Event Modeling Overview

Key Concepts

Event: Represents something that has happened in the system. These are factual occurrences and are described in the past tense. For example, "Order Placed" indicates that an order has been placed by the user.

Command: An action or decision that triggers an event. Commands are typically named with a verb in the imperative mood, present tense. For example, "Place Order" is a command issued by the user to place an order.

State: The condition of the system at a given time, influenced by events. Each event alters the system's state, creating a new snapshot of the system.

Read Model: A projection of the state used to query the system. It provides a way to view the state based on the events that have occurred. For example, "Order Summary" shows the details of an order based on the events that have updated the order state.

Example: Online Shopping System

1. **Command**: "Add Item to Cart"
    - **Description**: User decides to add an item to their shopping cart.
2. **Event**: "Item Added to Cart"
    - **Description**: System records the addition of the item to the cart.
3. **State**: "Cart State"
    - **Description**: The cart now contains the new item, reflecting the latest state.
4. **Command**: "Place Order"
    - **Description**: User proceeds to checkout and places the order.
5. **Event**: "Order Placed"
    - **Description**: System records that an order has been successfully placed.
6. **State**: "Order State"
    - **Description**: The system updates to reflect the new order status.
7. **Read Model**: "Order Confirmation"
    - **Description**: User can view a summary of their placed order.

Timeline of Events

1. **Add Item to Cart** (Command) -> **Item Added to Cart** (Event)
2. **Place Order** (Command) -> **Order Placed** (Event)

State Changes:

- Initially, the cart is empty.
- After "Item Added to Cart", the cart includes the new item.
- After "Order Placed", the order state includes the placed order.

Read Models:

- **Cart Contents**: Displays items currently in the cart.
- **Order Confirmation**: Displays details of the placed order.

Event Storming

Purpose

Event Storming is a workshop-based method to collaboratively discover business processes and user flows. It applies to any technical or business domain, particularly those that are large and complex. The emphasis on events makes it an excellent tool for event-based system engineering.

Flavors

- **Big Picture Event Storming**: This approach involves all stakeholders to collect viewpoints and visualize the entire business flow as a series of events. It provides a broad overview of the system.

Getting Started

1. **Set Expectations**: Start small and aim to get familiar with Event Storming. The goal should be to create a shared understanding of the domain, not to derive actionable stories immediately.
2. **Facilitator**: Ensure a facilitator with Event Storming experience is available to guide the session, manage time, and mediate discussions.
3. **Split Sessions**: For remote workshops, split the sessions into at least two days with breaks. This helps maintain focus and manage fatigue.

Process

1. **Prepare Board Workspace**: Create a new board, invite participants via email, and set up the workspace for the session.
2. **First Event**: Start by asking about the goals of the company or business unit. Identify initial events that serve user needs and how the company makes money.
3. **Chaotic Mode**: Attendees write down events in parallel without worrying about order. This captures a wide range of ideas and possibilities.
4. **Tell a Story**: Order the events chronologically to construct a coherent story. This helps in understanding the sequence of operations and interactions.
5. **Group By Context**: Group related events and name these groups. This helps in organizing the workflow into manageable sections.
6. **Explore The Domain**: Introduce additional concepts like Information, Actors, External Systems, and Commands. Dive deeper into each group to understand the specifics.
7. **Identify Hot Spots**: Use Hot Spots to highlight questions, discussions, and bottlenecks. This helps in identifying areas that need further exploration or improvement.
8. **Retro**: Conduct a retrospective at the end of each day to gather feedback and discuss improvements for future sessions.

Cards and Their Roles

Domain Event

Color: Orange
Represents a fact that has happened. Use past tense and place events in chronological order. Examples include "User Browsed Online Shop," "Item Put Into Cart," "Order Placed," "Order Paid," and "Order Shipped."

Command

Color: Blue
Represents a decision that something should happen. Use imperative mood, present tense. Examples include "Start Engine," "Buy House," and "Share Picture." Commands cause events and are often in a 1:1 relationship with them, but can also cause multiple or no events depending on business rules.

Actor

Color: Lemon Yellow (Portrait)
Highlights user roles involved in a process. Write the user role on the card and place it next to a UI or Command card.

Aggregate

Color: Sunny Yellow
Represents business rules that ensure a command can be executed. For example, "An order can only be shipped if it is paid." Write these rules down and place them between commands and events.

Information

Color: Green
Represents information needed to make a decision. Information can be displayed in a UI, an Excel sheet, a PDF document, or fetched via an API from a database. Information changes should be described as Domain Events.

Policy

Color: Lilac
Represents automated decision and coordination logic, often "If this, then that." Policies trigger new commands. For example, in a ticketing system, a policy might automatically change a reserved seat to a booked seat after a ticket is bought.

Hot Spot

Color: Red
Highlights questions, discussions, and bottlenecks. Use Hot Spots to mark areas that need further exploration or have unresolved issues.

External System

Color: Dusky Pink
Visualizes third-party systems involved in a process. Digitizing workflows often involves connecting different digital services.

UI / API

Color: Black
Represents public interfaces, either for humans (UI) or machines (API). Use icons on UI Cards to sketch what is visible on the screen.