Service levels represent an important concept that can be applied toward the solution of difficult problems surrounding communications between users and providers of data processing services. In this paper, this concept is described in terms of the architecture, which defines the scope and structure of service information. The paper further translates the architecture into data processing terminology by presenting the data-base structure and data elements related to service levels. The paper also addresses the post-processing of the data base, a step essential to properly communicating service-level information.

Service levels: A concept for the user and the computer center

by L. J. Lewis

In the past decade, engineers, programmers, and administrative/manufacturing people have become increasingly dependent on data processing to perform their work. The manner in which computers are used has also changed. In addition to the typical batch service offering, where a user's job is sequentially processed to completion in an input, process, output sequence, a variety of interactive service offerings have been developed. What is common to such functionally different but terminal-oriented service offerings as, for example, APL-SV, OS/Time Sharing Option (TSO), and Virtual Machine Facility/370 (VM/370), is the direct extension of the basic data processing function into the user's working environment. One significant result has been increased awareness by the users of the available data processing services and the measures of service upon which they can rely.

The type of service offering selected determines the manner in which the workload is processed. Without elaborating on the global benefits of interactive processing versus batch processing, the contrast between these two data processing methods is presented from the viewpoint of how the user goes about obtaining the service and his relationship with the computer center.

Batch processing requires that users submit their jobs (e.g., card decks) to a computer center operations area. In most cases, this area is typically a service window where the user later returns to retrieve his processed output. A variation of this procedure is to use a remote work station for job input and output. The work station is a low-to-medium volume card reader and printer that is located close to the user's work area. Input and output for batch jobs is transmitted between the work station and the host computer by voice grade telephone lines. Another option is to use a terminal function of an interactive service offering to place a job in the batch processing queue.

The significant communications aspects of a batch service offering are: (1) The user generally interfaces with the service offering through another person who represents the computer center. (2) When the user experiences a problem between his submission of the job and retrieval of the output, computer center personnel usually provide some degree of immediate satisfaction which is often perceivable by the user. The relationship between the user and the provider of the service is on a person-to-person basis, and the solution to problems is facilitated by face-to-face communications.

Interactive services, however, require the user to interact with the "system" via a terminal. In this mode, the user is more aware of the "system" because of the contention factors associated with getting a terminal, getting a line, being able to sign-on, and receiving a measure of service that is satisfactory for accomplishing the workload.

The expanded use of interactive service offerings, along with the increasing acceptance of remote work stations that are usually operated by the user, tend to create a negative environment for user and supplier communications. The difficulty occurs because users have no readily accessible outlet for complaints to the providers of the service. Service-related problems, no matter how trivial, will tend to accumulate until the user's frustration threshold is exceeded. Communications between the user and the supplier at this point will be impersonal, inflated by emotion, and often occur long after the complaint. The underlying problem is that users of a data processing service are guided by the pressures associated with meeting their workload schedules along with some human factor considerations. These considerations tend to influence the users' beliefs along the lines that data processing services should be offered as an unlimited resource and be at their immediate disposal. In this context, users generally express a requirement for a level of service that often exceeds the level that is economically justified. The suppliers of the service, however, operate under an almost diametrically opposed set of pressures. They are often asked to reduce data processing

expenditures significantly until the user's truly required level of service is achieved. This level is often judged by the user's threshold of pain as denoted by a possible schedule slip, by a user's unhappiness, etc. The overall result is that the level of service tends to fluctuate over time depending on which set of criteria is on top management's priority list.

It is in this environment that the concept of service levels is most important. Service levels are valuable to top management as a quantified method of balancing the users' requirements against the cost and value of a level of service. Service levels also represent an important vehicle to the computer center for communicating with its users. The overall purpose of providing service levels is to: (1) establish an agreed upon benchmark of expectation and a comparable measure of achievement, (2) provide a definable structure for evaluating users' requirements in terms of the human factors and economic considerations[1,2] that would justify a given level of service, and (3) provide a track record against which complaints can be objectively compared for merit and subsequent action.

Implementing a service-level scheme is not a trivial task. For example, in IBM, programmers, engineers, and planners use multiple service offerings in performing their daily work. A fundamental requirement of service levels is to consistently report service information between functionally different service offerings such as TSO, batch, VM/370, and APL. This requirement is especially necessary in situations where the same service such as TSO is provided from different host locations but is made equally available to a remote user. Another requirement of service levels is to relate a user's experience in using a service to the information that reports how the service was used.

Many computer applications that use IBM equipment are based on the use of the OS/VS Systems Management Facility (SMF) data for reporting availability, utilization, and performance measurements to users and management. A particular application in this area is the Boeing Computer Services Systems Accounting and Resource Analysis (SARA) application.[3] The major drawbacks of these applications are that they collect and report service-oriented information at the most detailed level in strictly data processing terminology to which users cannot easily relate their requirements, or the information is summarized into accounting-oriented periods that obscure the time perspective of the users' experience.

Optimizing the performance of a computing facility by computer performance evaluation has been widely presented.[4-6] An excellent state-of-the-art review of computer performance evaluation is contained in the *Proceedings of the Eighth Meeting of Com-*

puter Performance Evaluation Users Group.[7] Although this paper has a large amount of information that is common to both computer performance evaluation and service levels, the significant difference is that the concept of service levels emphasizes the relationship between the user of and the provider of data processing services. The paper also expounds upon the concepts of service levels and not the implementation of a service-level application. The concepts presented were derived from empirically researching and implementing a service-level application. It should be noted that this application has evolved through two significant redesigns since its initial implementation in 1973. Each rewrite was necessitated by positive and negative experience gained in utilizing the prior version. The state-of-the-art as presented in this paper exceeds the design point of the service-level application currently installed.

In this paper, the first section presents the architecture of service levels. The architecture separates the body of service-related information into a service description and several measures of service. The latter is presented for clarity as separate elements, each of which is described and related to other elements. Then the discussion translates the architecture into data processing terminology. The service description and measures of service are defined as data elements. The data elements are structured into a data base for service-level information. This data base is ancillary to the system, such as SMF itself, and is totally independent of the users' data base.

Architecture

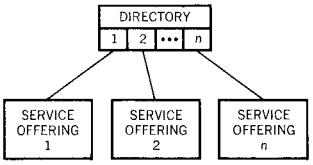
A service level is a structured set of information pertaining to: (1) a description of the service offering, (2) measures of service which define the level of service a user can rely upon, and (3) a record of what transpired during the use of the service. The role of the architecture is to define the scope and structure of this information. The architecture becomes a prerequisite to defining the structure of a data base for service-level information. It is also necessary for developing a capability of selectively capturing the measures of service. Presenting the architecture first enables us to view the concept of service levels from its global perspective. When individual topics such as computer performance and installation accounting are presented out of context to the overall architecture, invalid service-oriented conclusions may be drawn.

In a service-level application, a service level would be established for each service offering provided. The architecture has a nucleus that can be represented by a directory that simply contains a pointer to each service offering. Assume that the over-

Table 1 Service-offering description information

Description	*Service-offering 1*	*Service-offering n*
• Names of host and serviced locations.	Kingston (host) Poughkeepsie Endicott	Raleigh (host) Kingston Manassas
• The service type identifier and name.	VM/370	BATCH
• The local identification and name.	VM1	ADM-MFG
• Responsible manager's name.	A. B. Jones	C. D. Smith
• Operating system version and level.	VM 21.1, CSL11	MVS3.1, JES3
• Classes of service supported.	CMS MVS Driver	Priority Prime Shift Overnight

Figure 1 Overview structure of service levels



view structure in Figure 1 will be subsequently enhanced in detail throughout this paper for purposes of discussion.

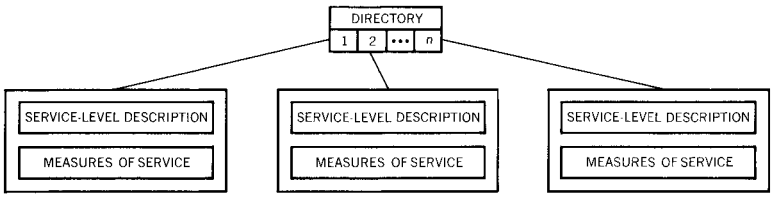
Early in the paper, a requirement was stated that each service offering be uniquely identified and consistently described. This requirement is satisfied through the service offering description portion which includes, but is not restricted to, the items in Table 1.

Service-level description information as illustrated in Table 1 is alphanumeric in composition. It is retained in the data base primarily to be used in the heading portion of the service-level reports. However, by looking upon this information as a list of data variables, it is possible to apply them to a common reporting format that encompasses widely different service offerings. The overview structure upgraded to reflect this information would appear as shown in Figure 2.

The information associated with the various measures of service has been grouped by function for ease of presentation, and the measures are defined in summary form as follows.

- Availability—the measures of time that define the interval during which a service offering is fully operational.
- Capacity—a measure of the potential amount of total workload that could be processed during the period when the service offering is available.
- Utilization—a measure of the amount of workload actually processed during the availability period.
- Performance—a measure of the amount of a user's workload that will be processed in a fixed period of time.
- Accessibility—a measure of how the workload capacity is distributed to the users in a manner that is consistent with the performance commitment.

Figure 2 Upgraded overview structure



- Reliability—a measure of future successfulness or difficulty in processing a user's workload relative to the prior measure of availability, capacity, and performance.

Accepting the gross definitions presented in this paper, even though they may vary in orientation from the commonly applied definitions associated with specific topics, such as performance, is essential. The definitions as applied enable the formulation of a service-level concept that represents at a very high, and perhaps over-simplified, level the integrated complexities of time, resource, demand, distribution, and reliability as these aspects of service relate to a user. The definitions as presented also enable the concept of service levels to be structured in a manner that is definable and can be tracked by the computer installation.

Each of the above measures of service is interdependent on one or more of the other measures. Together they define in a structured manner both a service-level commitment and an after-the-fact track record of achievement. In establishing a value for any aspect of a service-level commitment, it is important to ensure that all measures of service are synchronized. This step must be an integral part of the service-planning process. This process is complex and requires several iterations before a reasonable balance is achieved between the components.

The global aspects of this process can be illustrated by the following steps: (1) reliability is a function of the prior reliability measures for availability, capacity, and performance; (2) utilization, the amount of capacity that will be used, is calculated by dividing the workload that must be successfully completed by the reliability measure; (3) the measure of accessibility for a given kind of workload (k) is equal to the percentage of user demand multiplied by a term consisting of the capacity divided by the sum of the multiplication of the performance measures for each kind of workoad and the percentage of user demand; and (4) availability, the largest variable in establishing a service level, will then be equal to utilization divided by the measures of accessibility. This process may be illustrated as:

$$\text{Reliability} = f(\text{Availability}_{pr}, \text{Capacity}_{pr}, \text{Performance}_{pr})$$

$$\text{Utilization} = \frac{\text{Required workload}}{\text{Reliability}}$$

$$\text{Accessibility} = \text{Demand}_k \times \left[\frac{\text{Capacity}}{\sum_{k=1}^{n} (\text{Performance}_k \times \text{Demand}_k)} \right]$$

$$\text{Availablity} = \frac{\text{Utilization}}{\sum_{k=1}^{n} \text{Accessibility}_k}$$

where:

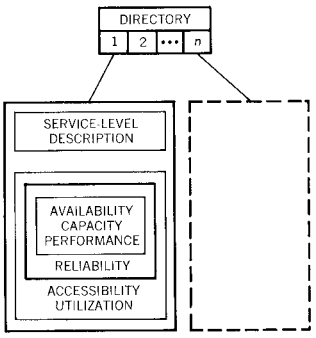
k = workload type
pr = prior measure

The service-level overview diagram can now be illustrated as in Figure 3. Each of the functionally grouped measures of service are now described in detail.

availability

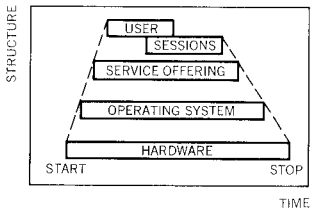
Users generally believe that a service offering is really available only after they have productively completed at least a portion of their workload. While there is sentiment for the users' position among the operations personnel, their reporting of availability has tended to be in terms of whenever the processor appears to be operating. The two perspectives result in a mismatched form of communications and a hassle as to when the service offering will be or was truly available. The solution incorporated into the service-level concept is to first recognize that there is an availability period associated with: (1) the hardware, (2) the operating system, (3) the service offering, and (4) the user's session (Figure 4). Computer performance evaluation applications have particular interest in the periods of *systems availability*, whereas installation accounting applications focus on the *hardware-availability* period. The service-level concept emphasizes the *service-availability* period, an interval of time defined to be when the service offering is truly available to act upon a user's workload. A subset of this interval is the period of time when a user interacts with the service offering via a terminal or through a batch-processing application.

Figure 3 Global service-level overview diagram



The scope of the service-level architecture is across all of the availability periods because there is a relationship between each of them and the other functional groupings. In this context, except for scheduled maintenance or an actual equipment failure, the hardware components are usually kept operational 24 hours a day. All other availability periods will be less than or at best equal to the hardware-availability period by the amount of time required to initially load a program plus any operating system outage time. Initializing a service offering once these tasks are completed requires only a few seconds.

Figure 4 Availability periods



Operating systems availability is not, however, a good approximation of service availability because other conditions, such as the system being in a closed loop or an application abnormal halt, further reduce the service-availability period. Each period has a beginning and ending point which can easily be measured in terms of a date and time of day. The latter measures are expressed in hours and hundredths of an hour.

capacity

Capacity is included in the service-level concept because capacity really represents the commodity that is offered by the service. Users are always concerned whether or not there is enough capacity available when they want it. The rate of capacity (amount per instant of time) is fixed by the mix of data processing equipment installed. In determining the size of computers to install, the computer center must focus on the trade-offs between service availability (time) and the rate of capacity. The total capacity is determined by multiplying the availability period by the rate of capacity. For purposes of the service-level concept, we define capacity as being the potential amount of workload processing that could occur during an interval of availability. A measure of capacity is associated with the availability of: (1) the computer hardware, (2) the operating system, and (3) the service offering. In addition, a measure of capacity will be created to reflect the user's workload. Historically, the subject of capacity and the related topic of utilization have been looked upon in terms of defining and applying standard accounting rates and installation recovery schemes.[8] In the concept of service levels, the measure of capacity must be relatable to the amount of work a user can expect to accomplish. The architecture suggests a methodology for measuring the basic data processing components such as CPU, memory, storage, and I/O and converting their units of hardware capacity into various kinds of user-oriented units of workload capacity.

The measure of *hardware capacity* is the lowest level at which capacity can be expressed in a data processing context. Each component type in a computer complex has a published data transfer rate or other measurable unit that best describes the device in terms of its function. For example, an IBM 2303 Model 3 Printer has a rated hardware capacity of 1,000 lines per minute based on the use of a 48-character print train. An IBM 3330 Model I direct access storage device has a hardware capacity of two packs, each containing 200 million bytes of data distributed across 15,352 addressable tracks. These measures of hardware capacity are obviously not additive across these devices. They also do not lend themselves to being directly expressed as a single unit of service that would be meaningful to a user. When resources are quantified at this level of detail, only the aggregate amount of service units by device class within a computer complex or installation can be established. A complex

may have, for example, six printers and 32 disk storage devices which would provide a hardware capacity of 6,000 print lines per minute and 12.8 billion bytes of storage. An expression of resource at this level of detail is, however, necessary to provide a benchmark for subsequent capacity measurements and for associating equipment rental cost with discrete units of capacity.

The measure of *systems capacity* is the next level in the capacity structure. It is a somewhat academic level used to clarify the net amount of hardware capacity by device type that remains after appropriate deductions are made for those devices dedicated to the operating system. Of the six printers defined in the previous example, one may be dedicated to the operating system. In addition, this level would factor in the reduction in the gross amount of capacity that must be made to account for channel contention and other configuration factors. This distinction is important because the same hardware complex will provide different amounts of system resources depending upon the particular operating system(s) being utilized. For example, the IBM 3203 Printer, Model 3 will have a rated line speed of 870 lines per minute when a 60-character print chain is used. Assuming a mix of print train usage, the effective system capacity could be about 4,600 lines per minute.

The measure of *service capacity* represents the portion of systems resources allocated to a particular service during the availability period. The exact capacity will vary as devices are varied on and off line, or are switched between services on a dynamic basis. The importance of this level is that it reflects from a data processing viewpoint the basic kinds and amounts of resources typically offered to the user set. The particular service offering may provide the user with access to only two of the attached printers. The service capacity would then be 1,850 lines per minute.

The measure of *workload capacity* is the highest level at which capacity can be communicated. What makes this level unique from the other measures of capacity is the fact that these units of capacity: (1) are not expressed in the usual data processing terms but are expressed in functional units that are meaningful to users and (2) are not directly measurable but are derived from the measures of service capacity and utilization.

An example of a workload capacity measure would be to describe the service offering as being able to provide 30 program compilations per hour. One of the factors, but not necessarily the limiting one, could be that each compilation requires an average of 3,500 lines of print.

Workload capacity for electronics engineers would be expressed in the number of circuit logic designs, layouts, simulations, and physical designs they could perform. A similar set of measures for programming developers would be expressed in terms of edits, compiles, builds, tests, etc. Workload capacity for each workload type would be assigned according to a functional definition; for example, circuit logic designs would be classed according to the technology used, average circuit density, and component function.

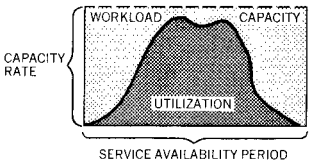
In the service-level concept, the measure of workload capacity also becomes the least common denominator for expressing the user's workload requirements in terms of the capacity of a service offering. This concept contrasts sharply to today's typical environment in which the engineer must think of his workload as being, for example, 18,000 I/O commands in the form of Execute Channel Programs (EXCPs) at a dollar rate of 1.192 per thousand EXCPs plus 338K bytes of core for 16 seconds at the rate of 0.2242 cents per thousand bytes per second plus X, Y, Z, etc.

The most significant problem to be addressed in implementing a service-level concept is the creation of the algorithms that would translate service capacity into workload capacity. Because of the mix of service offerings and spectrum of available hardware configurations, it would be impossible to develop a single or even a handful of shareable algorithms. The solution is arduous and requires the detailed analysis of how a service offering is actually used.

utilization

The portion of capacity that has been productively applied toward processing a user's workload is termed utilization as illustrated in Figure 5. The measures of utilization have a structure identical to the measures of capacity presented above. Of the two topics, utilization has received a greater amount of attention in today's environment. The reason is perhaps that while every computer installation uses some meaure of utilization for cost recovery, very few service offerings are described in terms of their capacity.

Figure 5 Availability, capacity, utilization relationship



Unfortunately, the popularity of utilization has not been a plus factor toward advancing the concept of service levels. Today utilization measures are almost always stated in the very detailed data processing terms of EXCPs, amount of main storage occupied, CPU seconds, etc. Although the reporting may be by job or project number, the user is left to his own initiative to relate these measures back to his processed workload. User dissatisfaction with this detailed level of reporting has caused many computer installations to develop cost recovery algorithms in which the hardware measures are converted into common billable units. An example would be to have each 1,000 EXCPs or

500 lines of print equal one billable unit. The user of, say 500 billable units, which may have an expense value of 20 dollars, would still have difficulty in relating his utilization to his processed workload except in gross monetary terms. The generally accepted theme among providers of data processing services remains that as users get experience relating to these nonservice-oriented measures, the users' expertise for estimating their workload will somehow improve. Along these lines of thought, Gladney et al.[9] take the viewpoint that the billing systems conveniently provide system performance information and demonstrate utilization and cost trends.

performance

Performance for interactive services is typically thought of as the amount of time the user must wait to have a unit of workload processed, e.g., response time. Bard[10] illustrates the commonly accepted relationship between system load and performance with a graph that depicts response time increasing as a function of the increase in the number of active users. While the relationship is true, the definition is not suitable to the overall service-level concept for the following reasons: (1) The system load is expressed in terms of the number of active users, a measure which cannot be easily related to capacity, as that term was previously defined. (2) Response time cannot be measured at the terminal, and system measurements of response time differ from the user's perception of response time experienced at the terminal. (3) The service-level commitment for performance must be stated in terms applicable to each workload type. In the concept of service levels, we view the measure of performance as the amount of a user's workload that will be processed in a fixed period of time as illustrated in Figure 6. This definition is also quite different than the meaning of the term as it is applied in the context of computer performance evaluation.[5]

Users are individually interested in how long it will take to process a workload that has certain characteristics, such as the Initial Program Loading (IPL) of an MVS system under VM/370. The time can be varied by controlling the amount of capacity and priority assigned to a user. Of course, given too few resources, the user will not be productive, and provided too many resources, the user will in effect waste a portion of capacity. Although we have used a grossly narrow definition of performance, our definition does enable the computer center to negotiate with the users a level of performance in terms of a rate of service that best satisfies the users' overall time requirements and which is economically justified. The point to be made is that good performance is not necessarily a 15-minute turnaround of a batch job or a two-second response time. The sequence in which a service offering processes a user's workload can be visualized as a pattern of alternating user and service interactions, or cycles. A batch service offering would have a pattern of only one cycle per

job, whereas an interactive service offering would have a pattern that reflects a large and variable number of cycles per session.

Performance, or the rate of processing, is calculated by dividing the workload processed (utilization) by a *portion* of the user and service interaction cycle. It is first necessary to divide the cycle into discrete intervals so the appropriate portion is included in the performance calculation. Boies[11] defines the interaction cycle as consisting of two components, user response time (URT) and system response time (SRT). The user's portion (URT) begins when he receives either the output from a completed batch job or is given control from an interactive service (by unlocking the terminal keyboard). The service portion (SRT) begins when the user either submits a batch job or presses the carriage return key at a terminal after completing his data entry. These points in time may be recorded over a limited time period using the Generalized Trace Facility of OS/VS or the equivalent support function for a particular service offering. This data may be subsequently processed for inclusion in the data base of the System Management Facility (SMF).[12]

Two items of information associated with each SRT must be captured. One is the duration of the SRT; the other is a measure of the workload processed during the SRT interval. The duration of the SRT can be calculated from the system time information recorded in each SMF record. The workload utilization information must be recorded in common numeric terms for the required processing. For purposes of performance measurement each workload type or class is expressed by a service value. The larger the service value, the greater the portion of capacity required to process it. Performance can then be calculated by the following algorithm.

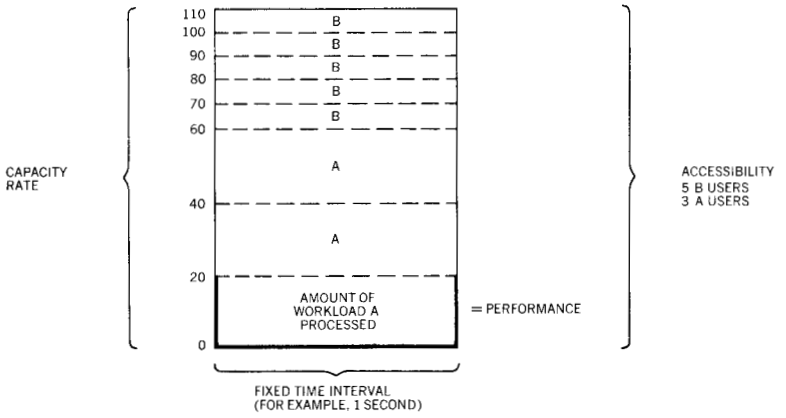
$$\text{Performance} = \frac{\sum_{\text{SRT}=1}^{n} \text{service value}_{\text{SRT}}}{\sum_{\text{SRT}=1}^{n} \text{duration}_{\text{SRT}}}$$

accessibility

User management is very interested in maximizing the number of their personnel that simultaneously receive service. The computer center personnel are also interested in optimizing the load on the computer to improve upon the utilization of capacity. The problem in either case is that increasing the users' access to the service offering beyond a certain point impacts the performance provided to an individual user. Simply, the pie (capacity) is cut into smaller pieces. Identifying the break point is a problem in itself since it varies based on the workload mix.

Accessibility is a service-level measure of how the workload capacity is distributed to the active users in a manner consis-

Figure 6 Performance, capacity, accessibility relationship



tent with the performance commitment for each workload type. In this context, accessibility actually serves as the governor, assuring the overall service level will be achieved in a dynamic environment. As previously stated, there are numerous classes of workload, and each workload class may have a different performance commitment. At any point in time, a different mix of workload can be processed and the mix will tend to change during the availability period. Achieving the performance commitment therefore requires that the access to the service offering by users within workload type be managed. The accessibility measure is expressed in the number of on-line users or batch processing jobs, per workload type, that can be simultaneously processed, which is illustrated in Figure 6. The algorithm for calculating accessibility is:

$$\text{Accessibility} = \text{Demand}_k \times \left[\frac{\text{Capacity}}{\sum_{k=1}^{n} (\text{Performance}_k \times \text{Demand}_k)} \right]$$

where workload capacity is expressed in a service value of 1000 units and k equals workload type. An example of this algorithm is illustrated in Table 2.

In the previous example, and as illustrated in Figure 6, the service has committed to provide a user who has a workload of type A, a rate of service of 20 units of work per defined time interval, such as a second. Furthermore, the service offering has planned that workload type A would approximate 25 percent of the simultaneous user demands. Following the algorithm for accessibility, the service would allow 13 users having Type A workload to access the service simultaneously. Substitutions between services is possible. For example, one additional Type A user could be signed on in lieu of two Type D users and have the performance level maintained.

Table 2 Example of calculating accessibility

Workload type	*(A) Performance value*	*(B) User demand*	*(A × B)*	*Accessibility*
A	20	0.25	5	13
B	10	0.10	1	5
C	30	0.20	6	11
D	15	0.45	6.75	24
		Total	18.75	53

Accessibility has two important roles related to service levels. First, it is the factor that allows widely different kinds of workloads to be concurrently processed but integrated so as to achieve a single service-level commitment. Second, service levels are entirely separate from the operations function associated with providing the service offering. Accessibility provides an operating plan by which access to the service can be managed so as to achieve the other service levels. It becomes the basis for scheduling the overall workload from a service-level perspective. It also relates to the service control mechanism implemented that dynamically balances the workload mix[5, 11] to ensure that utilization and performance commitments will be achieved.

reliability

Users are very concerned when a component of the service offering fails and their session or batch job is terminated. Often they will lose a portion of the workload processed since the last checkpoint, and this workload will have to be redone. There is no way of predicting exactly when a failure will occur. There is also no foolproof way of guaranteeing a failure will not occur. Reliability in the service-level concept is aimed at assisting the user to overcome these obstacles. Reliability is a measure of the success or difficulty associated with using a service offering. This measure only quantifies the confidence users can place on completing their workload on a timely basis. Since failures are random occurrences, a commitment as to a level of service for reliability is impractical. It is expected, however, that users having knowledge of the failure rate or the effective yield of the service will then intelligently plan their overall workload and schedule their use of the service accordingly.

Reliability is different from the other measures of service in the following ways: (1) The measures of reliability have the greatest value to the user prior to his utilizing the service offering. (2) Reliability encompasses each of the other measures of service that have been presented. For example, there is a measure of reliability associated with the availability, capacity, and perfor-

mance attributes of a service offering. (3) The measures of reliability are expressed in statistical rather than absolute values. For example, the availability-oriented statement for reliability would express, as a function of elapsed time, the probability that (a) the service will remain available without a service interruption which is called continuity and (b) the duration of an interruption will exceed a certain amount of time which is called outage.

In each of the following measures, the probability, p, associated with each unit of service would be calculated based upon the most recent measures of achieved service. An illustration of the format of these measures of reliability is:

Hours	4	5	6	7	8
p (continuity)	0.6	0.5	0.45	0.4	0.3

Hours	0.5	1	1.5	2
p (outage)	0.7	0.3	0.1	0.05

The measure of reliability concerning capacity can be called successfulness, defined as the probability of successfully completing a percentage of the total submitted workload. An illustration of the format of this measure of reliability is:

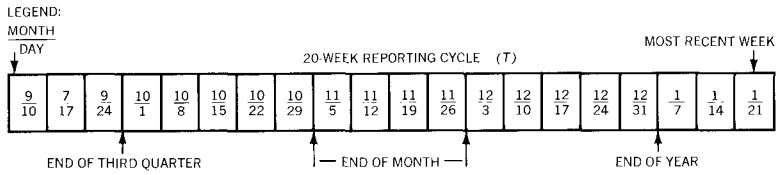
Percent of total workload	70	75	80	85	90	95	100
p (successful)	1.0	1.0	0.95	0.90	0.8	0.6	0.2

We have thus far presented the scope of the service-level concept in terms of its components, such as availability, capacity, performance, reliability, etc. Within each component, several items of information were called out as being important in terms of defining and measuring a service level. To place the service-level concept in a more practical light, the next step is to describe a data-base structure suitable for the kinds of information and special relationships just presented.

Data-base structure

Determining the size and structure for a service-level data base (not user application data) is the next crucial step toward the implementation of a service-level reporting system. The size of

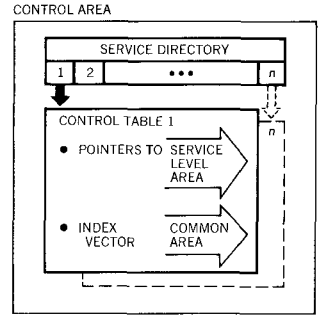
Figure 7 Service-level time dimension



the data base is grossly determined by the following three factors: The first is the number of service offerings to be included in the application. In a company of small or moderate size there may be only a handful of unique computer services that are provided to employees. In contrast, a multilocation company may need to provide a greater number of different service types (TSO, VM/370, etc.) as well as provide more service offerings. In the IBM System Communications Division Computation Network (SCD-C/N), for example, there are five major service types and over 60 different service offerings provided. The formality of the service-level concept is most suited to the latter environment. The second data base size factor is determined by the number of reporting periods (T).

In the service-level application developed for IBM's SCD-C/N, a time span of 20 weekly intervals was chosen. The data base is conceptually rotated about the "T" axis and then updated to contain the most recent and the prior 19 weeks of information. Twenty data intervals are sufficient for data plotting and to visually observe significant deviations in the service level. In this scheme, typical reporting events, such as the end-of-year, end-of-quarter, and end-of-month times, are absorbed into the ongoing 20-week cycle. Assume each reporting period is noted by its week-ending date. The time dimension can be illustrated as in Figure 7.

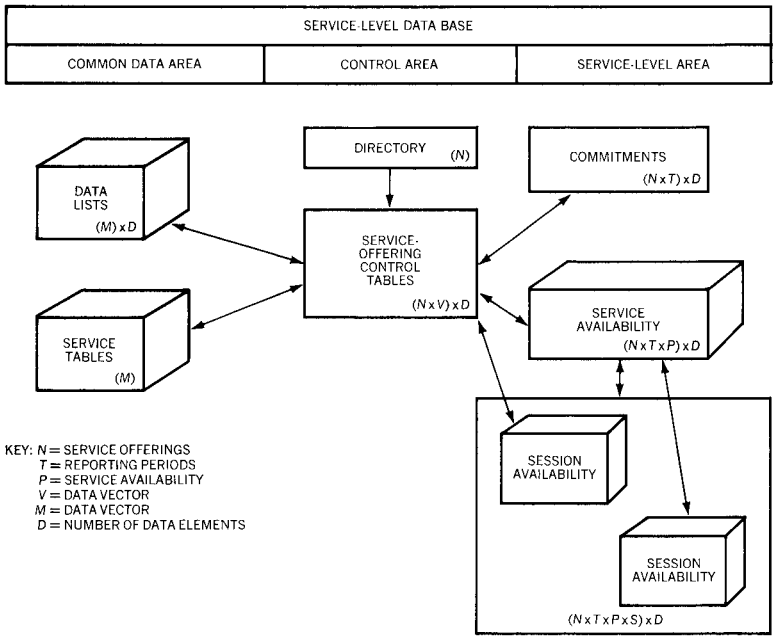
Figure 8 Service-level data-base control area



The third data base size factor is determined by the number of data elements (D) associated with a single service and a single reporting period. The number of data elements is determined by the logical structure of the data base.

The overall data base may be logically divided into the following three data areas: (1) the control area illustrated in Figure 8, (2) a common data area, and (3) the service-level area. Within each of these global areas, illustrated in Figure 9, several additional levels of structuring will be indicated. For purposes of structuring the data base, the data elements will be grouped and illustrated as data blocks. The logical linkage and association of information between blocks of data will be referred to as data pointers.

Figure 9 Data-base structure for service-level application



The control area illustrated in Figure 8 represents the nucleus of the service-level data-base structure. It serves as the service-level application's control block for data-base management. Each of the service-level application programs, such as those which update or report service-level information, access the data by indexing the data base through the control area. In the control area (Figure 8) are: (1) the service directory which consists of data pointers to a control table associated with each service offering and (2) the control tables which have data pointers to the service-level areas and the common area. The service-level area pointers link the commitments, service availability and session availability data blocks together. The index vector contains pointers to each of the data lists in the common data area.

A design point of the data-base structure is to have the data content be completely table-driven. This facilitates the data management function and enables the data base to be virtually open-ended. For example, a data list exists in the common area that contains the names of all locations serviced by the application. If the sixth element in the list contained the name "Kingston" and a service offering had that location as its host site, then the index vector associated with the host location would have a value of six. The variable "Kingston" would be stored only once in the data base. By use of the control area, the size of the data base may also be changed by the application's data-base administrator without requiring programming changes.

The common area of the data base contains many blocks of data of varying length and format. These blocks of data are really data lists used in the description of the service and to describe the many different numerical measures of service. These data blocks also serve as tables of information that can be shared across the service offerings. In addition to saving data-base space, this structure simplifies the data-base management task. For example, the name of a new location has to be added to the data base in only one place.

The service-level area is the most complex of the data-base areas in terms of structure. In addition to the dimension of repetitive data blocks caused by the number of service offerings, in this area of the data base, each set of data blocks will be repeated for each of the reporting intervals. One such group of data blocks, which have an $N \times T$ dimension, where N is the number of service offerings and T the number of reporting periods, contains those data elements that are used to define the level of service.

Since service-level commitments can change from one data collection period to another, it is necessary to have a separate block of commitment data for each week of retained actual data, thus allowing a valid comparison to be made between the commitment and the achievement information on a per-period basis. Certain elements of commitment data, such as the information pertaining to reliability, have no correlation to measures of service achievement.

Another set of data blocks is used to record those measures of service that are associated with each period of achieved service availability. In addition to an $N \times T$ dimension, these data block sets have an $N \times T \times P$ dimension where P represents the number of availability periods. A typical service will have about eight to 12 periods per week. The last set of data blocks is used to record the measures of service that are associated with each user session within each service-availability period. The dimension of this portion of the data base will be $N \times T \times P \times S$ where S represents the number of user sessions. For a heavily used interactive service offering, S will have a value in the hundreds.

The thing that distinguishes a service-level-oriented data base from a data base applicable for systems performance evaluation or installation accounting is the inclusion of the session-availability data structure. The logical data-base structure for a service-level application would appear as in Figure 9. Given this data-base structure, the next task is to determine the data content and format.

Data-base content

This section of the paper is concerned with translating the architecture into data elements. The functional grouping used in the discussion on architecture is retained here for ease of description.

availability

Commitment data elements define the starting and ending times that are planned for a particular service to be available for productive use. The commitment of weekly service availability is in scheduled hours per week.

The service-level commitment data elements are:

- Scheduled start of service (work day and time of day)
- Scheduled end of service (work day and time of day)

Assume the scheduled start time to be 8 am daily and the scheduled end time to be 5 pm daily. The scheduled availability would be 45 hours per week.

Against these benchmarks the actual service and session-availability measurements will be captured and reported. The service-availability data elements contain the measured starting and ending times for the service. Accurately measuring the ending or point in time that the service could no longer have supported a user is quite difficult. If the service was normally concluded, sufficient System Management Facility (SMF) records are produced that yield the exact time. If, however, the service abnormally terminates where no audit trail was created or if a system component fails that effectively renders the system unusable but statistically available, accurate information will not exist beyond the last checkpoint. In this situation, such as a looping condition would produce, the actual availability data elements may have to be updated with manually recorded operator information in lieu of SMF. The three service-availability data elements are:

- Actual start time (work day and time of day)
- Actual end time (work day and time of day)
- Actual service time (hours per week)

A typical week of service may produce the measures of availability in Table 3.

For each user session three data elements are created to record the session start time, end time, and duration or connect time. For most services, a log-on, or accounting record, is created every time a user signs on the service. A similar SMF data record is created when a user logs off or abnormally ends his session. If the service or system crashes, however, both the log-on and/or

Table 3 Typical measures of availability

Period ID	*Actual start time (Time of day)*			*Actual end time (Time of day)*			*Service availability interval*	
	Workday	*Hr.*	*Hundredths*	*Workday*	*Hr.*	*Hundredths*	*Hr.*	*Hundredths*
1	112	15	19	112	19	03	3	84
2	112	19	51	112	20	76	1	25
3	112	21	05	113	1	15	4	10
4	113	6	35	113	11	50	5	15

Table 4 Measures of session availability

Session ID	*Start time*		*Termination time*		*Session length*	
	Hour	*Hundredths*	*Hour*	*Hundredths*	*Hour*	*Hundredths*
AZ	8	70	10	21	1	51
QL	8	91	9	15	0	24
JX	10	21	11	50	1	29

log-off data may be unrecoverable. The log-on information is actually stored and reported on the SMF log-off record which accounts for there being no log-on data when the system crashes. The mapping of session availability into service availability must be accomplished using the time and date information in each of the respective records since a unique record identifier is not assignable to the service-availability period that can be carried forward to each of the session-availability records. The three user session-availability data elements are:

- User session start (session identification (ID) and time of day)
- User session termination (session identification and time of day)
- User session length (hours)

For a typical period of service availability such as for Period ID4 in Table 3, the measures of session availability in Table 4 could have been produced. The session ID may be the user's sign-on identification.

capacity

Commitment data elements are used to define the workload capacity that can be provided to each user based on the defined accessibility mix. Workload capacity is expressed in terms of units per hour of service availability. A pair of data elements is required for each measure of capacity. One element would de-

Table 5 Example of data elements for capacity

Data element 1 *Description of workload unit*	*Data element 2* *Capacity measure*
TSO Trivial transaction	425
TSO Moderate transactions	60
TSO Complex transactions	2
TSO Account ADD	30

Table 6 Data elements for utilization

Description	*Utilization*
TSO Trivial transactions	1,621
TSO Moderate transactions	218
TSO Complex transactions	10
Compiles Class 'A'	86

scribe the workload unit, and the other would contain the capacity measure. An example of these data elements is listed in Table 5.

utilization

The major difference between a service-level data base and the data bases used in the ordinary data processing accounting application is in the level at which utilization information is stored. The architecture for utilization describes the hierarchy beginning with hardware data, systems data, service data, and finally, user work-defined data.

Workload-defined utilization data is not easily captured nor is it in a form available directly from the SMF. Extensive preprocessing is required to accurately translate measures of hardware utilization collected by the SMF and by the other ancillary applications into the various defined units of workload utilization associated with each user session.

Measures of utilization would be recorded in both the service-availability and session-availability data-base areas. In the service-availability area, utilization would be stored as a table of description data elements and their associated units-of-measure data elements. An example of such a table forms Table 6.

The values represent the cumulative utilization for all users within a single service-availability period. In the service-availability data-base area, an open-ended vector of data elements

Table 7 Example of service table

Workload type	*Service value*
ACCOUNT ADD	0.6
ALLOCATE	0.1
ASM	2.7
DELETE	0.3

Table 8 Example of workload service values

Workload classes	*Service value*
COMPILE–A	570
COMPILE–E	1005
SORT–Y	875
SIMULATION–A	6000
ASSEMBLY–QX	110

would be created to record each workload unit. If each element is assumed to have a value of one workload unit, only the description list pointers need be recorded.

performance

For each service offering and workload type or class, a service table is established in the common area of the data base. Data elements in the service table would reflect the workload identifier and the related service value. For an interactive service offering, a service table entry would represent a command or high-level function. A portion of the service table would appear as in Table 7.

The service values for units of workload associated with a batch-processed service offering will be significantly larger because they must reflect the total job. However, instead of a list of values common to each TSO application, a single service value will be stored for each class of workload, as in Table 8, for example.

The actual measure of performance will require several data elements. The quantity depends upon the level of detail at which performance is recorded. Assuming a summary level of one measurement per session-availability period, the following data elements are required: (1) an element that contains the cumulative service value and (2) an element that contains the cumulative service interval. For a batch service offering, the latter would be reflected by the elapsed time for the job. For an interactive service offering, it would be the sum of the queue and

Table 9 Relationship of the probability distribution

Descriptive elements		*Data vectors*						
(1) Hours		3	4	5	6	7	8	9 (etc.)
(3) ≥								
(2) Continuity		0.8	0.6	0.5	0.45	0.4	0.3	0.2

process intervals defined in the discussion of performance in the section on architecture earlier in this paper. With the use of the above data elements, Streeter's algorithm for calculating relative value of the service[1] can be applied.

reliability

The data-base structure for reliability measures consists first of multiple pairs of data vectors that are used to describe the anticipated reliability associated with some aspect of availability, capacity, or performance. Associated with each pair of data vectors are three descriptive data elements. The first element describes the scale represented by the first data vector in each pair. The second element describes the meaning associated with the second vector in each pair. The third element describes the mathematical relationship of the probability distribution, which is contained in the second data vector to the scale as, for example, in Table 9, which is the data-base recording of the belief that a "user" can expect to achieve a continuous session length of between six and seven hours only 45 out of every 100 attempts.

The levels of service reflected by the reliability measures usually do not have a linear distribution. If the user is provided with a distribution in the format described above, he can easily establish his degree of confidence in the service offering based upon where he typically intersects the distribution.

The second set of reliability measures consists of data elements that record in summary form the attained level of service not recorded by the other service levels. For example, the actual continuity, or measure of service availability, is recorded in the availability portion of the data base. The measure of the actual duration of a service outage is not, however, recorded elsewhere. This measure and similar measures would be recorded in the data base under reliability. Often it is desirable to record, in addition to the arithmetic mean, additional measures such as the mode, standard deviation, and range values. In the service-level concept, this information provides the input for statistically creating the anticipatory kinds of reliability measures.

We have discussed the existence of a well-structured data base that reflects the service-level architecture. We now focus on the last of the service-level concepts, the communications aspect.

Communicating service information

When a data base has been created that contains service-related information, there is a tendency to create a variety of reports. Usually these reports are implemented to satisfy a specific requirement of the computing facility. Often they are also circulated among users in an attempt to squelch the user's need for service-level information. The fact is, regardless of how superb the report format is or how encompassing the report content is, the same report cannot satisfy the needs of both the service user and the computer center personnel. Users require reports that have a different level of information and different time orientation of the information than an installation manager would need.

Service-level information can be communicated at three distinguishable plateaus. The first would be in terms of the information associated with each period of session availability. This detailed level necessitates a separate report to be created for each interactive session and batch job. The significant difference between these reports and the currently available SMF detailed reports is in the level of the information. The service-level-oriented reports would address the user's workload utilization and not simply report EXCPs, etc. In addition, the deviations from the service-level commitments would be included.

The advantage of this plateau is that the user would be able to relate his experience in using the service to the service-level information being reported to him. The disadvantages are: (1) a large volume of reports would be produced each week, (2) they do not provide an overview perspective of the level of service provided, and (3) prior session information cannot be combined for trending because of the varying intervals of the measurement periods. These reports, while ideal for a user, would not be suitable for the computer installation's personnel except as a reference to a user's complaint.

The second plateau represents the summary of all information associated with each period of service availability. A separate report would be created for each such period and would reflect a summary of all user sessions. Each report would provide the computer installation with an excellent picture of how well the level of service was relative to their commitments. These reports would be detailed enough to indicate which component (availability, capacity, performance), if any, is responsible for a degradation in the level of service. They would also provide management with an indication as to how well operations adhered to the service plan.

The disadvantages of these reports are: (1) individual user identity is lost along with the ability to focus on his service problem,

Figure 10 Overview of the time perspective

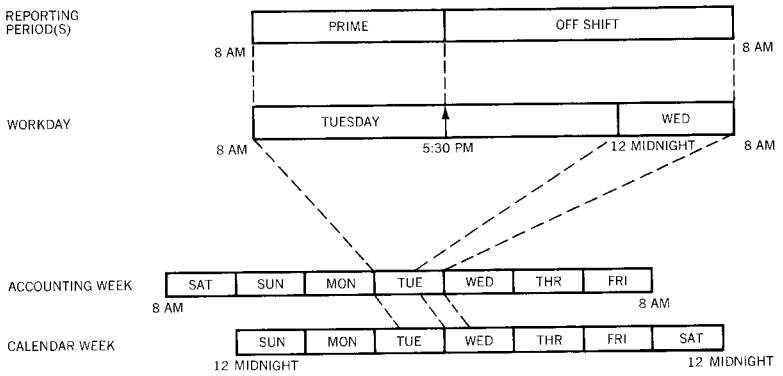
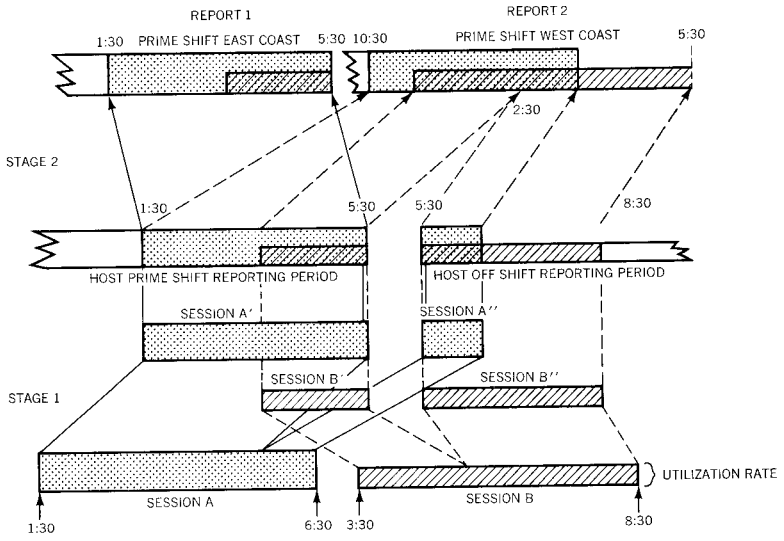


Figure 11 Mapping of session data into reporting information



(2) the reports still portray the level of service out of context with the overall level of service, and (3) different reports cannot be easily relatable because of the different lengths in the measurement intervals.

The third plateau represents the reporting of all service information aggregated into a reporting time interval such as illustrated in Figure 10. These reports would be most useful to a level of management that is interested in the gross service-level perspective as it relates to the cost of service, be they representative of the users or of the providers of the service. The most notable advantage of this level of reporting is the ability to present the information in an historical context, e.g., what has happened over the past 10 weeks, etc.

The disadvantage is that the session and service data must be post-processed into an artificial time-structured relationship, as illustrated in Figure 11. Unless this transformation is properly implemented, readers of these reports will lose their orientation to, acceptance of, and belief in the report should they attempt to reference a specific service problem.

The importance of defining the appropriate reporting time structure and properly post-processing the data-base information is underscored in the remainder of this paper. In the service-level concept, the time orientation of the information becomes the keystone in meaningfully communicating service-level information. Figure 10 presents an overview of the time perspective discussed in this portion of the paper.

The most popular time-related reporting interval is the calendar week. Many companies have established an accounting week for record-keeping purposes that may be offset from the calendar week and could begin, for example, at 8 am on Saturday and continue through 8 am the following Saturday. Within this period, employees typically think of the normal workweek as beginning at 8 am on Monday through 8 am on the following Saturday, a time span of 120 hours. The starting times are synchronized to the start of a major work shift.

A logical subset of the reporting week is the *workday*. The workday for reporting purposes, like the week, begins at 8 am and ends at 8 am on the following calendar day. The subtlety of the time offset and significance to service-level reporting results from the fact that, in contrast, SMF and other computer system-captured information is recorded by calendar days and system clock time.

The workday is usually further divided into multiple reporting periods or shifts to distinguish differences in the use of the service, such as prime and off-shift, or to denote a change in the service-level commitment. A reporting period, as illustrated in Figure 10, becomes the interval within which the data base of service and session availability information must be post-processed for reporting. Reporting periods are therefore mutually exclusive and necessitate the allocation of collected service-level information which may have occurred across the boundaries of reporting periods into the individual periods. The boundaries, however, must first be aligned with the report reader's perceptions.

To place each report in proper perspective and to establish a base for data analysis, the following additional three data elements are necessary.

- Period/Shift Start (time of day)—the moment of time in which each measurement period begins. It is significant to note that the value for this data element and the next is usually determined by the data processing organization and not the users of the service who rely on the reports.
- Period/Shift End (time of day)—the moment of time in which each measurement period ends. Both this and the prior data element are important to understanding and evaluating the achieved availability measurements and should be included in each service-level report, a procedure not usually implemented.
- Reporting period (hours)—the calculated difference between the period/shift and stop times.

In addition to defining the bounds of the reporting periods, it is necessary to establish the data collection starting and/or cutoff date and time. The date should be equivalent to the host location's date for the beginning of an accounting week. The time should correspond to the earliest period/shift starting time of the remote locations being serviced. This procedure will ensure that all of the data will be included in the post-processing of the data base to account for the time orientation.

If we utilize the above time structure illustrated in Figure 10, the following stages of post-processing must be performed. First, the data for each session-availability period in the service-level data base must be mapped into one or more discrete reporting periods. Where a session overlaps two or more reporting periods, it is necessary for consistent reporting to artificially terminate and originate the sessions at the reporting period boundaries. Other measures of service for capacity, utilization, and performance must be adjusted accordingly, as illustrated in Figure 11. In this illustration, the length of the shaded area represents the availability period, whereas the height represents the other measures such as utilization. Summary information is illustrated by the overlapped shaded area.

Stage two of the post-processing involves logically orienting a reporting period to the reader's time perception. For example, the prime-shift reporting period may be defined at an East Coast host to be the interval between 8 am and 5:30 pm EST. In communicating the data-base information to readers at the host location, the reports would have the proper time orientation. However, to a reader on the West Coast, the same reports would not have the proper time orientation. In stage two, either the reporting period time must be modified, such as redefining the prime shift to be 5 am to 2:30 pm PST or the contents of the reporting period relocated to include the West Coast perception of service between 8 am and 5:30 pm PST. The point which is visually depicted at the top of Figure 11 is that the information requirements of the users of Report 2 cannot be satisfied by Report 1.

The trend in data processing is toward computation networks in which users at one site have a high probability of being serviced at another host site. In the future, use of communication satellites will further increase the potential that a service offering will cross time zones. It is suggested that, in order to eliminate the above kinds of problems, the emphasis in service-level reporting be on the post-processing aspect of communicating the information and not on the report formats.

It should also be noted that while the report format, in the above context, is not a major factor in the concept of service levels, several excellent ideas in service-level report formats are presented in a paper entitled, "A Graphical Computer Performance Report for Management."[13] Whatever format is selected, it is essential that the information be presented relative to the service-level commitment.

Conclusion

A major intellectual step forward in the relationship between a computer center and its remote users is possible through the establishment and tracking of service levels. Remote users generally complain about the service and the computer centers usually ignore them. The result is a lot of finger pointing and very little gathering of factual data. The environment surrounding the use of computers is rapidly changing, and the users and providers of the service must seek a common ground.

It is acknowledged that the concept of service levels is ancillary to the actual operations of a computing facility, and this cannot directly affect the quality of a service offering for better or for worse. The belief is, however, that where service levels are implemented, they make a positive contribution toward a computer center's communications with its users. In addition to providing facts about the service in very objective terms, service levels aid in properly orienting the user's perceptions about the service offering. A well-implemented service-level application would therefore enable the user to plan his workload more productively and thus improve upon his utilization of a given service offering. In this context, experience has proven service levels are an important and, perhaps, new kind of tool for both computer center management and users, be they engineers, programmers, or administrative personnel.

The concept of service levels has been presented in terms of (1) the architecture, (2) the data-base content and format, and (3) communicating the service-level information.

The architecture as presented evolved over a two-year period out of the necessity of combining the separate service-reporting

efforts of eight computer centers into a single methodology which could then be implemented. The guidelines established were very simple: (1) Avoid collecting data and reporting it as information simply because it is capturable by SMF or some other system facility. (2) Each item of data must fit into the overall relationship of the measures of service. (3) The content of a service level must be of some decision or planning value to the user.

The data base for a service-level application has less than 100 data elements. The data-base structure is, however, complex and quite important for the proper implementation of the service-level concept. The major data-base-oriented implementation tasks were, and still are, related to (1) logically auditing all commitment and actual measures of service prior to having the data base updated and (2) developing consistent data collection and reduction programs across the various service offerings for the eight host locations involved.

Communicating the service-level information has been the most challenging task. Every computer installation publishes some kind of status report that is generally shared by the service personnel and the users. When there are no constraints on the measures of service obtainable, i.e., when capacity far exceeds utilization, when availability is not a problem, and when performance is excellent, users may glance at these reports and accept the information. However, when a service problem occurred, users agree that such reports were worthless in identifying the problem and denied their accuracy. A case-in-point: before the introduction of the service-level application, an availability problem occurred with one service offering. The host installation published reports showing the cumulative availability period was, for example, 80 hours per week, which exceeded the 72 hours per week commitment. Users at another location were complaining to management that because of poor availability the workload was not being processed. They presented counter-availability measurements of about 38 hours. Who was correct? Well, in a way both were. The host measured availability between 8 am and midnight, whereas the users measured it between their normal work hours of 7 am to 3:30 pm.

The emphasis thus far in communicating service-level information has been on eliminating the above kinds of problems. To this end the application and concepts presented have been highly successful. A continuous dialogue was also held with users to determine their requirements and preferences. A significant contribution toward a single report format cannot be presented as the users were quite indifferent about the data arrangement. Their emphasis was, and remains, on accurate data which is presented so that it is relatable to their experience. They also de-

sired to see the information only when a commitment measure of service was missed and then on a timely basis. Additional development is being performed on an inquiry form of exception reporting to satisfy this requirement.

In summary, the service-level concept has been more than an exercise. The implementation has been met with the anticipated resistance on the part of computer center personnel, because of (1) its contending for scarce resources, (2) exposing the computing facilities to being closely measured, and (3) necessitating the publishing of realistic service commitments. Time, education, and management direction have cleared away some of the resistance. Advancing the service-level concept will also clarify a large portion of the remaining problems. By-products of the service-level application are a base for a user workload forecasting application and a base upon which to develop a meaningful cost recovery application.

CITED REFERENCES

1. D. N. Streeter, "Cost-benefit evaluation of scientific computing services," *IBM Systems Journal* **11,** No. 3, 219–233 (1972).
2. S. B. Ghanem, "Computing center optimization by a pricing-priority policy," *IBM Systems Journal* **14,** No. 3, 272–287 (1975).
3. M. M. Morris, "System accounting and resource analysis (SARA)," *Proceedings of SHARE XLIV,* Los Angeles, Session B307, 141–170 (March 6, 1975).
4. P. H. Callaway, "Performance measurement tools for VM/370," *IBM Systems Journal* **14,** No. 2, 135–159 (1975).
5. Y. Bard, "Performance criteria and measurement for a time-sharing system," *IBM Systems Journal* **10,** No. 3, 193–216 (1971).
6. H. W. Lynch and J. B. Page, "The OS/VS Release 2 System Resources Manager," *IBM Systems Journal* **13,** No. 4, 254–291 (1974).
7. *Proceedings of the Eighth Meeting of Computer Performance Evaluation Users Group (CPEUG),* Institute for Computer Sciences and Technology, National Bureau of Standards, Special Publication 401, Washington, D.C. 20235 (September 1974).
8. R. C. Rettus and R. A. Smith, "Accounting control of data processing," *IBM Systems Journal* **11,** No. 1, 74–92 (1972).
9. H. M. Gladney, D. L. Johnson, and R. L. Stone, "Computer installation accounting," *IBM Systems Journal* **14,** No. 4, 314–339 (1975).
10. Y. Bard, "Performance analysis of virtual memory time-sharing systems," *IBM Systems Journal* **14,** No. 4, 366–384 (1975).
11. S. J. Boies, "User behavior on an interactive computer system," *IBM Systems Journal* **13,** No. 1, 2–18 (1974).
12. *OS/VS System Mangement Facilities (SMF),* IBM Systems Library, No. GC35-0004, IBM Corporation, Data Processing Division, White Plains, New York.
13. D. Schumacher, "A graphical computer performance report for management," *Proceedings of SHARE XLV,* New York, Volume 1, 106–113 (August 1975).

GENERAL REFERENCE

D. N. Streeter, "Productivity of computer-dependent workers," *IBM Systems Journal* **14,** No. 3, 292–305 (1975).

Reprint Order No. G321-5040.