Github Link: https://github.com/kadhikari9/url-processor

This Simple URL processor reads the URLs from a file and makes HTTP calls to those urls and also logs the failure and success while processing. This system has following components.

FileReader: File I/O Utility class which is used to read Urls from URL files

HttpService: Simple service to make HTTP GET call to URL

FileProcessingQueue: Bounded LinkedBlockedQueue, A thread-safe FIFO data structure provided by JDK that will block the incoming thread if it tries to fetch when queue is empty or add when queue is full. This queue will store URL files to be processed by FileProcessor. This data structure helps to synchronize the speed at which File Processor and URL processor work together.

URLProcessingQueue: Another bounded LinkedBlockedQueue that stores the URLs to be processed by URLProcessor.

DeadLetterQueue: A syncronized LinkedHashMap customized to use as LRU Cache. It is used to temporarily store the failed URLs for re-processing. It will store URL and processing attempts of URL in LRU Cache. It is bounded by size because if there are too many failures, it will discard old failures while alerting developer to look into issue.

AsyncDirectoryProcessor: This will poll the directory for new incoming URL files, it will then put the file names for processing into the FileProcessingQueue.

AsyncFileProcessor: This will take the URL file names from FileProcessingQueue, read the actual file from disk and put the URLs in the file to URLProcessingQueue. It will also move the file into another directory (processed) once it finish reading all contents of file so that same file is not read twice.

UrlProcessor: This is the actual URL processor which will poll the URLProcessingQueue, make the HTTP GET call to URL using HttpService and Publish the URL processing events to EventReporter. If there is failure on processing URL it will add failed URLs to DeadLetterQueue incrementing the retry attempts. It will also remove URL from DeadLetterQueue if processing succeeded.

DeadLetterProcessor: The purpose of this processor is to handle failures. If for some reason there is Error on HTTP call, UrlProcessor will put the failed URL to DeadLetterQueue. This processor will periodically check the queue and add back those URL to URLProcessingQueue for retry. If it has already tried maximum amount of times, it will discard the url and log the failure events.

EventReporter: EventReporter is responsible for logging the URL processing events. For every URL processing success and failure, UrlProcessor sends events to EventReporter, It will accumulate the failure, success count and periodically logs total success and failures on that interval. Default reporting interval is 60 seconds.

All the services run on separate thread and there is scheduler for each service that will periodically run the process again. By default, AsyncDirectoryProcessor runs every 2 minutes, AsyncFileProcessor every minute, EventReporter also every minute, DeadLetterProcessor every 10 secs and UrlProcessor is

continuously running. All these parameters can be configured changing properties on application.properties. The block diagram of the system is given below. Each service use `CompletableFuture` to run the process using it's worker thread.

Block Diagram