ProductScope™

JAMS Scheduler

commissioned by

concentrated technology

maximum knowledge • minimum time

MVP Systems' JAMS Scheduler is a job-execution scheduling engine designed for heterogeneous environments of all sizes. It offers a full-featured replacement for traditionally-used native scheduling components such as the Windows Task Scheduler, Unix/Linux cron jobs, SQL Server Agent jobs, and more. It does so in a highly-configurable way that supports robust reporting and monitoring, complex multi-step jobs, and more. The version reviewed for this analysis (January 2011) supports leading-edge job execution technologies, including Microsoft Windows PowerShell, putting the product into a market leadership position.

The Need for Job and Task Scheduling

There has never been a time when systems administrators did not need the ability to schedule various tasks to run in an automated, recurring fashion. The oldest computer operating systems, going back to the earliest multi-user mainframes, incorporated at least some ability to schedule tasks. Modern operating systems tend to provide fairly simple time-based task scheduling, such as the Windows Task Scheduler or Unix/Linux cron jobs. Many application platforms, such as Microsoft SQL Server, Oracle, SAP, and more all offer in-platform task scheduling capabilities of some kind.

The types of tasks that administrators schedule are infinite in their variety, ranging from routine periodic maintenance tasks such as log cleanups, to complex, multi-step jobs that automate entire business processes, such as provisioning new users from a personnel management system.

Weaknesses of Native Scheduling Solutions

Native scheduling solutions tend to be simplistic, offering the ability to run jobs based upon a simple schedule, with fairly basic monitoring of whether or not a job succeeded or failed. Some native solutions, such as SQL Server Agent's job system, offer multi-step jobs with some success/failure logic, and incorporate reasonably complete logging so that a job's progress can be tracked and archived.

Even if all of the native scheduling features that an organization uses are able to completely meet that organization's needs, there's still a weakness in multi-point management: SQL Server jobs are managed in SQL Server. Windows jobs are managed in Task Scheduler. Oracle jobs are managed in Oracle. Different systems have differing capabilities for managing the jobs being executed on remote machines. Job management quickly becomes disjointed and haphazard, making it extremely difficult, if not entirely impractical, to get a cohesive view of what tasks are running when, and where.

Minimum Requirements for Job Scheduling

We spoke with seven administrators in a variety of organizations, all of whom were familiar with job scheduling challenges and solutions used within their organization. After doing so, we believe that the following eight features are the minimums needed by an organization-wide job scheduling and management system:

1. Multiple technologies. A system should be able to execute tasks in as many technologies as possible, to include scripting languages such as Perl, Python, Windows batch, and so forth, as well as in proprietary systems commonly used by businesses, such as SAP, PeopleSoft, SQL Server, VMware, SAS, and more.

2. Central or distributed execution. Execution of jobs should be flexible. The solution should offer the ability to execute jobs on a central server, when that is appropriate, or on a collection of remote computers, when the need calls for it.

3. Job logic and dependencies. Multi-step jobs should offer flexible workflow, so that specific job steps proceed only when preceding steps have completed successfully. Jobs should also be able to run based on external dependencies, such as the exclusive availability of a file or collection of files.

4. Meta-Automation. A job scheduling solution must itself be manageable through automation, such as Application Programming Interfaces (APIs).

5. Coordination. A job scheduling solution must be able to help administrators visualize the entire automation runspace, so that they can avoid job conflicts, manage job execution centrally, and so forth.

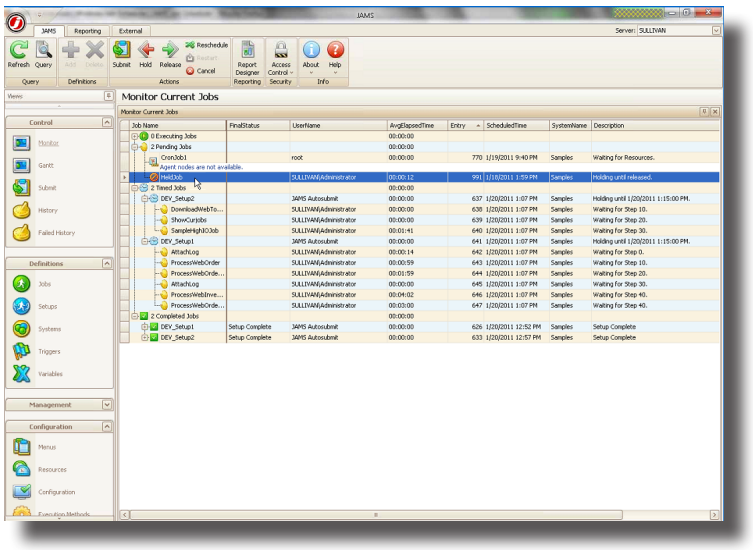
6. Reporting and Monitoring. Administrators must be able to view current job status and report on future and historical job scheduling and results.

7. Security. Jobs must be securable on a per-job basis, ideally assigning permissions to security principals from an existing authentication system, such as Microsoft Active Directory.

8. High Availability. Because scheduled jobs are often used to carry out mission-critical tasks, a scheduling solution must offer some form of failover or redundancy. We feel that it is acceptable for this to be an optional feature delivered by the vendor, as some environments will not feel that this is a high priority for them. If delivered as an optional feature, however, high availability must be easily added to an existing solution deployment if desired.

Potentially, some organizations will want also a system that can migrate their existing natively-scheduled tasks into itself with a minimum of re-working. We do not consider this a mandatory feature, as the administrators we spoke with were of mixed opinions. Many felt that, even given the ability to import existing tasks, they might still choose to re-construct them from scratch in order to leverage better features and flexibility if such were available.

JAMS Scheduler



MVP Systems' JAMS Scheduler is a Windows-based centralized job-scheduling solution. It supports both local job execution and, through the use of agents, remote job execution. Agents are available for versions of Windows, Linux, Unix, OpenVMS, and IBM Power Systems. We feel that Windows, Linux, and Unix will comprise the majority of most customers' use of the solution.

JAMS Scheduler supports an extremely wide range of execution technologies. From a scripting language perspective, it supports Windows batch, Unix shell, SQL queries, VBScript, Perl, Windows PowerShell, Python, AutoIt (which can enable automation of graphical user interfaces, or GUIs), and more. It also supports execution of tasks within SAP, PeopleSoft, VMware, JD Edwards, SAS, and other platforms. It supports direct execution of FTP tasks for file transfers, and can also execute SQL Server DTS/SSIS packages.

JAMS Scheduler is manageable through a custom GUI console, as well as through a Web-based console, making it accessible throughout an organization. Meta-automation is available through a Windows PowerShell snap-in and a .NET Framework API, giving administrators the ability to automate and manage the product itself through either PowerShell or .NET. All of these management options are available for unlimited use within an organization as part of the main JAMS Scheduler license.

The solution supports high availability through the use of a primary/backup execution engine.

Working with JAMS Scheduler

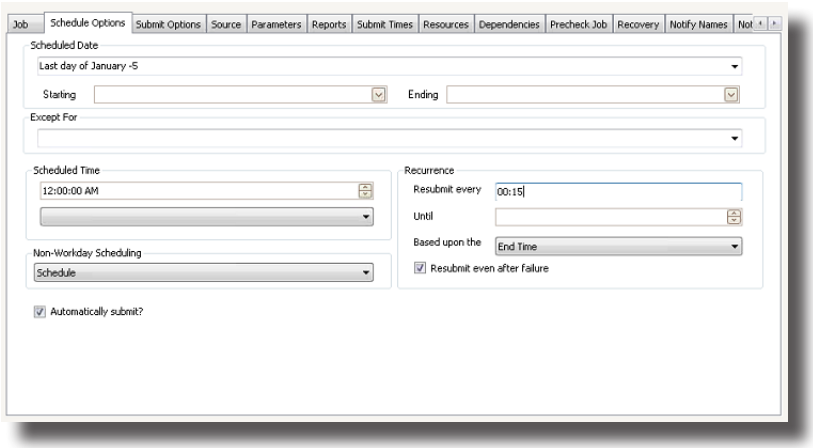
A *job* is the basic unit of work within JAMS Scheduler. A job is tied to an execution method – such as a shell script or SQL statement. JAMS Scheduler can handle the code or commands for these jobs in one of two ways: It can store them internally, or simply reference an existing script file. That means longer scripts can be developed in an external editor which is better-suited to whatever language you're working in. In fact, JAMS Scheduler integrates with certain external editors. For Windows PowerShell, for example, it can launch either Microsoft's own Integrated Scripting Environment (ISE) or the free PowerGUI editor. Scripts edited in this fashion are transferred back into the JAMS Scheduler for storage.

Setups link one or more jobs in a logical sequence, and can include workflow logic and sequencing to make sure jobs execute in the desired order. *Systems* provide a way to group execution targets, and a way to specify settings like security permissions, notifications, and

so forth, which will all be applied to the jobs within that system.

The product includes a large library of sample jobs, including jobs for managing JAMS Scheduler itself, such as purging job history from its database.

Job Flexibility



Jobs can be configured with an enormous array of options – so many that newcomers might find all the choices bewildering. Options are logically grouped on tabs within a properties dialog, so it's easy enough to ignore the more advanced options, and simply exploring the options by tabbing through the dialog is also straightforward.

Scheduling is obviously one of the main and most-used job options. The product uses a natural-English parser, allowing you to enter schedules by typing phrases such as "every other Monday" or "workdays." You can also create custom scheduling terms, permitting you to type a phrase like "last day of quarter" and have the product understand what that means to your company. Jobs can also be given execution windows, and will only run within the window provided.

Jobs can be parameterized, and those parameters given names, default values, explanatory text, and so forth. Parameters can draw their values from global variables within JAMS Scheduler (think of variables as a kind of global, custom configuration setting), or jobs can prompt for parameters if run interactively.

Jobs can, as already mentioned, be configured with dependencies. Jobs will only execute once their dependencies are met. Generally speaking, JAMS Scheduler uses these dependencies as *response* triggers. That means, for example, that it doesn't continually poll to check if a dependency file is available, but instead receives notification from Windows when the file is available and triggers the job accordingly. Other dependencies can include waiting for other jobs to complete and so on.

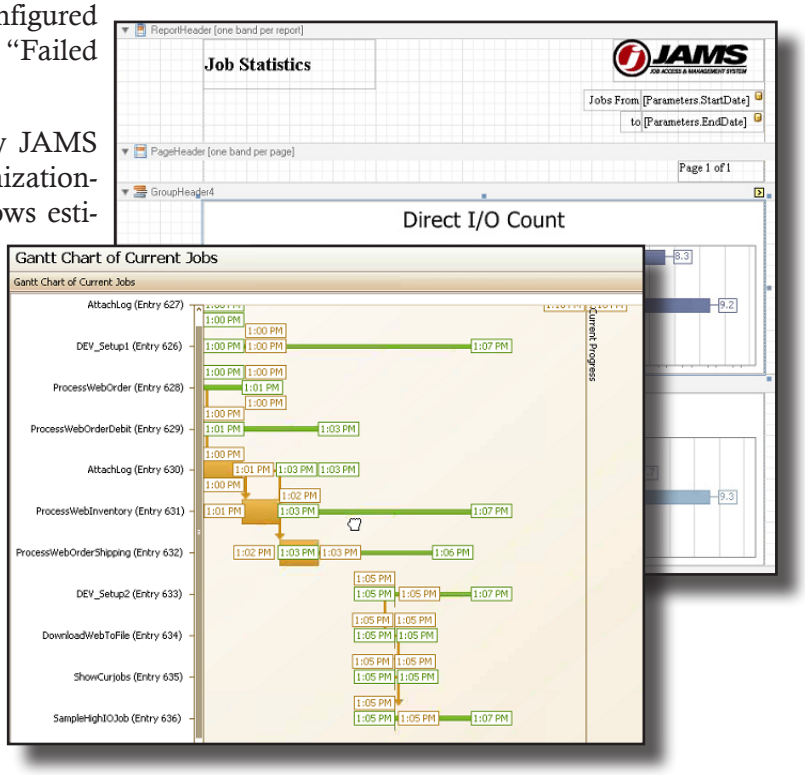
Notifications are another option of a job, and include e-mails for a variety of situations: A job that's running too long, that's been sitting and waiting too long, that didn't run long enough, that's using too many resources, and so forth. Since administrators are unlikely to routinely examine jobs once they're set up and working, this notification system (if used) can be an effective way to learn about jobs that are encountering problems and that may need attention.

Reporting

JAMS Scheduler stores extensive information about job execution in a history database stored in SQL Server. A SQL Server license is not required; JAMS Scheduler is capable of operating with the free SQL Server Express edition if you don't have another suitable SQL Server instance that it can use. Job history is stored for as long as you want it to be, limited only by the amount of disk space you're willing to devote to that database.

Numerous in-the-box reports are available for a variety of job statistics, including run time, resource consumption, and so forth. Custom reports can be created using any of the data from the database, and can include tabular reports, charts, and the like. The GUI

management console is customizable, and can be configured with icons that provide quick access to queries like "Failed Job History" or whatever you might like.



Especially effective is the Gantt view provided by JAMS Scheduler, which shows a chart-style view of organization-wide job scheduling and execution. This view shows estimated job run times as well as actual run times, and permits administrators to avoid job conflicts, manage the overall automation runspace, and so forth. This view is perhaps the most compelling argument for an all-in-one job scheduling solution, since it creates an intuitive visualization that simply isn't practical when you're using a variety of native scheduling solutions throughout the organization.

JAMS Scheduler also supports logging and notification via Syslog and SNMP, and can be integrated with Microsoft System Center Operations Manager, HP OpenView, and IBM Tivoli for monitoring purposes.

Triggers: Responding to Events

JAMS Scheduler can also register for specific kinds of events, such as job completion, various file system-related events, and so forth, and can trigger a job in response to an event occurrence. For example, JAMS Scheduler could monitor a "drop box" folder, and automatically kick off a job when files appear in that folder. That job might then run a translation utility on the file, and then FTP the file to some other location.

This ability to respond to external events pushes JAMS Scheduler into the realm of a true runbook automation and orchestration solution, giving it the ability to act as the "middleman" or "glue" that ties multiple disparate systems together into a unified business process.

Migration from Native Scheduling

JAMS Scheduler can natively import Windows Task Scheduler tasks from remote machines, as well as SQL Server Agent jobs and SAP tasks. With an add-in, it can also import cron jobs. While not useful for every organization, these import capabilities do make it easier to migrate to JAMS Scheduler for organizations that already have a heavy investment in these native scheduling mechanisms.

Windows PowerShell Provider

It's not uncommon for a job scheduling system to need interconnections with other technology processes. For example, an administrator may wish to have a job scheduled in response to an Operations Manager alert – perhaps scheduling a server to be restarted at a given time to help correct a problem. For that reason, some kind of externally-accessible API is often desirable in job schedulers.

JAMS Scheduler exposes its internals through a Windows PowerShell *provider.* This provider essentially "adapts" the product's internal configurations so that they appear within PowerShell as a disk drive. Administrators can then use well-known commands – the same

ones used to manipulate the file system – to explore and manipulate anything exposed by that provider.

We feel that the inclusion of this provider is a good strategic move, both for MVP Systems and for JAMS Scheduler customers. Windows PowerShell is clearly Microsoft's way forward for administrative automation. By adopting this PowerShell interface, MVP Systems enables their product to integrate more tightly with a growing number of other products and systems – without requiring administrators to learn a proprietary API, scripting language, or other mechanism.

Support

MVP Systems offers 24/7 technical support for JAMS Scheduler. The company informed us that many of their customers are in "always-on" industries such as financial services and healthcare, and that their support organization was built accordingly. They also maintain an online knowledge base that offers not only solutions to problems, but also tips, tricks, and other information that can help administrators solve especially tricky challenges more easily.

It should be noted that, in our conversation with administrators, contacting technical support or using a knowledge base for a task scheduling product seemed counterintuitive to them. We suspect that their reliance on native scheduling mechanisms – which obviously don't come with that kind of dedicated support – causes that mindset. Administrators who begin using a centralized scheduling solution will need to take care not to go too far down the "do it yourself" path when they run into problems or challenges, and instead take advantage of vendor-provided support resources when available.

Competition

This analysis did not include a comparative look at competing job scheduling products. However, there is a fairly rich marketplace for these products, with a handful of companies competing at the level where JAMS Scheduler is marketed.

We believe that the existence of a diverse and competitive marketplace is desirable. It indicates that there is significant customer demand for this kind of product, and it drives vendors in this space to continually innovate and improve their products, lest they fall behind in the marketplace. JAMS Scheduler has shown a steady stream of updates and improvements, suggesting that MVP Systems monitors their competition and responds accordingly.

Summary

We believe that JAMS Scheduler meets all eight of the key requirements that organizations will have for a centralized job-scheduling system, along with the ability to import existing natively-scheduled jobs from numerous sources.

We particularly like the generally agnostic approach that MVP Systems has taken with the product, supporting a variety of operating systems for agent deployment and supporting a plethora of execution technologies and scripting languages. Of special note is JAMS Scheduler's support for Windows PowerShell: Not only does the product execute PowerShell scripts through its own in-process shell host, but it is itself manageable via PowerShell, by means of a provided shell snap-in. This ability aligns precisely with Microsoft's own manageability strategies and priorities, and makes JAMS Scheduler especially suitable for Microsoft-based environments that are investing in PowerShell as a means of task automation.

About Concentrated Technology

Concentrated Technology, LLC is a technology education and analysis consulting firm founded by industry luminaries Don Jones and Greg Shields. The company provides strategic consulting services to companies around the world, and performs market research and analysis for technology vendors and customers. The company distinguishes itself through its motto, "Maximum Knowledge - Minimum Time," focusing on producing concise, accurate materials that help modern businesses accomplish more, with less effort.

About MVP Systems

For more than 20 years, MVP Systems Software, Inc. has provided leading-edge batch job scheduling and workload automation solutions to its more than 750+ customers. Customers include household names like Bank of America, JPMorgan Chase, Boeing, FINRA, Manulife Financial, Kaiser Permanente, The Hartford, and the US Postal Service. MVP's solutions are delivered in traditional software as well as SaaS models. For more information, please visit www.JAMSScheduler.com.