Apache Kafka 勉強会

Kafkaで実装するための前提知識2

土田晃司

2021/07/30

Kafkaで実装するための前提知識2

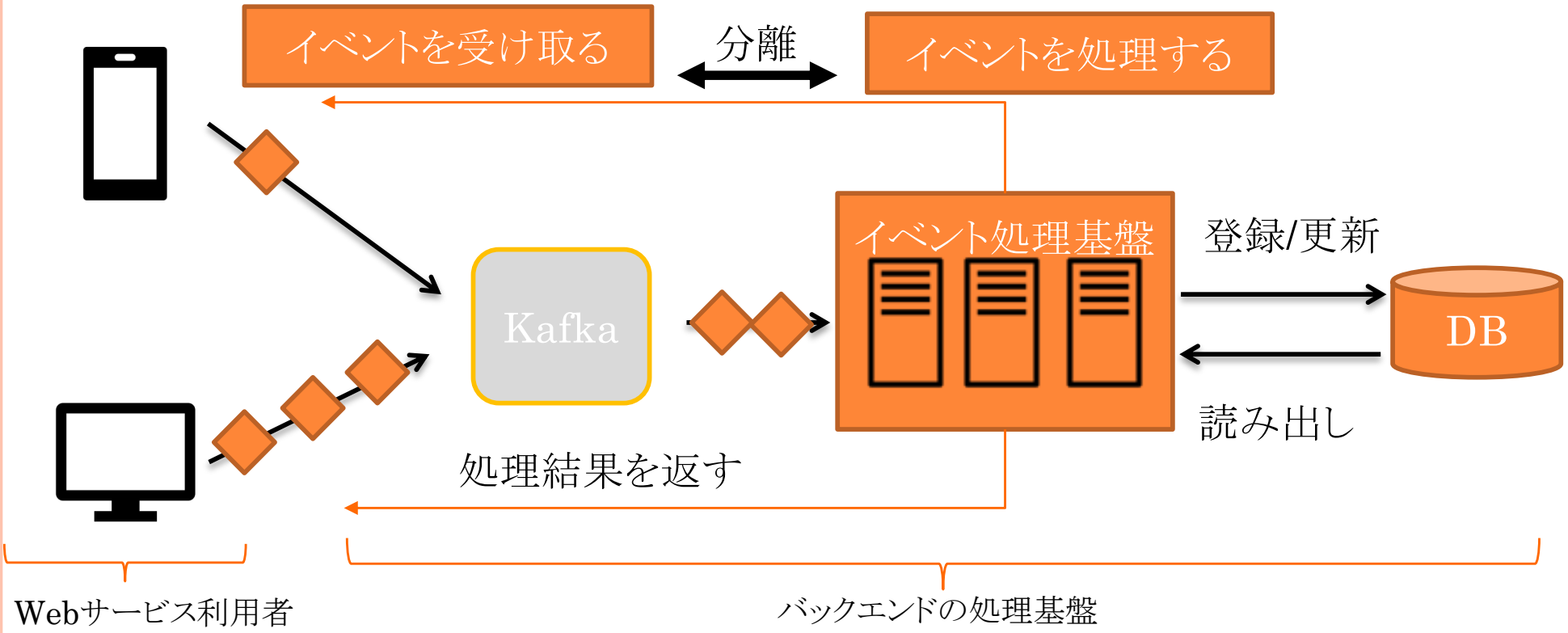
1. イベントソーシング
2. Kafkaを用いたデータフローの構成要素
3. Kafka Connect とは
4. Kafka Connectのユースケースでの実装
5. Kafka Streamとは

1. イベントソーシング＋CQRSとは

イベントソーシングとは、状態の変化をイベントとして扱い、発生したイベントを逐次記録してくものです。DBMSのトランザクションログのレコード書き込みと同じような考え方ができます。

CQRS(Command Query Responsiblitiy Segregation)とは、データの更新処理と参照を分けて処理をすること

1. イベントソーシング＋CQRSとは



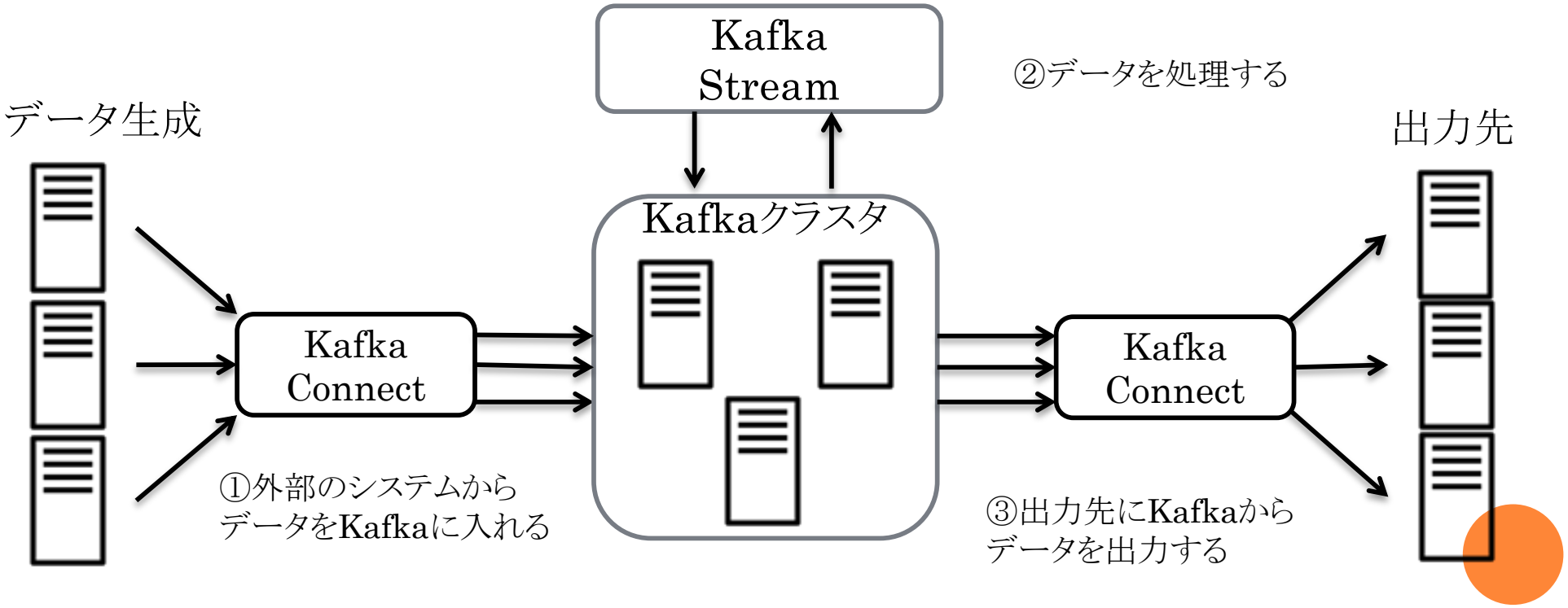
イベントの受け取りと実際の処理を分けることで汎用性の高いアクセスパターンに対応することができる

2．Kafkaを用いたデータフローの構成要素

デザインパターン Kafka ConnectとKafka Stream

データベースなどのデータストアを使用しないでシステムを構築することを目標にしています。

Kafka上のデータをSQLライクに処理できるKSQLを利用できます。



3. Kafka Connect とは

Kafka Connectとは、Kafkaと他のシステムとのデータ連携を行うためのフレームワーク

1. ProducerとConsumerの両方を構成することができる
2. 複数のサーバでクラスタを組むことができる
3. Brokerと同じサーバ上で動作することができる
4. RESTAPI形式で設定をこなうことができる

データハブアーキテクチャを組むことができる　例

4. Kafka Connectでの実装 ケース1　あとで実演

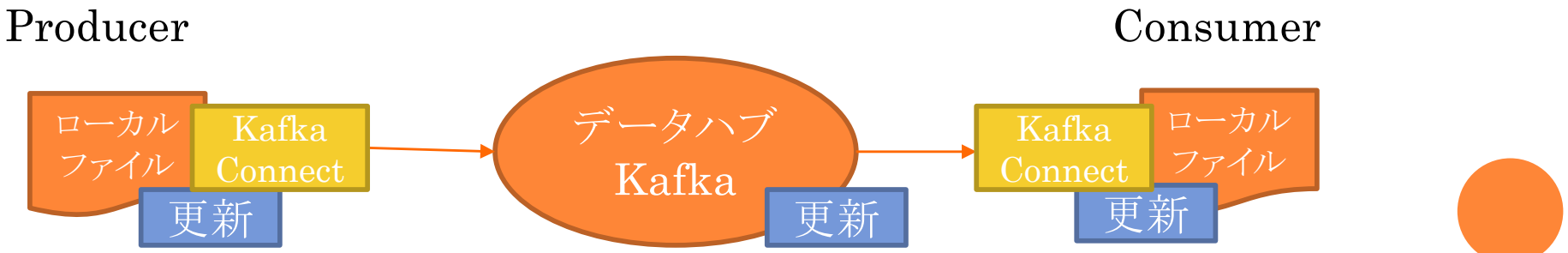
1.Kafka Connectを使用して、Producerとなるローカルファイルを指定し、KafkaのTopicへデータを送信するための設定を行う。

2.Kafka Connectを使用して、Consumerとなるローカルファイルを指定し、KafkaのTopicからデータを取得するための設定を行う。

3.Producerのローカルファイルにデータを追加する。

4.KafkaConnectにて、Kafkaにデータが送信される。

5.KafkaConnectにて、KafkaからConsumerのローカルファイルにデータが追加される。



4. Kafka Connectでの実装 ケース2　あとで実演

1.Kafka Connectを使用して、ProducerとなるMariaDBを指定し、KafkaのTopicへデータを送信するための設定を行う。

2.Kafka Connectを使用して、ProducerとなるPorstgresを指定し、KafkaのTopicへデータを送信するための設定を行う。

3.Kafka Connectを使用して、ConsumerとなるAWS S3のローカルファイルを指定し、KafkaのTopicからデータを取得するための設定を行う。

4.ProducerのMariaDBにデータを追加する。

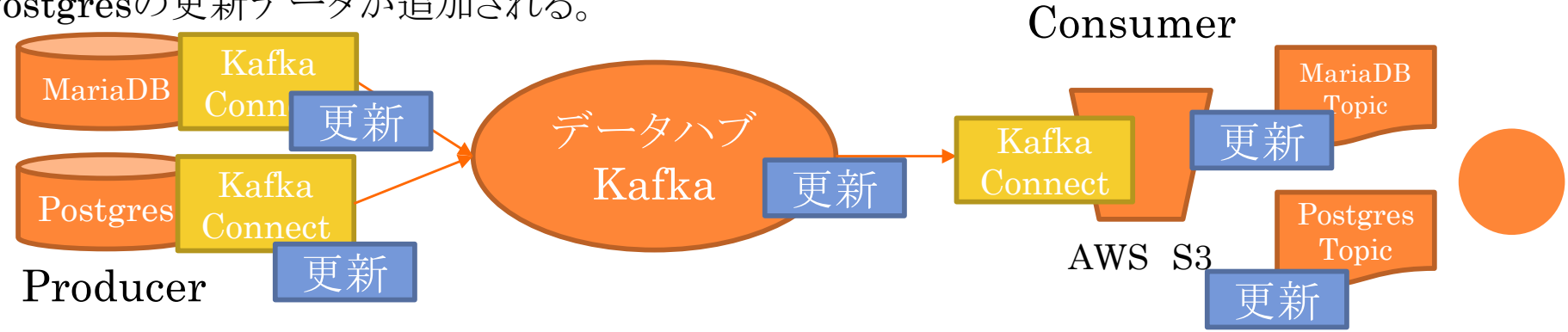
5.Kafka Connectにて、MariaDBの更新データがKafkaにデータが送信される。

6.Kafka Connectにて、KafkaからConsumerのAWS S3のローカルファイルにMariaDBの更新でが追加される。

7.ProducerのPorstgresにデータを追加する。

8.Kafka Connectにて、Postgresの更新データがKafkaにデータが送信される。

9.Kafka Connectにて、KafkaからConsumerのAWS S3のローカルファイルにPostgresの更新データが追加される。



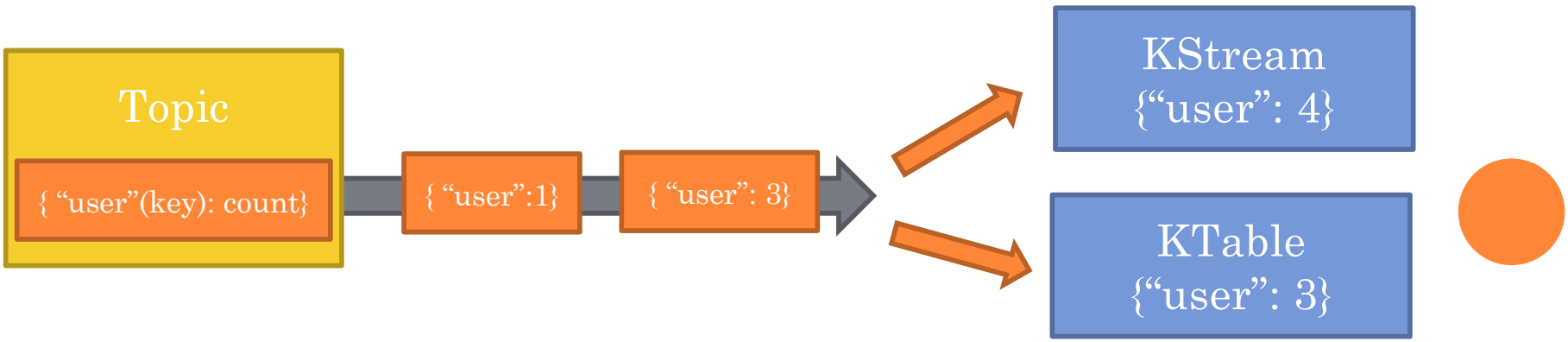
5. Kafka Streamとは

Kafka Streamとは、Kafkaで分散ストリーム処理を記述するためのフレームワーク

1. ProducerとConsumerの両方を構成することができる
2. Topicからデータを取得して、別のTopicにデータを送信できる
3. Kafkaに流れるStream(key-value形式)には「KStream」「KTable」という2つのStreamタイプがある

   KStreamはキーに対するデータを加算する（record stream）

   KTableはキーに対するデータを更新する（changelog stream）



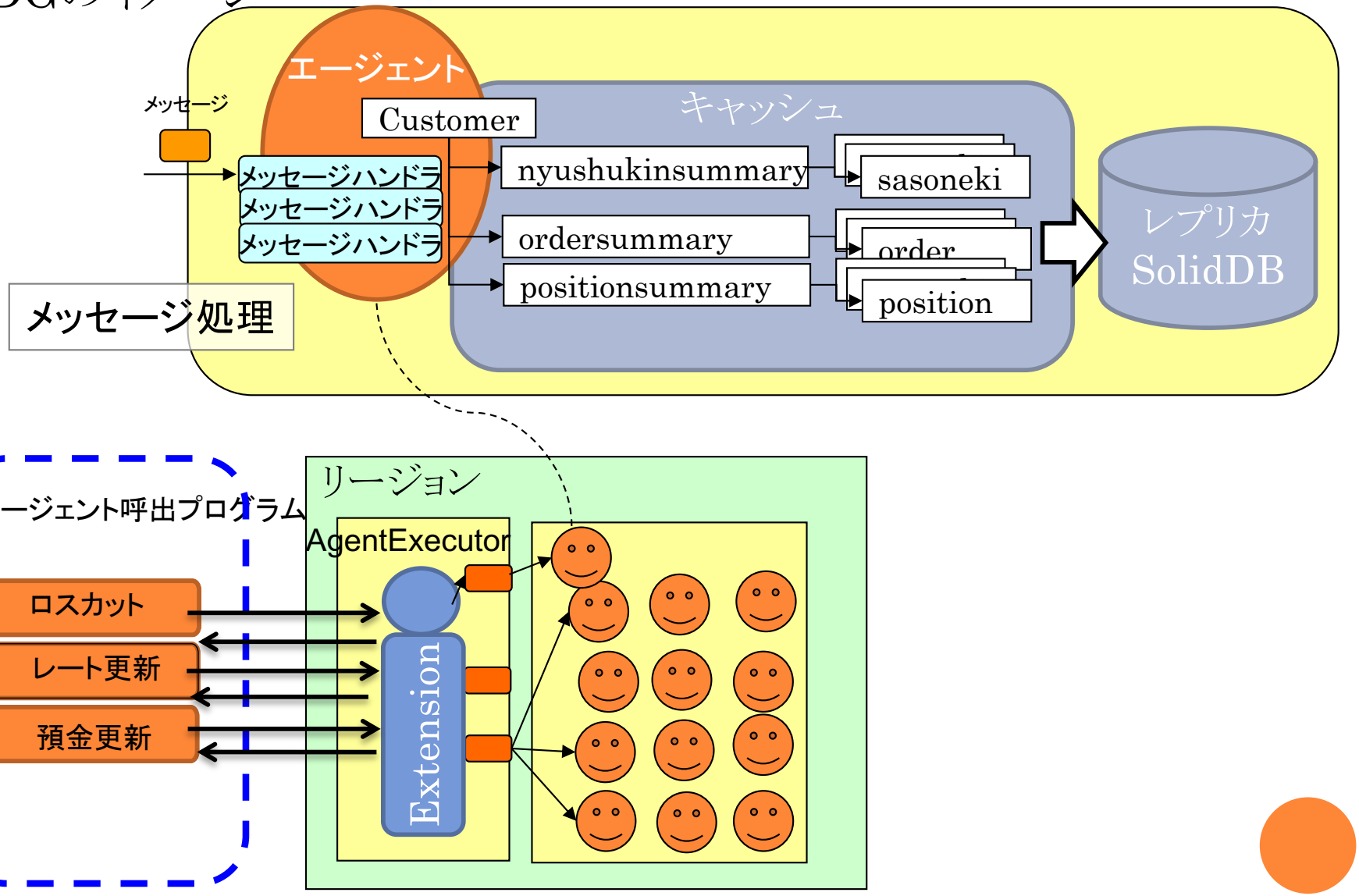
補足資料 AFDGとKafkaの違い

AFDGのエージェントは

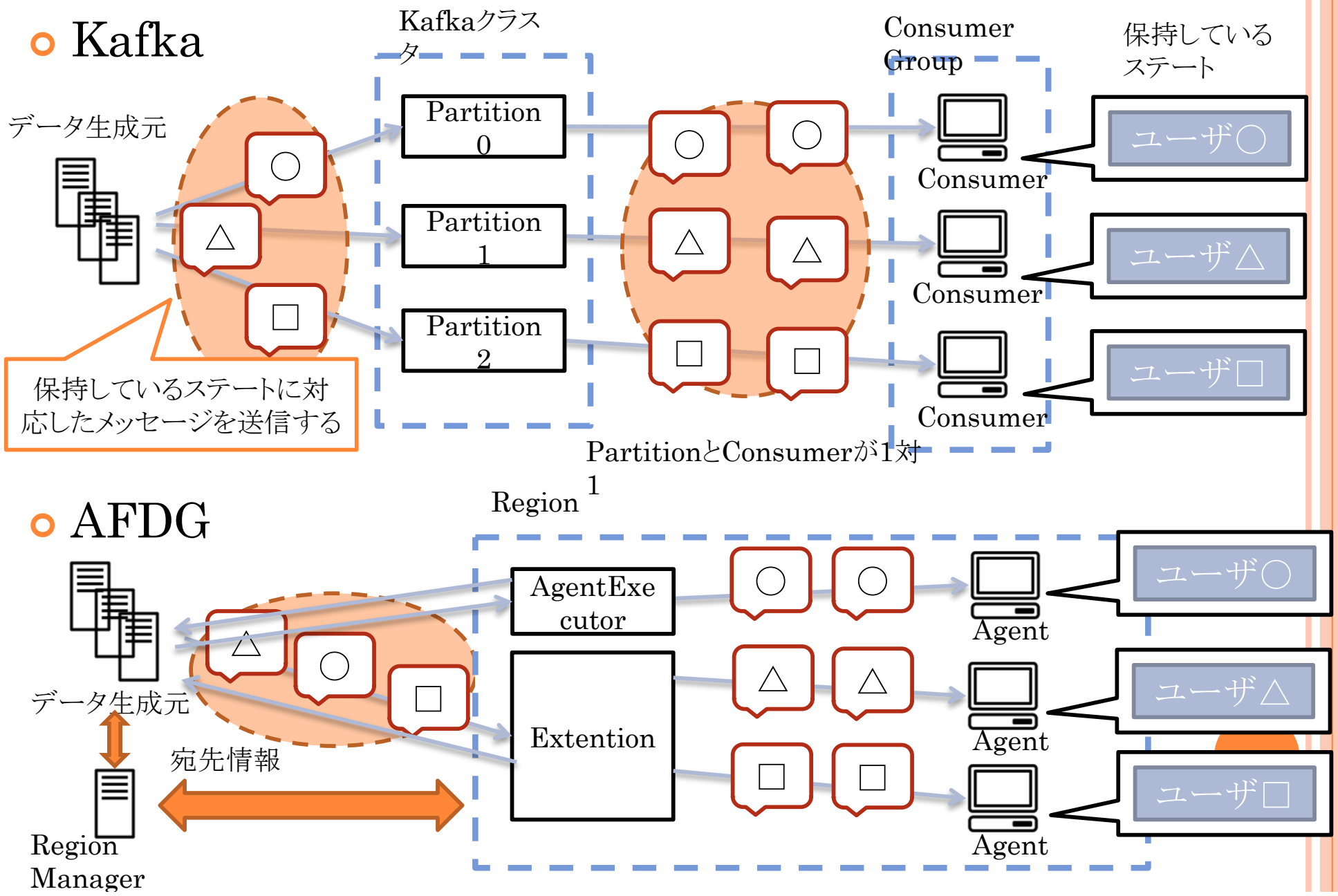
- メッセージを受信して、自分のデータにアクセスしながら処理を実行する
- あるエージェントは他のエージェントのデータにアクセスしない
- 複数のエージェントにメッセージ送信ができる
- 要求・応答処理、非同期処理、並列処理ができる
- エージェントへのメッセージ送信は、複数のエージェントが存在するリージョンへ送信し、リージョン内のAgentExecutor かExtentionがエージェントへ届ける
- エージェント内のデータ構造はエンティティ(キー + リスト)構造
- 送信先の詳細はカタログサーバで管理している

補足資料 AFDGとKafkaの違い

AFDGのイメージ



補足資料 AFDGとKafkaの違い



補足資料 AFDGとKafkaの違い

AFDGにできてKafka単体でできないこと

- 処理結果をリターンできる
- キャッシュでユーザ単位のデータを保持している
- レプリカを作成してデータ保全をしてる

KafkaにできてAFDGにできないこと

- メッセージ自体のレプリカを作成できる
- メッセージを取得するタイミングや取得する件数などを制御できる