How to run

Prerequisites

This project depends on and was tested with the following Node.js libraries:

- fivebeans version 1.3.1
- js-yaml version 3.3.1
- mongodb version 2.0.39
- yargs version 3.15.0

Before you run anything

Frist, configure the connection to beanstalk and mongo DB in the configuration file (**config.yml**) and place the configuration file in the main directory of the project. All parameters for job management (such as the maximum number a job is allowed to run before it fails or the delay that it incurs in the tube as a result) can be configured in **config.yml**. For all parameters, please, see **config.yml**.

```
beanstalkd:
# IP and port of beanstalkd server are used to open a plain TCP connection to server
    host: "192.168.1.11"
    port: 11301
handlers:
# the class that handles the jobs from the tube (no need to change this)
    - "./lib/quote_retriever_job.js"
watch:
# name of tube.
    - 'ivanRachevTube'
ignoreDefault: true
job:
# the class that handles jobs looks for this type of jobs
    type: 'quoteRetrieverJob'
    priority: 1000
    timeToRun: 20
    delayOnFailure: 3
    delayOnSuccess: 60
    maxSuccessfulRuns: 10
    maxFailedRuns: 3
mongodb:
    connectionUrl: "mongodb://username:password@domain.com:port/dbname?uri.ssl=true"
    collection: "mycoll"
feeder:
    url: "http://themoneyconverter.com/HKD/USD.aspx"
```

Running the project

You can run the following modules:

- Consumer worker: run **consumer_worker.js** in the main directory. To run consumer_worker.js in multiple instances of Node.js, specify an ID for each instance as follows (see appendix for sample output):

```
> node consumer_worker.js --id=worker2
```

  This allows the tube to know who is watching it. If running worker from different computers, the ID of each worker can be the IP of the computer.

- Producer worker: in order to put an initial seed into the tube, run **producer_worker.js** in the main directory.

```
> node  producer_worker.js
```

- Show tube status: run **list_tube_info.js** in the main directory.

```
> node  list_tube_info.js
```

NOTE:

If running any of the above modules from a directory other than the main directory, you need to pass the configuration file to the module as follows:

```
> node /dev/quoter/producer_worker.js --config=/dev/quoter/config.yml
```

Design

Introduction

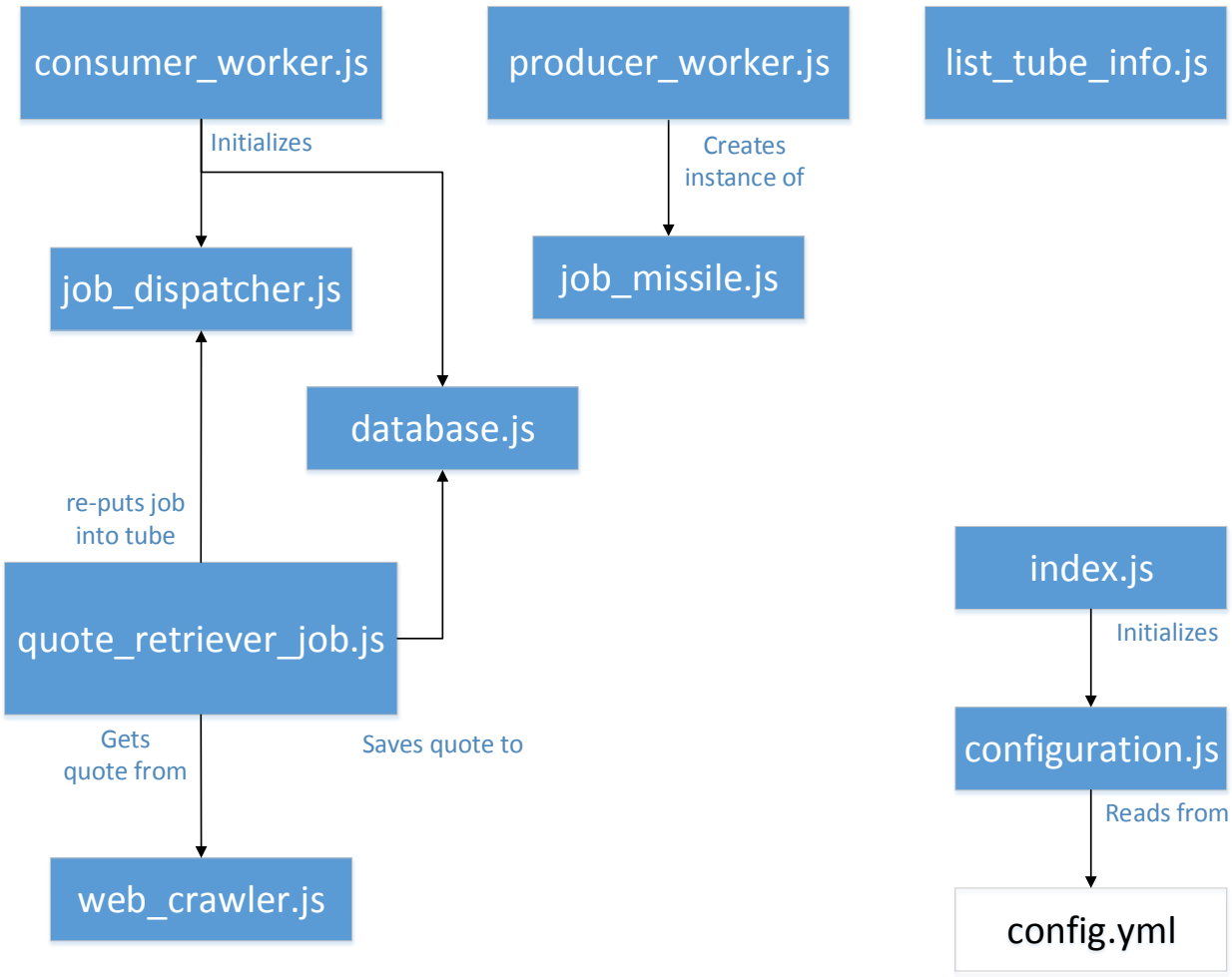
This is my first JavaScript project, using MongoDb for the first time, and using beanstalkd for the first time, and Node.js. I guess, I have a few firsts here. So, it was exiting.

Here are a few design considerations:

1. Fivebeans lib performs asynchronous execution of **asynchronous** jobs (emphasis on asynchronous). So, Fivebeans can be executing many jobs at the same time within the same Node.js instance. And our quote job is asynchronous by nature since it issues requests to the web (which are asynchronous themselves).
2. Database connection pool keeps 5 active connections to the database.
3. Javascript closures are used to avoid sharing data between asynchronous jobs executing in parallel.
4. I know Java object oriented design so I tried to follow the Java object oriented design patters. I recognize that there is an entirely different set of unique design patters for Javascript. I tried a few and I am learning …
5. Plain TCP is used for transport to beanstalkd. I couldn't get the HTTP POST to Aftership working. See section at end. Maybe I need help here.
6. Quotes are retrieved from http://themoneyconverter.com using an HTTP GET. I do not have a key for XE.com
7. Error handling is implemented for the web component that retrieves the quotes and for the database component partially. Losing connection to beanstalk will yield undefined behavior.
8. Testing: I performed manual testing with 3 jobs in one Node.js and with 2 jobs in two Node.js instances. I could write unit tests as a next step but it will be another first for me in Javascript.
9. I left some @todo-s that are still outstanding. Following up on these will improve design and error handling.

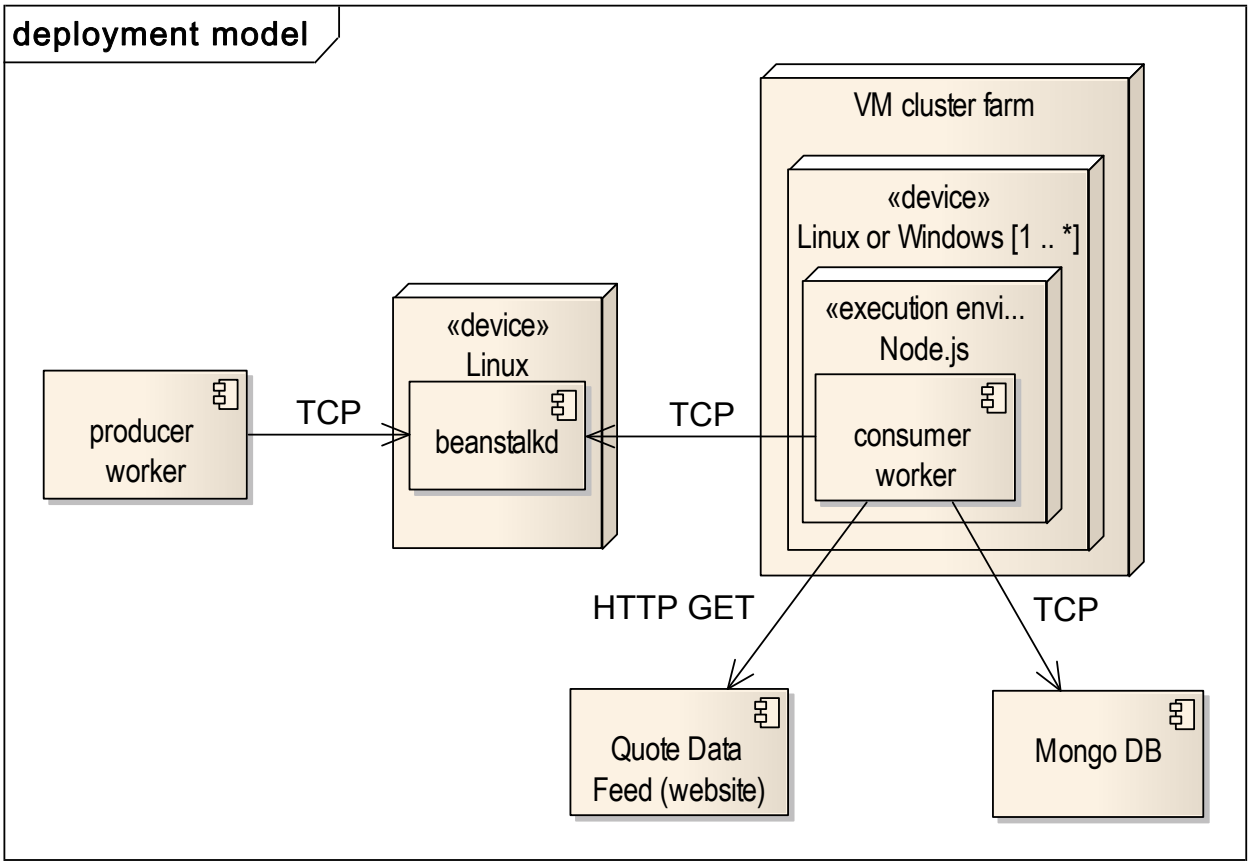
Package Diagram

Below is a package diagram showing the relationship between files in the project. Consumer_worker.js producer_worker.js and list_tube_info.js are the main files that initiate execution of the relevant module. Consumer_worker.js initializes connections to the database and to beanstalkd tube. Quote_retriever_job.js manages the state of the job, calling web_crawler.js to get the quote, calling database.js to save the quote and calling job_dispatcher.js to save the job back to the tube. producer_worker.js creates job_missile and puts into the tube. Index.js and configuration.js are used by all the rest of the modules.



Deployment Diagram

Below is a specification-level deployment diagram. It shows that the consumer worker can run on multiple environments within a cluster.



Problem connecting to Aftership API

I could not receive an OK message when putting jobs into Aftership's beanstalkd. I used the code snippet below and I receive the following response. What should I change so I can use Aftership's beanstalkd?

```
>echo output from Aftership
>node aftership.beanstalk.js
{"meta":{"code":200},"data":{"host":"challenge.aftership.net","port":11300}}
{"meta":{"code":200},"data":{"host":"challenge.aftership.net","port":11300}}
{"meta":{"code":200},"data":{"host":"challenge.aftership.net","port":11300}}
```

```
// code snippet from aftership.beanstalk.js
var http = require('http');
function performRequest(command, data) {

	var dataString;
	if (data)
		dataString = command + data.length  + '\r\n' + JSON.stringify(data);
	else
		dataString = command + '\r\n';

	var options = {
		hostname: 'challenge.aftership.net'
		, port: 9578 //11300
		, path: '/v1/beanstalkd'
		, method: 'POST'
		, headers: {
			'Content-Type': 'application/json'
			, 'aftership-api-key': 'a6403a2b-af21-47c5-aab5-a2420d20bbec'
			, 'Content-Length': dataString.length
		}
	};
	var req = http.request(options, function(res) {

		res.setEncoding('utf-8');
		var responseString = '';

		res.on('data', function(data) {
			responseString += data;
		});
		res.on('end', function() {
			console.log(responseString);
		});
	});
	req.write(dataString);
	req.end();
}

performRequest('use ivanrachev');
performRequest('put 1000 0 1 ', {"from": "HKD", "to": "USD"});
performRequest('list-tubes');
```

Appendix

Below is sample output from processing two jobs in parallel with consumer worker in one Node.js. For the given run, the following configuration parameters were used:

```
job:
    delayOnFailure: 3
    delayOnSuccess: 60
    maxSuccessfulRuns: 10
    maxFailedRuns: 3
```

Here is the command to start the consumer worker:

```
> node consumer_worker.js --id=worker2
```

And here is the output:

```
yaml config file /dev/quoter/config.yml
{"beanstalkd":{"host":"192.168.1.11","port":11301},"handlers":["./lib/quote_retrieve
r_job.js"],"watch":["ivanRachevTube"],"ignoreDefault":true,"job":{"type":"quoteRetri
everJob","priority":1000,"timeToRun":20,"delayOnFailure":2,"delayOnSuccess":6,"maxSu
ccessfulRuns":2,"maxFailedRuns":1},"mongodb":{"connectionUrl":"mongodb://url/ivan?ur
i.ssl=true","collection":"mycoll"},"feeder":{"url":"http://themoneyconverter.com/HKD
/USD.aspx"},"tubeName":"ivanRachevTube"}
Connected to mongo db server @
mongodb://url/ivan?uri.ssl=true
JobDispatcher connected
JobDispatcher using tube:ivanRachevTube
listening for new jobs
consumer info
{"clientid":"worker2","message":"connected to beanstalkd at 192.168.1.11:11301"}
consumer info
{"clientid":"worker2","message":"watching tube ivanRachevTube"}
consumer info
{"clientid":"worker2","message":"ignoring tube default"}
consumer worker started
QuoteRetrieverHandler payload:
{"UID":1438062975933,"from":"HKD","to":"USD","successfulRunsCount":0,"failedRunsCoun
t":0}
Got response for http://themoneyconverter.com/HKD/USD.aspx code:200
BODY length: 4779 UID: 1438062976667
BODY length: 8820 UID: 1438062976667
BODY length: 3780 UID: 1438062976667
BODY length: 12600 UID: 1438062976667
BODY length: 5040 UID: 1438062976667
QuoteRetrieverHandler payload:
{"UID":1438062977104,"from":"HKD","to":"USD","successfulRunsCount":0,"failedRunsCoun
t":0}
```

```
BODY length: 5040 UID: 1438062976667
BODY length: 5040 UID: 1438062976667
BODY length: 5040 UID: 1438062976667
BODY length: 5040 UID: 1438062976667
BODY length: 3922 UID: 1438062976667
responseBody length: 59093 UID: 1438062976667
The whole webpage received!
The new quote is 0.13
Inserted one quote into mycoll collection. Quote : 0.13
queued a Quote job into tube:ivanRachevTube jobid: 176 with delay: 6
reput successful job:
{"type":"quoteRetrieverJob","payload":{"UID":1438062975933,"from":"HKD","to":"USD","
successfulRunsCount":1,"failedRunsCount":0}}
Got response for http://themoneyconverter.com/HKD/USD.aspx code:200
BODY length: 4779 UID: 1438062977589
BODY length: 8820 UID: 1438062977589
BODY length: 6300 UID: 1438062977589
BODY length: 5040 UID: 1438062977589
BODY length: 3780 UID: 1438062977589
BODY length: 1260 UID: 1438062977589
BODY length: 3780 UID: 1438062977589
BODY length: 1260 UID: 1438062977589
BODY length: 12600 UID: 1438062977589
BODY length: 8820 UID: 1438062977589
BODY length: 2662 UID: 1438062977589
responseBody length: 59093 UID: 1438062977589
The whole webpage received!
The new quote is 0.13
Inserted one quote into mycoll collection. Quote : 0.13
queued a Quote job into tube:ivanRachevTube jobid: 177 with delay: 6
reput successful job:
{"type":"quoteRetrieverJob","payload":{"UID":1438062977104,"from":"HKD","to":"USD","
successfulRunsCount":1,"failedRunsCount":0}}
QuoteRetrieverHandler payload:
{"UID":1438062975933,"from":"HKD","to":"USD","successfulRunsCount":1,"failedRunsCoun
t":0}
Got response for http://themoneyconverter.com/HKD/USD.aspx code:200
BODY length: 4779 UID: 1438062984026
BODY length: 8820 UID: 1438062984026
BODY length: 21420 UID: 1438062984026
QuoteRetrieverHandler payload:
{"UID":1438062977104,"from":"HKD","to":"USD","successfulRunsCount":1,"failedRunsCoun
t":0}
BODY length: 3780 UID: 1438062984026
BODY length: 13860 UID: 1438062984026
BODY length: 6442 UID: 1438062984026
responseBody length: 59093 UID: 1438062984026
The whole webpage received!
The new quote is 0.13
Inserted one quote into mycoll collection. Quote : 0.13
Configuration.app.job.maxSuccessfulRuns (2) reached. So, job deleted. Deleted
payload:
{"UID":1438062975933,"from":"HKD","to":"USD","successfulRunsCount":2,"failedRunsCoun
```

```
t":0}
Got response for http://themoneyconverter.com/HKD/USD.aspx code:200
BODY length: 4779 UID: 1438062984948
BODY length: 8820 UID: 1438062984948
BODY length: 8820 UID: 1438062984948
BODY length: 12600 UID: 1438062984948
BODY length: 15120 UID: 1438062984948
BODY length: 5040 UID: 1438062984948
BODY length: 2520 UID: 1438062984948
BODY length: 1402 UID: 1438062984948
responseBody length: 59093 UID: 1438062984948
The whole webpage received!
The new quote is 0.13
Inserted one quote into mycoll collection. Quote : 0.13
Configuration.app.job.maxSuccessfulRuns (2) reached. So, job deleted. Deleted payload:
{"UID":1438062977104,"from":"HKD","to":"USD","successfulRunsCount":2,"failedRunsCoun
t":0}
```

References

https://raw.githubusercontent.com/kr/beanstalkd/master/doc/protocol.txt

http://www.maritimejournal.com/__data/assets/pdf_file/0020/1033940/Javascript-The-Good-Parts.pdf

http://javascript.info/tutorial/closures

http://stackoverflow.com/questions/111102/how-do-javascript-closures-work

http://www.sitepoint.com/understanding-module-exports-exports-node-js/