

CS529

Fundamentals of Game Development

Lecture 4a

Antoine Abi Chakra
Karim Fikani

Questions?

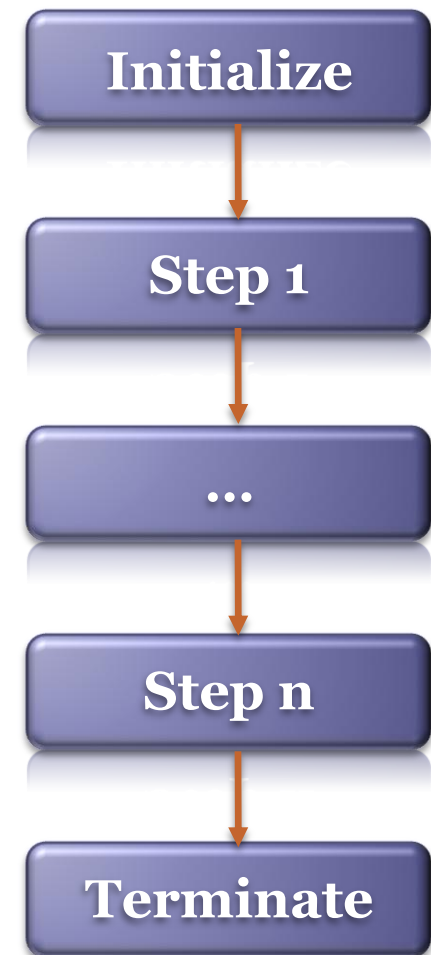
- Game State Manager
- Function Pointers
- Frame Rate Controller

Overview

- Procedural Programming
- Control Driven Programming
- Event Driven Programming

Procedural Programming (1/2)

- Definition:
 - Divide and conquer
- Example:
 - Cooking
 - Making a car
 - Making a game
 - Etc...

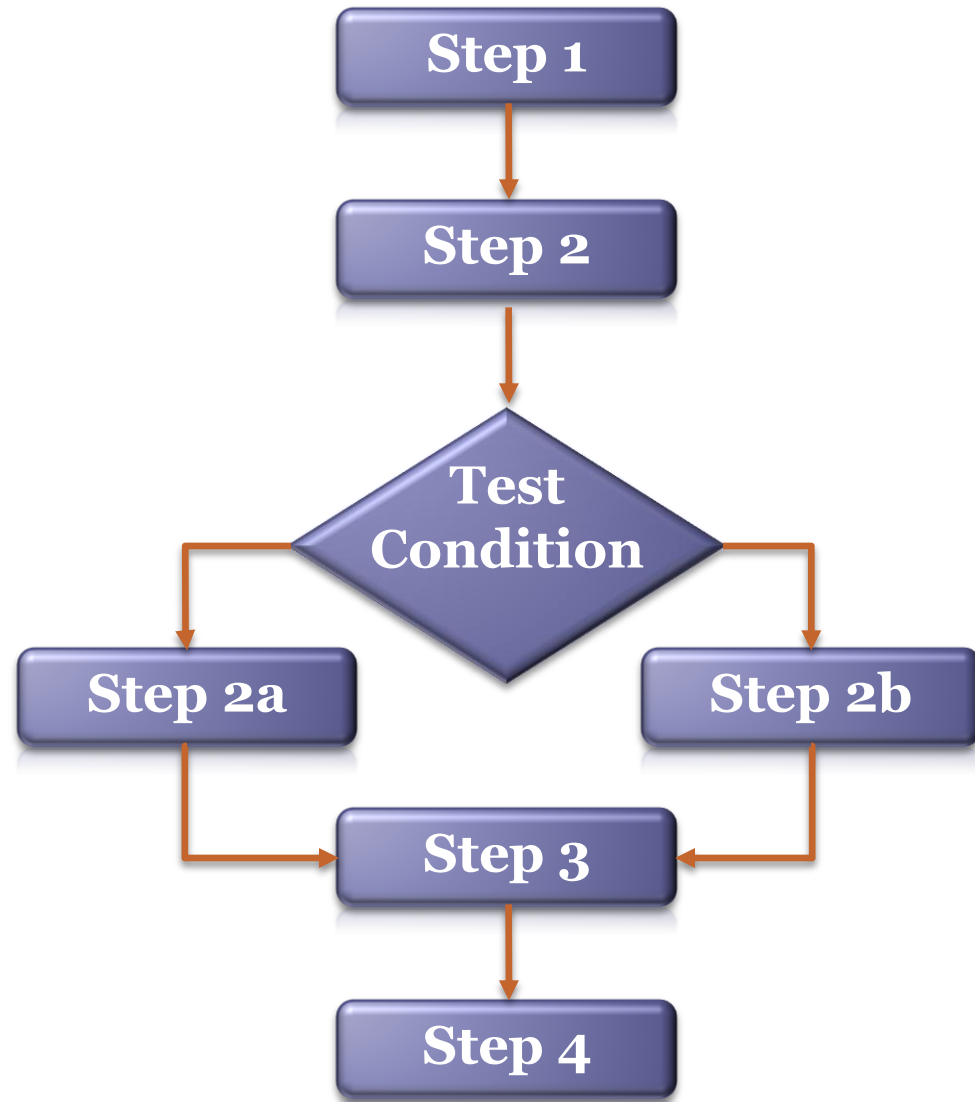


Procedural Programming (2/2)

- Also known as structured programming (concept by Edgar Dijkstra)
- In this concept, a program has a basic structure and facilities such as:
 - Branching
 - Looping
 - Functional decomposition

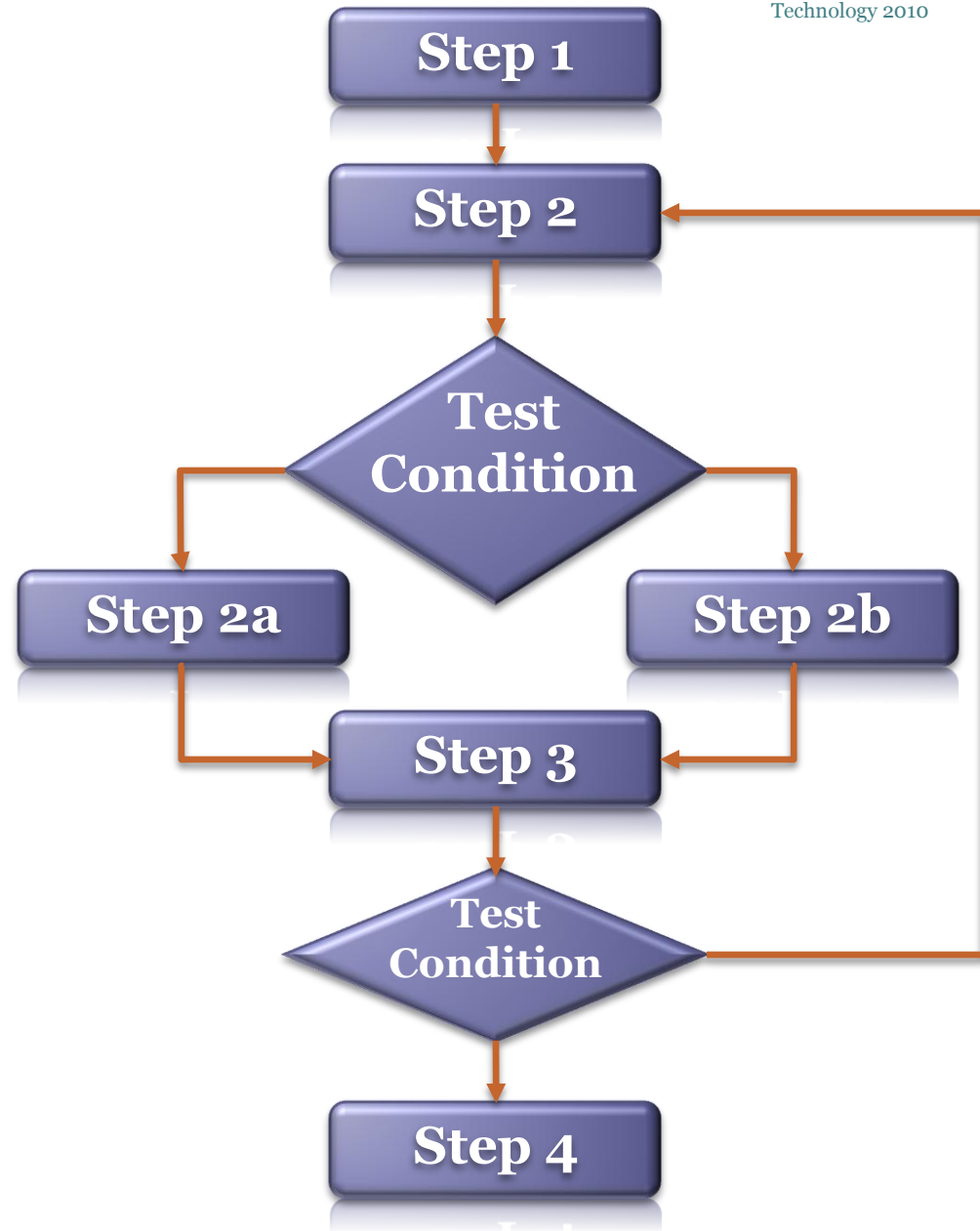
Branching

- Flow of control is dependent on results of conditions.



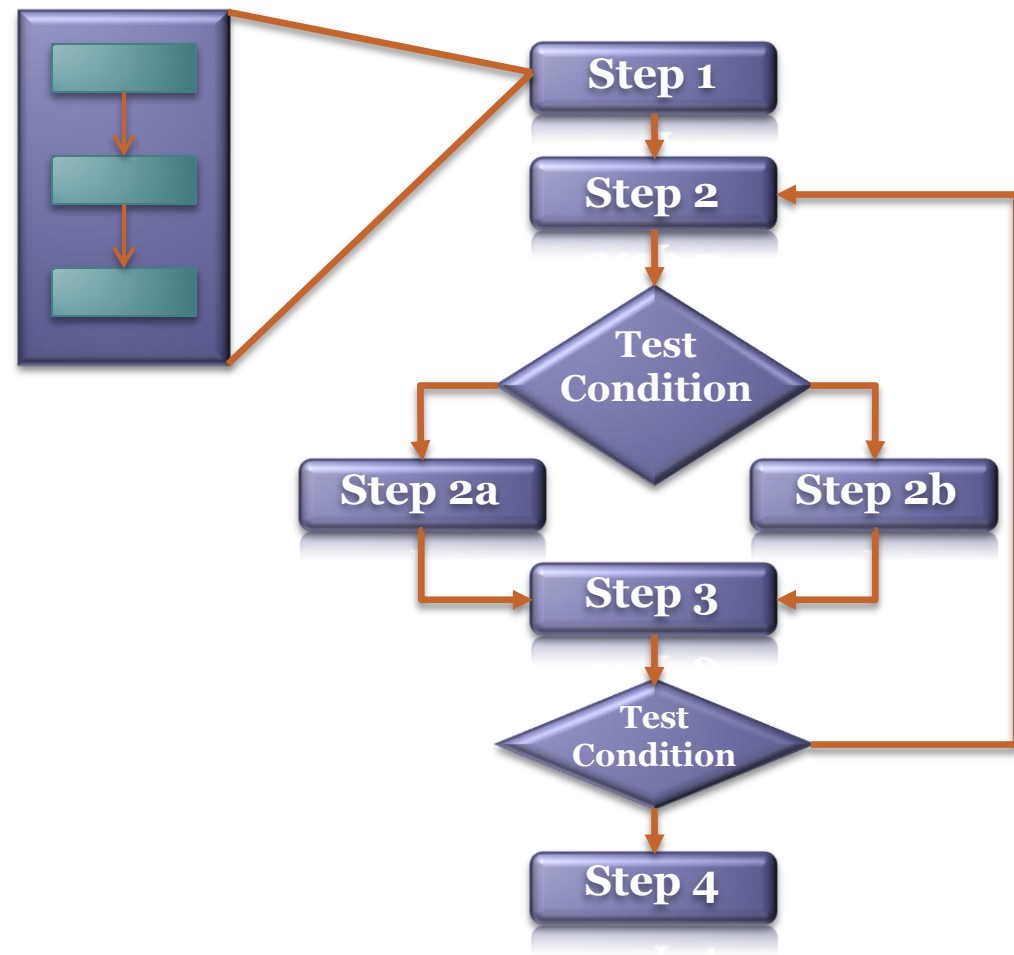
Looping

- Repeat certain steps continuously until some test condition is reached.



Functional Decomposition

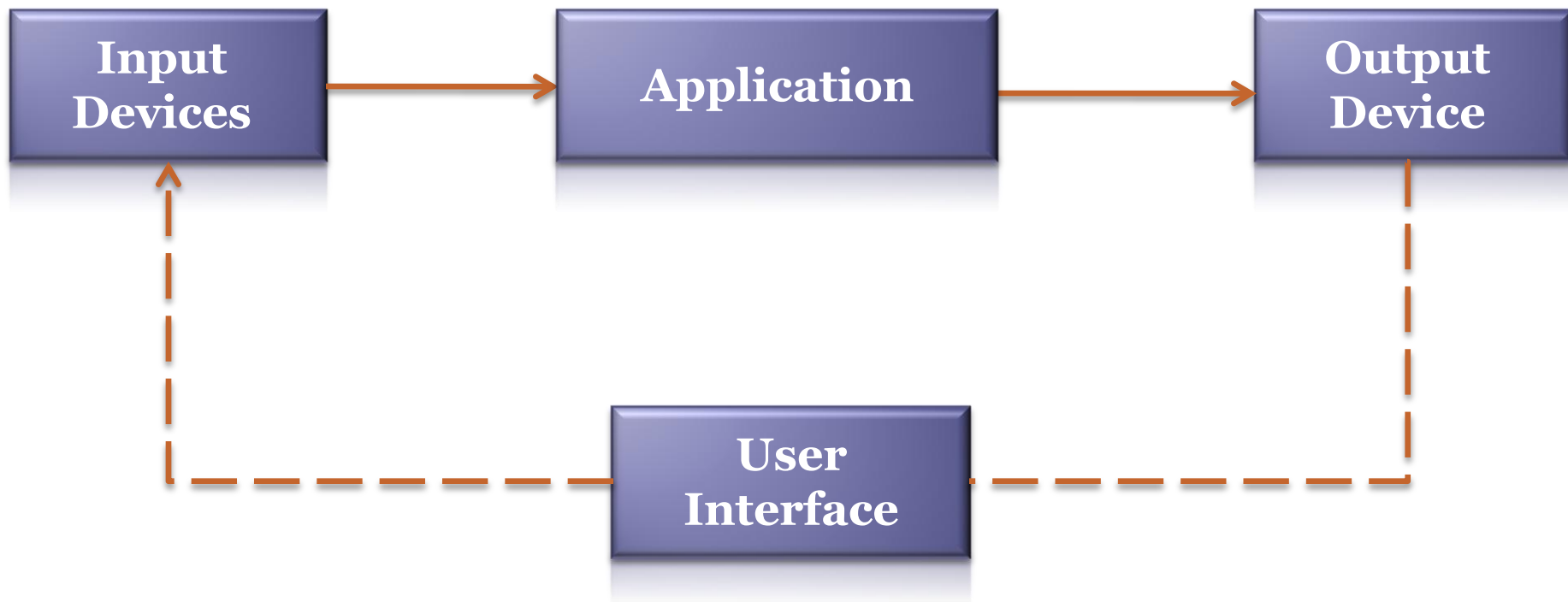
- Problems are decomposed into functional steps



Overview

- Procedural Programming
- Control Driven Programming
- Event Driven Programming

Interactive Applications (1 / 2)



Interactive Applications (2/2)

- They require three functionalities:
 - Reading input
 - Writing or drawing output
 - Data handling and management
- Ex:
 - Games
 - Training Simulators
 - GUI-Based Application (MS Word)

Control-Driven Programming

- The control of execution is within the program

Initialize the Application



```
while (!quit)
{
    Prompt the user
    Read input from keyboard or mouse
    Parse user input to determine user choice or action
    Generate output
    Write output
}
```

Advantages

- Straightforward development
- Interactions between users and programs can be easily modeled

Disadvantages

- Polling (continuous checking) for user input leads to wasting system resources.
- Complex interfaces and asynchronous interactions cannot be implemented.
 - **Example:**
If an application is waiting for a key press and a mouse is clicked, the mouse click is ignored

Overview

- Procedural Programming
- Control Driven Programming
- **Event Driven Programming**

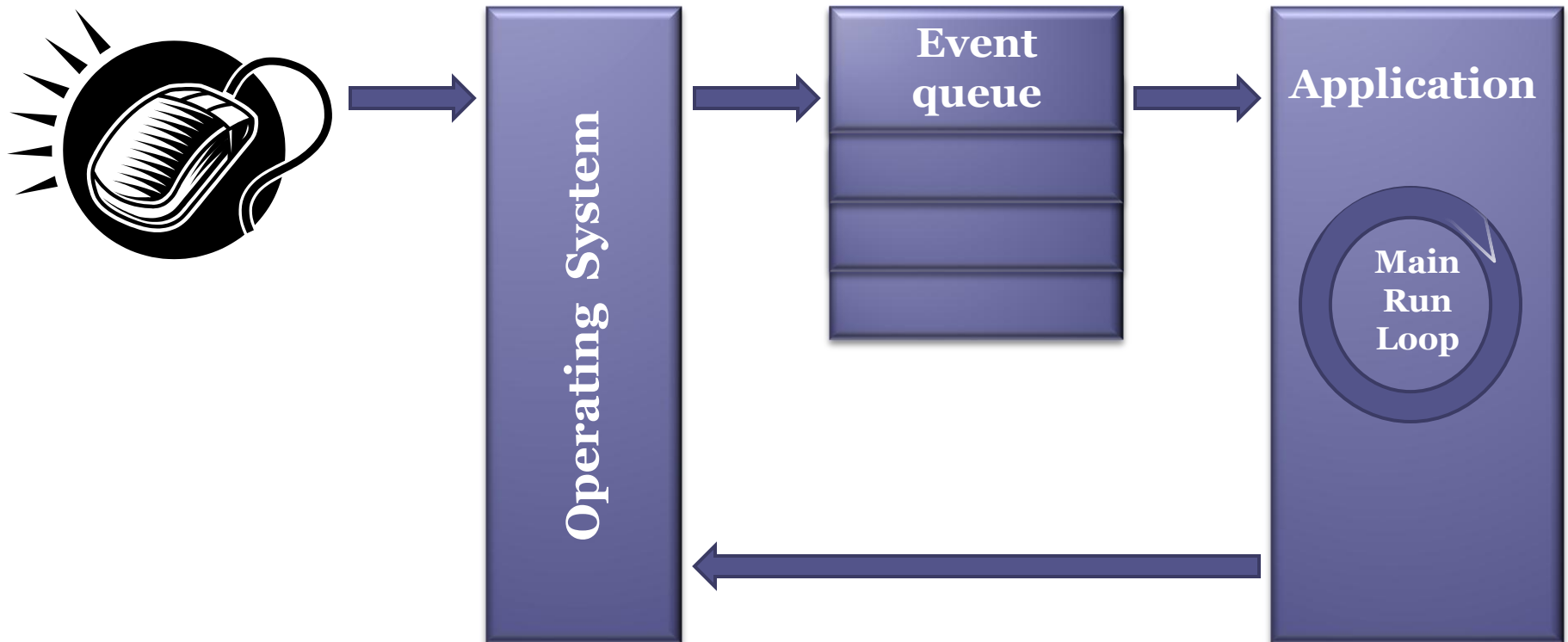
What is an Event?

- Defined as a type of signal to the program that something has happened
 - **Example:**
 - Mouse click
 - Joystick movement
 - Keyboard is pressed
 - Etc...

What is an Event Handler?

- Defined as functions that encapsulate response to events of that type.

Event-Driven Programming (1 / 3)



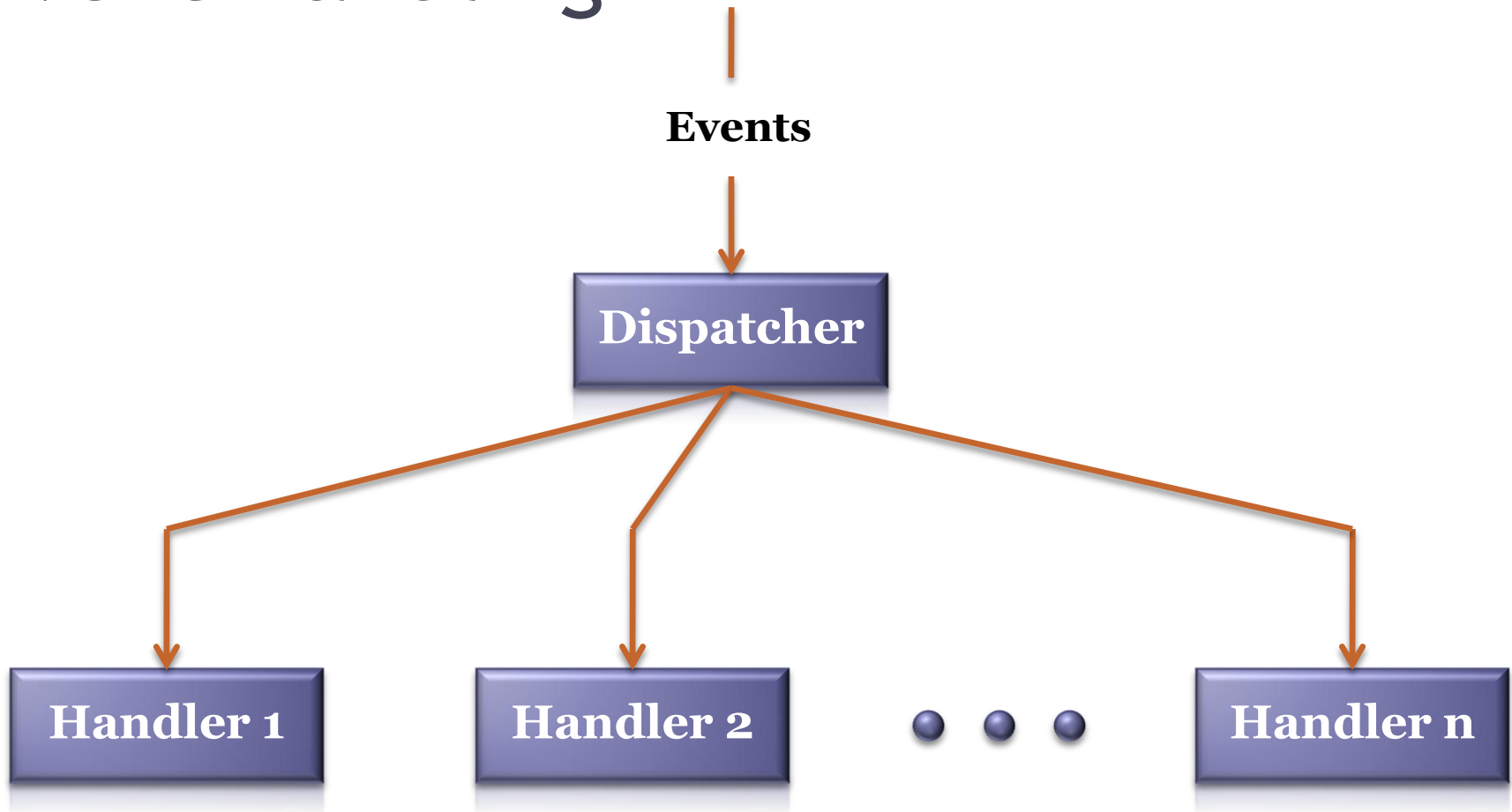
Event-Driven Programming (2/3)

- Functions are executed when the user interacts with the program's interface.
- User is in control of the application
- The so called Hollywood principle
“Don't call us, we'll call you”

Event-Driven Programming (3/3)

- Instead of the programmer dictating the flow of control, the programmer simply writes and registers the functions that are to be executed when a user interacts with the program.
- An application can decide what events to handle **but** it won't know in advance the exact order in which those events will occur.

Event Handling



What is the Dispatcher?

- The job of the *dispatcher* is to take each item that comes to it, analyze the event to determine its event type and then send each event to a handler that can handle events of that type.

Implementation

- The program structure is divided roughly into two groups:
 - Detect actions performed by the user
 - Contain the code associated with these interactions.

Pseudo-Code

```
while (!quit) // do forever
{
    get event from input stream
    if( event type == EndOfEventStream)
        Quit() //Break out of event loop
    else if( event type == EventTypeZero)
        ExecuteEventTypeZero(event information);
    else if( event type == EventTypeOne)
        ExecuteEventTypeOne(event information);
    else    // handle unrecognized event type
            // ignore the event or raise an exception
}
```


Control-Driven VS Event-Driven

- Follows steps (step1...final step)
- Divide and conquer
- Application is in control

- Instructions are **not** executed sequentially from first to last.
- Users are in control of the application