Realtime
publishers
"Leading the Conversation"

The Shortcut Guide™ To

IT Workload Automation and Job Scheduling

sponsored by

Advanced Systems Concepts

Mark Scott

Introduction to Realtime Publishers

by Don Jones, Series Editor

For several years now, Realtime has produced dozens and dozens of high-quality books that just happen to be delivered in electronic format—at no cost to you, the reader. We've made this unique publishing model work through the generous support and cooperation of our sponsors, who agree to bear each book's production expenses for the benefit of our readers.

Although we've always offered our publications to you for free, don't think for a moment that quality is anything less than our top priority. My job is to make sure that our books are as good as—and in most cases better than—any printed book that would cost you $40 or more. Our electronic publishing model offers several advantages over printed books: You receive chapters literally as fast as our authors produce them (hence the "realtime" aspect of our model), and we can update chapters to reflect the latest changes in technology.

I want to point out that our books are by no means paid advertisements or white papers. We're an independent publishing company, and an important aspect of my job is to make sure that our authors are free to voice their expertise and opinions without reservation or restriction. We maintain complete editorial control of our publications, and I'm proud that we've produced so many quality books over the past years.

I want to extend an invitation to visit us at http://nexus.realtimepublishers.com, especially if you've received this publication from a friend or colleague. We have a wide variety of additional books on a range of topics, and you're sure to find something that's of interest to you—and it won't cost you a thing. We hope you'll continue to come to Realtime for your educational needs far into the future.

Until then, enjoy.

Don Jones

Copyright Statement

THE MATERIALS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. The Materials are subject to change without notice and do not represent a commitment on the part of Realtime Publishers, Inc or its web site sponsors. In no event shall Realtime Publishers, Inc. or its web site sponsors be held liable for technical or editorial errors or omissions contained in the Materials, including without limitation, for any direct, indirect, incidental, special, exemplary or consequential damages whatsoever resulting from the use of any information contained in the Materials.

The Materials may contain trademarks, services marks and logos that are the property of third parties. You are not permitted to use these trademarks, services marks or logos without prior written consent of such third parties.

Realtime Publishers and the Realtime Publishers logo are registered in the US Patent & Trademark Office. All other product or service names are the property of their respective owners.

If you have any questions about these terms, or if you would like information about licensing materials from Realtime Publishers, please contact us via e-mail at info@realtimepublishers.com.

[**Editor's Note:** This eBook was downloaded from Realtime Nexus—The Digital Library. All leading technology guides from Realtime Publishers can be found at http://nexus.realtimepublishers.com.]

Chapter 1: Challenges of IT Workload Automation and Job Scheduling

The information technology (IT) landscape for many organizations has grown into a complex web of servers, networks, locations, products, and processes. Simple, siloed applications now need to interact with systems scattered across the country and across the globe. The process of helping all these systems, processes, and procedures work in an efficient, orderly, and secure manner has become an enormous task.

The purpose of this guide is to identify the issues surrounding the design, control, and management of jobs within an organization. It provides insight into challenges and recommends best practices for establishing an efficient, maintainable system for scheduling, monitoring, and adjusting jobs within the enterprise.

The guide is organized into four chapters. Each chapter will explore different facets of the complex topic that is IT workload automation:

- Chapter 1 examines the various challenges of job scheduling. By identifying the current challenges and looking toward those that loom in the future, an organization can take steps to assume control of job automation and remain its pilot rather than its victim. The chapter addresses the complexity of planning and overseeing the execution of jobs in heterogeneous environments. It looks at working with diverse platforms and the trials of disparate data types, connectivity issues, languages, and control paradigms. It considers issues with remote locations, including time zones, synchronization of data, and delivery options. It also reviews the factors of managing such interconnections of systems.
- Chapter 2 addresses the specific complexity involved in job scheduling itself. The chapter begins with a discussion of the process of defining a job and its requirements. It explores job scheduling based on time, running jobs at a specific point of the day and/or within the constraints of a specified window of time. It looks at scheduling that is event driven, often employing resources dynamically to meet the current needs. It delves into job execution, using the right resources at the right time, and the intricacies of error detection and recovery. It speaks to monitoring and setting alerts to keep the operational staff well informed and ready to intervene as required.

- Chapter 3 dives into the specific details of job design and configuration. It looks at job modeling to encapsulate business processes and design a system that is compatible with the business need for the job. It looks at target platforms and discusses the consideration of data formats and execution languages. It considers the choices in communications solutions between systems, from file transfers to queues to Web services and other networked communications. It reviews the incorporation of conditional logic and security in job flows. It also addresses enterprise service architectures, including enterprise services buses, single points of deployment, and the interaction of existing business processes with the requirements for IT workload automation.
- Chapter 4 tackles best practices for IT workload automation and job scheduling throughout the enterprise. This includes establishing effective service level agreements (SLAs). It discusses job deployment and execution, and suggests best practices for monitoring and auditing job execution. It explores resource management and improving efficiency to help build "greener" computer operations and data centers. It also explores systems architecture issues from scalability to portability to resiliency.

Dealing with Complexity

Computer systems have grown over time from a single mainframe to a complex ecosystem of mainframes; mini-computers; and UNIX, Linux, and Microsoft Windows servers. They host a variety of applications. These independent applications are often purchased and installed over a span of years. Complexity is born of the variety of applications, communications mechanisms, and host systems that need to be coordinated in order to facilitate information flow throughout the organization.

The formidable task facing IT is to orchestrate the series of jobs that allow these systems to share information. The wide variety of mechanisms used to coalesce shared information and move it between systems makes managing this interaction daunting. The challenges can be seen in the difficulty in scheduling, synchronizing, monitoring, and troubleshooting the flow of information.

Scheduling Automation Tasks

The concept of “batch processing” is ancient, extending back as far as the first time someone decided to break a job into discrete steps, for instance making all the arrowheads needed for arrows before making the shafts or preparing the fletching. There is a greater efficiency in completing one type of task while all the elements to complete that task are available and at hand.

Many computer systems use batch processes to organize information. Accounting systems perform day-end reconciliations, data warehouses import the daily data loads, and inventory systems do final tallies on products at hand. These tasks are frequently performed during times when humans interact less with the server systems because it is easier to deal with the data when it is not being changed and to keep the capacity of the system available to human operators when they require it. Many server systems perform a series of tasks at the end of the business day. These jobs include backup, data reconciliations, imports and exports, and similar tasks.

Although scheduling each job may be simple, the sheer number of jobs to be scheduled may make the process complex. The tasks typically need to be executed in a specific order. The reconciliation needs to be performed before the backup (or after the backup—depending on the system and circumstances in question). Data may need to be imported before the reconciliation can take place. Tracking the order is crucial, and as the number of interrelated tasks grows, tracking becomes more difficult.

The time in which the tasks must be performed soon becomes an issue as well. Many automation tasks are performed during hours when the systems are not being used by humans. This helps the users by preserving system capacity for their use. It also helps the processes by working on data while it is less volatile. Most shops have a nightly processing window during which these tasks are performed.

The nightly processing window presents its own challenge. First, these windows have a habit of shrinking. As operations grow to cover more area, often in different time zones, the period during which the server is available to perform these tasks shrinks. The tasks themselves become longer as there is more data to process. As the need to share data with other systems increases, there are more jobs to process. The ability to change the schedule to align with the changing circumstances can help organizations gain the maximum return for their hardware investments.

The methods used for scheduling also add to the complexity of the problem. Many applications have their own job schedulers for performing internal tasks. For instance, Oracle can execute packages on a scheduled basis to perform tasks within the database. Similar capability exists to perform tasks on other Oracle applications, such as Oracle Financials and PeopleSoft. This setup works well with Oracle-oriented applications, and leaves scheduling within the family of related applications. It becomes more difficult if there are other applications involved, such as mainframe production systems or Microsoft applications that must also share the data. If an enterprise owns a variety of applications from different vendors, using the applications’ internal scheduling engines becomes impractical for an enterprise solution to job scheduling.

Most OSs provide the capability to schedule. Mainframe systems have long provided dedicated solutions for controlling the scheduling and automation of jobs. UNIX derivative systems have relied on basic systems such as cron to remember what time it is and launch jobs at scheduled intervals. More sophisticated systems have developed to help provide better control of these scheduling activities. As the number and diversity of servers grows, the number of diverse systems needed to schedule jobs tends to grow as well.

In a heterogeneous environment, it is preferable to find a system that can schedule jobs on a wide variety of systems—from mainframes to mini-computers to UNIX, Linux, and Microsoft Windows servers—from a single interface. An interface that clearly displays the tasks each server will perform and the order in which they will be performed helps operators achieve optimal results. The ability to schedule many different types of servers with a common interface helps contain training costs and may help reduce the staff of systems operators. There is less training involved and less need to maintain esoteric server-specific or application-specific skills (which often are more costly to acquire).

Synchronizing Automation Tasks

Closely related to scheduling a series of tasks on a single server is synchronizing tasks across servers. As previously noted, the individual tasks often need to be executed in a specific order. Tallying the orders for the day to update inventory might need to wait until the shipping system reports what actually went out the door and was shipped that day. The system that provides order status for the customers cannot be processed until the order tally is provided. The accounts receivable system cannot generate invoices for the latest orders shipped until it knows what orders shipped.

If these systems are all integrated, the server can be programmed to execute each task in order to accomplish the desired goal. If the data is generated on independent systems, synchronizing the tasks becomes more complex.

The simplest solutions to this challenge are schedule based. Using the previous example, if the shipping system data can be made available by 9:00 PM, the shipping and inventory reconciliation can be completed from 9:00 to 9:30 PM. The data can be exported to the accounts receivable system by 10:00 PM, and accounts receivable can process the invoices. Many organizations who have used this type of scheduling soon understand the shortcomings. This type of scheduling invariably includes idle time to ensure that the previous process can be completed before the next one starts. As the time allocated to processing shrinks and the time to process increases, this “slack” time grows scarce. This can affect SLAs and drive the need for additional hardware resources.

If the schedule is disrupted—for instance, the inventory and shipping reconciliation overruns the allotted time and cannot deliver the export on schedule, the accounts receivable system will be unable to process the invoices. The schedule can be disrupted for a wide variety of reasons

- Changes in priorities on one server can change the times at which it processes jobs. This change can affect the other servers that depend on a particular job in that server's queue.
- Server resources (CPU cycles, memory, disk I/O, network bandwidth) can be shared between multiple jobs. This can make the time to complete any given job unpredictable.
- People can fail to complete tasks and thus prevent the scheduled job from starting on time.
- Simple failures of software to run or system disruptions (for example, network links) can prevent the job from executing.

A solution to this challenge of disruption is event-driven processing. A job can be triggered when the previous job is completed. It is not uncommon for an extraction, transformation, and loading (ETL) package to be triggered when an exported file arrives in a specified location. The arrival of the file itself is the scheduling mechanism.

Event-driven scheduling has its own set of difficulties. There is no real guarantee when an event will occur, so there is no way to guarantee that resources will be available to execute the job when it is available. The best way to get the most work from a server is to supply it with a consistent workload running 24 × 7. If the workload runs in peaks, there is a tendency to size the server for the peak utilization. During lull times, this can leave a great deal of unused computing power sitting idle.

There are also times that input from multiple sources is required. A job may take input from several systems. For example, a data warehouse may receive input from several ERP systems. If the ETL package requires the input from all these systems before it can run its load correctly, there must be a mechanism that tracks each data export and triggers the ETL system when it has all the components that it needs to complete its work.

An effective technological solution is a system that can coordinate the execution of the individual jobs on each server. If a series of jobs can be linked, with the completion of one task triggering the commencement of the next task, orchestration of the chain of jobs becomes much easier.

A good solution should span all the independent systems within the enterprise. It should help the operators not only understand the flow of work throughout all systems but also the jobs scheduled on each individual server within the environment. This will help operators balance the load and keep servers running nearer their full capacity.

Monitoring Automation Tasks

The key to controlling automation tasks is monitoring. The operators need to know what tasks are being performed successfully, and which tasks are failing. The challenge is derived by the wide variety of ways in which the individual tasks can be monitored.

There is no single reporting system that organizes the information into a single form that is easy to comprehend. There is no common thread that ties the job on the order entry system to the process on the ERP system to the business integration on the message queues to the reporting system. There is no simple way for an operator to see the chain of related jobs as a whole and see how each step affects the other steps. Auditing and reporting of automation jobs provides many benefits.

- Proof that the system is operating as designed
- Guidance for the staff in isolating and correcting problems
- Performance baselines that can be used for future planning

The challenge for corporate IT operations then becomes how to knit these disparate sources of information together into a comprehensive view of how jobs are executing across the entire organization. Many of these jobs are part of a chain of operations that work in sequence, so it becomes important not only to know whether a job ran successfully but when. An operator might need to track a specific chain of jobs across multiple applications, servers, and communications mechanisms to ensure all the steps in the chain executed successfully.

Monitoring is required when jobs must be audited for corporate policy or regulatory reasons. IT may be required to provide a detailed history of the data flow for each day in order to demonstrate that the job integration is operating as designed. These reports will need to incorporate data from each system on which the job steps executed and format that information into clear, concise reports.

A system that can track the execution of each job and provides that information in a consolidated format is invaluable to the IT operations staff. The system should track each scheduled job and provide a concise, timely report about their execution. It should show start and stop times, and make note of any errors. Systems that notify operators of job failures help operators respond to problems quickly and keep the systems operating on track.

Troubleshooting Automation Task Scheduling

It is inevitable that things go wrong. No matter how well engineered the system; something will occur that will create an error. When systems are comprised of multiple steps, it can be difficult to locate the source of the problem. Consider the following example: An ERP system creates a data export file. The file is picked up by the ETL system, which adds the data to the data warehouse. The Online Analytical Processing (OLAP) system processes the new data in the data warehouse, and the reporting system then uses the data in the OLAP system to create a daily report on shipping.

Imagine that the FTP server that moves the data from the ERP system to a file folder where it is picked up by the ETL system encounters an error. The ETL system looks for a file but does not find it. This may not cause the ETL system to generate an error. The OLAP system processes the data, but because the data does not change, it has no new data in the data warehouse. When the report runs, it shows that there were no shipments. Phone calls start as operators begin trying to determine what went wrong. They check the ERP system, ETL system, OLAP database, and reporting system and find no errors.

A system that can list each of these steps in a chain of events can help troubleshoot the problem. If the ETL system, which is triggered by the arrival of a file, never executes, the system would note that. Operators would be able to quickly note that the ETL job did not run. They would look for the file and find that it never moved. The fault could be quickly found and corrected, in spite of the fact that the major systems involved in the process did not register any specific errors.

A system that looks at a series of jobs as a related whole and tracks each of the individual steps helps operations track activity through the system and locate errors, even when they hide and do not appear as errors. Conversely, they can explain a cascade of errors.

A reconciliation process exports a file that is moved by a Web service to another system. This file is used to update an ERP server and create another file that is sent to a vendor. If the initial reconciliation process fails, it might cause failures in the Web service, ERP system, and the system that moves the file to the awaiting vendor. These steps are organized into a single process, so operators can quickly track back through the cascade of errors and find the original cause.

The ability to visualize the related steps in a process and track through each step is invaluable for troubleshooting. It can help locate errors more quickly and correct them.

Building Robust Automation Systems

The conditions in a vibrant enterprise are in constant flux. If a system can respond to changes in the environment and job conditions, it can make the most efficient use of system resources. To accomplish this, the job sequences need some form of conditional logic.

For example, a job may need to be run once the accounts have been manually reviewed and reconciled. This may not be a regular event, but it may involve touching multiple servers and applications when it occurs. Many IT organizations have been frustrated by a series of jobs that have to be executed manually because they do not occur on a regularly scheduled basis. A system that can be triggered and executed on demand can be very helpful when executing a series of jobs on multiple servers.

Sometimes when job errors occur, a compensating job can be run to correct the error and complete the original series of tasks. If a system can detect the error and then run the compensating job, the tasks can be executed on schedule without additional operator intervention or delay. By automating the process of error detection and compensating automatically, automation tasks can be kept on schedule.

A system that can respond to the availability of resources can make best use of those resources. Systems that run compensating job sequences when they encounter errors can be self healing and keep information flowing. Systems that operate in response to available data will need to execute only when there is work to be done.

Playing Well on Different Platforms

The challenge of scheduling enterprise IT workloads stems from the diversity of platforms on which the jobs must run. Some systems store jobs in databases on the mainframe. Data is stored by the ERP system in Oracle and SQL Server. Messaging systems house data in Lotus Notes and Microsoft Exchange. Teradata and Cognos data warehouses provide reports through Microstrategy and Crystal Reports. For all this diversity, IT must find the way to get these diverse data sources and repositories of corporate information to share and integrate their wealth of corporate intelligence.

This diversity requires operators to get computers with very different operating and communications paradigms to interact with one another. It increases the complexity of monitoring the flow of interrelated jobs. It tests the capacity of connectivity.

Scheduling on Distinct Platforms

Although all computers evolved from a similar source, the mediums of systems architectures, computer applications, and OSs have allowed for a great deal of variation and creativity. Sometimes to try to improve things, sometimes to avoid patents, they tend to do the same things but do them in different ways. The divergence of common tasks makes them difficult to manage as a group.

Mainframes were originally designed to operate in batch patterns, taking large volumes of data and performing operations on them. Many mainframe jobs still perform these massive batch operations. Systems evolved to work more closely with users and to interact with people more directly. From green screen CICS interfaces to graphical user interfaces (GUIs), the computer worked more closely with people.

As the paradigm of the computer changed, and the interfaces changed, so did the manner in which they were controlled. This has led to a wide variation in the manner in which jobs can be scheduled and how they operate on a computer. The ability to create a job schedule on an IBM mainframe may not provide any of the skills used to set up jobs to run on an AS400 let alone a Microsoft Windows server. The skills to operate on each platform tend to be unique. Thus, people tend to specialize in one platform or another.

This specialization presents a challenge when scheduling jobs across the enterprise. The mainframe operators can keep their jobs in line. The Linux systems administrators can control the scheduling of jobs on their servers. When enterprise jobs must be shared between platforms, the task becomes much more difficult.

The choreography of job steps on different systems is inherently difficult. It becomes more difficult if the people managing the project have to deal with the intricacies of scheduling those jobs on a wide range of different platforms. The Oracle DBA who knows exactly how to import data into the warehouse may have no idea how to get the COBOL data extract program running on the Hitachi mainframe.

An automation solution that can abstract the execution of the individual packages into a common interface allows a single operator to schedule and adjust the job steps across all the related systems required. The operator need not be an expert on each system on which the job steps execute. If control of those job steps can be abstracted and handled by agents on the individual platform, the operator need only be concerned with scheduling the tasks on each server so that the enterprise as a whole is properly served. This simplifies the process and allows the scheduler to concentrate on the formidable task of getting all the jobs executed without getting lost in the minutiae of how to schedule each job on each individual system.

Monitoring on Different Platforms

Different tasks on different platforms have different monitoring mechanisms. Most applications will have some type of log that chronicles their activity. Database-centric applications, such as ERP systems, tend to keep these logs in database tables. Other systems will write a text file that tracks key events. Microsoft Windows–oriented software often uses the Windows Server event logs to store application information. Some systems track information only if applications settings require them to do so. The thought behind this is that if things are running smoothly, it can be a waste of resources to log successes; thus, they log only errors.

Many automation tasks are written with command-line scripts and shells. These allow simple execution of commands from simple text files. Most applications provide a mechanism that allows an external script command, executed from the OS command shell, to execute jobs within the application. Most OS functions can be executed from these scripts. They form the basis for much of the system automation in use today. There are many scripting languages—such as JavaScript, Perl, Python, and VBScript; with the wide variety of available means to script jobs, it becomes a complex task to be able to read, configure, and troubleshoot them all.

Command-line scripts, however, do not have a specific mechanism for reporting how the script operated—whether it succeeded or failed. As they are very flexible, these reporting mechanisms can be added to the scripts, but there is no uniform standard for how or where they should report. Many such scripts are written with no reporting or monitoring at all. They either execute or they do not. They may or may not leave behind any evidence of their execution, depending on the whim of their programmer.

Some of the automation required is simply moving files from one location to another. This may be done through services such as file transfer protocol, or message queues, such as Microsoft Message Queue or IBM’s MQ Series. These services will each have their own tracking and monitoring services.

The automation system can provide a unified mechanism for reporting the individual steps in the jobs. As the scheduler executes each job, it can track the progress of that job. The system stores the information in a common data source, shared by all the jobs regardless of the system on which they run. This eliminates the need to find all the various means by which each system stores its monitoring and tracking information.

The common data source for IT automation tracking provides operators with a wealth of information. They can use the information to measure the time spent on executing the jobs in the schedule. They can track when servers are struggling to complete the workload that has been assigned to them. This information may be used to re-balance workloads to take full advantage of the capacity found across systems within the enterprise.

Connectivity and Security on Different Platforms

As computer platforms have developed, so have the means for connecting to those platforms. Consider all the means of connecting to different servers:

- Usernames and passwords
- Digest credentials
- Kerberos tickets
- Security certificates

Each step in a process may require its own credentials. A job may need to execute on the mainframe, requiring a username and password. That job might produce a file that is processed by a Java application that uses LDAP authentication to access the file system and store information. That data may then be accessed by a Windows service that uses Kerberos authentication. Operators who learn to assemble these packages across servers must learn to deal with this variety of security mechanisms. The issue is further exacerbated because good security practices advocate that credentials be changed on a regular basis. When credentials expire, jobs fail.

Centralization of security credentials can help ease this burden. A secure, centralized store of credentials helps operators keep all the jobs executing as designed. A system that alerts operators to job failures will help the operator identify the issue and minimize disruption of the job automation schedule.

Another aspect of security is granting rights to those who need them. Operations personnel often oversee the processing of data that they are not cleared to see themselves. If they operate the jobs with their own security credentials (an all-too-common occurrence), such personnel will need to be granted permission to use the data. A system that allows a job to be executed in a privileged security context by an operator who has security only to execute the job can help secure this data. Most applications provide operator roles that allow people to execute jobs without directly accessing the data. Extending the concept across jobs operating on different platforms in different security contexts can help extend that security mechanism throughout the enterprise.

A more secure technology will allow the operator to execute scheduled jobs on the system platforms without granting the operator permission to access the same data or applications that the job can access. The operator can perform the tasks of running the automation jobs without opening security holes in firewalls, routers, perimeter servers, databases, or applications.

Operating in Diverse Locations

As organizations grow, they expand into new territories. They open new offices in new locations. They might buy or merge with other businesses. This growth and expansion puts increased demands on IT job automation. There are remote servers with information that needs to be consolidated into a coherent base. New divisions and businesses need to cooperate and work together. The expansion of the enterprise into multiple sites will add new dimensions to the complexity of job automation:

- Locations
- Time Zones
- Culture

Working in Diverse Locations

Servers work best when they are located near the people who use them. Although broadband links can help systems work globally, local area networks (LANs) still work much faster, more economically, and more reliably than wide area links. Because of this, most enterprises distribute their server infrastructure to support the locations where they do business.

Systems operators who oversee the execution and interaction of automation tasks must be able to work on servers scattered throughout the organization. Although most servers provide the means of remote access and control, each system and each platform will do so with a unique interface and a unique control system.

This becomes a magnification of the diverse platforms issue. Without a common interface, the system operators must learn not only to schedule jobs on each platform but also to create multiple interfaces to connect to those systems. They need terminal emulators to work with the mainframes, telnet to work with the UNIX and Linux servers, and remote desktop access to work with the Windows servers. The operator must work across all these diverse interfaces and attempt to coordinate their activities to get the steps to operate in order.

Systems operators need to schedule across many servers to coordinate their activities. A common interface that can provide a single point of control for all servers, regardless of their location, is invaluable. A technological solution that provides a single point of control for automation is very useful.

Conversely, the system should be able to run in a distributed manner. If communications are down between facilities, the automation jobs should be able to complete their work (provided the communication issues do not interfere with it directly). Furthermore, local resources often have better knowledge of when IT automation tasks should be scheduled. If the control can be executed from a common console but distributed throughout the enterprise to ensure redundancy and more efficient localized control, the best of all worlds can be realized.

Diverse locations add to the scheduling issues. Remote connections will use WAN links to connect. The WAN links typically have limited bandwidth, so scheduling jobs when utilization is lower can help improve the performance of the system throughout the organization. It can also help reduce the need to purchase additional bandwidth.

Some organizations do not remain connected all the time—for example, remote locations that need to connect only to report on an occasional basis. Organizations may use dial-up links or VPN connections that do not remain connected at all times. When connections are not always on, scheduling and error detection becomes more critical. If systems operators can flexibly schedule jobs and carefully monitor results, they have better, more reliable control of these types of systems.

Remote connections are not always reliable. There can be delays in connections or complete failures. As previously stated, systems that can monitor activity and adjust when connectivity problems are encountered can help mitigate disruption. They may be able to use conditional logic to correct the issue or use alternative means to accomplish the task. For example, if the primary link is down, a system can be programmed to use an alternative link, such as a backup network or dial-up. The alternative may be slower or more expensive that the primary link, but a more desirable alternative to having the job fail altogether. Automation systems that are aware of job status and able to execute optional means of completing jobs can be used to help overcome these types of faults.

Working in Different Time Zones

As the organization spreads, it will begin to open or acquire facilities in different time zones. In most companies, there is a time each day when the servers are not working directly with the corporate users. This is the typical window for batch processing and the completion of many of the IT automation tasks. If the office works from 9 AM to 5 PM, there are 16 hours during which the servers can chew away at the automation tasks uninterrupted.

If a company on the North American East Coast opens an operation on the West Coast, there is now a 3-hour difference in when the servers are likely to hit their “idle” time. Moreover, the servers on the East Coast will still likely interact with the servers on the West Coast, and the active period extends from 9 AM to 5 PM to 9 AM to 8 PM. The processing window shrinks. This trend continues as operations integrate in Europe, Asia, and Africa. International operations have servers and users operating around the clock.

IT automation becomes more complex when jobs scheduling encompasses multiple time zones. The times that servers have available resources for processing will not be consistent across the organization. Users in India may be using their servers while automation tasks are running in South America. A centralized interface that helps the operators understand when servers are available for use will help them schedule automation tasks while minimizing the impact to other operations within the organization.

Since the days end at different times, the availability of data becomes an issue. Many systems process a day’s activity and make the compiled results available to other systems. As different time zones will reach day’s end at different times, the availability of this data will have a direct affect on when the data can be collected and exported to other systems.

Time zone issues will tax the scheduling and data synchronization capabilities of the job scheduling system. Operators need to know when data is available in order to combine it with other data and complete any given sequence of jobs. They need the ability to easily adjust or modify the schedule to accommodate the availability of that data. They need to be able to schedule work across the enterprise to fit together the pieces of the schedule. An automation system that provides this type of flexibility helps operators maintain the flow of information throughout the organization.

Working with Diverse Cultures

It can be difficult enough to get the messaging staff to talk to the DBAs. When the working groups do not speak the same language (natively), the challenge increases. There are cultural and linguist barriers that need to be addressed. Technologies that can abstract the scheduling into a localized interface and provide a common platform for scheduling can help keep everyone on the same page.

The interface can help manage control of the system in a language native to the systems operator. It can help keep control in a single, uniform format that can be learned quickly and help communicate the needs of the organization to everyone concerned.

Helping make job automation a utility function can also help overcome internal corporate cultural issues. It is not uncommon for the mainframe staff to be unenthusiastic about helping the guys on the UNIX servers running the data warehouse to get changes to their data feeds. The people supporting the ERP system may not be as cooperative in helping the Customer Service Relationship report team as they might. By making these tasks utilities that do not necessarily require the active support of the individual application support teams, some of the intra-departmental cultural clashes can be mitigated.

Managing Job Automation

To manage job automation, an organization needs a cohesive view of the entire enterprise. They need to know what work is happening on which servers. They need to know where they have spare capacity and how to use that capacity to meet the needs of the organization.

Balancing the Workload

Many systems are designed to perform the same tasks on the same servers, day after day, week after week. But workloads shift and grow. Servers take on new responsibilities or are consolidated into fewer servers with each server doing more. Applications upgrade and migrate, changing the nature and the timing of the workloads.

A job scheduling system needs to be flexible and able to change with the changing conditions within the organization’s IT infrastructure. Part of that flexibility comes in determining where jobs are executed. There is a tendency to put a job on a server and leave it there (if it is not broken, do not fix it). As additional servers are added to help with the workload, the original server remains scheduled with the majority of the work. Small pieces are parceled off to the new servers, but the total workload is never objectively examined nor judiciously balanced.

The difficulty in configuring and scheduling jobs is contributory to this. It can be difficult to get jobs up and running on a server. Once a job is delegated to a server, that server becomes its home (for example, the “FTP” server, the “data extraction” server, and so on). The risk of moving a job and having it fail in its new location and the overhead of changing the specifics of the way the job is modified all contribute to this inertia. An overworked IT staff might well begrudge the time it takes to shift jobs around and make certain they still work correctly as well as troubleshoot the issues when they do not.

A system that helps delegate jobs from one server to another can help overcome this. If it becomes simpler to change where a job executes and keep that execution integrated with the flow of the interrelated jobs within the system, operations becomes much freer to balance the workload across the available servers.

Another issue is that the size of the jobs themselves does not remain static. A database backup takes 15 minutes to complete. 18 months later (and 1.5TB more data) and the same job takes 90 minutes. The backup consumes most of the disk I/O bandwidth and RAM in the server. The other jobs scheduled on the server begin to slow with fewer available resources. Processes do not complete and begin to lock one another, creating even more delays.

Monitoring the time each job takes to complete can help provide a baseline. That baseline can be used to project the demands on the server and adjust the schedule accordingly. It can be used to project when jobs should be moved from one server to another, or when more server capacity is required.

The converse is also true. Sometimes jobs get smaller. They take less time to complete. This can make additional resources available on that server—resources that can be effectively used to complete other workloads. A solution that tracks the performance of jobs on servers throughout the enterprise can help identify which servers are running at or over capacity and which servers are available to help manage the workload. If the solution can dynamically monitor the workload and adjust on a continuous basis, the workload can be balanced throughout the enterprise. This can help get more work done with fewer servers and less wear and tear on the operations staff that needs to keep everything operating on an even keel.

Regulatory Compliance

The rules governing IT operations have tightened. Many organizations find they must carefully track the changes made within their IT processes. The Sarbanes-Oxley Act (SOX), Euro-SOX, Payment Card Industry Data Security Standard (PCI DSS), Basel II, Health Insurance Portability and Accountability Act (HIPAA), and other similar regulations for the handling of data and changes to corporate IT systems have made flexible and auditable systems mandatory. Corporate policies often require that processes be carefully monitored and change processes be duly governed and recorded to ensure that systems operate as approved and designed. All this adds up to a growing need to monitor and validate the execution of internal IT processes.

Automation systems run across platforms and use a wide variety of tools. In spite of the varied processes used to execute the results, these jobs form systems that disperse information throughout the organization. They can be difficult to govern and track.

An automation system that can show the individual steps in a clear manner can help control these systems. The reports can validate that the system is operating as designed and data is moving according to corporate policy or regulatory guidelines. It can be used to execute change control of the process and satisfy the organization’s need to govern the process.

Putting Together the Pieces of an Effective Job Scheduling Initiative

The government of most IT functions can be seen as a convergence of policy, process, people, and products. Policies help govern the direction of the function. Processes perform measurable actions that accomplish the stated purpose of the policy. People craft both the policy and processes to ensure that goals are met. And products provide the mechanisms through which the processes are executed.

A good set of policies helps everyone in the organization understand the reason that the automation is required and how it will be executed. Policy is the primary tool for governing the operation of the automation system. It also provides the chief tool for helping independent groups work together. With appropriate executive sponsorship, the policy can be crafted to gain the required access to the data and a verifiable goal that the investment spent in automating IT workloads is returning on the investment.

The processes should verifiably reflect the policy. They should demonstrate that operations were executed in a secure and reliable manner. They should report on their activity in a clear and timely manner.

People should be trained in the meaning and intention of the policy and the products used to execute the processes. They should be able to manage the system and have at their disposal the tools required to both execute the policy and demonstrate its success. Tools that help them remain nimble, effective, and remove the tedium of the mundane tasks will help improve morale and efficiency within the staff.

The products should be chosen carefully to meet the needs of the organization. First, they should be able to demonstrate their ability to execute the processes required by the organizational polices for job automation. They need to be able to execute on all the organization’s target platforms. The organization may also look ahead to future expansion or acquisition and find products that will support their future as well as their current needs.

The products should provide a clear ability to manage the process, including the ability to move workloads easily to the servers that have the greatest capacity to perform the tasks. They should abstract the security concerns of the organization and help maintain credentials in a trustworthy manner. They should help operational personnel quickly identify and correct problems and minimize downtime when automation tasks run into problems.

The products should also fully support monitoring operations. It should help with the planning of workload distribution. It should meet any regulatory requirements for auditing and historical tracking of operations and data integrity.

Summary

Automating workloads within an enterprise is a daunting challenge. It can require building systems that span applications, servers, server platforms, network barriers, physical locations, time zone, cultures, and departments. It requires the knitting together of people and technology to synergize a cohesive structure that can become more than the sum of its parts.

Because the data is typically scattered into individual silos of information and business process, automating tasks and integrating information can become an orphan child who is loved by none and dealt with reluctantly by many. IT can be unruly and tough to monitor, manage, change, or mature.

The proper approach to IT workload automation can change that. Systems that provide policies that demonstrate the value of connecting systems and automating tasks can help bring people on board. Addressing the operational and security concerns of the constituent data owners can ease their fears. Implementing processes that can execute at the best time on the hardware with the most bandwidth can optimize IT operations. Monitoring the system can ensure its health, it meets regulatory compliance, and it is easy to quickly troubleshoot errant processes. Providing the operational staff with the tools to do their job efficiently and reliably will allow them to provide their best value to the organization.

Choosing the right products to support the automation effort can have a dramatic impact on the process. Finding products that centralize operations across the full range of tasks, applications, servers, and OSs is critical. Abstracting tasks to simplify where and when they run helps make full use of the resources in the enterprise. Reporting that is clear, timely, and insightful helps the operation reach its full potential.

Download Additional eBooks from Realtime Nexus!