Migrating AI-infused chat to Kubernetes

Steven Ware Jones: Lead Architect, Senior Software Engineer, Messaging and AI, IBM
Nicholas Fong: Full Stack Engineer, Messaging and AI, IBM
Inbound communication channels on IBM.com

Chat with a chatbot
Build and maintain a large set of intent and entity data, train models for natural language classification

Chat with an agent
Enterprise chat-routing across geographies and languages, integrating different chat vendors within a single experience

Book a consultation
Allow customers to schedule time with IBMers based on expertise, scheduling preference, and customer’s interest

Traditional contact
Manage additional email, phone, and informational links to show users, configured centrally across tens of thousands of pages
Existing Chat Bots

You are now chatting with a Virtual Agent

Hi, I'm a chat bot, trained to help answer questions about IBM Blockchain. How can I help you?

how do i get started with block chain

The best way to get started with IBM Blockchain is the Hyperledger Composer playground.

Go to the sandbox
Start modeling now
IBM Cloud Trial Registration
Email a sales rep

Was this helpful? 😊😊
AI-Infused Boost

Document Discovery Watson Discovery

Product, Topic Classification Watson NLC

Entity, Concept Extraction Watson NLU

Knowledge Center APIs Support Center APIs

Document Re-Ranking

Language Classifiers

Dialog Watson Assistant

Existing Chat Bot

Leveraged Services

Additional Value Add
Chatbot architecture
kubernetes
Configuring Kubernetes

- Dockerize our application
- Set up our pods
- Networking
- Set up volumes for our model files
- Performance/Scaling
- Worker hardware requirements
Traditional Linux containers vs. Docker

LXC

DOCKER

APP

PROCESS

PROCESS

PROCESS

LIB

LIB

OPERATING SYSTEM

OPERATING SYSTEM

DOCKER ENGINE
FROM node:10
COPY . /express-api
WORKDIR /express-api
RUN npm install
EXPOSE 5555
CMD [ "bash", "-c", ". run.sh" ]
Configuring Kubernetes

- Dockerize our application
- Set up our pods
- Networking
- Set up volumes for our model files
- Performance/Scaling
- Worker hardware requirements
Deployments

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fast-api-deployment
spec:
  selector:
    matchLabels:
      app: fast-api
  replicas: 2
  template:
    metadata:
      labels:
        app: fast-api
    spec:
      containers:
        - name: fast-api
          image: us.icr.io/ibm-chatbot/fast-api:1.0.0
          imagePullPolicy: Always
          ports:
            - containerPort: 8000
```

Configuring Kubernetes

- Dockerize our application
- Set up our pods
  - Networking
  - Set up volumes for our model files
  - Performance/Scaling
  - Worker hardware requirements
“Kubernetes pods are mortal. They are born and when they die, they are not resurrected.”

-- https://kubernetes.io/docs/concepts/services-networking/service/
ClusterIP service

```yaml
apiVersion: v1
kind: Service
metadata:
  name: express-api-service
spec:
  type: ClusterIP
  selector:
    app: express-api # pod to target
ports:
- port: 80
  protocol: TCP
  targetPort: 5555
  name: http
- port: 443
  protocol: TCP
  targetPort: 5555
  name: https

<serviceName>.<namespace>.svc.cluster.local:port
```
Ingress

apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: ingress-chatbot-api
  annotations:
    ingress.bluemix.net/rewrite-path: "serviceName=express-api-service rewrite=/;serviceName=fast-api-service rewrite="/"
spec:
tls:
  - hosts:
    - ibm-chatbot.us-south.containers.appdomain.cloud
    secretName: ibm-chatbot-secret
rules:
  - host: ibm-chatbot.containers.appdomain.cloud
    http:
      paths:
        - path: /express-api/
          backend:
            serviceName: express-api-service
            servicePort: 443
        - path: /fast-api/
          backend:
            serviceName: fast-api-service
            servicePort: 8000
Configuring Kubernetes

- Dockerize our application
- Set up our pods
- Networking
  - Set up volumes for our model files
  - Performance/Scaling
  - Worker hardware requirements
IBM Cloud
Object Storage
Cloud Object Storage

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: model-pvc
  annotations:
    ibm.io/auto-create-bucket: false
    ibm.io/auto-delete-bucket: false
    ibm.io/bucket: ibm-chatbot-model
    ibm.io/secret-name: model-pvc-secret
    ibm.io/endpoint: "https://s3.private.dal.us.cloud-object-storage.appdomain.cloud"
spec:
  accessModes:
    - ReadOnlyMany
  resources:
    requests:
      storage: 8Gi
  storageClassName: ibmc-s3fs-standard-cross-region
```
initContainer

template:
  metadata:
    labels:
      app: fast-api

spec:
  initContainers:
    - name: fast-api-init
      image: us.icr.io/ibm-chatbot/fast-api-init:1.0.0
      imagePullPolicy: Always
      volumeMounts:
        - mountPath: /files
          name: fast-api-data
  volumes:
    - name: fast-api-data
      emptyDir: {}  
  containers:
    - name: fast-api
      image: us.icr.io/ibm-chatbot/fast-api:1.0.0
      imagePullPolicy: Always
      volumeMounts:
        - mountPath: /files
          name: fast-api-data
      ports:
        - containerPort: 8000
Configuring Kubernetes

- Dockerize our application
- Set up our pods
- Networking
- Set up volumes for our model files

- Performance/Scaling
- Worker hardware requirements
Rolling updates

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: express-api-deployment
spec:
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 1
      maxSurge: 50%
  selector:
    matchLabels:
      app: express-api
  replicas: 2
  template:
    metadata:
      labels:
        app: express-api
    spec:
      containers:
        - name: express-api
          image: us.icr.io/ibm-chatbot/express-api:1.0.0
```
Health checks

- name: fast-api-data
  image: us.icr.io/umx-cognitive/sherlock:<version><env>
  imagePullPolicy: Always
  volumeMounts:
  - mountPath: /files
    name: fast-api-data
  ports:
  - containerPort: 8000

startupProbe:
  httpGet:
    path: /api/v1/status
    port: 8000
    # worst case is failureThreshold * periodSeconds
    failureThreshold: 30
    periodSeconds: 10
    timeoutSeconds: 1

livenessProbe:
  httpGet:
    path: /api/v1/status
    port: 8000
    periodSeconds: 10
    timeoutSeconds: 5
Load testing

Locust

```python
from locust import HttpLocust, TaskSet, task, between

class Status(TaskSet):
    @task(1)
    def status(self):
        self.client.get("/api/status")

class Route(TaskSet):
    @task(1)
    def route(self):
        payload = {
            "input": "hello",
            "context": { "product": "blockchain", "language": "en" }
        }
        self.client.post("/api/message", json=payload)

class botApi(HttpLocust):
    task_set = Route
    wait_time = between(10, 20)
    max_wait = 10
```

Artillery

```python
config:
    target: "https://ibm-chatbot.us-south.containers.appdomain.cloud"
    phases:
        - duration: 60
          arrivalRate: 1
        - duration: 60
          arrivalRate: 1
          rampTo: 20
        - duration: 120
          arrivalRate: 20
    scenarios:
        - flow:
            - post:
                url: "/express-api/api/message"
                json:
                    "input": "hello"
                    "context":
                        "product": "Blockchain"
                        "language": "en"
```
Performance profiling

cProfile

```python
python -m cProfile [-o output_file] [-s sort_order] myscript.py
197 function calls (192 primitive calls) in 0.002 seconds
Ordered by: standard name
ncalls  tottime  percall  cumtime  percall filename:lineno(function)
1      0.000    0.000    0.001    0.001 <string>:1(<module>)
1      0.000    0.000    0.001    0.001 re.py:212(compile)
1      0.000    0.000    0.001    0.001 re.py:268(_compile)
1      0.000    0.000    0.000    0.000 sre_compile.py:172(_compile)
1      0.000    0.000    0.000    0.000 sre_compile.py:201(_optimi:
4      0.000    0.000    0.000    0.000 sre_compile.py:25(_identit:
3/1    0.000    0.000    0.000    0.000 sre_compile.py:33(_compile
```

line_profiler

```python
python -m line_profiler script_to_profile.py.lprof
Pystone(1.1) time for 50000 passes = 2.48
This machine benchmarks at 2061.3 pystones/second
Wrote profile results to pystone.py.lprof
Timer unit: 1e-06 s

File: pystone.py
Function: Proc2 at line 149
Total time: 0.006656 s

<table>
<thead>
<tr>
<th>Line #</th>
<th>Hits</th>
<th>Time</th>
<th>Per Hit</th>
<th>% Time</th>
<th>Line Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>149</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>@profile</td>
</tr>
<tr>
<td>150</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>def Proc2(IntParIO):</td>
</tr>
<tr>
<td>151</td>
<td>50000</td>
<td>82003</td>
<td>1.6</td>
<td>13.5</td>
<td>IntLoc = IntParIO + 10</td>
</tr>
<tr>
<td>152</td>
<td>50000</td>
<td>63162</td>
<td>1.3</td>
<td>10.4</td>
<td>while 1:</td>
</tr>
<tr>
<td>153</td>
<td>50000</td>
<td>69065</td>
<td>1.4</td>
<td>11.4</td>
<td>if CharGlob == 'A':</td>
</tr>
<tr>
<td>154</td>
<td>50000</td>
<td>66354</td>
<td>1.3</td>
<td>10.9</td>
<td>IntLoc = IntLoc - 1</td>
</tr>
<tr>
<td>155</td>
<td>50000</td>
<td>67263</td>
<td>1.3</td>
<td>11.1</td>
<td>IntParIO = IntLoc - Int</td>
</tr>
<tr>
<td>156</td>
<td>50000</td>
<td>65494</td>
<td>1.3</td>
<td>10.8</td>
<td>EnumLoc = Ident1</td>
</tr>
<tr>
<td>157</td>
<td>50000</td>
<td>68001</td>
<td>1.4</td>
<td>11.2</td>
<td>if EnumLoc == Ident1:</td>
</tr>
<tr>
<td>158</td>
<td>50000</td>
<td>63739</td>
<td>1.3</td>
<td>10.5</td>
<td>break</td>
</tr>
<tr>
<td>159</td>
<td>50000</td>
<td>61575</td>
<td>1.2</td>
<td>10.1</td>
<td>return IntParIO</td>
</tr>
</tbody>
</table>
```
Pod Resources

```yaml
initContainers:
  - name: fast-api-init
    image: us.icr.io/ibm-chatbot/fast-api-init:1.0.0
    imagePullPolicy: Always
    volumeMounts:
      - mountPath: /files
        name: fast-api-data
    volumes:
      - name: fast-api-data
        emptyDir: {}

containers:
  - name: fast-api
    image: us.icr.io/ibm-chatbot/fast-api:1.0.0
    imagePullPolicy: Always
    volumeMounts:
      - mountPath: /files
        name: fast-api-data
    resources:
      requests:
        cpu: "500m"
        memory: "512Mi"
      limits:
        cpu: "1000m"
        memory: "1024Mi"

ports:
```

Pod scaling

Pod 1 → Pod 2 → Pod N

RC / Deployment

Scale

Horizontal Pod Autoscaler
Pod scaling

```yaml
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: fast-api-autoscaler
  namespace: default
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: fast-api
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: Resource
      resource:
        name: cpu
      target:
        type: Utilization
        averageUtilization: 50
```
Configuring Kubernetes

- Dockerize our application
- Set up our pods
- Networking
- Set up volumes for our model files
- Performance/Scaling

Worker hardware requirements
<table>
<thead>
<tr>
<th>Machine</th>
<th>2 vCPUs 4GB RAM</th>
<th>4 vCPUs 16GB RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Virtual - shared</td>
<td>Virtual - shared</td>
</tr>
<tr>
<td>Size</td>
<td>u2.xlarge</td>
<td>b3.8xlarge</td>
</tr>
<tr>
<td>RAM</td>
<td>64 GB</td>
<td>16 GB</td>
</tr>
<tr>
<td>Storage</td>
<td>256 GB SSD</td>
<td>256 GB SSD</td>
</tr>
<tr>
<td>Use cases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced Cores and RAM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra local storage for SSD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>Small (4)</td>
<td>Small (4)</td>
</tr>
<tr>
<td>Operating System</td>
<td>Ubuntu 16 (37)</td>
<td>Ubuntu 16 (37)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine</th>
<th>8 vCPUs 32GB RAM</th>
<th>8 vCPUs 64GB RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Virtual - shared</td>
<td>Virtual - shared</td>
</tr>
<tr>
<td>Size</td>
<td>b3.8xlarge</td>
<td>c3.8xlarge</td>
</tr>
<tr>
<td>RAM</td>
<td>64 GB</td>
<td>64 GB</td>
</tr>
<tr>
<td>Storage</td>
<td>256 GB SSD</td>
<td>256 GB SSD</td>
</tr>
<tr>
<td>Use cases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced Cores and RAM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra local storage for SSD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>Medium (36)</td>
<td>Medium (36)</td>
</tr>
<tr>
<td>Operating System</td>
<td>Ubuntu 16 (37)</td>
<td>Ubuntu 16 (37)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine</th>
<th>16 vCPUs 16GB RAM</th>
<th>32 vCPUs 64GB RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Virtual - shared</td>
<td>Virtual - shared</td>
</tr>
<tr>
<td>Size</td>
<td>c3.16xlarge</td>
<td>c3.32xlarge</td>
</tr>
<tr>
<td>RAM</td>
<td>64 GB</td>
<td>64 GB</td>
</tr>
<tr>
<td>Storage</td>
<td>256 GB SSD</td>
<td>256 GB SSD</td>
</tr>
<tr>
<td>Use cases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced Cores and RAM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra local storage for SSD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>Large (34)</td>
<td>Large (34)</td>
</tr>
<tr>
<td>Operating System</td>
<td>Ubuntu 16 (37)</td>
<td>Ubuntu 16 (37)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Machine</th>
<th>32 vCPUs 64GB RAM</th>
<th>16 vCPUs 128GB RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Virtual - shared</td>
<td>Virtual - shared</td>
</tr>
<tr>
<td>Size</td>
<td>c3.32xlarge</td>
<td>c3.16xlarge</td>
</tr>
<tr>
<td>RAM</td>
<td>64 GB</td>
<td>64 GB</td>
</tr>
<tr>
<td>Storage</td>
<td>256 GB SSD</td>
<td>256 GB SSD</td>
</tr>
<tr>
<td>Use cases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balanced Cores and RAM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extra local storage for SSD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data-intensive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>Large (34)</td>
<td>Large (34)</td>
</tr>
<tr>
<td>Operating System</td>
<td>Ubuntu 16 (37)</td>
<td>Ubuntu 16 (37)</td>
</tr>
</tbody>
</table>
Node Affinity

apiVersion: v1
kind: Pod
metadata:
  name: with-node-affinity
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: beta.kubernetes.io/instance-type
            operator: In
            values:
            - bc3.8x32.encrpted
      preferredDuringSchedulingIgnoredDuringExecution:
        - weight: 1
          preference:
            matchExpressions:
            - key: another-node-label-key
              operator: In
              values:
              - another-node-label-value
  containers:
    - name: with-node-affinity
      image: k8s.gcr.io/pause:2.0

<table>
<thead>
<tr>
<th>Labels</th>
<th>Ready</th>
<th>CPU requests (cores)</th>
<th>CPU limits (cores)</th>
</tr>
</thead>
<tbody>
<tr>
<td>arch: amd64</td>
<td>True</td>
<td>2.56 (32.00%)</td>
<td>5 (62.50%)</td>
</tr>
<tr>
<td>beta.kubernetes.io/instance-type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta.kubernetes.io/arch: amd64</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta.kubernetes.io/hostname: 10.0.0.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>failure-domain.beta.kubernetes.io</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/enclave-0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/ha</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/iaaserver</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/marketing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/os</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/signtool</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm-cloud.kubernetes.io/worker-wp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kubernetes.io/arch: amd64</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kubernetes.io/os: linux</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>privateVLAN: 2291501</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>publicVLAN: 2291499</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>show less</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pod Affinity

```yaml
matchLabels:
  - app: express-api

replicas: 2

template:
  metadata:
    labels:
      app: express-api

spec:
  affinity:
    PodAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        - podAffinityTerm:
            labelSelector:
              matchLabels:
                app: fast-api

containers:
  - name: express-api
    image: us.icr.io/ibm-chatbot/express-api:1.0.0
    imagePullPolicy: Always

    ports:
      - containerPort: 5555

    readinessProbe:
      httpGet:
        path: /api/v2/status
        port: 8000
```
Cluster scaling

**Worker Pool 1**
- 4x16 Worker
- 4x16 Worker
- 4x16 Worker
- 4x16 Worker

**Worker Pool 2**
- 16x128 GPU Worker
Single zone cluster

Region

Zone

Single zone cluster

Worker pool

Worker 1

Worker 2

Worker 3

Master

ALB
Three single zone clusters in separate zones within one region

Global load balancer

Region

Zone 1
Single zone cluster 1
Worker pool
Worker 1
Worker 2
Worker 3

Zone 2
Single zone cluster 2
Worker pool
ALB
Worker 1
Worker 2
Worker 3

Zone 3
Single zone cluster 3
Worker pool
Worker 1
Worker 2
Worker 3