Scheduling Reports: More Than Just Fire and Forget

Moving beyond Scheduling Reports to Orchestrating Them

Reading Time: 5 minutes

Scheduling reports is a common use case for job schedulers. Such scheduling is often seen as a simple matter of setting a timer to fire off a report at a given time. Available open source projects and even Windows Task Scheduler are often used. But this simplistic approach is often found wanting once the true requirements are better understood. A more realistic set of report scheduling requirements would include, for example:

1. User can specify a specific schedule to run their reports.
2. Schedules may vary in the kind of reports run, and the parameters passed to the report creation software.
3. User can specify a set of dependencies that must be satisfied before running their reports. An example dependency may include associating a point in time with a database status with the availability of files before a report can be run.
4. User can specify error situations associated with report execution that require forwarding to IT staff for resolution. In some instances user can define exception handling processes that are initiated automatically.
5. User defines status and tracking metrics for report execution, success and failure.
6. Service level agreements are specified and associated with the reporting process.
7. Report execution may need to be assigned to a specific reporting server or node where the required resources are available for processing.
8. Reports may need to be generated and coordinated across multiple reporting systems, e.g., Cognos, Actuate, Oracle, and J.D. Edwards.

Reviewing a more complete set of requirements for report scheduling exposes the need for a more comprehensive reporting solution. Embedding report generation and distribution into a workflow management solution is an increasingly common means to address these requirements. As opposed to scheduling, workflow management addresses job dependencies, monitoring externally-supplied events such as incoming messages or file availability, time based events, error handling, and monitoring.

Many reporting software packages provide some basic scheduler functionality. Often these schedulers are limited in capability and flexibility. Such report-vendor provided schedulers often address items 1 and 2 above, but leave the other requirements unsatisfied. Items 3 through 7 can be addressed via workflow and file orchestration capabilities generally not provided by the reporting vendor.

Flux supports reporting within the context of workflow. Flux supports schedules not only based on time, but also on database conditions, the existence of files, the arrival of incoming web service messages and mail messages, the state of other workflows, and other channels, all within the context of an overall workflow.

Workflows containing reports can be very complex. For instance, many financial institutions utilize their workflow engines to control generating reports within the overall context of their payments processing. These workflows tend to be large, configurable, and contain many dependencies and control points that trigger the creation of a variety of reports. One flow in one of these larger workflows may initiate a file transfer of a payment file into the institution, validate the file, output a report of errors and omissions, trigger a reject and correction process, dynamically configure a set of customer specific processing parameters and merge those with general processing rules from a configuration database, archive the corrected file to a document archive, and initiate payment reporting and downstream audit and compliance processes. Status is messaged and emailed to the institution's customer and internal operations staff throughout the entire process. Numerous reports are created during the process.

The general approach to such systems requires a design where a workflow engine acts as the system orchestrator. External components message the workflow engine via files, web services, database contents, and the workflow engine's API. The workflow engine itself messages and directs external components, such as a reporting system, via that component's command line, or web API, or other vendor-provided API. A number of workflow engines can easily wrap a reporting vendor's Java or web API to orchestrate report generation within the context of the larger workflow. Many reporting vendors provide APIs that allow external workflow engines to direct their report execution in the context of a more complete workflow perspective.

Such report-vendor-provided reporting APIs generally provide facilities to:

- Configure a report dynamically.
- Populate a predefined report template with workflow-supplied parameters.
- Submit a report for execution.
- Get status updates on the report's execution status.
- Format the report into some format (e.g., PDF or CSV)
- Distribute the report via email, SFTP, or some other distribution mechanism.

Using these APIs in conjunction with a workflow engine provides the ability to provide sophisticated and dynamic reporting capabilities within business process workflows. As the workflow progresses, for instance,

- It can collect and configure parameters for reports specific to the information available at the time. Parameters may include selection criteria, date ranges, distribution lists, and the schedule on which the report should run.
- Information generated from a report's execution can actually be injected back into the workflow, driving complex workflows through multiple paths. This use cases surfaces when the report is the output of some data analysis, and the report's output may drive the workflow to recurse through portions of the workflow to get to a recommendation or other decision.
- The workflow can ensure that required dependencies are met before executing the report. Or instance, reporting periods often involve many dependencies. "End of Financial Period" (e.g., end of month or end of quarter) frequently involves dependencies that include a specific date and a job dependency graph that ensures all needed reports and predecessor processing are complete before the specific report is generated to ensure consistent and complete data is available to the report.
- The workflow in some instances can solicit customer input via a client facing web form, MQ message, or email structure that allows the user to specify needed parameters and pass them to the workflow engine using workflow engine's API set.
- Data can be extracted, moved, or translated in preparation for report creation. In some cases images may need to be transcoded, needed files transferred and centralized, or database extracts performed.
- Finally, report jobs are configured for execution on specific nodes – in effect 'pinning' them to the resources they require for processing. Some workflow management solutions allow configuring unique execution requirements based on the 'namespace' the workflow executes within. A namespace can be any organizing structure, such as client, department, month, or business event. The same generic report workflow can be deployed to many namespaces where each name space has its own unique and specific configuration.

Generating reports in the context of a comprehensive workflow provides needed context to ensure report data is timely, accurate, and complete. Using a fire and forget approach, while satisfying the need to create reports, may not necessarily be an effective or efficient approach at getting needed information to your clients in a manner that will satisfy them.

In Summary

Flux's workflow and scheduling features extend enterprise reporting platforms with automated scheduling, workflow dependencies, and automated error recovery and exception handling. As such, Flux offers a platform-agnostic means to integrate reports into an enterprise's processing flows.

About Flux

Flux assists enterprises in provisioning, onboarding, scheduling, tracking, and reporting an enterprise's file orchestration processes. These orchestrations vary from simple file transfers to highly complex workflows involving extensive processing, many routes, varied alerts, and complex decisioning. First released in 2000, Flux has grown into a file orchestration and workflow scheduling platform that enterprises rely on daily for their mission critical systems.

Contact Flux

Contact sales at: sales@flux.ly
For further information browse: flux.ly