Coffee Shop

Software Architecture

March 31, 2026

Brae Webb & Richard Thomas

1 Brief

This week we have introduced the concept of event-driven architectures. In this tutorial, we will explore and document a project which implements this type of architecture.

https://github.com/CSSE6400/scalable-coffee-shop

The project is an implementation of a coffee shop, implemented by Sebastian Daschner. The coffee shop exposes a HTTP API, which can be used to stock beans and place drink orders.

Our goal is to run the software locally, explore the implementation, and create C4 diagrams to document the software architecture.

2 Installation and Running

In the first 15 minutes of the tutorial, we will install the application. To install, we recommend that you use Docker Compose. The project has a `docker-compose.yml` configuration in the top-level folder.

```
$ git clone https://github.com/CSSE6400/scalable-coffee-shop
$ cd scalable-coffee-shop
$ docker-compose up
```

Ideally this command will execute without error and start servicing requests as shown in the examples below.

Info

The `docker-compose up` command produces a lot of interleaving output. If you need to debug why one particular service, e.g. the `beans` service, is not working, you can use the `logs` command to inspect it.

```
$ docker-compose logs -f beans
```

Add some coffee beans to the beans storage, and confirm they have been created.

```
$ curl http://localhost:8002/beans/resources/beans -i -XPOST \
    -H 'content-type: application/json' \
    -d '{"beanOrigin": "Colombia", "amount": 10}'

  HTTP/1.1 204 No Content
  X-Powered-By: Undertow/1
```

```
  Server: WildFly/10
  Date: Fri, 17 Nov 2017 20:49:26 GMT

$ curl http://localhost:8002/beans/resources/beans
{"Colombia":10}
```

Create a new coffee order, and confirm it has been created.

```
$ curl http://localhost:8001/orders/resources/orders/ -i -XPOST \
    -H 'Content-type: application/json' \
    -d '{"beanOrigin": "Colombia", "coffeeType": "Espresso"}'

  HTTP/1.1 202 Accepted
  Connection: keep-alive
  X-Powered-By: Undertow/1
  Server: WildFly/10
  Location: http://localhost:8001/orders/resources/orders/c099c158-b748-4115-bed3-7
     b5dfff70771
  Content-Length: 0
  Date: Fri, 17 Nov 2017 20:51:16 GMT

$ curl http://localhost:8001/orders/resources/orders/c099c158-b748-4115-bed3-7
   b5dfff70771
{"status":"started","type":"espresso","beanOrigin":"Colombia"}
```

3 Documentation

By now, you should be able to create simple C4 diagrams of a software architecture. You will use these skills to document the coffee shop application's architecture. You can use your favourite diagram software for this task. For the tutorial, you should produce:

1. A context diagram for the system.
2. A container diagram of the system.
3. A component diagram for at least the Kafka server.

For reference on how event-driven architectures should look in C4, please refer to the Event Driven Architecture handout [1].

Info

You do not need an in-depth understanding of Kafka to document the architecture. For a simple introduction to the concepts of Kafka see the Hortonworks documentation [2]. For a more in-depth look at how Kafka actually works, read the article by Stopford [3].

References

[1] R. Thomas, "Event-driven architecture," March 2023. https://csse6400.uqcloud.net/handouts/event.pdf.

[2] H. D. Platform, "Basic kafka concepts." https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.2.4/bk_storm-user-guide/content/basic-kafka-concepts.html.

[3] B. Stopford, "Apache kafka as an event-driven backbone for service architectures." https://www.confluent.io/blog/apache-kafka-for-service-architectures/, July 2017.