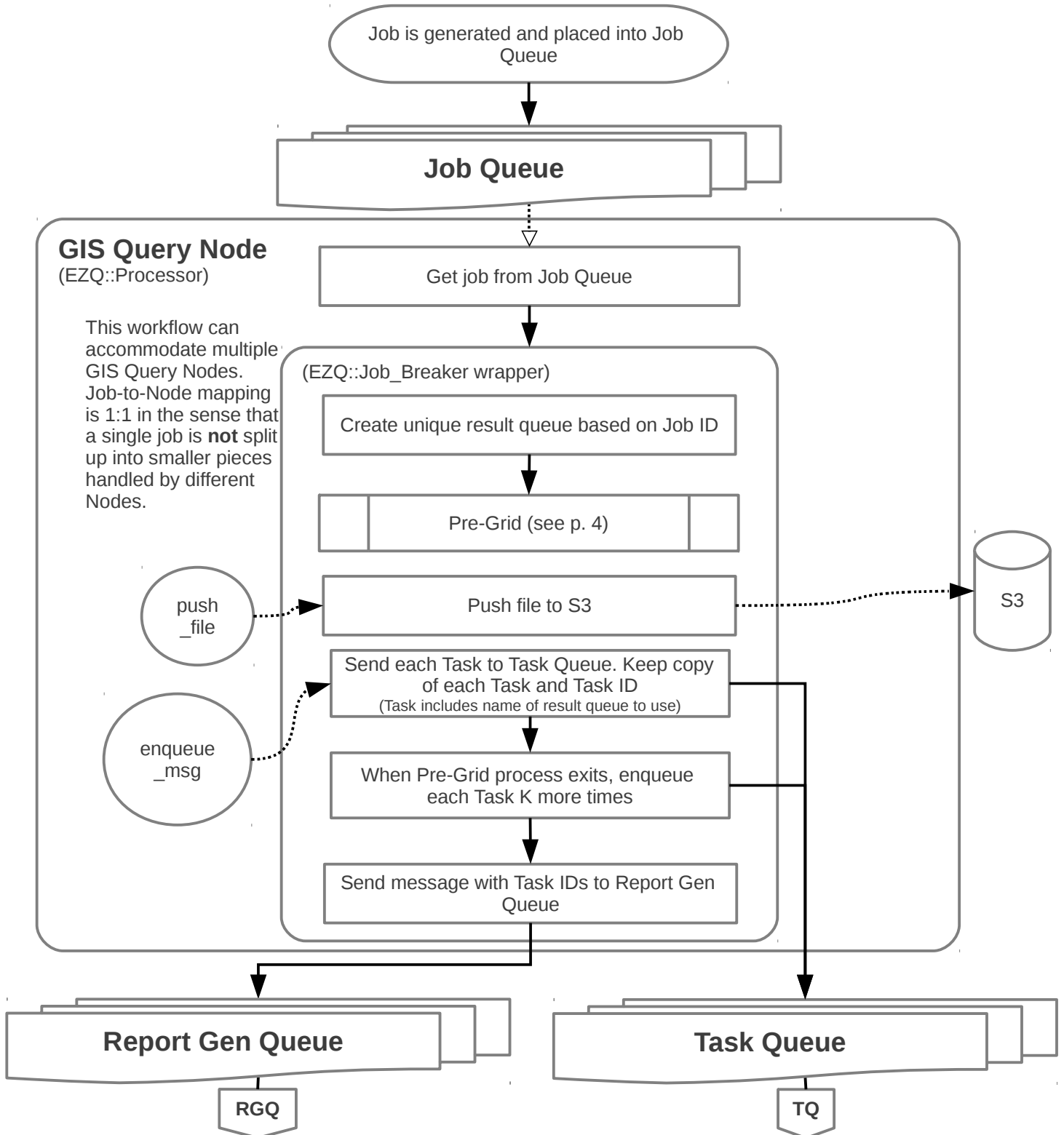


Modified workflow #1 for 6k worker case

This workflow replaces direct network connections with queues, and uses processes wrapped inside an EZQ::Processor to handle queued messages. The queuing provides task/job/etc. persistence in the case of node failures, as well as allowing easy scaling via running multiple instances of GIS Query Node and Report Handlers.

- A.....> B indicates B pulls data from A.
- A.....➡ B indicates A pushes data to B.
- A➡ B indicates flow of execution sequence from A to B.



Worker Manager

(Singleton Instance)

Inspect Task Queue size and start up workers as needed

This would request the approx. queue size every 60 (?) seconds, and start up an appropriate number of workers. This behavior could be accomplished without a dedicated Worker Manager node by using AutoScale on the Task Queue, but more complicated logic and tracking is possible with a dedicated node.

T
Q

Worker Nodes will run in batches of 44 nodes on a single EC2 instance.

Worker 0

(EZQ::Processor)

Get task from Task Queue

Process task

Send results to specified Results Queue

Worker N

(EZQ::Processor)

Get task from Task Queue

Process task

Send results to specified Results Queue

Results Queue A

Results Queue ZZZZZ

RQA

The results coming out of each worker which is working on the same Job will go to the same Results Queue. Task results for a different Job will go to a different Results Queue. This allows for multiple Jobs to be processed simultaneously, and for multiple Report Handlers to operate.

