

Batch Processing in a Services World

Subtitle: BPEL and Job Schedulers are both relevant even as IT infrastructures migrate to real time services-based processing

The job scheduler and associated run time are architectural components that are typically associated with outdated batch processing systems. However, job schedulers are very important in services based infrastructures and cloud computing environments. This is because anytime you buy something with a credit card, make a phone call, pay for a doctor's visit, or pay your mortgage there is a high probability the transaction is processed on a mainframe. There is also a high probability that the transaction was processed in a nightly batch cycle using a job scheduler.

There is some discussion in the industry that BPEL engines can provide the same type of functionality that a job scheduler provides. Fundamentally, job schedulers and BPEL process managers are machine and human work flow engines. However, this greatly simplifies the number of duties performed by job schedulers to provide you with enterprise-wide batch load scheduling and management.

This article will explain how BPEL and job schedulers (most recently branded as Workload Automation suites) provide an integrated solution that can satisfy the needs of batch and real time processing in a services-orientated infrastructure. Industry leading distributed job schedulers, workload automation (WLA) products, are offered from UC4, Orsyp, CISCO and Advanced Systems Concept, Inc. Oracle offers an industry leading BPEL Process Manager that runs on a variety of Java EE containers.

Why both?

We are all familiar with the functionality and features of BPEL Process Managers. It would seem that this functionality makes the need for a job scheduler supercilious in service-orientated applications. However, job schedulers provide functionality that is not core to BPEL:

- ? Platform support: You must be able to execute jobs (do not to have to be web services) on any popular hardware or OS platform. This also includes integrated scheduling of z/OS, iSeries, Tandem, Unix, Windows, and Linux workloads.
- ? Heterogeneous batch job support: Since batch still rules most of the mainframe and mid range world, the ability for job schedules to execute jobs (whether they are CA, Autosys, UC/4, ControlM) and integrate these different jobs into one job stream is necessary.
- ? Interface with job execution engine: This is different from calling a job execution engine to execute a batch job on the mainframe. This is the ability to interface with the mainframe job run time environment (JES) during processing. Job Entry Subsystem (JES) is the IBM mainframe job execution engine. JES2 and JES3 are the two JES execution engines from IBM. Sometimes, it is necessary to call JES2 or JES3 during processing of applications on open systems.
- ? Sophisticated job monitoring and management: The system manager and developer must be able to set and change job priorities, cancel jobs on the fly, restart jobs, change the frequency of jobs, and more.
- ? Service classes: Service classes (job priority) are built into JES2 and JES3 from the outset, to

provide a mechanism for prioritizing batch processing based on resource requirements and job priority.

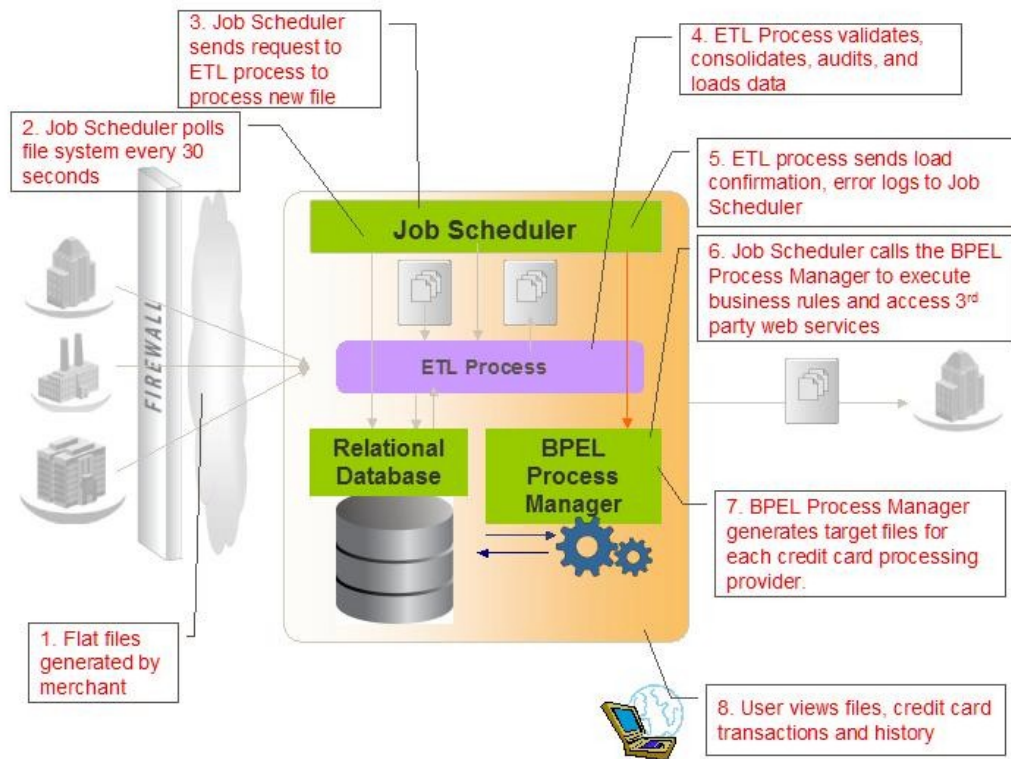
- ? Restart: Job Schedulers have work flow restartability in the event of job, system, or application failure. JCL and the JES engines support multi-step processes with a sophisticated checkpoint and restart capability.
- ? Calendaring: This is not just a simple system that allows the job to start at a specific time or date. This is a complex calendaring system with the ability to change dependencies and view daily, monthly, quarterly, and yearly job schedules. It is also a calendaring system allowing for easy configuration and maintenance of even the most complex schedules, allowing users to maintain a single job schedule calendar across the enterprise.
- ? Proactive critical path analysis: Job Schedulers automatically calculate and displaying job dependencies, as well as analyze the critical path. Proactive notification, in the event it appears, means the critical path is in jeopardy of completing on time. Also, the ability to analyze the impact of a change in the job schedule or a new job being added before it is put into production are tasks automatically performed by the job scheduler.
- ? Management by Exception: The job scheduler must be able to manage by exception. Management by exception means constantly monitoring the millions of executing jobs, alerting only on those that fail, run long, or start late.
- ? User and process-based time limits: The ability to set execution time allocated to a user, number of simultaneous jobs allowed for a user, and execution time allocated to a process.
- ? Capturing and storing of historical job data: The job scheduler keeps historical information on jobs, so it can estimate the job execution time. It can also be set to alert you if a job runs over a specific threshold of the estimated time. The job scheduler provides specific details of elapsed execution time of each job and job step.
- ? Interface to hardware devices: The job scheduler is proactive and can check on the availability of peripheral devices such as storage and printers before it executes a job that is dependent on a device that is offline, or not functioning properly.
- ? Dependency checking: Before the next job in a job flow runs, it checks to make sure all dependencies and prescribed events have completed successfully. The next job in the schedule will not 'kick off' unless all the prerequisites of the job flow are completed successfully.

BPEL and Job Schedulers working together

BPEL is a tool offering enterprises the ability to streamline and rethink existing business processes. BPEL coordinates web services actions. Where business processes are extremely complicated, BPEL on its own is not always enough. In addition, it is important to remember that not all processes are web services.

Most real-world integration scenarios involve multiple applications (or application components) which run on distinct physical machines across an enterprise network, and may be developed in different languages and run on different operating systems. The typical scenario for work flow integration

involves the flow of events and request/reply interactions between service-components distributed across a heterogeneous network. With this in mind, a typical file batch processing use case provides a graphical view of how a BPEL process manager and job scheduler would work together to process credit card transactions:



You can see how background job schedulers / workload automation fills the gap between traditional job scheduling and business process management (delivered by BPEL) by providing needed background flow control capabilities. This use case also demonstrates a number of capabilities that BPEL Process Managers, Business Rules engines, ETL tools, Job Schedulers and relational databases provide by working in unison:

- ? Automated and adaptive flat file processing: The processing that takes place is very data centric so it makes sense that it is very close to the database. Also, the use case here is very much a situation of extract, transform and load (ETL) data. The file is received (or extracted from the source system), it is transformed (validated and reformatted) and loaded (into a relational database). One of technology vendors' inventions is ETL software that is great at performing a specific set of data processing and loading use cases, is easy to maintain, and reduces developer costs. Why reinvent the wheel?
- ? Business Rules processing: There are many rules associated with processing credit card transactions. These business rules no longer need to reside in application programming

languages like COBOL, Java or .NET as Business rules engines are integrated with BPEL Process Managers.

- ? Integration with third party web services: The banks or merchants that are part of this scenario may already have business processes and/or logic that they require be used during the processing of the credit card transactions. BPEL makes it easy to integrate these services into the overall solution.
- ? Near real time file processing: The files can be processed as they arrive instead of waiting until the nighttime. This is largely to do with the fact that relational databases do not need to off line at night in order to perform maintenance, back ups and load large amounts of data.
- ? Ad-hoc reporting of file processing: Exception handling and reporting is now written in Java using Java Server Pages (JSPs), Java Server Faces (JSF) and relational database persistence products. This allows all exception handling to happen in a browser, and in near real time. Because all input file records are stored in the database, they can be edited in the browser.

Summary

Job schedulers are as important to modern IT technical infrastructure as they were in legacy systems. Open system job schedulers actually become more important and must be more robust as compared to mainframe job schedulers. This is because they must interface with distributed platform ERP, CRM, and other COTS applications. They have to be web service ready as they are expected to communicate in a bi-directional manner with service orientated technologies such as BPEL and ESBs. Today, BPEL Process Managers have become a core component for business process execution in service-orientated infrastructures. The business information processing of the future is business service automation via the Grid, Cloud Computing, dynamic workload automation, and job management as a service. This means that job schedulers and BPEL Process Managers will both play a significant role in IT infrastructures for years to come.

Author summary

Tom Laszewski is currently the Director of the Oracle Platform Migrations Group. Tom works on a daily basis with global SIs, technical architectures, CTOs and CIOs, and account managers to ensure the success of migration projects. Before Oracle, Tom held technical and project management positions at Sybase and EDS. He has provided strategic and technical advice to several starts up companies in the Database, Blade, XML, and storage areas. Tom holds a Master of Science in Computer Information Systems from Boston University. Tom is co-author of the [Oracle Modernization Solutions](#) book found on Amazon and other book web sites. He also contributed two chapters to a [legacy modernization book](#) released in February, 2010.