independIT Integrative Technologies GmbH
Bergstraße 6
D–86529 Schrobenhausen

BICsuite!focus

The independIT BICsuite Scheduling System in Data Warehouse Environments

Dieter Stubler Ronald Jeninga

November 25, 2016

The independIT BICsuite Scheduling System in Data Warehouse Environments

Introduction

A multitude of processes have to be executed in data warehouse environments on a daily basis.
Although an enterprise scheduling system is to be found in many companies, it is frequently not used for controlling the individual processes in data warehouse operation. In the majority of cases, larger and more complex batch processes are triggered by the enterprise scheduling system, but the workflow control within these high level batch processes takes place using alternative solutions outside the enterprise scheduling system. The reasons for this are usually the lacking features in the deployed enterprise scheduling system or its complex and complicated handling.
The data warehouse batch processes are, for the most part, internally controlled using the internal tool scheduling functionality of the utilised tools and/or the use of scripting (sh, Perl, Python, ...).

Internal tool scheduling

The use of internal tool scheduling functions is problematic because external tool processes frequently have to be integrated as well. This is often not possible or must regularly be resolved using workarounds. Where multiple tools are being used for different tasks in a 'best of breed' environment, monitoring the active workflows in the various systems becomes a problem. When a tool has to be replaced, the workflow controls realised in the tool need to be re-implemented as well.

Scripting

Even more problematic here is the use of scripting. To achieve a minimum of stability involves significant development time and costs, and the result is frequently the development of a bespoke, small-scale scheduling system.

Necessity for a scheduling system

The two frequently encountered solutions described here are neither efficient with regard to the required development resources, nor are they suitable for guaranteeing a medium and long-term stable operation of the data warehouse environment. Operating a data warehouse environment without a scheduling system is always too expensive and unreliable.

Practical application of enterprise scheduling systems

Enterprise scheduling systems have their roots in controlling production workflows on mainframe systems. These production workflows are typically extremely

static, not so complex, stable, and generally require few or no operator actions. Since development and production are usually separate entities, it is not necessary for developers to familiarise themselves with the enterprise scheduling system.
This is different in a typical data warehouse environment, however. Data warehouse process workflows change almost on a daily basis. Errors caused by resource bottlenecks when processing large volumes of data occur much more frequently. In most cases, there is no strict separation between development and production. This means that each developer, alongside performing his own work, is also involved in monitoring and operating his data warehouse processes. The complex dependency relationships in data warehouse workflows place heavy demands on the functionality of the deployed scheduling system.
Utilising an enterprise scheduling system to control all the data warehouse (sub-) processes is therefore often the wrong solution because these systems are not designed to meet the requirements of data warehouse operations.

Requisite capabilities of a scheduling system in data warehouse environments

A scheduling system in a data warehouse environment must therefore feature the following capabilities:

- Handling the system (job/batch definition, execution, monitoring, operation) must be easily and quickly learnable even without the need for any programming skills. To achieve this, the system's underlying concepts have to be simple, clear and understandable.

- The scheduling system must incorporate all the features required to cope with even complex task definitions for mapping the workflow controlling to avoid having to resort to scripting solutions.

- In order to be able to respond quickly to new demands that arise in the data warehouse, it needs to be possible to modify the workflows at all times without affecting any of the active workflows.

- The operator must always be able to exercise control over the active workflows so that he can respond to emergency situations in an appropriate manner.

- The installation, administration and operation of the system must not require any special system privileges (root privileges, etc).

- Definition, execution, monitoring and problem analysis have to be possible without any additional software installation and at any workstation (web application server).

- As data warehouse environments always tend to test the limits of the hardware, the system must provide appropriate mechanisms for controlling the system resources in order to avoid or diminish errors caused by resource bottlenecks.

- Access to workflows with regard to their definition, monitoring and operation, as well as the capability for executing jobs in certain environments (job server), must be safeguarded by user privileges.

Established enterprise scheduling systems do not (or only in part) provide the capabilities stated here, and are therefore only suitable to a very limited extent for deployment in data warehouse environments.
We will now go into more detail about some of the points we have already mentioned.

Ease of operation

Ease of operation was a focal issue in developing the BICsuite Scheduling System. A web application server requiring no software installation equips the workplace with an easy-to-use, browser-based GUI covering all aspects from the workflow definition, scheduling, monitoring and operation, through to administration of the job server. Experience gathered at customers shows that a short familiarisation session lasting about 3 hours can put each team member in a position to use the system productively. The capability for defining templates for recurrent complex tasks, based on which such tasks can be performed with just a few actions, means that it is unnecessary for most of the team to acquire more in-depth knowledge of the system.

Complete feature set

Data warehouse workflows are typically characterised by complex dependency relationships and a workflow logic. To avoid having to resort to scripting solutions to implement these workflows, the scheduling system must incorporate a complete feature set. The BICsuite Scheduling System was designed based on the complex requirements of a large data warehouse, and it has been steadily improved over several years of productive operation.
Here are just some of the features of the BICsuite Scheduling System:

- An arbitrary number of user-defined exit statuses allow for a flexible response to the results returned by a job (not just failure or success!).

- In addition to the exit status, jobs can also hand over additional result data âĂŃâĂŃto the scheduling system which can be displayed in the monitoring module to give the operator supplementary details about the jobs.

- Based on the exit status and result data, the downstream processing of workflows can be controlled using dependencies and triggers. This also includes the conditional or repetitive execution of sub-workflows.
- Parameters and result data from jobs can be used by subsequent jobs for internal control purposes.
- The capability for giving workflows a hierarchical structure significantly simplifies the definition of dependencies and triggers for controlling the workflows, enhances the reusability of (sub-) workflows, and ensures that even large-scale workflows remain manageable.
- Any parts of the workflow environment (tables, data ranges, data marts, files, ...) can be mapped as status and feature-laden resources to enable workflows to be controlled dependent upon the status and actuality of the required resources (for example: Creating a report only when the required data range is 'VALID' and it has been updated within the past day).
- The dynamic submit allows jobs to trigger other jobs or sub-workflows at runtime. This facilitates the programmatical controlling of workflows and parallelisation of (sub-) workflows.
- A warning system allows the operator to be advised about potential problems without influencing the workflow.
- A full API enables all the BICsuite Scheduling System functions to be controlled programmatically.
- As all the definition, configuration and runtime data is stored in an RDBMS, user-defined valuations can be run whenever required.
- ...

A complete list of all the features is beyond the scope of this documentation. Representative enterprise scheduling systems are usually overextended at this point and force the use of a workaround with scripting solutions. Consequently, task definitions that cannot be accomplished using features in the scheduling system are outsourced to the job implementation. This results in inflexible, non-standardised, unstable and scarcely maintainable customised solutions.

Flexibility

Workflows in data warehouse environments have to be modified on an almost daily basis. Not only that, but they frequently also have an extremely long runtime. This means that it must be possible to modify a workflow instance even while it is active. To ensure that this does not cause any problems in already active and partially

executed workflows, any changes must only be allowed to take effect with subsequently started workflows. The scheduling system must therefore be capable at any one time of handling multiple versions of a workflow definition. For this reason, the BICsuite Scheduling System saves versions of all the aspects of a workflow definition, and runs the workflows using the version that was valid at the start (submit) of the workflow. Ad hoc actions (error fixes, analyses, migrations, ...) frequently have to be realised in the data warehouse. Since it only takes a few operations to create jobs and workflows in the BICsuite Scheduling System, such ad hoc actions can be quickly and easily placed under the control of the scheduling system and are thus subject to the system's resource control mechanism. Consequently, the risk of these ad hoc actions interfering with other crucial production workflows can be significantly reduced.

Operator functions

To be able to react quickly to exceptional situations (the rule in data warehouse environments), the operator is provided with functions for intervening in active workflows.
These are:

- Stop and restart (sub-) workflows (Suspend/Resume)
- Restart the rerun of failed (Restartable) jobs (Rerun)
- Set the exit status for jobs
- Discard (sub-) workflows (Cancel)
- Ignore dependencies
- Ignore resource requirements
- Change the scheduling priority of (sub-) workflows
- Quit active jobs (Kill)
- Comment on (sub-) workflows
- Reset warnings
- Change the availability and quantity of resources
- Start/stop the job servers

All operator interventions are logged in an audit trail for subsequent traceability.

No restraints with system privileges

The members of a data warehouse team do not usually have any system privileges (root access,
. . .). If the scheduling system requires such privileges for operation or administration purposes, any changes that have to be made to the system will cause delays and extra costs in implementing them. This applies in particular where, as is frequently the case, the operation of the hardware and operating system has been outsourced. The BICsuite Scheduling System does not require any such privileges for installation, administration or operation. This means that fast and cheap access to all aspects of the scheduling system is always guaranteed.

Closing remarks

Even if not all the aspects and features of the BICsuite Scheduling System have been discussed in this documentation, you will still have gained an insight into the powerful capabilities of the BICsuite Scheduling System in data warehouse environments. Our team will be glad to provide you with more information.