Automated Job Scheduler Basics

From Scheduling Tasks to Automating Jobs

Scheduling tasks in organizations is a common need. Grouping these tasks into cohesive units called 'jobs' and then automating these jobs allows IT staff to manage thousands of individual tasks with higher levels of quality, reliability, and productivity. Automating job scheduling reduces manual interventions and unnecessary delays while increasing the efficiency and utilization of computing resources.

There is no lack of opportunity to utilize automated job scheduling. Consider, for instance:

- A customer relationship management (CRM) job that contacts customers periodically during the sales cycle or,
- A file transfer job used to transmit, validate, catalog, and receives files on company business days, or
- An HR job that sends reminder emails to notify employees and customers about important events.

As the IT staff workload inevitably increases the justification for automated job scheduling becomes increasingly apparent.

But How?

Job scheduling has been around since the early days of computing. The Unix cron command has always been a popular job scheduling solution. It has a powerful facility for determining when Unix commands or scripts should execute. Cron is used typically to automate system maintenance or administration—though its general-purpose nature is utilized in a wide variety of tasks. Windows machines have the Windows Task Scheduler available which provides a subset of cron time scheduling and a set of Windows specific capabilities.

Software development teams and System Administrators often write scheduling code for their particular needs. These are often delivered as Unix or Windows code snippets delivered in shell scripts or batch files although sometimes the code is embedded into software applications directly. Such development is justified when the task at hand is considered too specialized or the scheduler needs are rudimentary and not needing an off-the-shelf solution.

Over the years many commercial and open source job scheduler offerings became available. The list of vendors is too large to mention here, but solutions can generally be filtered by their supported platforms such as mainframe, Linux, and Windows. There also exists offerings that schedule jobs across platforms. Other distinctions can be found in their support for graphical user interfaces, reports, data analytics, workload distribution, and of course, price.

Characterizing Modern Job Schedulers

Most modern schedulers provide capabilities that include the following:

Automated job scheduling

- Flexibility in scheduling, and rescheduling, jobs
- Unattended (or attended) job processing 24 hours a day, 7 days a week, with total compliance to set schedules
- Control of how, when, and where a job is submitted
- Job dependencies such as the existence of a file, the state of a database query, or the receipt of an email, the activity or inactivity of other jobs
- Business Calendar support, including fiscal and holiday calendars
- Multiple runs per day

System and user-defined variable

- Current date, submission date, previous date, and current time can be passed into application programs
- User-defined variable values can be created, changed, and passed into application programs

Workload/history forecasting

- Forecasts all scheduled jobs to be run next week, next month, or next day
- Optimize production requirements
- Historical tracking and logging of all scheduler activity

Workload Distribution across the Network

- Jobs can be set up to run on any network node
- Provides complete job history of the job on the submitting system
- Group and dependent jobs can be submitted through the network

Report distribution and management

- Routing, monitoring, and controlling of all output reports generated by the scheduler
- Report file distribution to multiple systems
- Output can be duplicated or sent to any user

Security

- Existing LDAP or Active Directory security can be utilized within the scheduler
- Specify who in your organization has authority to set up or change information about scheduled jobs
- Authority can be specified for either the individual functions of the scheduler or for specific jobs

Graphical user interface

- Point and click capabilities when scheduling or designing a job
- Manage jobs
- Maintain dependencies
- Track scheduler activity and log information

Other key features

- Multiple commands per job
- Console monitor to run jobs in restricted state
- Check maximum run time for each job
- Interface directly to a message-based third-party paging system or issue tracking system
- Provisions for full online documentation of each job
- Help text on all displays

Kinds of Scheduling

Automated schedulers provide various schemes to schedule jobs and their underlying steps. Support for these schemes vary by scheduler. Some schemes to consider include (this list is by no means exhaustive):

- Time such as a specific time or day. A business or holiday calendar may impact when a timed job is allowed to start
- Job priority
- Resource availability (e.g., CPU, memory, disk, network bandwidth)
- Number of simultaneous jobs or daily allowed jobs for a user or type of job. Sometimes referred to as concurrency constraints
- Estimated execution time of a job
- Elapsed execution time of a job
- Availability of peripheral devices such as printers
- Occurrence of prescribed events such as a database query returning a specific result or a mail from a specific sender being received
- Job dependency such as a job or particular step having completed or failed
- File dependency as when a file arrives
- Operator prompt dependency to allow a job to continue

From Descriptions to Definitions

Transitioning from hand-drawn flowcharts and manual procedures to an automated job scheduler's definition takes time and effort. The degree of effort varies depending on the complexity of the process being automated and the skills of the IT staff performing the transition. Migration utilities and converter routines may import some information but seldom deliver a complete representation. Additional work and definition is generally required. During this definition phase using existing process diagrams (be they flowcharts, UML diagrams, a Business Process Modeling Notation (BPMN) graphic or other sequence flows) can be instructive. Consider the following for each job.

- What steps are involved in the job?
- Which step(s) are started when the job starts?
- What triggers the start of each step?
- Is the step initiated strictly by time event (e.g., 12 AM), the completion of a prior step, or some other condition (e.g., the presence of a file in directory perhaps)?
- What are the sequence of steps? Are steps in the job linear, or do steps split and merge into other steps during the job's execution?
- Is the result of a step used to determine the next step to execution?
- Does the step require a manual intervention, a response to a prompt, or some form of review before allowing the job to continue to the next step?
- Are the results from a step used as variables to direct the processing in a later step?
- What possible errors can a step generate and how will errors be recovered from?
- What step(s) are the last steps in the job?
- What steps can trigger the automatic cancellation of the job?

Ensure that the above information is captured and documented in a form usable for your translating the information to your automated job scheduler definitions. Using a spreadsheet or standardized Word template are frequent medium for this information capture.

From Existing Process Definitions to Automated Job Scheduler Definitions

Automated job schedulers present facilities for designing and defining jobs. The vocabulary of the job scheduler may not necessarily map to the vocabulary used to describe existing processes since the job scheduler may introduce new concepts, or have a different name for existing concepts, or provide enhanced features not understood or used in the existing scheduling environment. Consider preparing a mapping of existing to new terms as follows using a simple table:

Existing Term	Scheduler Term
Job	Workflow

Step	Action
Timed Event	Timer Trigger
Shell Script	Process Action
SQL Query	Database Query Action
Run	Run
Calendar	Business Interval, Business Calendar
Operator Prompt	Provision
…	…

The above addresses the terminology to be addressed. The next step is to then start to develop the jobs in the scheduler. This goes beyond mapping terms to addressing matters of sequence, looping, timing and timeouts, error and exception handling, templates of reusable jobs, and numerous other topics. Ideally the selected job scheduler allows one to visually construct and review the job as well as describe the job in a text document. These job scheduler definitions need to be reviewed and compared with the existing definitions to assure completeness and consistency.

From these definitions one then proceeds into testing and finally production rollout. During testing and rollout issues of metrics collection, failover and recovery, and exceptional and adhoc conditions needs to be designed and documented.

Conclusion

Automated job schedulers have been available for decades and their value and utility continue to improve. The process of moving from manual or scripted job execution to automated job scheduling is not necessarily easy or straightforward. Effective adoption and utilization of automated job schedulers requires that IT staff pay attention to job design, recovery, and error handling. Partnering with your scheduling vendor through their training and consulting, as well as leveraging already tried and tested job patterns and templates can significantly improve the likelihood of success in improving IT delivery through job scheduling automation.