A Broadcom Company
technologies

White Paper

Key Requirements for a Job Scheduling and Workload Automation Solution

Traditional batch job scheduling is not enough.

Executive Summary

Challenge

While traditional batch job scheduling still has a place in many enterprises, the task is now far more complex than it used to be, due to the use of various computing platforms, multiple computing environments and business processes that run 24/7. Add to that the convoluted interdependencies that often exist between different processes and the frequent need to trigger jobs dynamically based on changing circumstances, and it is clear that the requirements of a traditional job scheduler have evolved into a broader workload automation capability to handle today's complex circumstances.

Opportunity

But what are the key capabilities that you need to address the modern day demands of an enterprise job scheduling or workload automation solution? We have collated the top requirements from the global enterprises that we have worked with over the past 15 years. These enterprises have successfully automated the core data center processes supporting their core business applications and services.

Benefits

The enterprise customers that we worked with were able to reduce processing time by 70% and eliminate 90% of IT process errors while reducing manual effort by 90%. You can do this too. This white paper explains how.

It Must Be Able to Schedule Jobs

Although the demands on job schedulers have increased over time, one fundamental truth remains: They must still be able to schedule jobs. At its most basic, job scheduling is the orderly, reliable sequencing of batch program execution. To handle these duties properly, a modern job scheduler must provide a host of features that allow distributed workloads to be scheduled efficiently and in a manner consistent with the operational goals of the business.

For instance, all businesses rely on one or more business calendars. Calendars govern such things as when payroll is run, when corporate reports are generated, and when fiscal months and quarters are closed. The key issues with calendars and other important scheduling features are flexibility and reliability. Can the scheduler adapt to the changing needs of the business and deliver predictable results time after time?

In today's cloud-first world, it is critical that any solution can extend its reach to all of your cloud-sourced solutions working well with both REST and SOAP integration frameworks. If you are only a casual user of SaaS-based cloud solutions today, that will not be the case in the future; so, it is essential to make certain you are protected with a deep level of control.

Additionally, programs and applications need to run on a variety of platforms, at specified times, in specified orders, and with varying levels of resource demands and prioritization. Schedulers need to be flexible enough to accommodate these varying technology, business, and resource demands. If the job scheduler is able to sequence processes and manage resource contentions, applications will execute faster and more predictably, and scheduling throughput will increase. Unfortunately, most scheduling products only focus on sequential activities and ignore other opportunities to balance and prioritize workload.

Built-in Managed File Transfers

Whether it is moving data between remote locations, external business partners or simply between internal systems, most workloads will rely on secure and fast file transfers.

A separate, non-integrated and uncoordinated file transfer is an *unmanaged* file transfer approach, which introduces yet another disjointed *island of automation* and another weak link in the chain that makes up the enterprise process. Management of file transfers typically relies on manual coordination or time-based scheduling, which is not only inefficient but also introduces inaccurate and incomplete data into the process.

An enterprise job scheduler or workload automation solution must have integrated managed file transfer capabilities so that it will enhance the flow and the quality of data across enterprise processes. Having a fully automated, coordinated managed file transfer provides consistent and reliable straight-through processing, making enterprises more predictable, with fewer errors.

Central, Scalable, and Multi-tenant Architecture

An enterprise job scheduler or workload solution with a centralized architecture takes advantage of the data being stored in a centralized database management system. All scheduling data, including any scripts, is always centrally available, providing a comprehensive overview of past, present and future processing and ensuring the highest possible operational safety. Centralized control reduces maintenance effort to an absolute minimum and eliminates any potential for desynchronization of the scheduling rules.

A multi-tenant architecture reduces hardware and administration costs, lowering TCO.

Because many products are still restricted to a single-engine architecture rather than an active-active configuration, they cannot scale up in peak load situations. This means you have to purchase new hardware or set up an additional scheduler or workload system to handle the additional workload.

Another relevant factor is the programming language used to develop the automation engine. For example, engines that are developed in Java often have performance issues with high workload and require additional hardware.

Multi-tenancy refers to a software architecture approach in which a single instance of the software serves multiple client organizations (tenants). With a multi-tenant architecture, a software application virtually partitions its data and configuration so that each client organization works with a customized logical instance of the application. Because IT is viewed as a service today, being able to logically separate client organizations is important, because it simplifies reporting and chargeback of delivered services to the appropriate department. Such separation also enhances security.

Object Orientation to Enable Fast, Flexible Implementation

An object-oriented design has enormous time-saving potential because it lets you use a single generic job definition for jobs that run with different parameters on hundreds of target systems. This method is based on the *module* concept. For example, consider a backup process that needs to run on all UNIX servers in your data center once a week as a full backup, and every other day as an incremental backup. For solutions that are not object-oriented, you would need hundreds of job definitions. That approach would also result in a real maintenance nightmare because you would need to make any changes to hundreds of job definitions. And, the one that you would be likely to miss is also likely to be the most important one. An object-oriented approach speeds up deployment and creates a flexible platform for rapid changes in the environment.

Native Application Support

Whether applications are bought or built, they still have a requirement for job scheduling. Historically, job scheduling and systems management emphasized the management of the various components in the IT infrastructure. Networks, CPUs, databases, controllers, programs, disks, and other components require management, monitoring and scheduling. Over time, IT personnel realized that focusing strictly on a given component without regard to the applications that these components supported was missing the big picture, which is monitoring and managing the enterprise.

But the goal of a modern job scheduler is to manage the application, not just the underlying components that support it. For many IT organizations, managing and monitoring core business applications and maintaining service level agreements based on those applications is the essential service they provide to the rest of the organization. Most packaged applications have some type of application program interfaces, such as APIs, that are designed to be used by programmers and other technical personnel. These interfaces allow the scheduler to communicate directly with the packaged application to launch and monitor jobs.

Your solution must be able to natively integrate the variety of the applications in your IT environment into a comprehensive, seamless process flow. It should let you easily manage external interfaces so that one application can efficiently exchange information with all your other applications. It should eliminate the need for custom scripting and routine manual checks between process steps, saving you time and reducing errors.

Dynamic, Event-Driven Automation

Traditional schedulers completely ignore the application’s data as a source for automating business processes, missing the opportunity to remove processing latency and to respond dynamically to the changing needs of the business. Simple job schedulers don’t fulfill today’s increased requirements for automating, integrating, and accelerating complex business processes.

Application data found in flat files, reports, and relational databases represents the current state of the enterprise. Cash fluctuates, inventories rise and fall, and invoices are paid on a day-to-day, hour-by-hour basis. To accelerate application processing, automation products must respond dynamically to the changing state of the business as represented by changes in the corporate data.

Take the case where the inventory of a part has dropped below its reorder level. A typical scheduling approach would run a job at some interval, daily or weekly, to determine inventory. If the corporation wanted to accelerate this process, it would need to write a program to check reorder quantity and take some subsequent action, such as to order more parts. Depending on the nature of the application, several programs might need modification, which could potentially mean significant time and money.

The modern job scheduler must be able to accelerate this application process. With minimal effort, process flows must be able to be configured to analyze changing application data and to trigger events based on that data. Design tools should allow data values to be interrogated, and their status to be checked, to initiate events immediately or at some future time. IT personnel are then in a position to accelerate business processing without making expensive modifications to program source code. The opportunity to incorporate complex conditional processing to create custom solutions that further automate and integrate corporate business processes must also be available. Dynamic application automation contrasts with the capabilities of typical job scheduling tools that rigidly schedule system events or scripts. To move to intelligent business automation that supports the growth of the business, it is essential that processes be run when the business requires them, not simply at a particular time interval. For this reason, it is essential that any underlying application data or business event can be used to instigate processing.

Conditional Business Rules

Another important requirement for the enterprise job scheduler is the ability to implement conditional business rules. These rules can be based on the underlying application data or other factors. You need powerful tools for modeling these conditional business rules.

When implementing business rules, logical operators are used to check for a variety of business conditions, including *greater than*, *less than*, *equal to*, *is like*, and more. A wide variety of conditional operators for querying data values and for making decisions based on the results of those queries need to be provided. Creation of these conditional operations needs to be straightforward and supported through a natural language interface so that non-technical employees can create business rules without coding. Simple examples of the use of conditional logic include:

- *If the value in row x of table y is greater than 100, start process A.*
- *If there are more than 50 transactions in table y, start process A.*

The ability to add additional logical operators such as *and* and *or* means it is easy to construct more complex business rules to model the dynamics of the business.

The ability to apply this conditional logic before, during and after task execution is crucial because it provides tremendous flexibility when attempting to model complex business processes. With this capability, any process can be checked to ensure that all precedent conditions, execution conditions, or successor conditions are valid. Often this conditional logic will eliminate required manual intervention or the writing of program code. A simple example of this would be to check whether a task or requirement has completed prior to starting a process, or possibly checking to see how long the current job has run, and canceling it if it has run in excess of some predetermined time. More complex processes can be constructed around a variety of inputs, temporal conditions, and events.

Conditional logic also allows us to maximize the re-use of workflows across our business. If a particular operating unit requires an extra step within a process, it can be optionally modeled. The result is a standard workflow that can be used multiple times across the organization, reducing maintenance and, more important, guaranteeing that consistent processing happens for every unit of the business.

Workload Balancing

Application processing today cannot be separated from the notion of workload balancing. Workload balancing ensures that scarce computing resources are allocated appropriately among competing tasks and applications. Ignoring the effects of these resource contentions can severely impact processing and cause service levels to suffer. You need efficient workload automation to provide technology for managing these resource contentions.

Balanced workloads ensure resource and service efficiency, recouping the cost of the managed solution.

Workload balancing provides not only the ability to set priorities for processes, but also to create multiple queues so that jobs of similar priorities can be grouped together and can be run according to a predetermined algorithm. Additionally, priorities and queues must be adapted to accommodate the changing needs of the business. Creating and managing queues, changing queue priorities, and assigning priorities to processes and schedules should all be features of a modern-day job scheduler or workload automation solution. These capabilities optimize application throughput and balance the load across systems and applications.

Managing End-User Involvement

Many of today's enterprise applications are designed to empower the end-user. Empowered users submit their own requests, generate their own reports, and are free to make demands on the system around the clock. This poses a problem for the operations staff because these user requests create fluctuations in system load that impact overall system performance. These fluctuations are often severe enough that operations personnel can no longer meet their service-level objectives. Some user requests also collide with other mission-critical business processes. The challenge is to continue to allow end-users the freedom to maximize their use of the system while maintaining a reasonable system load and managing resource contentions.

Like the enterprise applications themselves, an enterprise job scheduler or workload automation product must be flexible and easy to use so that an end-user can exploit the product's features. The end-users know how their business processes can be accelerated or streamlined. Therefore, a self-service capability is crucial to allow end-users to submit and monitor jobs or processes to completion.

The key to making this all work in the enterprise is being able to set the correct processing and prioritization policies for managing user-requested jobs. Operations personnel can create and maintain process queues, priorities and end-user profiles. These parameters can be tuned to eliminate undue system load and resource contention. Queues can be established that only handle certain classes of requests or that operate only during certain times of the day, which in turn allows users the flexibility to submit jobs for execution at any time without IT staff needing to worry about unmanageable processing loads. From an end-user's perspective, he or she has gained an unprecedented level of control and visibility. Launching, monitoring, and canceling requests at will no longer require intervention from IT staff. This eliminates an unnecessary burden on IT staff in terms of handling end-user requests by phone or email. Moreover, since end-users control the submission, the chance for costly errors being introduced when the user passes the request to the IT group is eliminated. Additionally, because input data sources can be validated at the time of the user's request, the simple but potentially costly errors associated with running submissions with incorrect data can be avoided.

Additionally, in today's fast-paced world, it is essential to turn data into information that can be placed in the hands of users. Because our workload automation has all the data about our processing position, and how it impacts the business, this can be turned into actionable information and presented to users in dashboards. These dashboards should augment the information about process states with real business data to create meaningful information.

Security and Compliance

If the programs that run your payroll processing or your money transfers, together with their outputs, are stored on the same servers where they are executed, then any administrator or hacker may get unauthorized access and make changes without you noticing it. This situation poses a huge security risk and can have a very negative impact on your business.

To avoid this risk, you need to take the following precautions.

- All programs and scripts are stored and encrypted in a central database, to which only a few authorized persons have access.
- Every change to the database is documented, so that you can easily track who changed what and when they changed it.
- You have archiving mechanisms that enable you to quickly look at past processing data.
- Any communication between components is highly encrypted, so that no one can view the output of sensitive data, such as payroll runs. If such communications are not secured, hackers can easily gather this information by using network sniffer tools.

Conclusion

If you are looking for a job scheduling or workload automation solution that is built to meet the demanding enterprise requirements of a modern day business, it is important to consider the following key capabilities:

- Job scheduling capability
- Built-in managed file transfers
- Central, scalable, and multi-tenant architecture
- Support for object orientation to enable fast, flexible implementations
- Native application support
- Support for dynamic, event-driven automation
- The ability to embed conditional business rules
- Support for advanced workload balancing
- Self-service capability to enable end-user involvement
- Ability to enable security and compliance across the enterprise

For more information, please visit ca.com/automation.