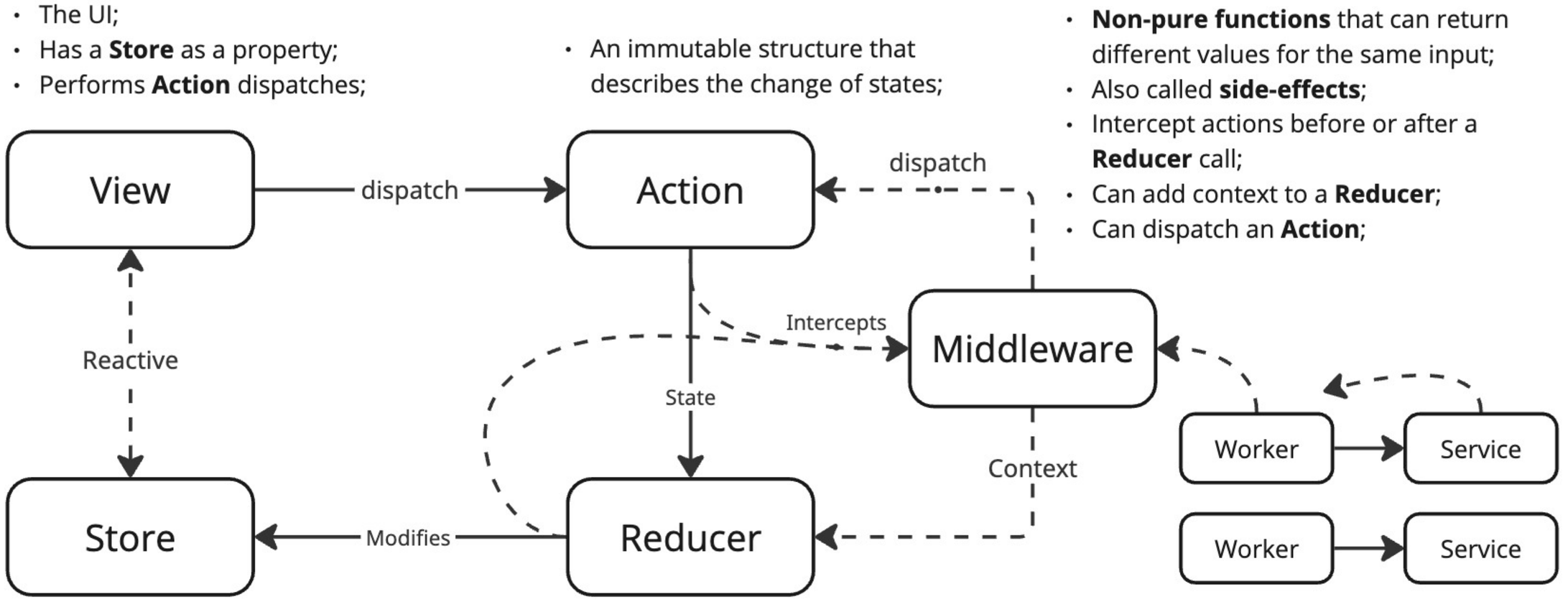
Redux - The "MV" that worked

That's not even the goal.

Redux doesn't reduce code and increase bureaucracy.

Created by:
@pedro0x53



- The UI;
- Has a **Store** as a property;
- Performs **Action** dispatches;

- An immutable structure that describes the change of states;

- **Non-pure functions** that can return different values for the same input;
- Also called **side-effects**;
- Intercept actions before or after a **Reducer** call;
- Can add context to a **Reducer**;
- Can dispatch an **Action**;

- A **snapshot** of the application's states;
- Single Source of Truth (**SSoT**);
- Contains all the states of the **View**;

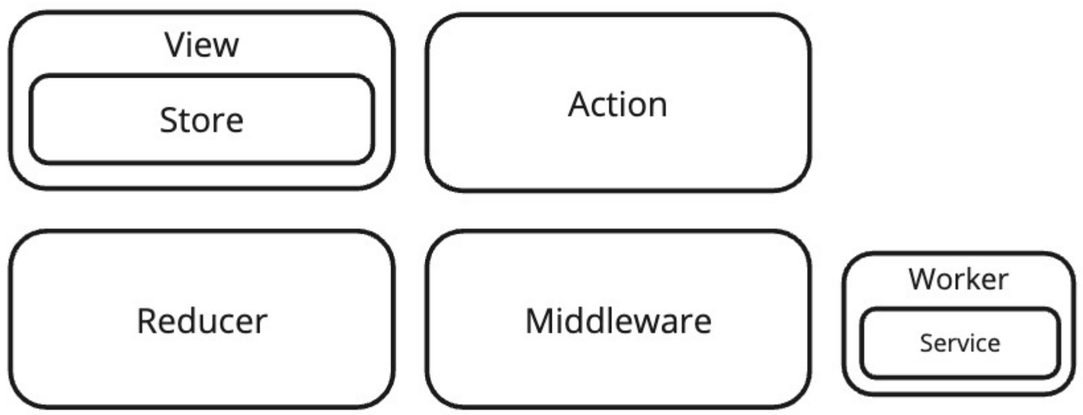
- **Pure functions** that return the same value if they receive the same input;
- It's the only place in the code where modifications to the **Store** occur;
- It's the implementation of the Business Rule;

- **Facades** for more complex subsystems/services such as Web Requests, Databases, file handling, mathematical calculations;

*__Actions__, __Middlewares__, and __Reducers__ can literally be **"standalone" functions**, not associated with *classes*, *structs*, or any other construct in the code;

Architectural Composition

Who contains whom?



From a "static" point of view (defining a class, constructors), there is little composition in the architecture (ironic, isn't it?), but the use of its parts is dynamic and can take on many different compositions (not so ironic).

Sequence Diagram