An application called GrepTheWeb, discussed in [360], is now in production at Amazon. We use it to

illustrate the power and appeal of cloud computing. The application allows a user to define a regular

expression and search the Web for records that match it. GrepTheWeb is analogous to the Unix grep

command used to search a file for a given regular expression.

This application performs a search of a very large set of records, attempting to identify records that

satisfy a regular expression. The source of this search is a collection of document URLs produced by

the Alexa Web Search, a software system that crawls theWeb every night. The inputs to the applications

are a regular expression and the large data set produced by theWeb-crawling software; the output is the

set of records that satisfy the expression. The user is able to interact with the application and get the

current status [see Figure 4.7(a)].

The application usesmessage passing to trigger the activities ofmultiple controller threads that launch

the application, initiate processing, shut down the system, and create billing records. GrepTheWeb uses

Hadoop MapReduce, an open-source software package that splits a large data set into chunks, distributes

them acrossmultiple systems, launches the processing, and, when the processing is complete, aggregates

the outputs from different systems into a final result. Apache Hadoop is a software library for distributed

processing of large data sets across clusters of computers using a simple programming model.

The details of the workflow of GrepTheWeb are captured in Figure 4.7(b) and consist of the following

steps [360]:

1. The startup phase. Creates several queues – launch, monitor, billing, and shutdown queues. Starts

the corresponding controller threads. Each thread periodically polls its input queue and, when a

message is available, retrieves the message, parses it, and takes the required actions.

2. The processing phase. This phase is triggered by a StartGrep user request; then a launch message

is enqueued in the launch queue. The launch controller thread picks up the message and executes the

launch task; then, it updates the status and time stamps in the Amazon Simple DB domain. Finally,

it enqueues a message in the monitor queue and deletes the message from the launch queue. The

processing phase consists of the following steps:

a. The launch task starts Amazon EC2 instances. It uses a Java Runtime Environment preinstalled

Amazon Machine Image (AMI), deploys required Hadoop libraries, and starts a Hadoop Job

(run Map/Reduce tasks).

b. Hadoop runs map tasks on Amazon EC2 slave nodes in parallel. A map task takes files from

Amazon S3, runs a regular expression, and writes the match results locally, along with a description

of up to five matches. Then the combine/reduce task combines and sorts the results and

consolidates the output.

c. Final results are stored on Amazon S3 in the output bucket.

3. The monitoring phase. The monitor controller thread retrieves the message left at the beginning of

the processing phase, validates the status/error in Amazon Simple DB, and executes themonitor task.

It updates the status in the Amazon Simple DB domain and enqueues messages in the shutdown and

billing queues. The monitor task checks for the Hadoop status periodically and updates the Simple

DB items with status/error and the Amazon S3 output file. Finally, it deletes the message from the

monitor queue when the processing is completed.

4. The shutdown phase. The shutdown controller thread retrieves the message from the shutdown queue

and executes the shutdown task, which updates the status and time stamps in the Amazon Simple DB

domain. Finally, it deletes the message from the shutdown queue after processing. The shutdown

phase consists of the following steps:

a. The shutdown task kills the Hadoop processes, terminates the EC2 instances after getting EC2

topology information from Amazon Simple DB, and disposes of the infrastructure.

b. The billing task gets the EC2 topology information, Simple DB usage, and S3 file and query

input, calculates the charges, and passes the information to the billing service.

5. The cleanup phase. Archives the Simple DB data with user info.

6. User interactions with the system. Get the status and output results. The GetStatus is applied to the

service endpoint to get the status of the overall system (all controllers and Hadoop) and download

the filtered results from Amazon S3 after completion.

To optimize the end-to-end transfer rates in the S3 storage system, multiple files are bundled up and

stored as S3 objects. Another performance optimization is to run a script and sort the keys and the URL

pointers and upload them in sorted order to S3. In addition, multiple fetch threads are started in order

to fetch the objects.

This application illustrates the means to create an on-demand infrastructure and run it on a massively

distributed system in a manner that allows it to run in parallel and scale up and down based on the

number of users and the problem size.