bulk of the workload adds to the complexity.

But the size of the workload is only the beginning of the problem. Most data processing jobs are really parts of 'job streams,' with defined precedence relationships. For any given application system, there will be (1) at least one type of input, and generally several types, that must be

made ready for data entry by a scheduled time, (2) data entry and the validation of input must then be performed, (3) errors detected by the validation check must be corrected and the transactions re-entered, (4) if all of this has been done at a remote site, the batch(es) of transaction data must be transmitted to the data center, (5) all input may be staged, to get it into the system, ready for processing, (6) sorting and processing is done, (7) output is staged and then printed, (8) the output documents are burst, decollated, collated, inspected, and bound, and finally (9) the output documents are delivered to the users.

Within this overall sequence, there are other precedence sequences that must be maintained. For batch processing, sorting of transactions must preceed file updating. Generally, too, any given transaction may affect several files; this is particularly true of the older, tape-oriented application systems. So there can be a sequence of sort-update, sort-update, etc. Each update produces outputs, some of which are printed and others of which flow into other sort-update sequences.

So if some input does not arrive by its scheduled start time, a whole series of subsequent jobs can be affected. If that input is essential, then all subsequent jobs that rely on it will have to be delayed. And it would be very desirable to re-schedule, so as to make the best of the situation.

The above discussion has dealt only with the *scheduled* production workload—the work that is known to come in on a regular basis. In addition, there is usually a fair amount of unscheduled work that must be performed in any data center. The general types of jobs may be known for this unscheduled work, but the specific jobs cannot be predicted in advance. An organization policy is then clearly needed: how soon will results be delivered to the users, after the input has been submitted, for (say) 95% of the unscheduled work? The policy might range anywhere from 30 minutes to "as soon as we can do it." The longer the users must wait, or the more variable the time that the results are delivered, the more these users will want to use other services over which they have more control.

The policy on handling unscheduled work probably will be based on both the volume of that work and its importance to the organization. For instance, in high technology manufacturing organizations, where engineering is a critical function, engineering computations probably will represent a substantial and important part of the total workload—and rapid response will be sought. In such an environment, computer capacity will be largely determined by this unscheduled workload. If the unscheduled work is run during the daytime shift, and the scheduled work is run at night, the capacity needed for the unscheduled work may be more than enough for the scheduled workload. So handling perturbations in the schedule may not be difficult.

But for most data centers, we gather, the bulk of the work is predictable, consisting of applications that are run on a regular basis. In this environment, computer capacity is chosen to handle the known workload. Not so much extra capacity is available for handling perturbations. The need to schedule the predictable workload becomes more intense.

What are these perturbations? They are anything that disrupts the regular processing of the work. They include hardware failures, software errors, operator mistakes, input that arrives too late, reruns, extra jobs with high priorities, peak loads, and so on.

One approach to handling the predictable workload and adjusting to the perturbations is that provided by Value Computing, Inc.

Scheduling daily production

Value Computing, Inc., of Cherry Hill, New Jersey, has developed and is marketing three major software products that are

appropriate for our discussion. These are (1) an OS/VS scheduler for use on IBM 360/370 and plug-compatible systems, in conjunction with most of IBM's operating systems, (2) a status and revision sub-system that keeps track of job status and supports revision scheduling, and (3) an automatic job submission sub-system. For more information, see Reference 1.

We will give a brief overview of these.

OS/VS scheduler. The heart of the scheduler is the 'workload control file' (WCF). The components of this file indicate VCI's approach to scheduling.

Job profiles. The job profile section of the WCF contains data about all scheduled jobs—those run daily, weekly, monthly, and such—for each data center that is being scheduled. All of the resources that are needed to perform a job, including the amount of a resource that is needed, are indicated. Really, streams of jobs are shown, giving the necessary sequence, from the submission of input to the delivery of output. For each job within the stream, the profile indicates what resources must be available before the job can run. Also indicated are the time (and date) when input data will be available and the time when output must be delivered.

Work center profile. This portion of the WCF lists all of the resources that are being scheduled, by type and by location. It can include clerical positions, for data control and error correction, as well as operator functions for bursting, collating, binding, and delivery. All major computer resources, such as memory, disk drives, tape drives, etc. are specified.

Yearly calendar. This part of the WCF indicates work days, week ends, holidays, etc. for up to one year in the future.

Forecast workload. As daily schedules are generated (to be described), they are stored in this portion of the WCF. Schedules for up to 62 days in the future can be stored in detail.

Revision and comparison section. This part of the WCF holds today's revised schedule(s), if any, plus the actual production results for the previous four schedules.

The scheduler uses iterative simulation for the scheduling operation. That is, using the WCF, it attempts to load jobs into work centers, according to their start times; in doing this, it checks to see that all needed resources will be available at this simulated time. If a work center becomes overloaded, some jobs may have to be moved ahead in the schedule so as to level the load. The objective is to get all jobs completed by their target completion times and at the same time make efficient use of the resources.

The scheduler can schedule all work stations, not just the CPUs. These include input preparation, data entry, staging, and so on. One study reported by VCI points up the need for this type of scheduling. This study found that the average time required by a data center—from the time that the user delivered input until the output was given to the user—was almost 15 hours. Of this time, 3 hours were consumed in input preparation, getting ready for entry into the computer. One hour was used for all computer processing. And then it took about 11 hours to print and distribute the outputs. Clearly, the big time losses occurred before and after the computer processing. Just scheduling the computer could thus have relatively little effect, as far as service to end users is concerned.

Using the WCF, the scheduler can 'roll ahead' for, say, one year to see what the workload looks like in the future. This can be useful for planning for additional computer capacity.

The scheduling system also keeps track of actual resource usage and compares this with the expected resource usage, as described in the job profiles. If there is a significant variation between actual and expected, the expected value is adjusted au-

tomatically. This adjusted figure in then used in future scheduling operations.

The scheduling system provides a wide variety of outputs. One type, of course, is the daily schedule. Typically it is printed out, but is also available via a terminal. A daily actual versus schedule report highlights the jobs that had significant variations from schedule. Bar graph reports show the scheduled resource usage, etc.

Status and revision. This sub-system (STARS) uses the schedule developed by the scheduling system and automatically updates a job status file for all CPU work, to show actual versus schedule. For non-CPU work, status data may be entered via a terminal.

The sub-system monitors the progress of each job stream, making sure that all steps have been performed in their specified sequence. If a job is missed, perhaps due to an operator error, this fact is reported by a message on the console.

STARS provides input to the revision scheduling portion of the scheduler. If an operator who is monitoring production determines that a revision schedule is needed, almost all of the needed information is available in STARS. It thus becomes feasible to reschedule numerous times a day, if the perturbations require it, so as to make as efficient use of the resources as is possible under the circumstances.

STARS also allows end users to find out the current status of their jobs directly, by accessing via TSO or ROSCOE.

Job submission. This sub-system (APOLLO) is also driven by the scheduling system. It automates a portion of the computer operator function. It stores job control (JCL) data and has access to the program library. When a job is in the job queue and its start time has almost arrived, APOLLO checks to see if all needed input resources are, in fact, available. If so, the data, JCL, and program are made ready for use by the computer. Jobs are thus submitted in schedule sequence, reducing schedule perturbations due to operator error.

APOLLO also allows for on-line changing of JCL and other control data. Further, manual job submission and entry of JCL can occur. An audit trail is made of all events, and an audit file of the current and six previous schedules, together with their JCL input, is maintained.

On-line production control. VCI suggests that the production control system of the future will use all three of these packages, for dynamic control. The scheduler would be used to create the daily schedules. The jobs would be released automatically, according to the schedule. Actual job status would be determined. If perturbations occur that require a revision of the schedule, that scheduling can take into account the actual status of all jobs.

Let us now take a look at how one user organization uses these tools for managing daily production.

Texas Instruments Incorporated

Texas Instruments Incorporated, with headquarters in Dallas, Texas, is a major manufacturer and supplier of electronic components and equipment, mini- and micro-computers, hand calculators, and digital watches. Annual sales exceed $2.5 billion and the company employs some 78,000 people world-wide. TI has 39 manufacturing plants in 18 countries.

As we discussed in our September 1979 report, TI is in the process of installing a world-wide distributed system. During the 1970s, the company consolidated almost all of its data processing into the corporate information center (CIC) in Dallas. The main goal of that consolidation was to remove the incompatibilities of data and programs that had built up during the 1960s, when the company had multiple data centers.

In 1966, TI began installing a data communications network to its various sites. By 1976, the network was connected to most TI locations world-wide, with inter-

continental transmissions via satellite. Also, by 1973, TI management began to see the advantages of moving to a distributed system. Since the incompatibilities had been largely eliminated, it appeared feasible to put processing power and data storage facilities at the end user locations—as long as corporate standards were observed.

The distributed system structure has four levels.

Level 1 is the CIC in Dallas. Presently, it consists of four IBM 3033s and two 370/168APs. The CIC performs the bulk of TI's batch data processing, plus interactive database queries (via IMS) and time sharing (via TSO).

Level 2 is in the process of implementation. TI plans to use intermediate size IBM systems for this level, to be installed at major sites around the world, mainly manufacturing plants. These computers will provide interactive processing services, including database queries and time sharing. In addition, where appropriate, some level 1 batch work will be off-loaded to these sites.

Level 3 consists of local work-area computers that serve clusters of terminals. TI currently has 87 of these computers installed world-wide. For this level, they have standardized on TI computers, primarily the TI 990 mini-computer (which uses the TI 9900 micro-processor). These level 3 computers provide the bulk of the interactive service. But some of them also provide remote job entry for batch work to be performed on higher level computers.

Level 4 is the work-station computers, for individual employee use. These generally are intelligent terminals, and are used in an interactive manner.

Managing the workload. In the present status of the TI distributed system, the bulk of the computing workload is batch work that is performed at the level 1 CIC in Dallas. And, as just mentioned, some of the 87 level 3 computers serve as remote job entry terminals for sending batch work to level 1 and for getting results back.

In all, CIC handles about 7000 jobs per day. Of these, some 1600 are predictable and are scheduled. These 1600 come from TI sites in North America, Europe, and the Far East. The remainder, the unscheduled jobs, consist of engineering calculations, program development and test, and ad hoc requests. Further, of these 5000+ unscheduled jobs per day, over 4000 occur between the hours of 7 AM and 7 PM Dallas time.

Most of the daytime hours—7 AM to 7 PM—are reserved for the unscheduled work, in order to provide good service to these users. Most of the scheduled work is done at night. Further, the remote batch transmissions, both to and from Dallas, are scheduled, where appropriate. The schedules are created in Dallas, converted to remote site local times, and then transmitted to those sites. And when level 2 computers are installed, it is expected that the predictable parts of their workloads also will be scheduled from Dallas.

Because the unpredictable workload dominates, TI determines its need for computer capacity largely from this workload. As it has worked out, the resulting capacity is quite adequate for handling the scheduled workload during the night shift.

TI has experienced a rapid growth in its CIC workload. In 1974, the center was handling about 250 scheduled jobs per day. By consolidating data centers plus the growth in usage, the load has grown since then to the current level of 1600 scheduled jobs (plus the 5400 unscheduled jobs) per day.

Five years ago, the Dallas center workload was scheduled manually. But due to the rapid growth, manual scheduling was becoming impractical. So TI assigned some people to the center who had production control experience. They decided to automate the production control function for the center. After considering several alternatives, including in-house devel-

opment, they chose the VCI scheduling system.

This scheduling system soon brought the daily production under control, even with the rapid growth in the workload. So TI then began building a number of other production control tools in-house, to go along with the scheduling system. One tool was used to schedule the key punch function. Another was developed to schedule financial closings for all TI accounting departments throughout the U.S. Still another tool was a system for automatic job submission. This one turned out to be the forerunner of VCI's APOLLO sub-system. TI has since replaced their in-house system with APOLLO to avoid the need for in-house maintenance.

TI has also installed VCI's STARS sub-system for tracking job status and for supporting revision scheduling.

With all of these production control tools, what have been the results at TI? Well, computer operations carefully monitors production performance, such as report deliveries. For a recent 52 week period, in only three weeks did they fail to deliver at least 95% of all reports on schedule. They also track 40 to 50 key jobs daily, most of them manufacturing control jobs. Here the goal is even more challenging. If any report is four or more hours late in a week, the performance is considered unsatisfactory. Over 95% of the key daily systems have acceptable performance, on a consistent basis.

In all, the people in computer operations monitor some 250 performance measures. The responsibility for this monitoring is spread throughout the organization. As the people in computer operations see it, they are very service oriented and they feel that they are meeting their performance goals. A part of the credit, they say, is due to the use of their production control tools.

But good performance is not just a function of these tools, we were told. Computer operations must also get co-operation from the end users. TI's charge-back system helps, they say. Different prices are charged for different priorities on different resources. On the average, resource prices that are charged for users have been going down steadily every year. But for specific resources in tight supply, prices can rise for a time. By charging different prices for different times of the day, peak loads are levelled somewhat and users can save money by making use of off-peak hours or longer delivery times.

As our interview was closing, one remark proved to be the 'frosting on the cake.' For the past five years, TI's computer workload has grown as described above. But the production planning and control staff budget has stayed practically flat over that period. Which speaks well for automated tools for computer workload control, we would think.

REFERENCES

1. For more information about their scheduling system, write Value Computing, Inc., VCI Building, West Marlton Pike, Cherry Hill, N.J. 08002.
2. For more information on BEST/1, write BGS Systems, Inc., Box 128, Lincoln, Mass. 01773.
3. For more information on the use of software physics for capacity management, write Institute for Software Engineering, Box 637, Palo Alto, Calif. 94302.

SUBJECTS COVERED BY EDP ANALYZER IN PRIOR YEARS

1977 (Volume 15)

Number

1. The Arrival of Common Systems
2. Word Processing: Part 1
3. Word Processing: Part 2
4. Computer Message Systems
5. Computer Services for Small Sites
6. The Importance of EDP Audit and Control
7. Getting the Requirements Right
8. Managing Staff Retention and Turnover
9. Making Use of Remote Computing Services
10. The Impact of Corporate EFT
11. Using Some New Programming Techniques
12. Progress in Project Management

1978 (Volume 16)

Number

1. Installing a Data Dictionary
2. Progress in Software Engineering: Part 1
3. Progress in Software Engineering: Part 2
4. The Debate on Trans-border Data Flows
5. Planning for DBMS Conversions
6. "Personal" Computers in Business
7. Planning to Use Public Packet Networks
8. The Challenges of Distributed Systems
9. The Automated Office: Part 1
10. The Automated Office: Part 2
11. Get Ready for Major Changes
12. Data Encryption: Is It for You?

1979 (Volume 17)

Number

1. The Analysis of User Needs
2. The Production of Better Software
3. Program Design Techniques
4. How to Prepare for the Coming Changes
5. Computer Support for Managers
6. What Information Do Managers Need?
7. The Security of Managers' Information
8. Tools for Building an EIS
9. How to Use Advanced Technology
10. Programming Work-Stations
11. Stand-alone Programming Work-Stations
12. Progress Toward System Integrity

1980 (Volume 18)

Number

1. Managing the Computer Workload

(List of subjects prior to 1977 sent upon request)

PRICE SCHEDULE (all prices in U.S. dollars)

	U.S., Canada, Mexico (surface delivery)	Other countries (via air mail)
Subscriptions (see notes 1,2,4,5)		
1 year	$48	$60
2 years	88	112
3 years	120	156
Back issues (see notes 1,2,3,5,)		
First copy	$6	$7
Additional copies	5	6
Binders, each (see notes 2,5,6)	$6.25	$9.75
(in California	6.63, including tax)	

NOTES

1. Reduced prices are in effect for multiple copy subscriptions and for larger quantities of a back issue. Write for details.
2. Subscription agency orders are limited to single copy subscriptions for one-, two-, and three-years only.
3. Because of the continuing demand for back issues, all previous reports are available. All back issues, at above prices, are sent air mail.
4. Optional air mail delivery is available for Canada and Mexico.
5. We strongly recommend AIR MAIL delivery to "other countries" of the world, and have included the added cost in these prices.
6. The attractive binders, for holding 12 issues of EDP ANALYZER, require no punching or special equipment.

Send your order and check to:
EDP ANALYZER
Subscription Office
925 Anza Avenue
Vista, California 92083
Phone: (714) 724-3233

Send editorial correspondence to:
EDP ANALYZER
Editorial Office
925 Anza Avenue
Vista, California 92083
Phone: (714) 724-5900

Name_____

Company _____

Address _____

City, State, ZIP Code _____