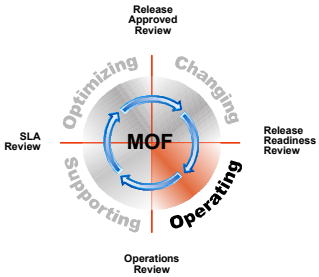
Microsoft®

MOF Service Management Function Job Scheduling

patterns & practices



Microsoft®
Solutions for Management

Contents

Document Purpose

This guide provides detailed information about the job scheduling service management function (SMF) for organizations that have deployed, or are considering deploying, Microsoft technologies in a data center or other type of enterprise computing environment. This is one of the more than 20 SMFs defined and described in Microsoft® Operations Framework (MOF). The guide assumes that the reader is familiar with the intent, background, and fundamental concepts of MOF as well as the Microsoft technologies discussed.

An overview of MOF and its companion, Microsoft Solutions Framework (MSF), is available in the *Introduction to Service Management Functions* guide. This overview guide also provides abstracts of each of the service management functions defined within MOF. Detailed information about the concepts and principles of each of the frameworks is also available in technical papers available at www.microsoft.com/solutions/msm/.

Executive Summary

The job scheduling service management function (SMF) is concerned with ensuring the efficient processing of data at a pre-determined time and in a prescribed sequence to maximize the use of system resources and minimize the impact to online users. A batch process is a system interaction with a database that runs in the background and in a sequential manner without interaction from an end user. The execution of batch processes may be automated or manually initiated. Batches are usually executed after business hours when user interaction with the system is low.

Batch runs typically require their own architecture, as they tend to be resource-intensive and long running, repetitive processes. The process usually involves reading large amounts of data from a database, processing the data, and returning the results back to a database. This process is accomplished through the execution of scripts.

Types of batch jobs that organizations execute include:

- Financial management reports
- Marketing reports
- Supply chain management reports
- Inventory reports
- Invoice reports

- Customer account processing (monthly account billing, and so on)
- Automated backups of system and application data
- System processing summaries and capacity planning reports

Process and Activities

Job Scheduling Overview

Job scheduling involves the continuous organization of jobs and processes into the most efficient sequence, maximizing system throughput and utilization to meet service level agreement (SLA) requirements. Job scheduling is closely tied to service monitoring and control and to capacity management.

Job scheduling entails defining:

- *Job schedules.* The workloads are broken down into time periods (daily, weekly, monthly, annually) and jobs are scheduled for execution according to business needs, length of job, storage requirements, and associated dependencies.
- *Scheduling procedures.* Schedules are set up and maintained, conflicts and problems pertaining to scheduling are managed, and special needs (such as as-needed jobs) are accommodated.
- *Batch processing.* Jobs are executed according to the work schedule, run priority, and job dependencies. Batch-processing procedures include:
  - Job documentation
  - Hardware instructions (for example, tape units, data cartridge units, and printers)
  - Console operations
  - Control checks
  - Problem management

Goals and Objectives

The goal of job scheduling is to ensure the successful execution of batch processes at a time that minimizes the impact on interactive users of system resources. The main activities of job scheduling include monitoring, analysis, tuning, and implementation of batch runs.

Scope

Job scheduling ensures the successful execution of batch runs while minimizing the impact on users of system resources. The main activities of job scheduling include monitoring, analysis, tuning, and implementation of batch runs:

- Job scheduling establishes batch schedules and maintains batch job standards, ensuring that operational level agreements (OLAs), service level agreements (SLAs), and resource thresholds are not breached.
- Job scheduling manages, tests, and implements changes to the schedule via change management, taking account of changes to the scheduling calendar (holidays, for example) and the targets detailed in the OLAs in place.
- Job scheduling monitors and controls the batch schedule, ensuring that batch runs do not affect the live service. It monitors the outcome of successful or unsuccessful batch jobs and establishes alert, error, and event management processes. It also identifies baselines for job scheduling performance.
- Job scheduling implements as-needed requests, ensuring that the existing schedule is not adversely impacted. It is responsible for the analysis and correction of batch process errors, implementing job performance analyses, and tuning batch processes. It also ensures that all changes to the batch schedule are implemented through change management processes.

Key Definitions

Alert. An indication of a significant event. Alerts are defined by processing rules.

Baseline. A frozen "picture" of the IT environment at a set point in time that identifies the structure of the IT environment and the underlying dependencies of the components of that environment. From an availability management perspective, the term also is used to identify an agreed set of availability definitions and targets for an IT service. Such definitions and targets normally will have been proved through modeling and, once defined, will be used as key availability design and reporting criteria.

Batch system. A system that takes a set of commands or jobs, executes them, and returns the results, all without human intervention. This contrasts with an interactive system where the user's commands and the computer's responses are interleaved during a single run. A batch system typically takes its commands from a disk file (or a

set of punched cards or magnetic tape in the past) and returns the results to a file (or prints them). Often there is a queue of jobs that the system processes as resources become available.

Error detection. A technique for detecting when data is lost during transmission. This allows the software to recover lost data by notifying the transmitting computer that it needs to retransmit the data.

Event. Any significant occurrence in the system or an application that requires users to be notified or an entry to be added to a log.

Job scheduling. A MOF service management function in the operating quadrant. It involves the continuous organization of jobs and processes into the most efficient sequence, maximizing system throughput and utilization to meet SLA requirements.

Task scheduler. System or application that automatically invokes scripts or programs at specified times

Major Processes

Job scheduling comprises four main processes and a number of subprocesses as follows:

- Batch architecture
  - Management server
  - Capacity database (CDB)
  - Application servers
  - Monitor and printer
- Batch processing
- Job scheduling activities
  - Monitoring
  - Analyze
  - Tuning
  - Implementation
  - Event management
    - Alerts
  - As-needed request handling
  - Schedule changing
  - System backup
  - Archiving

- Auditing
- Capacity manager log entry
- Reporting
  - Batch schedule
  - Error report
  - Performance report
- Documentation and training

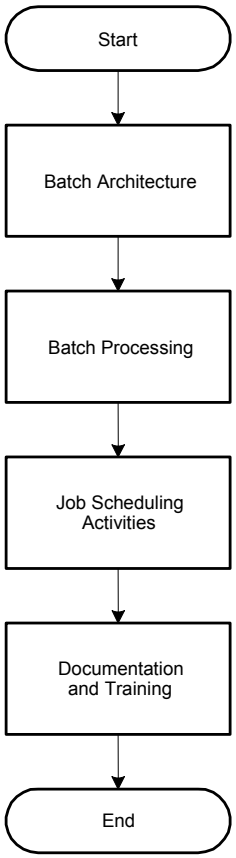


Figure 1
Process flow diagram

Batch Architecture

Before a discussion of daily operational batch activities can begin, the basic components that make up the batch architecture should be understood. A batch architecture consists of the processes and components used to effectively manage batch processing. This section provides a brief introduction to the typical components that can be found in a batch architecture.

The purpose of the batch architecture is to optimize processing (improve response time and utilization of system resources) by executing batch runs during off-peak periods. The architecture

should provide the capacity manager with an easy to use interface and permit a standard and centralized approach to batch scheduling, monitoring, control, and error correction. The architecture should be highly scalable in order to meet the future needs of the organization. It should also be highly available, with minimal downtime, and minimize impact to online operations, which usually are operating concurrently with the batch operations. Some organizations may decide to have backup components, such as the event server, to ensure the completion of all mission-critical batch jobs.

Figure 2 shows the basic components of the batch architecture, which include the management server, capacity database (CDB), monitor, printer, application servers, and databases. In addition to the monitor attached to the management server, each application server should have a monitor to permit viewing of local batch-processing activity; this also facilitates error analysis at the local level.

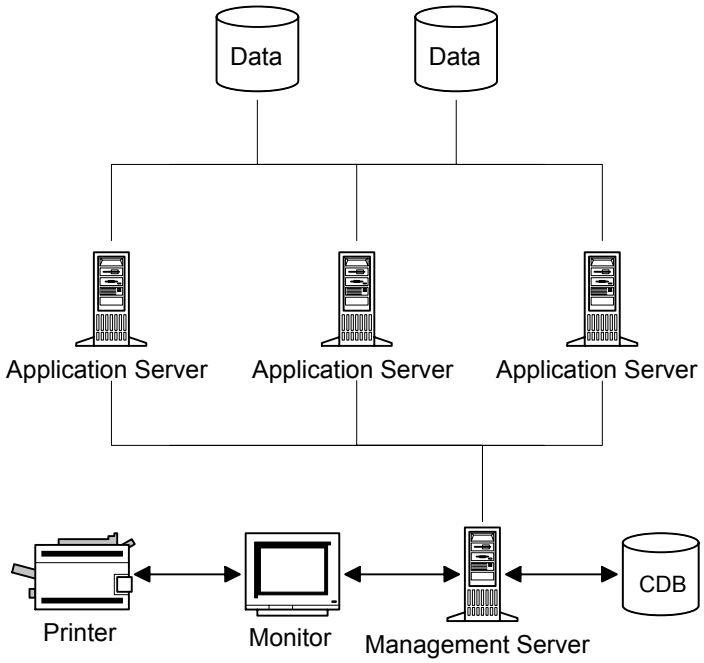


Figure 2

Simplified view of a batch architecture

Management Server

The heart of the batch architecture is the management server on which resides the batch scheduling tool. This tool permits the automatic execution of pre-determined scheduled batch runs. The scheduling tool typically has the capability to automatically perform the following functionality:

- Start and stop jobs based on date, time, day of the week, frequency, and so on
- Define, maintain, and manage job queues
- Prioritize jobs in queue
- Assign jobs to specific servers based on availability
- Track status of jobs and allow real-time monitoring
- Perform error recovery during batch runs
- Report and log errors
- Generate reports
- Archive reports and purge out-of-date reports and log files
- Graphically display all information
- Display job history

Note that the functionality of scheduling tools varies greatly. While some tools may be able to execute the tasks listed above, other tools may only be able to start a batch run at a particular time. Sophisticated scheduling tools should permit the capacity manager the ability to:

- Change schedules
- Change jobs
- Change job priorities
- Start and stop jobs
- Recover or restart failed jobs
- Initiate as-needed jobs
- Generate reports

The user interface of the scheduling tool should be easy for the capacity manager to interpret and use. The capacity manager should have the ability to perform the functional capabilities described above from a centralized location. From this location, the capacity manager should be able to access any information required to control batch processing and troubleshoot errors.

Capacity Database (CDB)

General information about each batch job and metrics that are collected by the scheduling tool are typically stored in a batch (or application) log. Scripts should contain a job step (portion of the script code) that records job execution information in three different log files: the batch log file, the error log file, and the system event log file, which stores significant system events including batch-processing errors and the successful or unsuccessful execution of batch jobs.

The CDB is the central repository for all capacity and performance-related information. Ideally, the batch and error logs should be part of or integrated with the CDB. The batch log contains general information about each batch job and information regarding system performance during the job execution. The error log records batch-processing errors and system component warnings that occur during batch processing. Storing log information centrally facilitates retrieval and management of the information. Keep in mind that the CDB is not necessarily a single repository, but may be a group of repositories that contain all capacity and performance information that is collected for the IT environment.

General information that should be collected about each job before the job is placed into production includes:

- Job name
- Description of the job
- Identification number
- Owner and contact information
- Batch run affiliation
- Batch job steps
- Time and frequency of execution
- Execution window
- Start and end conditions
- Steps where recovery can occur (which job step)
- Batch job duration
- Special conditions when job should not run
- Relationships to and dependencies on other jobs
- Job priority
- Expected results
- Problem resolution procedures
- The application servers that are utilized
- The databases that will be accessed
- Reporting requirements

Performance- and error-related information that should be collected includes:

- The impact of the job on the batch architecture:
    - Memory utilization
    - CPU utilization
    - Network utilization

    - Disk utilization
- The job start and stop times
- The job duration
- What system components were actually used to process the job
- Processing errors
- System warnings when thresholds are exceeded

Each application server that is involved in the batch run should record job processing results and performance metrics in a local log. It should then transfer the information to a central platform established to consolidate all logged information in a single database(s)—the batch log or CDB. Centrally storing batch/error log and descriptive job information allows easy access to and management of pertinent information by the capacity manager—information that is used to optimize system performance and analyze and correct errors. It also allows for easy backing up of important information. Operational reports are typically developed from information stored in this database.

Application Servers

Application servers that execute batch jobs are also part of the batch architecture. Each server has an instance of the scheduling tool installed, which allows the local monitoring and logging of batch-related information and the reporting of this information to a centralized location. The servers interact with the appropriate back-end databases to access the data that is used in the batch job. The local scheduling tool starts the job assigned to it by the event processor and sends completion information or errors to the centralized scheduling tool.

Monitor and Printer

A monitor connected to the management server permits interaction with the scheduling tool and thus the batch process. Scheduling tools typically have the ability to graphically display performance, status, and execution results information. From the monitor, the capacity manager should be able to manually control the batch process. The printer is used by the capacity manager to generate operational reports that are used to assess system performance and processing errors or any other information the capacity manager deems useful.

Batch Processing

This section provides an introduction to batch processing. It describes how a scheduled batch run is executed and what resulting information is recorded.

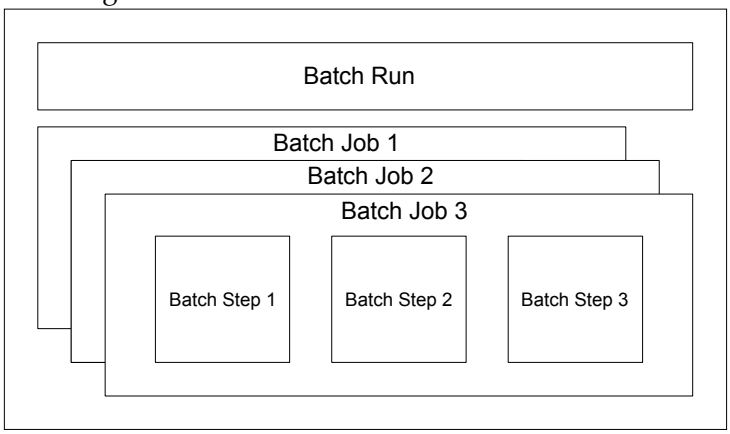


Figure 3

Batch process hierarchy

Before discussing scheduled batch runs, it is useful to first understand the hierarchy of the batch process and the contents of a batch script. Figure 3 shows the batch process hierarchy. A batch run consists of multiple independent batch jobs that are scheduled for execution on a recurring basis. A large organization may execute multiple batch runs throughout the day depending on the resources required to process them. Each batch job consists of multiple batch job steps that control specific activities of job execution.

An organization typically processes numerous batch jobs. To ensure a consistent approach to the execution of each job, a batch job skeleton should be devised that contains the standardized code required for each job; job-specific information should be coded into a designated area within the skeleton. The skeleton also serves to facilitate development and maintenance requirements by standardizing the content and structure of each job script. For example, any type of standardized actions, such as the notification of a successful or non-successful batch job execution and transaction data archiving and deletion, should be included in the code of every script that is executed.

Scheduled batch runs are initiated by a scheduling tool during a pre-defined batch window, typically when user activity of the system is at a minimum. After the scheduling tool has been programmed, capacity manager interaction should not be necessary during batch runs unless an error, from which the tool cannot recover, occurs.

Figure 4 is a flow chart of a typical scheduled batch process. This high-level overview is meant to give the reader a general understanding of how scheduled batch jobs are processed. Keep in mind that the capabilities of the scheduling tool determine exactly what functionality can be automated.

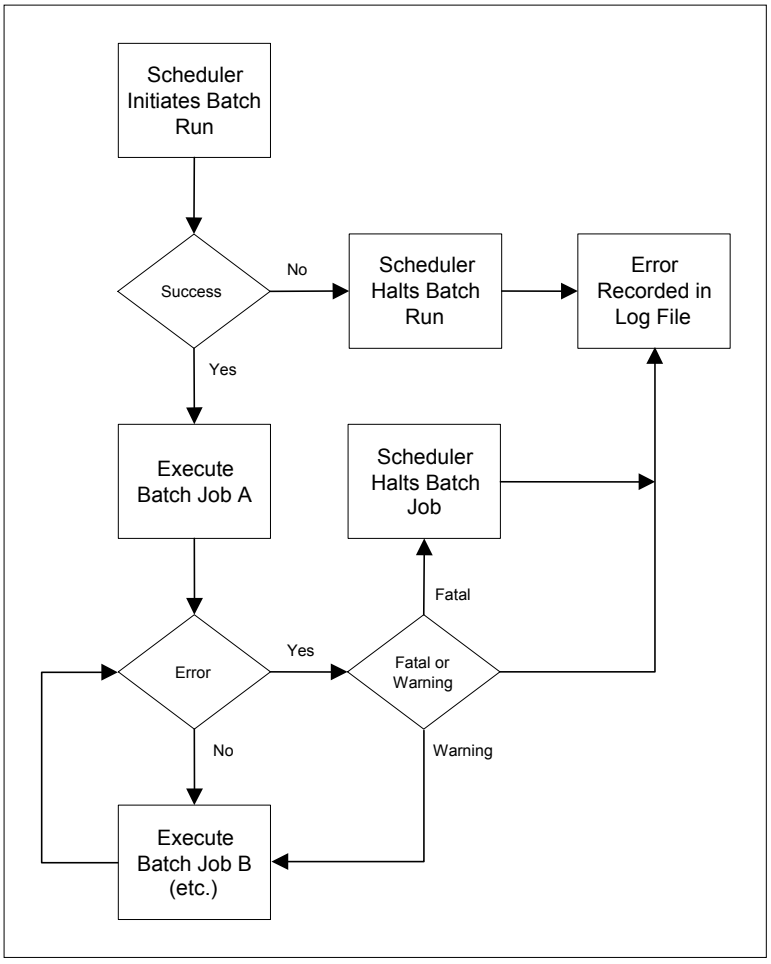


Figure 4

Batch processing flow chart

The scheduling tool initiates all batch runs. If the run does not begin as scheduled, the tool halts the run and records an error in the error log; some scheduling tools may have the ability to attempt to restart the batch run. If the initialization is successful, the first batch job is executed. The scheduler manages the execution of jobs on each application server targeted to perform the batch processing. If no errors are encountered during the execution of the job run, then the tool records the successful completion of the job in the batch log and the next job is executed, and so on.

If an error occurs during the execution of a batch job, the scheduling tool stops processing that job and an error is sent to

the error log. The actual execution of a batch job is dependent on a number of inputs that could be the reason that a job is not executed. For example, the following inputs may be required:

- Availability of production data
- Completion of a job on which the queued job is dependent
- Availability of resources to execute the job
- Priority of the job and the batch window

Warnings may also be generated during job processing if, for example, CPU or disk space capacity exceeds the threshold value for an application server processing the batch job. Warnings should not stop a batch job from being completed; however, the capacity manager should address all warnings as soon as possible because they may lead to future job processing failures. If an error causes a job to be stopped, some scheduling tools attempt to automatically recover from the problem and begin processing the job from the beginning of the job step that was executing when processing was stopped. Recovery is useful when processing very long batch jobs because jobs that are interrupted do not have to be restarted from the beginning. If recovery is not possible, the job needs to be restarted.

If the scheduling tool is not capable of restarting or recovering from a failed job (or able to restart or recover in a specific instance), the capacity manager has to manually initiate restart or recovery procedures. This situation is covered more extensively in the "Job Scheduling Activities" section below.

Job Scheduling Activities

Ideally, the batch architecture should be configured in such a manner that capacity manager interaction is kept to a minimum. However, there are still many daily tasks that the capacity manager must perform, including:

- Monitoring
- Analysis
- Tuning
- Implementation
- Event management
- As-needed request handling
- Schedule changing
- System backup
- Archiving
- Auditing
- Capacity manager log entry

- Reporting

Each of these activities is an integral part of the job scheduling process. The monitoring, analysis, tuning, and implementation activities form an iterative process as shown in Figure 5. Inputs to the process include resource utilization and OLA thresholds against which the batch architecture is monitored. These are ongoing activities whose goal is to optimize the performance and utilization of the architecture. The remaining activities are performed in response to an event, request, or requirement.

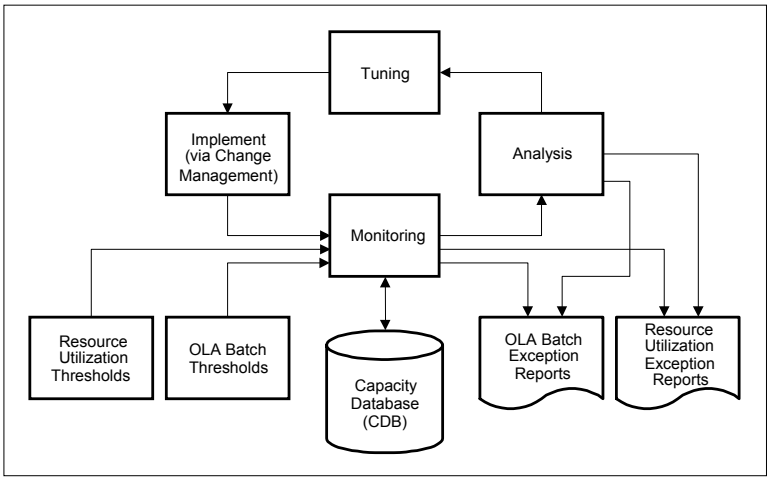


Figure 5

Iterative activities for resource management and service performance management of the batch architecture

Monitoring

Monitoring is a key activity involving the improving of daily batch-processing performance and the planning for future capacity requirements. It is important that the utilization of each resource and service is monitored on an on-going basis to ensure the optimum use of the hardware and software resources and that all agreed service levels can be achieved. The collected information (metrics) is typically stored in the capacity database (CDB). The metrics are analyzed for trends and are used to tune or adapt the system architecture and to aid in planning the job timing and interdependencies when adding new batch jobs in the future. Metrics that are typically collected include:

- CPU utilization
- Memory utilization
- Disk utilization
- Network utilization
- File store utilization

- Batch duration profiles
- Dependent subsystems (that is, database servers)
- Queue length (maximum and average)
- Transactions
- Transactions per second
- Transaction response time
- Percentage CPU usage per transaction

The data should be gathered at total resource utilization levels and a detailed profile for the workload for each system component, resulting from batch processing, should be developed. This information should be used in conjunction with similar information about system components designated for online users. This complete picture helps the capacity manager assess how batch processing affects the resource utilization and thus system performance of the entire IT infrastructure.

Part of the monitoring activity involves establishing a baseline, or profile, of the normal operating levels. For example, the capacity manager should develop a profile for each batch job in order to assess performance problems and how changes to the job affect resource usage, and so on. If thresholds beyond the norm are exceeded, alarms are raised and warnings are recorded in the error log.

It is also important to remember that monitoring takes up system capacity and can influence the performance of the system. Monitoring should be focused on performance measurement and OLA thresholds. Operating level requirements and other necessary elements for monitoring are often derived from their overall contribution to meeting the OLA.

During batch runs, the capacity manager should monitor batch process activity from the central scheduling tool interface, where the status of all scheduled jobs can be viewed (in Figure 6, this interface is represented by the monitor). The success or failure of batch jobs, log information, and details about each job step should be available at this interface. In addition, details about system performance and utilization should be available in a graphical format so that the capacity manager can assess trends in these metrics. The monitoring of performance information also helps the capacity manager to identify any bottlenecks that exist in the batch architecture. In the event of a batch-processing error, the capacity manager is notified of the error and is in a position to take corrective action if the scheduling tool is unable to restart or recover from the problem.

The metrics that are collected should be stored in the CDB so that other SMFs have access to the information. For instance, capacity

management can access historic metric information for the planning of future system capacity requirements. This information allows capacity management to plan system changes in advance of when they are needed, which results in better system availability and performance and fewer processing errors.

Analyze

Data monitored and collected is analyzed and used to carry out tuning of the batch architecture. Before tuning can begin, proper analysis must be conducted to ensure that the components being adjusted are in fact the items causing the problem. The data collected from the monitoring activity should be analyzed to identify trends from which the normal utilization and service level, or baseline, can be established. By regularly monitoring and comparing with the baseline, exception conditions in the utilization of individual components or service thresholds can be defined, and breaches or near misses in the OLAs can be reported and appropriate action taken. Analysis of the data should be conducted following each batch run and may identify such issues as:

- Contention (data, file, memory, processor)
- Inappropriate distribution of workload across available resources
- Inefficiencies in the application (batch script) design
- Inefficient use of memory

Before making tuning adjustments, the capacity manager must have a complete picture of how all system "customers" will be affected by batch architecture optimization techniques. As shown in Figure 6, all batch requests originate from three sources: online, scheduled, and as-needed requests.

Online requests can be further segmented into internal and external requests. For example, an external user may query a Web site for information pertaining to clothing that the organization sells, while an internal customer may submit a request for customer invoice information. Online requests have a high system impact that can be predicted but with a relatively large degree of error (as the result of online usage spikes). All online requests utilize system resources and must be accounted for when optimizing batch-processing performance. Metrics for resource utilization of online requests should be available from system monitoring information that is collected. To plan for future resource requirements, capacity management also uses this information.

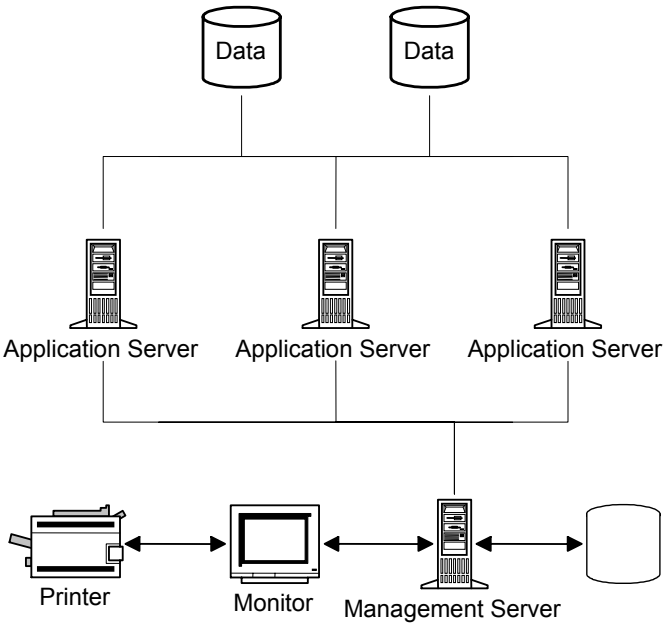


Figure 6

Sources of system resource utilization

Scheduled requests are the automated batch runs that have been discussed throughout this document. These requests have a high impact on system resources; this impact is highly predictable and the capacity manager should schedule batch runs in an effort to optimize system performance. As-needed requests are manually executed by a capacity manager in response to a user request for information, such as the number of users currently utilizing system resources. These requests are not predictable and have a low impact on system resources. Each of these three sources must be analyzed when attempting to determine how to optimize system performance for scheduled batch runs.

Keep in mind that the most important factor influencing optimization is the capacity that is needed to support online requests. Internal and external users must be given priority for the use of system resources because external users will not visit slow Web sites and internal users are less productive when they are hampered by a slow system.

Analysis is also instrumental in the resolution of exception reports and alarms that are signaled by the scheduling tool. Ideally, all thresholds should be set below the level at which the resource is over-utilized, or below the targets in the OLAs. This enables the capacity manager to take corrective action before the targets in the OLAs have been breached or the resource has become over-utilized and there has been a period of poor performance. However, in many cases, analysis must be conducted following the occurrence of an error or warning. The

handling of these conditions is covered in detail in the "Event Management" section below.

Tuning

Tuning for performance is the optimization of system resources to achieve acceptable transaction response times under maximum load conditions. The analysis of the monitored data may identify areas of the batch architecture or applications that can be tuned to better utilize the system resources and improve performance.

Once the performance affects have been assessed, the capacity manager can take steps to mitigate adverse affects to system performance. In a batch architecture there are primarily four areas that can be tuned:

- *Adjust the schedule of jobs.* The order of jobs can be adjusted to minimize the impact of changing the batch run or to improve the performance of the run. These options are available:
  - *Segment the batch into smaller batches.* Divide the job into segments that can be run at any time during the batch window. This segmentation gives the capacity manager more flexibility when setting up and prioritizing the job schedule. Restarting jobs is also easier and less time consuming, as the entire job does not have to be rerun.
  - *If possible, run parallel batches.* Split the batch into two or more segments that run simultaneously. This is only possible if transactions are not dependent on multiple records in the database (if they are, they must be segmented in such a way as to eliminate a dependency conflict).
  - *Change the job priority or job schedule.* Based on the batch run metrics, the capacity manager should determine the optimal scheduling of batch jobs. For instance, it might be better to run two extremely large jobs back-to-back instead of in parallel. The order of job execution affects resource usage; the optimal schedule can be determined by analyzing each job profile and assessing resource availability.
- *Tune at the application (script) level.* Examine the job script to determine if it can be altered to more efficiently manage the batch job.
- *Tune at the database level.* Databases can be indexed so that scripts are able to find information more quickly. The capacity manager must observe the queries that are run against the database, determine what data is accessed most

often, and index the database so that the script can quickly locate the applicable data.

- *Tune at the hardware level.* This involves adding more computing power to the architecture:
  - An additional processor can be added to a server.
  - An additional server can be permanently added to the architecture.
  - Servers can be load-balanced to distribute the transactions more efficiently. The capacity manager can assign some of the batch jobs to servers that are not technically part of the batch architecture to temporarily distribute the load and increase performance.

Keep in mind that tuning at the hardware level is not the "simple" solution that always increases performance. If bottlenecks exist in other parts of the architecture, the addition of more computing power might not increase system performance.

Each of these tuning options can be useful in improving batch-processing performance. However, the capacity manager must understand how tuning decisions will affect other parts of the IT environment before implementing these changes. For example, if load balancing servers, it might not be a good idea to assign scheduled jobs to a server that is primarily used by online users, because the performance of the system may be adversely affected. For this reason, no changes should be made until a request for change (RFC) is submitted to and approved by change management. The change process is in place to ensure that changes to the IT environment do not adversely affect the environment (for more information, refer to the change management SMF).

Implementation

After analyzing system metrics and error logs, if the capacity manager determines that a change needs to be made to the batch architecture, an RFC must be submitted to change management. Change management ensures that only authorized changes are made to the IT environment and that changes are implemented with a minimum impact to that environment. The change management process also ensures that changes to the IT environment are recorded in the configuration management database (CMDB), which is the central repository where information and relationships of all IT components (hardware, software, processes, procedures, documentation, and so on) is stored. As with the CDB, this database is not necessarily a single repository. The CDB might actually be part of the overall CMDB

(for more information, refer to the change management and configuration management SMF guides).

When submitting an RFC, the capacity manager should provide information concerning the impact(s) of implementing the change in the IT environment. This information should be obtained by testing the new components (for example, batch jobs) and assessing the impact they have on the existing architecture. Providing a complete impact analysis when submitting the RFC helps to speed the change review process. Change management evaluates the RFC and either approves or rejects it. If the RFC is rejected, an explanation should be provided describing the reason for rejection. Approved RFCs should be implemented prior to the next scheduled batch run. When implementing changes, the capacity manager should remember to update the batch log with any new information.

Change management directs the capacity manager when the change should be implemented in order to minimize the impact to operational systems. After the change is implemented, the capacity manager should test the change or observe operation of the changed components during the next batch run to ensure that the implementation was successful. If errors occur, the change should be backed out and the environment restored to its original condition. Change management should be notified of the results—successful or not. (For more information on implementing changes, refer to the release management SMF guide.)

Event Management

Event management is the monitoring of batch process successes and failures. This section focuses on what actions should be taken when batch-processing errors occur. Event management is typically part of the scheduling tool functionality.

Event monitoring verifies that production systems are continually functioning in accordance with defined service levels. The job scheduling tool receives events and then routes notifications to the appropriate components for further handling (for example, electronic message from applications, hardware, and so on). It also provides a historic record of events that have occurred in the system architecture.

Events that are tracked and recorded include errors and warnings. A fatal error is the result of a failed batch job and must be corrected before the job can be completed and dependent jobs can be run. Warnings indicate a problem with a system component (that is, thresholds being exceeded) that requires investigation. Warnings should not prevent the successful

completion of a batch job, but may indicate problems that will cause failures on successive and/or dependent jobs.

When a batch job is run, a success or non-success code should be returned to the scheduling tool. In some cases, the scheduling tool might be able to correct the error and restart or recover from the failed job. It may also have the functionality to adjust the batch run so that batch jobs that are dependent on the output of the failed job are not run until the failure is corrected. Otherwise, the capacity manager needs to manually restart or recover the failed job after the error condition is corrected. A set of procedures should be defined and documented for the manual restart or recovery and, if necessary, the escalation of failed batch jobs.

When an error occurs, the cause needs to be identified from information recorded in the error log file and affected end users and owners of the batch job(s) should be notified. In the event the capacity manager cannot correct the error condition, it is escalated as a service ticket to the service desk (for more information, refer to the incident management and service desk SMF guides). Depending on the cause of the error, this could in turn result in the identification of a problem and the submission of an RFC to change management if correction of the error is dependent on adding, removing, or changing a configuration item in the batch infrastructure (for more information, refer to the change management and configuration management SMF guides).

As with any system, proactive planning should reduce the frequency and impact of errors that occur during system operation. A concerted effort should be made by capacity management and storage management to plan for future batch-processing demands in an effort to alleviate such problems as insufficient processing capacity and storage availability. This proactive approach should help reduce the errors that the capacity manager faces on a daily operational basis. Proactive planning should thus reduce the need for reactive responses.

The event management activity should be highly automated. Depending on the system capabilities, either the scheduling tool or, if the tool has limited functionality, the application script, should perform the following tasks:

- Recognize that a failure event occurred.
- Log the failed event.
- Submit an error that the job failed.
- Automatically correct the failure or adjust the batch schedule.

Scripts can be written to accomplish some or all of these tasks if the scheduling tool is incapable of performing them. The objective is to automate the process as fully as possible. This prevents user error from entering into the batch-processing activity and ensures that corrective actions are taken with a minimum amount of delay.

Alerts

This section introduces the alerts that prompt automatic system and manual capacity manager event management actions. An alert is typically an audible or visual notification that some error condition (indicating a failed batch job) or warning (indicating that a threshold has been exceeded) has occurred. The capacity manager should respond to error and warning alerts in a timely fashion to ensure proper system operation.

Errors signal that a batch job was not completely executed. Typical errors that may be received include:

- The batch job exceeds the defined run time.
- The job cannot start.
- The job fails.

Warnings are an indication that thresholds are being exceeded and capacity manager response may be required to prevent a system failure. Threshold warnings may be received for:

- CPU utilization
- Network utilization
- Disk utilization
- Memory utilization

The capacity manager should attempt to correct warnings by optimizing the batch architecture as described in the tuning section. Errors and warnings that cannot be corrected should be escalated to the service desk in the form of a service ticket.

Thresholds for batch architecture component utilization are established by capacity management and documented as operational level agreements (OLAs). Care should be taken to properly input threshold levels when establishing system-monitoring points. Any changes made to thresholds should not be made until an RFC has been approved by change management.

After configuring the system to monitor specific threshold levels, it is important that qualified personnel perform the system monitoring activity. A capacity manager who is not able to recognize and diagnose system problems will not have the ability to correct errors. It is important that the capacity manager be

sufficiently trained and that documentation is available to assist the individual in identifying and correcting problems. If possible, it is useful to configure the scheduling tool to automatically generate a service ticket in the event of an error or warning. This helps to ensure that the problem is identified and should help to minimize the resolution time.

As-Needed Request Handling

As-needed requests are typically non-repetitive requests by users to manually run a batch job. Figure 7 shows the process for handling as-needed requests.

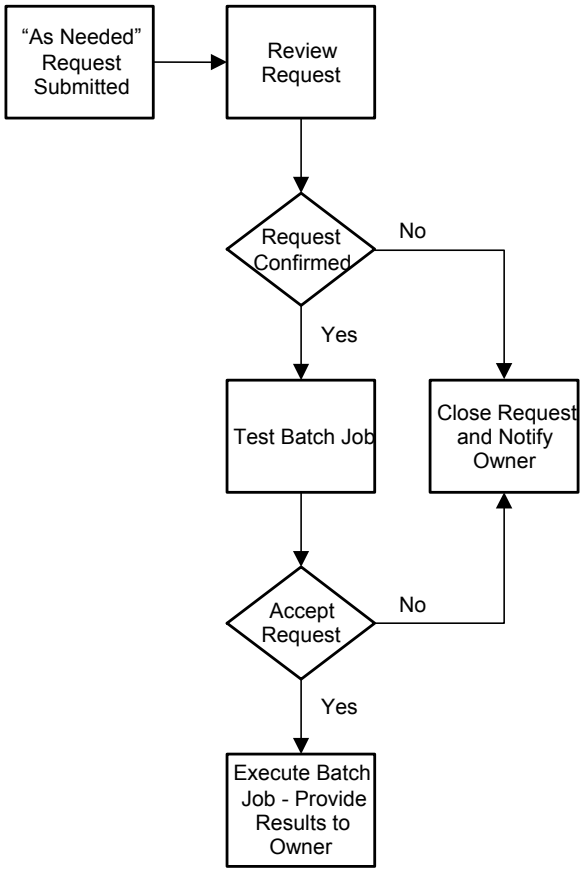


Figure 7

As-needed request process

All as-needed requests that are submitted should follow the process in Figure 7. All requests should be submitted using the batch skeleton format, which facilitates review and approval of requests. The capacity manager begins the process by assessing the submitted request to determine if all required information has been included. If information is missing, the request is cancelled and the owner is notified and provided with an explanation; the owner may resubmit requests at any time.

Accepted as-needed requests should be executed in a testing environment and a batch job profile, a set of metrics describing the job's execution performance, should be developed. The profile should be archived in the event the request is resubmitted. If the test is unsuccessful, an attempt should be made to determine the cause of the failure and the request should be returned to the owner and an explanation provided. Typically, errors result from incomplete or inaccurately coded batch job scripts.

Based on the profile, the capacity manager decides when the job should be executed, taking into account the resource requirements of the job and the available resource requirements of the batch architecture. The capacity manager should notify the owner when the job will be executed and provide this individual with the job results following completion of the run. Reports are printed, archived, or published as requested by the user and as executed by the script. Archived reports should be stored in a database external to the batch architecture.

Schedule Changing

Changes in the schedule may be planned or unplanned. Unplanned changes result from processing errors or system warnings. Planned changes are made in response to system optimization analysis. Planned changes to scheduled batch runs are tracked using the change management process because changes to runs that are in production require the submission of an RFC. This ensures that only authorized changes are permitted and provides a level of control and accountability to the daily operations of the job scheduling process (for more information, refer to the change management SMF guide). Figure 8 displays the batch schedule change process.

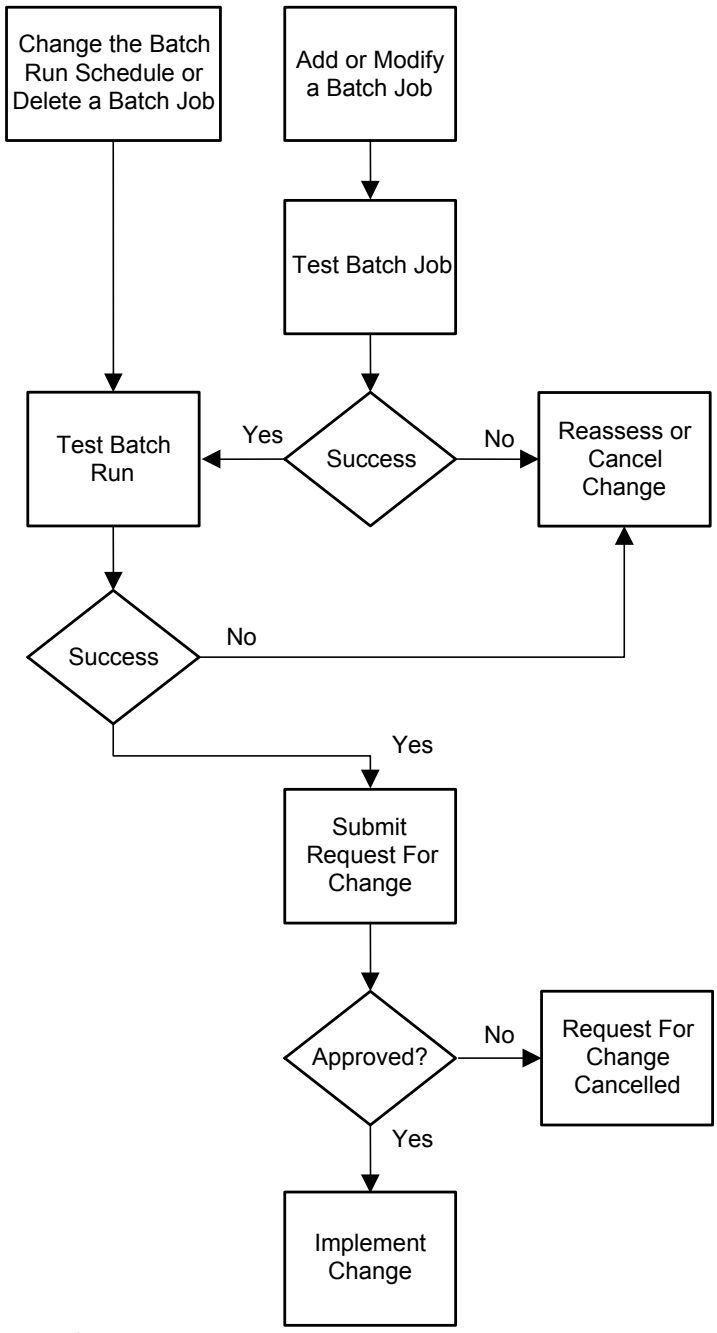


Figure 8

Batch schedule change process

Changes to scheduled batch runs typically take one of two forms:

- Batch run schedule changes:
  - Change the batch run start/stop time
  - Change to the batch jobs assigned to the run
  - Change a batch job priority
  - Change a batch job start/stop time
- Batch job changes:
  - Add a job
  - Delete a job
  - Modify a job

Any changes made to batch jobs will affect the processing performance of the batch run and must be tested and approved before being implemented in the batch environment. These changes must be accompanied by an approved RFC. Batch jobs that are added or modified should be tested in a test environment and, as with as-needed requests, a profile of the new job should be developed. Profile metrics collected include:

- Disk utilization
- Memory utilization
- Network utilization
- CPU utilization
- Time required to complete the job

The job profile can be extrapolated to assess the resource and time requirements if the job is altered. For example, if the profile is based on a job containing 1,000 records, the capacity manager can extrapolate the profile results to determine how long it would take to do the same job if it contained 3,000 records and what corresponding resource utilization would be required. The capacity manager can then determine if changes to the job inputs (that is, number of records) will adversely affect the completion of this job or other jobs. In other words, will the revised job prevent other jobs from having sufficient resources or the time required to complete execution? This is extremely useful if testing resources are limited.

If the job does not execute correctly, the change should be reassessed or cancelled. If it passes the test, the job should be assigned to a batch run. The entire run should then be tested and the impact of adding or modifying the job assessed. If batch jobs are deleted from a run or the batch schedule is adjusted in any way, the entire run should be executed and the impacts of the change assessed. In either case, if the batch run test is

unsuccessfully executed, the change should either be cancelled or reassessed. When batch runs are successfully tested, the capacity manager should submit an RFC to change management for approval to implement the change.

When testing batch runs, the capacity manager must remember to assess how system resources will be affected as they pertain to online users. It is worth mentioning again that online users should be afforded the highest priority when determining resource utilization requirements. Batch processing always takes a backseat to the utilization needs of the user. In addition to testing or if testing is not practical, the capacity manager can use workload forecasts based on historic online and batch metrics. Using the historic information and batch job profile information, the capacity manager should be able to extrapolate expected batch performance results. There is, of course, no substitute for actually testing the runs and monitoring the performance characteristics.

Depending on the organization's change process, temporary unplanned scheduling changes in response to errors should not need to be submitted as RFCs to change management. The capacity manager should record all temporary changes in the capacity manager's log for future review and reference. Temporary changes might include:

- Changing the allocation of resources to jobs
- Changing job priority
- Managing batch runs and jobs (starting and stopping)
- Changing the allocation of system resources

If possible, these changes should be avoided, because making changes without properly assessing the impact to the change can adversely affect the IT environment and possibly hamper online user performance. For example, the capacity manager should try to limit untested changes to situations where errors are preventing the execution of high priority batch jobs.

Unplanned permanent changes made in response to errors must be reported to change management. However, since changes in response to errors must be made immediately in order to complete batch jobs, change management should authorize the capacity manager to make necessary changes (as defined by change management) without first submitting RFCs. Every effort should be made to test the change prior to implementation to avoid adversely affecting the IT environment. The capacity manager should record all changes in the capacity manager's log and submit an RFC when time permits (as soon after the change as possible). This ensures that all changes to the environment are

recorded in the configuration management database (refer to the configuration management SMF guide).

System Backup

Transaction data and batch-processing log information should be backed up on a regular basis to ensure the availability and integrity of business data and other information and to provide a way to recover data in the event of a disaster. System backup activities should be focused on the ability to provide copies of critical data that have been destroyed and restore that information in an automated fashion.

As with all operational systems, the batch architecture should be backed up on a regular basis. The pertinent information that should be backed up and the suggested frequencies include:

- Batch and error logs (daily)
- Transaction data (daily)
- Logged information on the application servers (weekly)

In addition to the transition data, all information in the batch system architecture that would be required to restore the system following a system disaster should be periodically backed up. (For more information, refer to the service Continuity management SMF guide.)

Archiving

Data archiving is the process by which transaction data is written to an alternative storage facility and is then purged from the batch architecture databases. Data archiving is needed to minimize the long-term effects of a constantly growing database; as the production databases accumulate more and more information, there is a potential adverse impact to the performance of the batch architecture. Archiving is also required to meet legal requirements for record retention (for example, financial and tax reports).

The capacity manager needs to differentiate between transaction data that can be archived and data that should be deleted. The information that should be archived depends upon the business requirements of the organization (that is, financial data). Requirements should be documented in the batch log and in batch job OLAs. Users should also be made aware of what data has been archived and deleted, as access to this data is not available through the production environment.

When archiving information, it is important to understand how much information is being archived so that appropriate disk space can be allotted for the archived material and systems performance issues can be addressed. It is a good idea to have a

table, such as the one shown in Table 1, with information about the storage requirements for each report that is archived.

Table 1. Archiving Storage Information

Archived Object	# Of Documents Archived/Year	Bytes/Document	MB/Month	MB/Year
Batch Log	365	800	.02	.3
Financial Report	5,000	950	.4	4.8
Invoice Report	10,000	3,000	2.5	30.0
Inventory Report	30,000	2,500	6.3	75.0
Total	**45,365**	**7,250**	**9.4**	**110.1**

This table allows the capacity manager to provide storage requirement information to storage management, which is responsible for planning future storage capacity needs (refer to the storage management SMF guide).

Auditing

System audits should be conducted on a periodic basis. The goal of the audit should be to ensure that the batch architecture is operating as intended. The capacity manager should examine the metric information that is being collected during each batch run and assess if more or less information is required to perform daily tasks. This individual should also analyze the CDB, associated logs (batch and error logs), and all transaction data to ensure that historic data is being archived and deleted after the specified period (the time period varies depending on the requirements specified in the OLAs). Note that some information, such as financial reports, must be kept for a much longer period than less critical information.

Capacity Manager Log Entry

The capacity manager should maintain a capacity manager log to manually record all actions that are performed on a daily basis. Tasks that should be annotated in this log include, but are by no means limited to:

- Changes to batch runs (what changes were made and why)
- The receipt of an as-needed batch jobs request
- The results of as-needed request reviews and testing
- The execution of as-needed requests
- Starting or stopping batch runs

- Changes to batch jobs, including job scripts, schedule, priority, and so on
- The acknowledgment of errors and warnings
- The cause of errors and warnings and the steps taken to correct errors
- The escalation of errors and warnings
- The submission of RFCs
- The implementation of changes/backing out of changes/implementation results

When making entries to this log, also include the name of the individual performing the task.

Reporting

Batch job reporting requirements should be specified in the batch log and the batch script coded to automatically generate the appropriate reports. Some scheduling tools may have a report generating capability that can be configured to automatically generate reports for specified batch jobs or operational related reports. Generated reports are filed in the appropriate database and can be automatically printed if so required. Reoccurring operational reports that are generated include batch schedule reports, error reports, and performance reports. As-needed reports are also frequently generated as the result of a user request. These reports can contain any combination of data requested by the user.

Batch Schedule

The batch schedule report contains information about the batch runs scheduled for execution. These reports should be reviewed frequently, especially if batch runs are being altered, to ensure that the runs are properly scheduled for execution in order to optimize system resources. These reports typically include the following information:

- Batch jobs to be executed
- Start time and frequency
- Dependency on other jobs
- Contact and error recovery information

This report could also contain the general information about each job listed in the batch log.

Error Report

This report should be printed and reviewed following the recording of an error. The information in this report should assist

the capacity manager in determining the cause of the error. The information is read from the error log.

Performance Report

This report should contain information about the performance of the batch run execution. Specific information about each job should be reported including:

- CPU utilization
- Network utilization
- Disk utilization
- Memory utilization
- Job duration time

Information in this report should help the capacity manager to optimize system performance and identify performance degradation trends. Capacity management also uses this report to plan future capacity requirements for the batch architecture.

Documentation and Training

All policies and procedures should be clearly documented so that the capacity manager has a reference to refer to for daily operational guidance. Documentation should include information on:

- Procedures for starting and stopping batch runs
- Procedures for changing job priority
- Procedures for handling alerts and errors
- Procedures for handling common errors
- Procedures for analyzing the cause of errors
- Procedures for escalating errors
- Procedures for submitting RFCs
- Procedures for documenting tasks in the capacity manager's log
- Procedures for what should be monitored and when
- Procedures for handling as-needed requests including reviewing, testing, and running these requests

Without proper training, the capacity manager will not have the ability to conduct the activities discussed in this document. It is important that the capacity manager be properly trained so that processing errors can be responded to and corrected in a timely manner. If it is available, the capacity manager should attend vendor training on the scheduling tool utilized by the organization.

Roles and Responsibilities

Principal roles and their associated responsibilities for job scheduling have been defined according to industry best practice. Organizations might need to combine some roles, depending on organizational size, organizational structure, and the underlying service level agreements existing between the IT department and the business it serves.

Capacity Manager

The capacity manager is the owner of the job scheduling SMF. This individual is responsible for ensuring that all batch jobs are successfully completed without impacting user resource availability. In small organizations, the capacity manager may have one or two assistants who perform the daily job scheduling activities. However, in large organization that process thousands of batch reports on a daily basis, this individual is likely to be managing a large team of personnel (this document refers generically to the capacity manager) in the daily performance of these activities:

- Establish batch job standards (batch skeleton)
- Establish the batch schedule
- Change the scheduling calendar (that is, no runs during holidays)
- Make changes to the batch schedule and test the new schedule
- Monitor the batch process:
  - Success and failure of batch jobs
  - Resource availability and performance
- Capture metrics and create performance reports
- Tune batch-processing performance and submit requests for change (RFC) as necessary
- Troubleshoot and correct or escalate job errors
- Assist incident/problem management in the correction of errors that are escalated
- Review, test, and run as-needed requests
- Submit RFCs for schedule changes and the addition or removal of jobs from batch runs
- Ensure high availability (24/7)

Relationship to Other Processes

Job scheduling is a service management function (SMF) in the operating quadrant of Microsoft® Operations Framework (MOF). The main activities of job scheduling include monitoring, analysis, tuning, and implementation. As such, job scheduling is related to a number of SMFs in the MOF architecture, which are described below.

Every process within the Microsoft Operations Framework benefits from some aspect of monitoring and control, as these functions are inherent to ongoing process improvement. This is especially true in the operating quadrant of the MOF process model. Job scheduling is considered one of the foundational SMFs of the MOF operating quadrant. The graphic below depicts the relationship between job scheduling and other MOF SMFs.

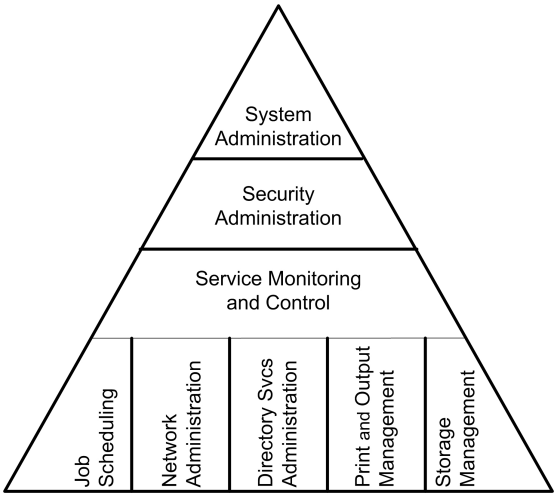


Figure 9

Relationship to other SMFs in the operating quadrant

System Administration

System administration deals with the administration model used by an organization. Some organizations prefer a model where all IT functions are performed at a single site with a team of IT professionals collocated at that site. Other organizations prefer a distributed branch office model where both technologies and support staff are geographically distributed. System administration examines the trade-offs of each model. Each type of system administration model has unique batch process administration requirements.

Security Administration

Security administration is an IT process concerned with implementing and managing security controls that enforce

corporate security policies, thereby ensuring data and system security within the production IT environment. Job scheduling and security administration have a relationship because some forms of corporate output (reports, for example) that are created via the job scheduling process must remain secure. System administrators and the security administrator need to work together to ensure that the corporate data security policies are followed closely.

Service Monitoring and Control

Service monitoring allows operations to observe the "health" of a service in real time.

Note Real-time as defined here is essentially "just in time." For example, timing may be dictated by the context of the source of data and the need for management information or corrective action. (Is the service up, down, or marginal?)

However, although important, this type of observation is purely reactive in nature, and in today's competitive marketplace, it is not enough for service monitoring to allow operations to merely react to service problems in the IT environment. In addition, it is extremely important that service monitoring provide operations with the ability to observe service behavior in such a way as to allow it to be proactive by, for example, finding problems and potential service outages before they occur. Providing operations with both a sound reactive and proactive monitoring capability can be defined as a service monitoring best practice.

But knowing the current health of a service or determining that a service outage may occur is not worth much unless operations has the ability to do something about it, or at the very least notify the appropriate group that a specific type of reactive or proactive action needs to occur. This is what is meant by the term control. When combined and implemented properly, service monitoring and control gives operations the critical capability it needs to ensure that service levels are always in a state of compliance. Without proper service monitoring and control, service level agreements are of little use.

Service monitoring and control provides the foundation for determining service performance levels. Optimization of service performance implies monitoring an application's end-to-end response times. In a well-run IT shop, performance levels are forecast and the monitoring system sets threshold alarms to trigger alerts "before" the customer of the service is aware of an issue. Monitoring is an important activity of job scheduling as batch processing must be evaluated and optimized to ensure that

online users are not affected by system degradation caused by the processing of batch runs.

Network Administration

Network administration is an IT process concerned with managing all production networks under change management and configuration management control. Network administration must work with job scheduling to coordinate efforts to optimize network resources and ensure strict adherence to network administration service level objectives.

Network administration should provide network usage, bandwidth, and trend analysis to enable predictive capacity planning information accessible from a central facility that may populate the capacity database (CDB). Job scheduling should provide network administration with network capacity requirements to ensure sufficient bandwidth is available to process batch runs.

Storage Management

Storage management covers the day-to-day activities required to effectively operate and maintain storage management in an IT environment. Batch processing requires significant disk availability for the storage of transaction data. The capacity manager should coordinate expected disk storage requirements with storage management.

Capacity Management

Capacity management is the process of planning, sizing, and controlling service solution capacity to satisfy user demand. Batch-processing system requirements and all operational metric information should be provided to capacity management so that future capacity requirements can be accurately planned.

Change Management

Change management deals with the coordination of any change that occurs within the organization, including software upgrades, entire system overhauls, organizational or personnel changes, business changes, and so on. Daily operational changes to the batch architecture may surface as attempts are made to correct system errors and tune the batch architecture to optimize system performance. Any change to the IT environment is channeled through change management as a request for change (RFC).

Since batch processing typically utilizes significant system resources, the timing of changes must be coordinated with availability management to establish the effect of executing batch

runs on the IT environment. Proper coordination is important to ensure that sufficient resources are available to execute batch runs and meet the service level requirement of online users.

Configuration Management

Configuration management is the process of tracking and accounting for hardware, software, documentation, and all other components of the IT environment. Configuration management is responsible for maintaining the configuration management database (CMDB), which is used to track all IT-related components. All changes that are implemented in the batch architecture environment must be recorded in the CMDB.

Service Level Management

Agreements between the provider and consumer of IT services exist within many organizations, yet in some they are unofficial, taken for granted, or unclear to one or both parties involved. Review and preparation of service level agreements (SLAs) and operational level agreements (OLAs) is a primary activity of service level management. Batch-processing guidelines are defined by the OLAs that are prepared by service level management.

Service Desk

The service desk is responsible for providing efficient, timely, and high-quality user support and incident management. All problems encountered with batch processing should be reported directly to the service desk and all affected owners and users.

Contributors

Many of the practices that this document describes are based on years of IT implementation experience by Accenture, Avanade, Microsoft Consulting Services, Fox IT, Hewlett-Packard Company, Lucent Technologies/NetworkCare Professional Services, and Unisys Corporation.

Microsoft gratefully acknowledges the generous assistance of these organizations in providing material for this document.

Program Management Team

Jeff Yuhas, Microsoft Corporation

William Bagley, Microsoft Corporation

Lead Writer

Curt Humes, Accenture

Contributing Writer

Vicky Howells, Fox IT

Editor

Patricia Rytkonen, Volt Technical Services