Event-Driven Architecture

CSSE6400

Richard Thomas

April 4, 2022

Definition 1. Event

Something that has happened or needs to happen.

Definition 2. Event Handling

Responding to notification of an event.

Definition 3. Asynchronous Communication

Sending a message to a receiver and not waiting for a response.

Comment on how this enables parallel processing.

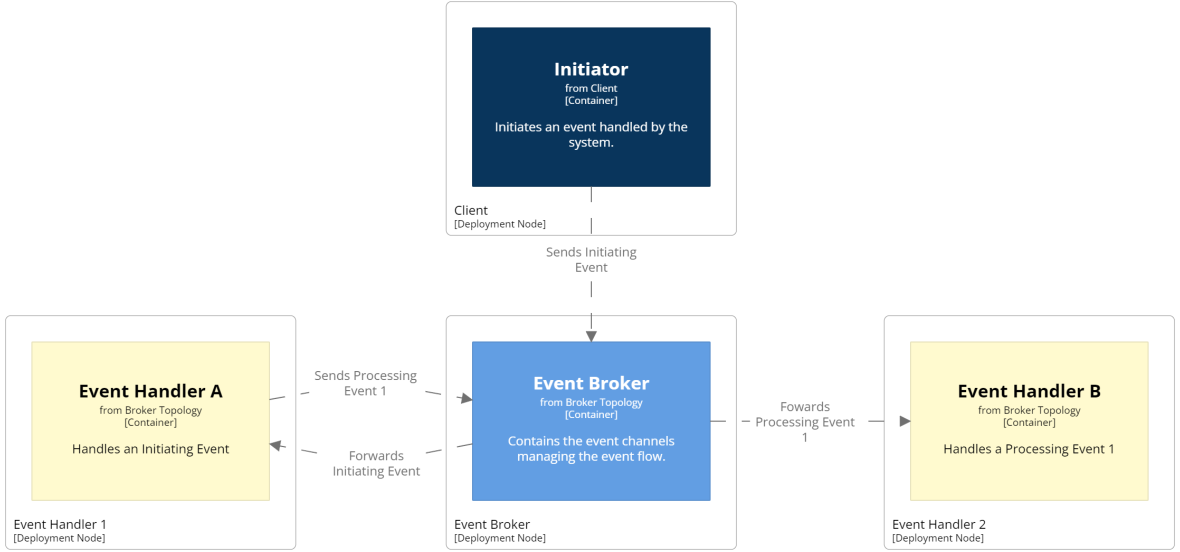
Responsiveness

- Synchronous Communication 👎
  - Send message
  - Wait for response
  - Continue processing
- Asynchronous Communication 👍
  - Send message
  - Continue processing
  - Optionally receive response
  - Complex error handling 👎

Definition 4. Event-Driven Architecture

Asynchronous distributed system that uses event processing to coordinate actions in a larger business process.

Event-Driven Architecture



Comment on how each container is deployed in its own compute node.

Terminology

Initiating Event Starts the business process

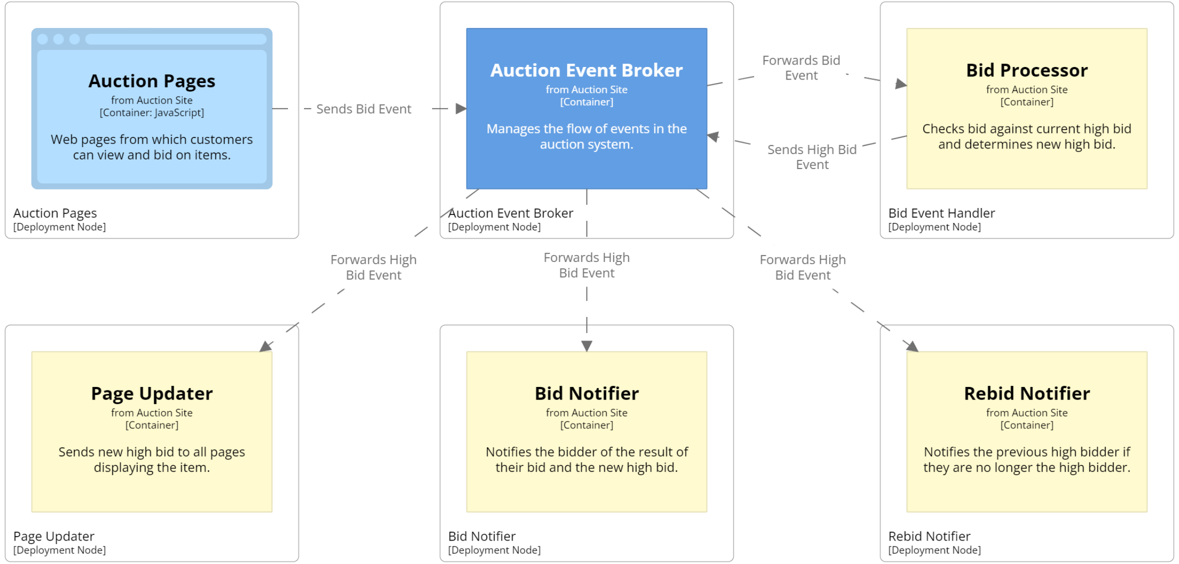
Processing Event Indicates next step in the process can be performed

Event Channel Holds events waiting to be processed

Event Handler Processes events

- Step, or part of a step, in the business process

Auction Example



- Step through event process.
- Highlight asynchronous messages and parallel processing.
- Bid Processor could send back a high bid event or an async message.

Definition 5. Event Handler Cohesion Principle

Each event handler is a simple cohesive unit that performs a single processing task.

Definition 6. Event Handler Independence Principle

Event handlers should not depend on the implementation of any other event handler.

Topologies

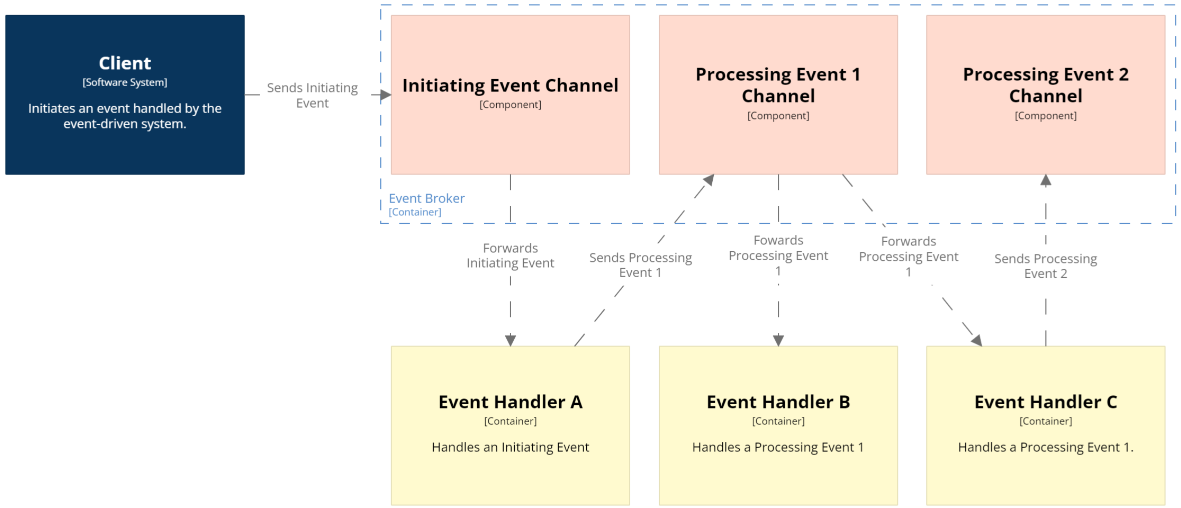
Broker All events received by event broker

- Notifies event handlers of events
- Event handlers send processing events when they finish processing

Mediator Manages the business process

- Event queue of initiating events
- Event mediator sends processing events to event handlers
- Event handlers send async messages to mediator to report process finished

Broker Topology



Client
[Software System]
Initiates an event handled by the event-driven system.

Sends Initiating Event

Initiating Event Channel
[Component]

Processing Event 1 Channel
[Component]

Processing Event 2 Channel
[Component]

Event Broker
[Container]

Forwards Initiating Event

Sends Processing Event 1

Fowards Processing Event 1

Forwards Processing Event 1

Sends Processing Event 2

Event Handler A
[Container]
Handles an Initiating Event

Event Handler B
[Container]
Handles a Processing Event 1

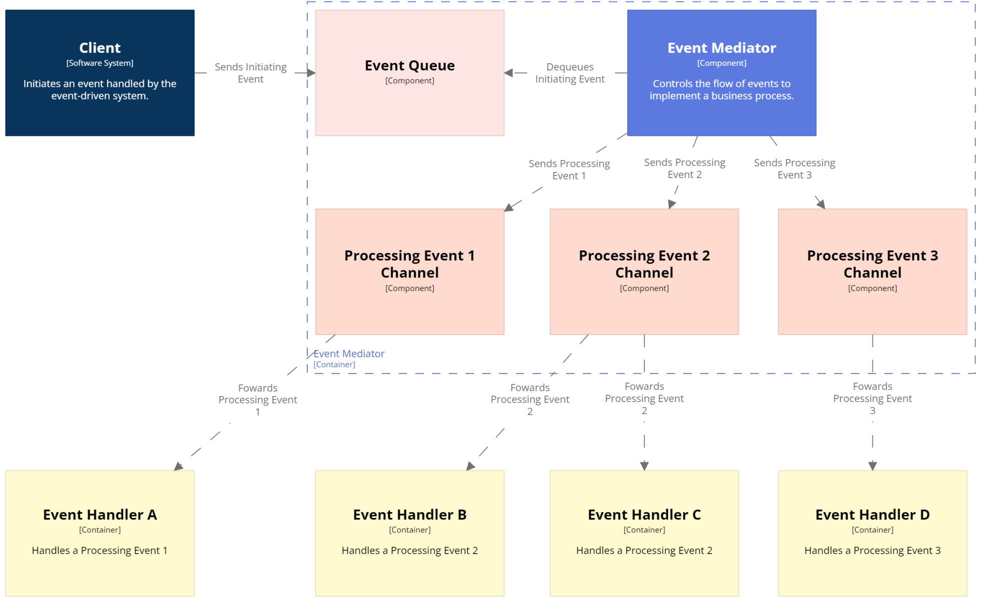
Event Handler C
[Container]
Handles a Processing Event 1.

- Step through event process.
- Describe idea of channels.

Event Broker Façade

- Event handlers can register to listen for events
- Receives events and directs them to the correct channel

Mediator Topology



Client
[Software System]
Initiates an event handled by the event-driven system.

Sends Initiating Event

Event Queue
[Component]

Dequeues Initiating Event

Event Mediator
[Component]
Controls the flow of events to implement a business process.

Sends Processing Event 1

Sends Processing Event 2

Sends Processing Event 3

Processing Event 1 Channel
[Component]

Processing Event 2 Channel
[Component]

Processing Event 3 Channel
[Component]

Event Mediator
[Container]

Fowards Processing Event 1

Fowards Processing Event 2

Fowards Processing Event 2

Fowards Processing Event 3

Event Handler A
[Container]
Handles a Processing Event 1

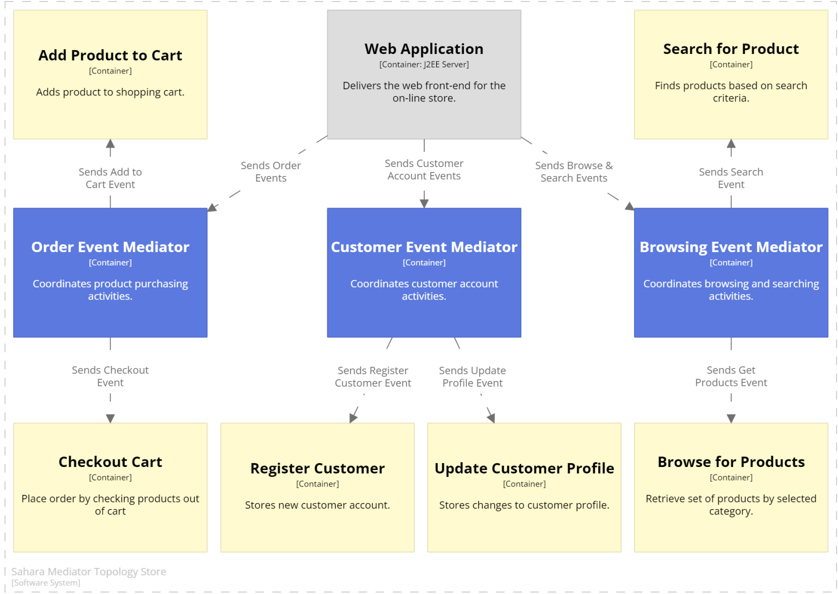
Event Handler B
[Container]
Handles a Processing Event 2

Event Handler C
[Container]
Handles a Processing Event 2

Event Handler D
[Container]
Handles a Processing Event 3

- Step through event process.
- Highlight process control performed by mediator.

Sahara Mediator Topology



Add Product to Cart
[Container]
Adds product to shopping cart.
Web Application
[Container: J2EE Server]
Delivers the web front-end for the on-line store.
Search for Product
[Container]
Finds products based on search criteria.
Sends Add to Cart Event
Sends Order Events
Sends Customer Account Events
Sends Browse & Search Events
Sends Search Event
Order Event Mediator
[Container]
Coordinates product purchasing activities.
Customer Event Mediator
[Container]
Coordinates customer account activities.
Browsing Event Mediator
[Container]
Coordinates browsing and searching activities.
Sends Checkout Event
Sends Register Customer Event
Sends Update Profile Event
Sends Get Products Event
Checkout Cart
[Container]
Place order by checking products out of cart
Register Customer
[Container]
Stores new customer account.
Update Customer Profile
[Container]
Stores changes to customer profile.
Browse for Products
[Container]
Retrieve set of products by selected category.
Sahara Mediator Topology Store
[Software System]

- Step through event process.
- Highlight multiple mediators as common implementation.
- Discuss internals of mediators: event queue and event channels.

Extensibility

- New behaviour for existing event

  Broker Implement event handler & register with broker

  - Existing ignored event hooks

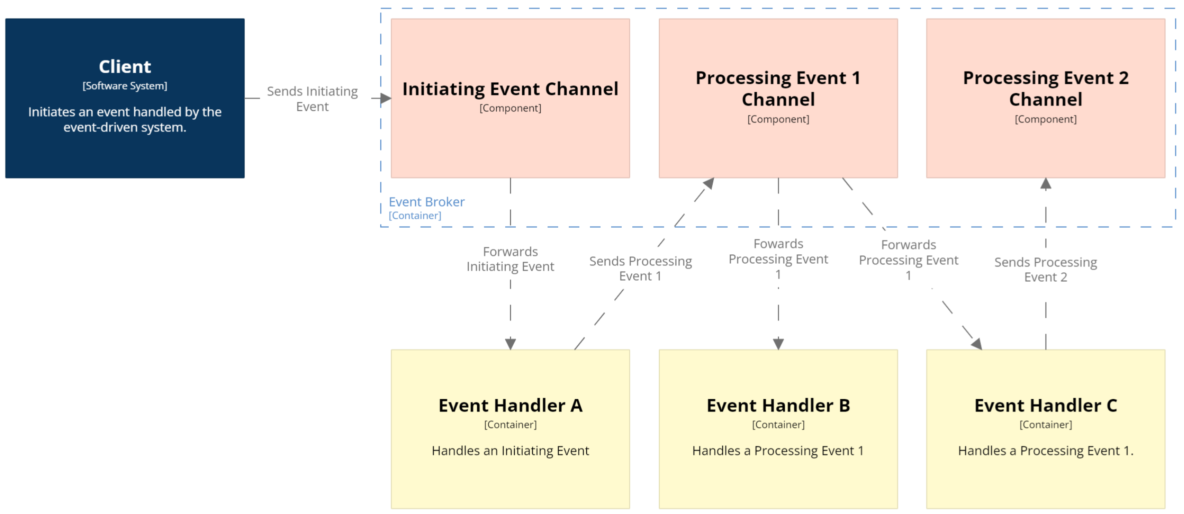
  Mediator Implement event handler & modify mediator logic

- New event

  Broker Implement event & event handler, create event channel, modify broker façade

  Mediator Implement event & event handler, modify mediator logic

Broker Topology


Client
[Software System]
Initiates an event handled by the event-driven system.
Sends Initiating Event
Initiating Event Channel
[Component]
Processing Event 1 Channel
[Component]
Processing Event 2 Channel
[Component]
Event Broker
[Container]
Forwards Initiating Event
Sends Processing Event 1
Fowards Processing Event 1
Forwards Processing Event 1
Sends Processing Event 2
Event Handler A
[Container]
Handles an Initiating Event
Event Handler B
[Container]
Handles a Processing Event 1
Event Handler C
[Container]
Handles a Processing Event 1.

- Step through event process.
- Describe idea of channels.

Scalability

- Event handlers deployed independently
  - Scaled independently to manage load
- Event broker federated
  - Distributed across multiple compute nodes
- Event mediators for different domains
  - Distributes loads by domain
    (e.g. browse & search, account, & order events)
    - Scaled independently to manage load

Queues

- Channels can be implemented as queues
  - FIFO behaviour
- Multiple front of queue pointers
  - For each event handler
- Event removed when event handlers finish
  - Retry if a handler fails
- Events persisted until removed
  - Recovery from broker failure

Streams

- Channels can be implemented as streams
  - Events are saved permanently
- Handlers notified when event added to stream
  - Observer pattern
- Handlers process events at their own pace
  - Cardiac arrest alarm vs. heart rate graph
- Events history
  - Redo processing
  - Review processing activities

Queues vs Streams

- Queue
  - Known steps in business process
  - Easier sequencing of steps in business process
  - “Exactly once” semantics
  - eCommerce system
- Stream
  - Very large number of events or handlers
  - Handlers can ignore events
  - Analysis of past activity
  - Event sourcing

Broker vs Mediator Topologies

Broker	dumb pipe
Broker	events have occurred
Mediator	smart pipe
Mediator	events are commands to process

Broker vs Mediator Topologies

Broker Advantages

- Scalability
- Reliability
- Extensibility
- Low coupling

Mediator Advantages

- Complex business process logic
- Error handling
- Maintain process state
- Error recovery

Pros & Cons

Modularity	Event Handlers	👍
Extensibility		👍
Reliability	Event Handlers	👍
Interoperability	Events	👍
Scalability	Event Handlers	👍
Security		😐
Simplicity		👎
Deployability		👎
Testability	Complex Interactions	👎