Elastic job or job agent

Last updated by | Charlene Wang | Jan 4, 2023 at 12:39 AM PST

Contents

Architecture

Elastic jobs is an application that runs in Sterling tenant rings. A customer can create a job account inside a SQL logical server using ARM APIs. When creating a job account, the customer specifies a database that they own that the job account is linked to. This control database is used for storing job definitions and status.

The request to create a job account results in the creation of a job account state machine, which manages the lifecycle of the job account. The job account state machine will create an instance of the Jobs winfab application. This Jobs application is the core of elastic jobs - it reads job definitions from the control database, runs the jobs, and writes status to the control database.

Customers can manage jobs using ARM APIs, which are serviced via Node Agent executing stored procedures in the control database. Customers can also directly connect to the control database and call these same stored procedures. This means that elastic jobs has both an ARM API and a T-SQL API.

Sterling components

There are multiple sterling components involved for elastic jobs.

- Resource provider
  - Receives web requests from ARM and passes them down to workflows layer. Basic input format validation only.
- Workflows
  - Fully validates requests and hands them to FSM/NodeAgent.
- Finite State Machine (FSM)
  - Handles creation/deletion of winfab resources (i.e. application instance) of a job account.
- NodeAgent
  - Lives in tenant ring on the same node as the job control database. Connects and executes commands on job control database.

Core design (Internal only)

Entities

Entity are code level concepts. Below entities are designed to implement the functionality of elastic jobs which are not customer sharable. Customer should follow out public documenations https://learn.microsoft.com/en-us/azure/azure-sql/database/elastic-jobs-overview?view=azuresql to configure/manage/use elastic jobs.

- Job
  - Contains a versioned list of job step definitions.
  - A schedule can be specified.
- Job step (not shown in diagram)
  - An action (T-SQL script) to be executed on a list of targets (databases) with specified credentials.
- Job execution
  - An instance of a running job.
  - The job version is specified, so if the job is updated while the job execution is running, the job execution is unaffected (the original version is used).
  - A child job execution is created for each step.
  - After a step has resolved its list of targets, a grandchild job execution is created for each target.
  - Has states including Created, InProgress, WaitingForChildJobExecutions, Succeeded, Failed, TimedOut, Canceled.
- Job task execution
  - Frequently referred to as "task" when discussing entities (not to be confused with .NET System.Threading.Task type)
  - An attempt to execute a unit of work in a job execution.
  - Not visible to customers.
  - If the in-memory knowledge of job task executions (in communicator queue and in job task host workers) gets out of sync with the database, we can get stuck jobs. Database is the source of truth, so if we detect this we will shutdown and restart the entire core to bring in-memory state in sync with the database.
  - Has states including Created, InProgress, Succeeded, Failed, TimedOut, Canceled.
- Job cancellation
  - A request to cancel a job execution.

In-memory objects

- JobHosting
  - Reads settings provided by Program
  - Creates, monitors, and starts/stops all the below objects
- Maintainer
  - Creates system jobs. Updates system jobs if necessary (i.e. if somehow they were edited on-disk).
- Scheduler
  - Periodically queries jobs to see if it's time for a scheduled execution.
  - Creates job executions.
- Controller
  - Queries for actionable job executions.

        - If a job cancellation exists and a task is in progress, cancellation is signaled to the communicator.
        - If the job execution has no active tasks, determines what is the next task (which may be a re-attempt at the last task), and enqueues it into the communicator. If there is no next task, completes the job execution.
- Communicator
    - Facilitates communication between Controller and JobTaskHost
    - Has queue for job task executions
    - Has collection for CancellationTokens
- Job task host
    - Contains many workers who are listening to communicator queue.
    - A worker dequeues a task from the queue, and executes it (with cancellation token from communicator), and then writes the outcome to the database.

Topics covered

- Check the Limitations to see if customer's issue hit any known limitations.
- For creation elastic job stuck issues, check ElasticJobs Agent Creation Stuck for help.
- For elastic job agent timeout issue, check ElasticJobs_Job_Agent_Timeout for help.
- For elastic job agent unhealthy issue, check ElasticJobs_JobAgent_Unhealthy for help.

How good have you found this content?

🙂 🙁