React Architecture Fundamentals

Areg Sarkissian

WebApp Builders (LA Eastside) meetup

March 4, 2015

React Advantages

▪ **Composable, Avoid manual DOM manipulation, Unidirectional data flow(Flux), Declarative(JSX)**

- □ ·Easier to reason about (especially for larger applications)
- □ ·Easier to prevent bugs ·

▪ **Speed (goals)**

□ 60 fps

□ Render < 16ms

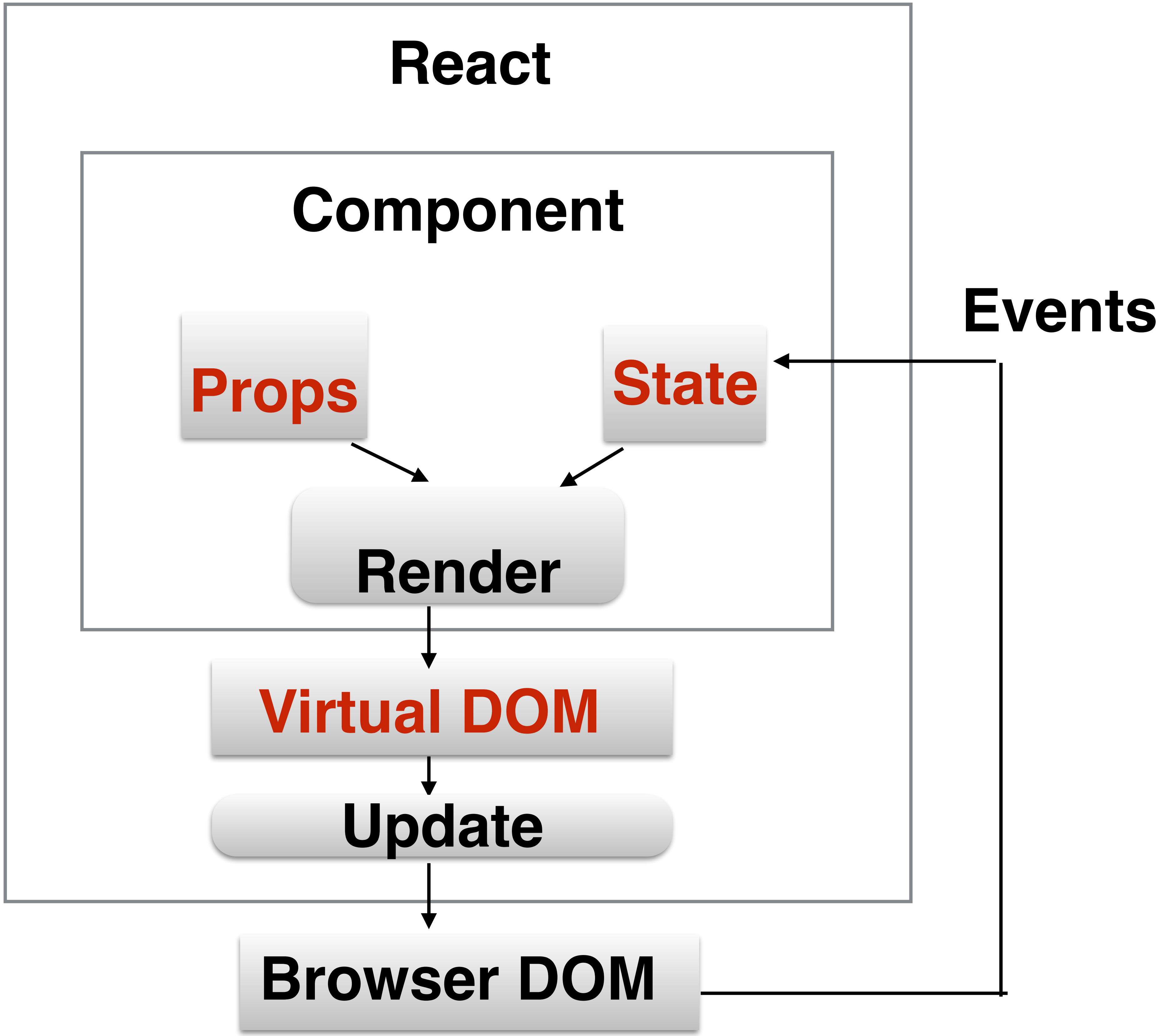
▪ **It’s only the View in MVC.**

□ Can mix with other front end technologies easily

▪ **Isomorphic**

□ React can easily render html content server side

React Rendering Flow



React

Component

Props

State

Render

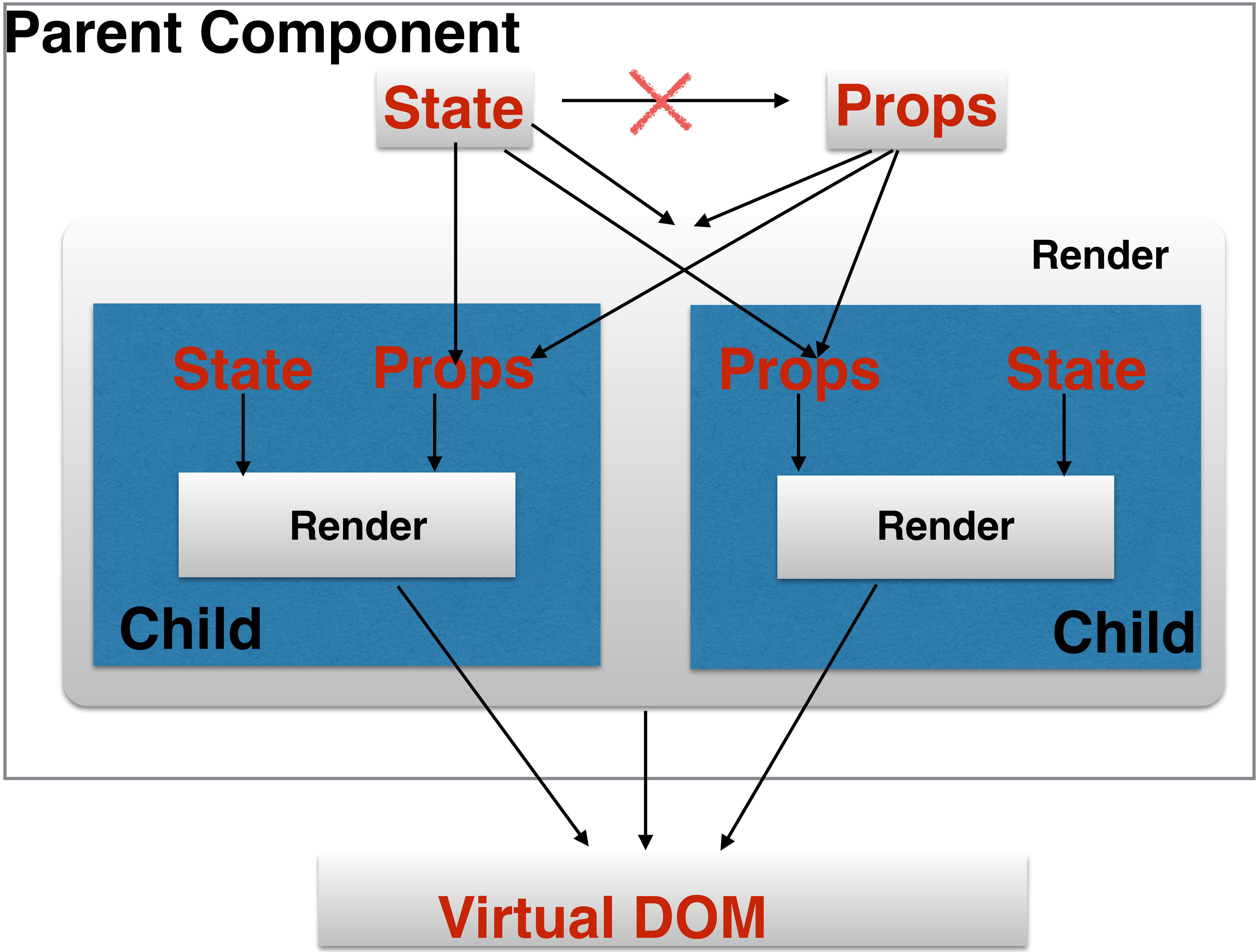
Virtual DOM

Update

Browser DOM

Events

React Component Hierarchy



rendering starts with one root component

bootstraping the rendering process

Single root parent component passed to bootstrap render function:

```
var child = React.createClass({
 render: function() {
  return (
   <div>child component {this.props.childprop}</div>
  );
 }});

var rootParent = React.createClass({
 render: function() {
  return (
   <div>root component</div>
   <child childprop={this.props.text}/>
  );
 }});
```

```
\\Bootstrap render function rendering root component into document.body
React.render(< rootParent text=“myson”/>, document.body);
```

Props and State

Declaring a Hello Component with props

```
<Hello message="hello"
now={new Date().toString()}
past={this.props.past}
future={this.state.future} />
```

Declare the component and initialize its props:

-in the render function of the parent (containing) component
OR
-inside the bootstrap render function if Hello is the root component

A component props can be initialized with the state or props of its parent component (unless the component is declared in the bootstrap render function)

Access a components props or state inside any of the components functions

```
{this.props.message}            {this.state.future}
{this.props.now}
{this.props.past}
```

State

getInitialState function sets up the initial state of a component

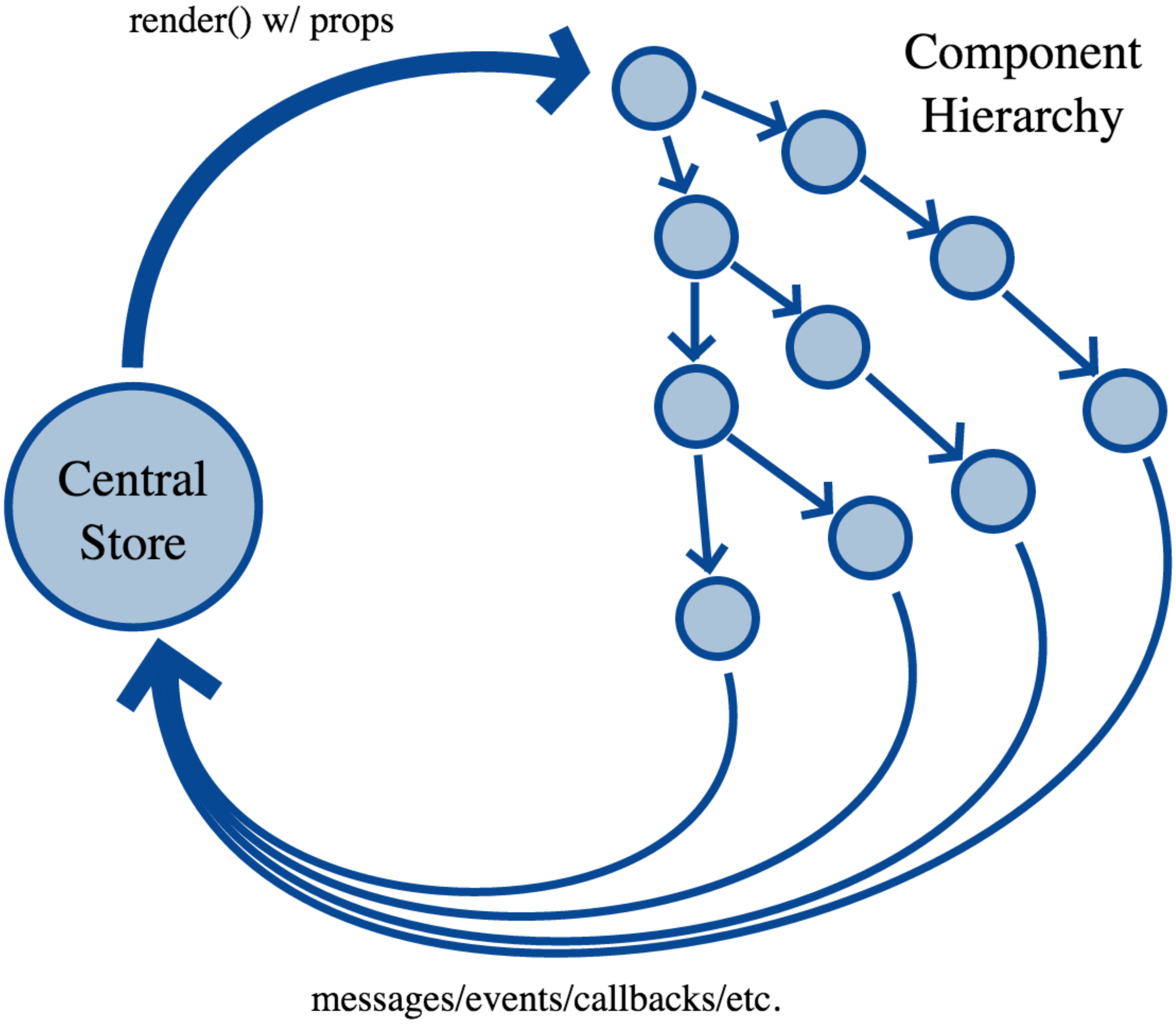
```
getInitialState: function() {
  return {secondsElapsed: 0};
}
```

setState function changes the state of a component
and triggers the render function of that component

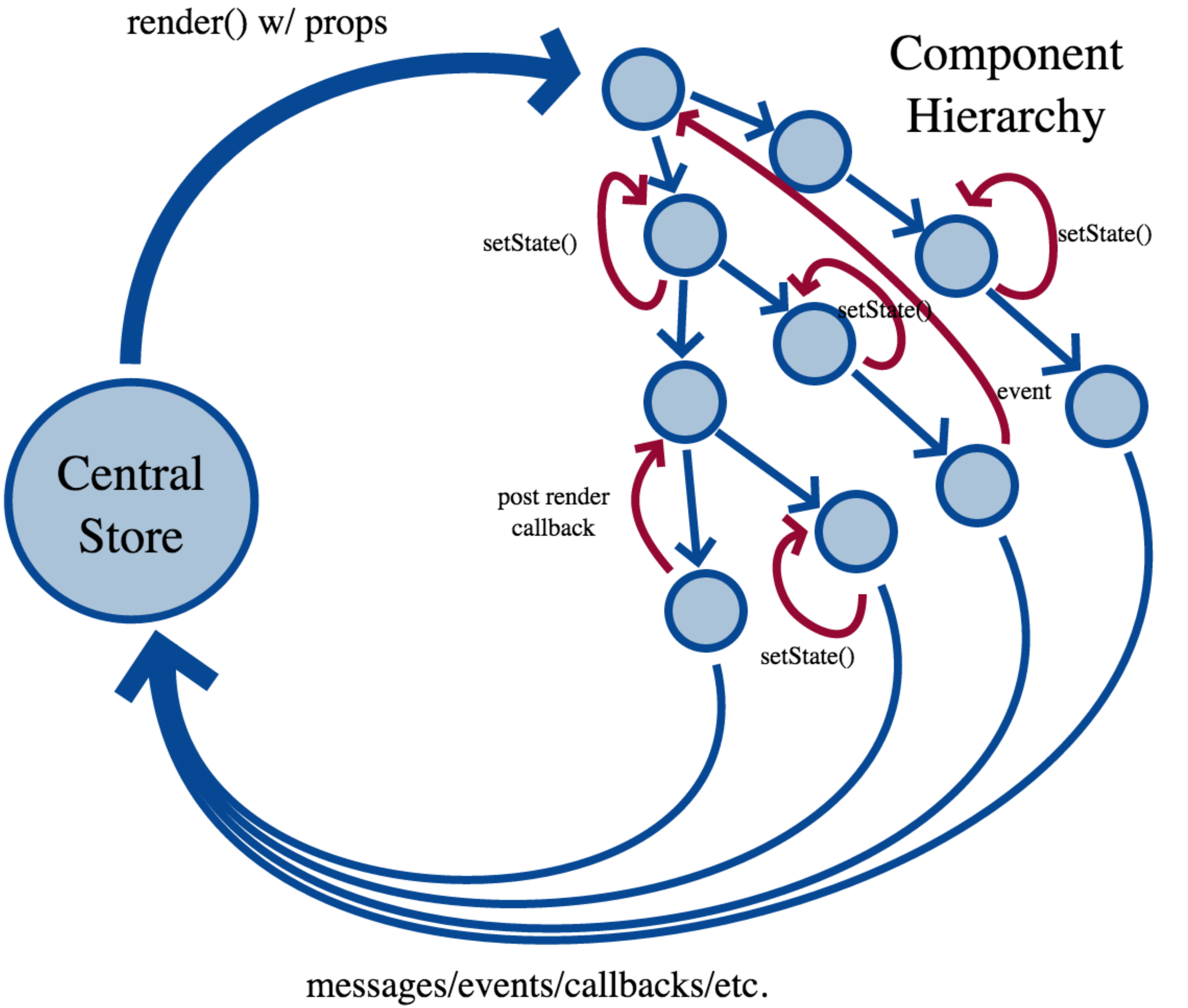
```
tick: function() {
    this.setState({secondsElapsed: this.state.secondsElapsed + 1});
}
```

Never call setState inside the render function

react one way data flow

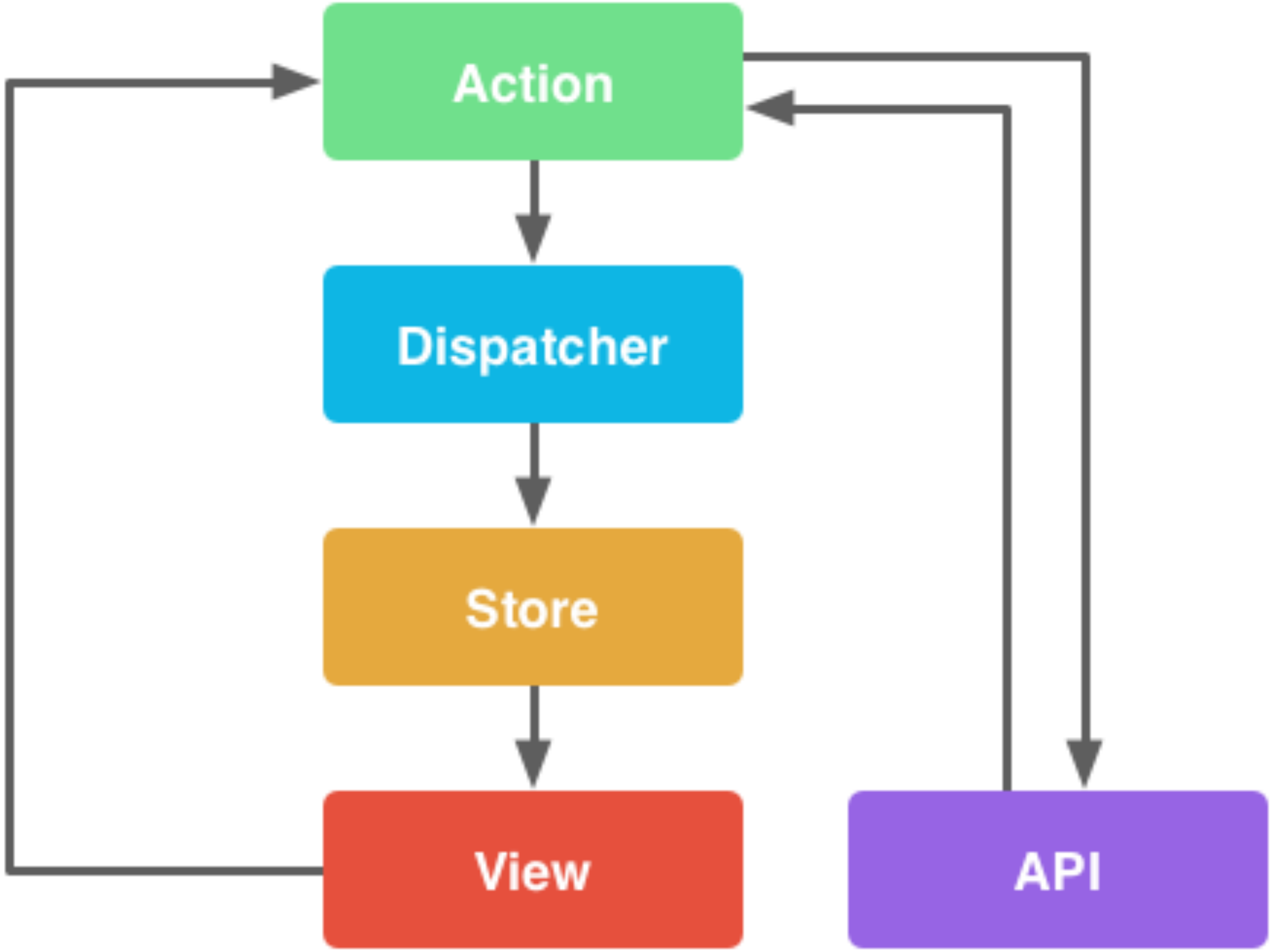


react two way data flow(not recommended)



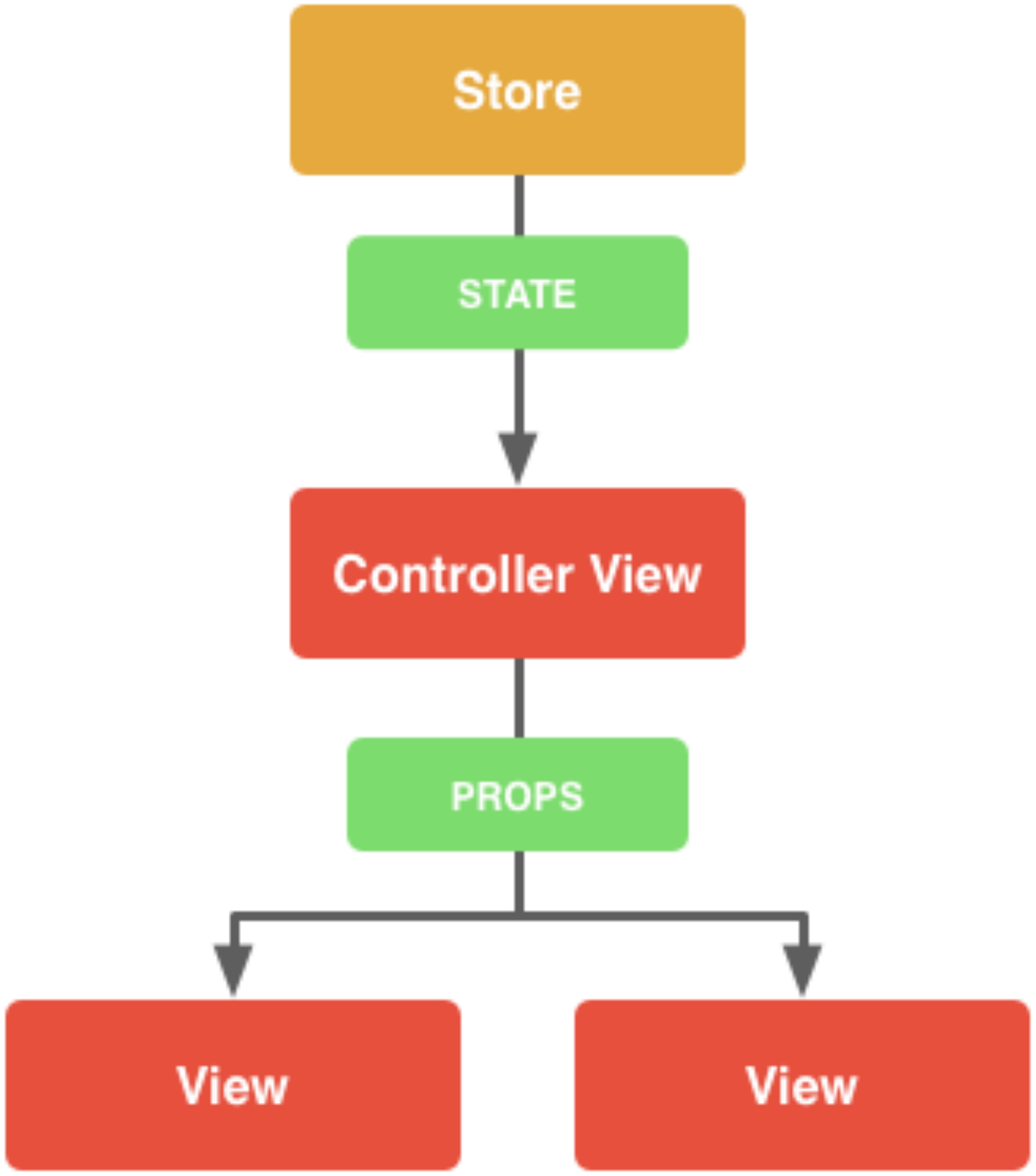
Flux data flow

One way data flow architecture for react

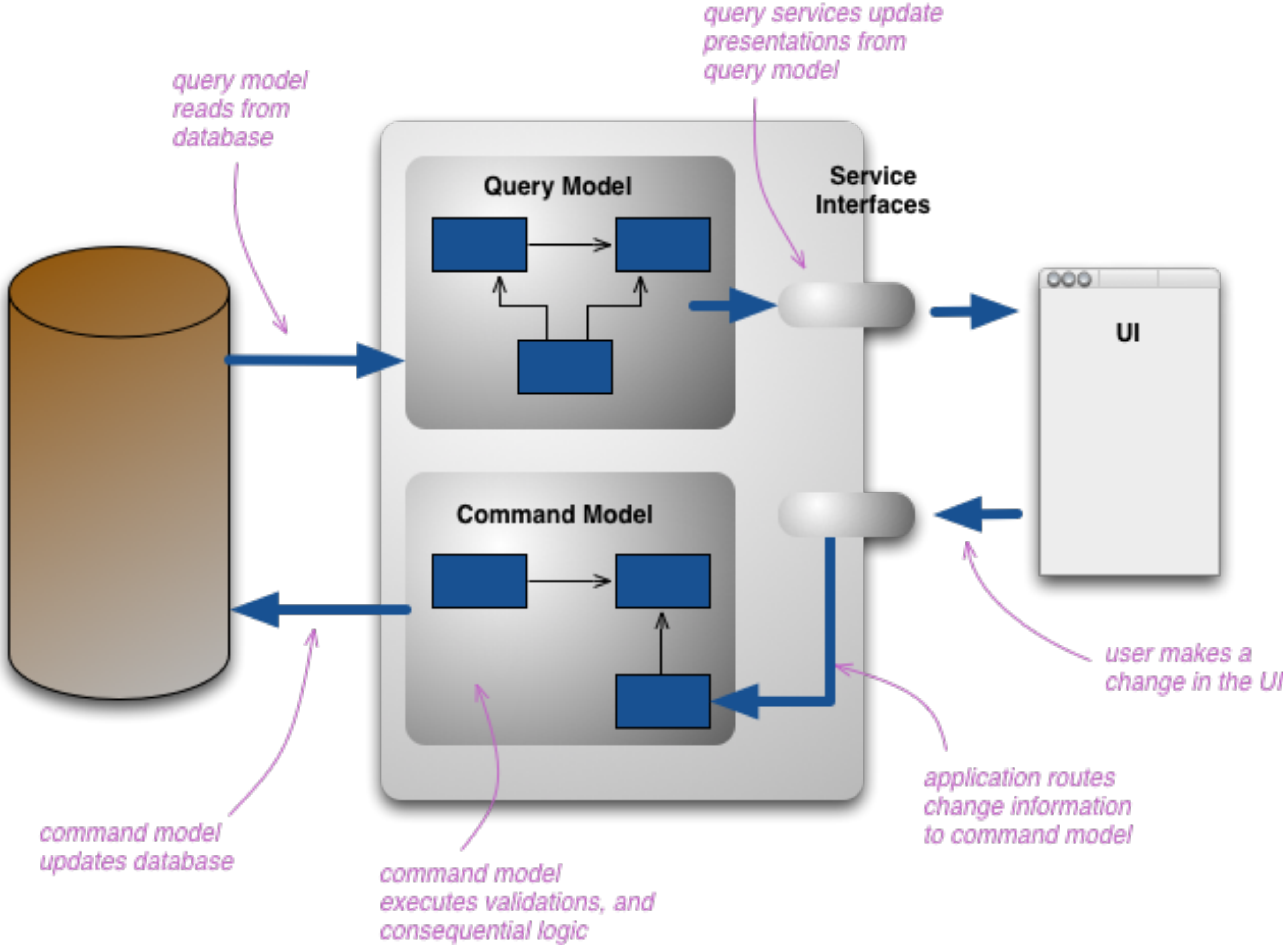


Flux state propagation

state propagates through parent component to child

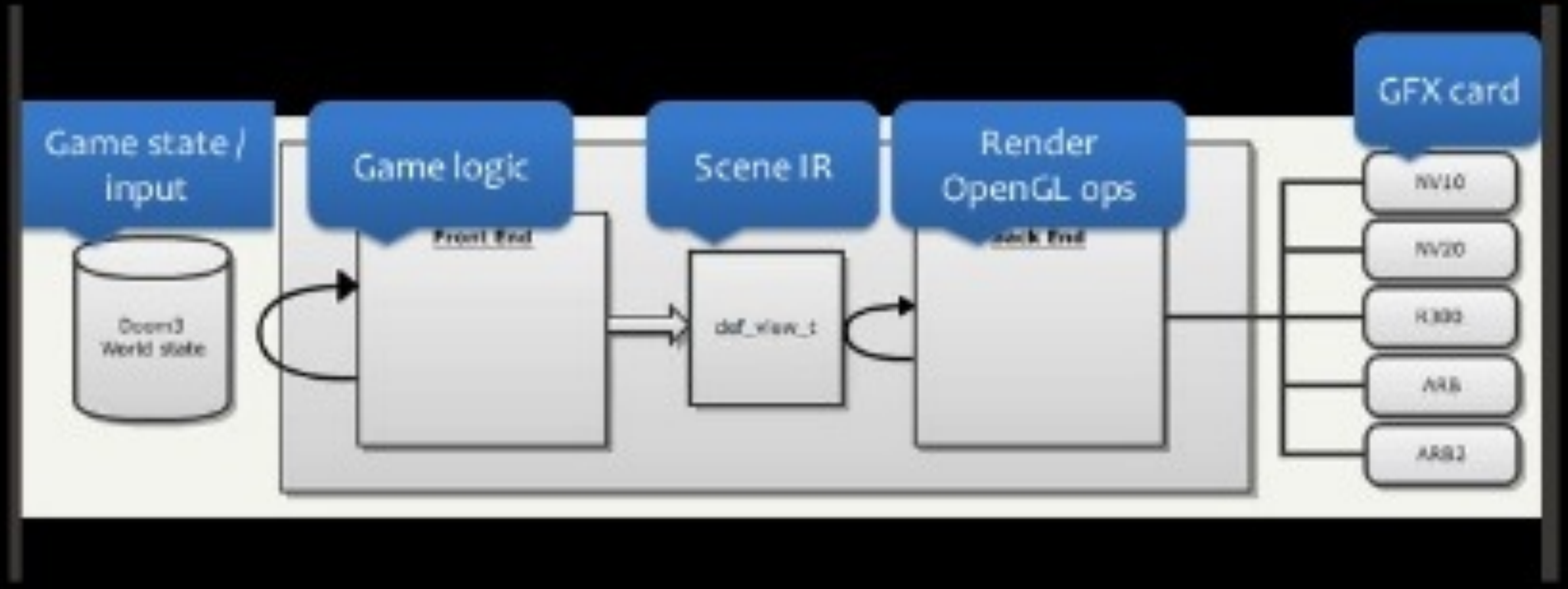


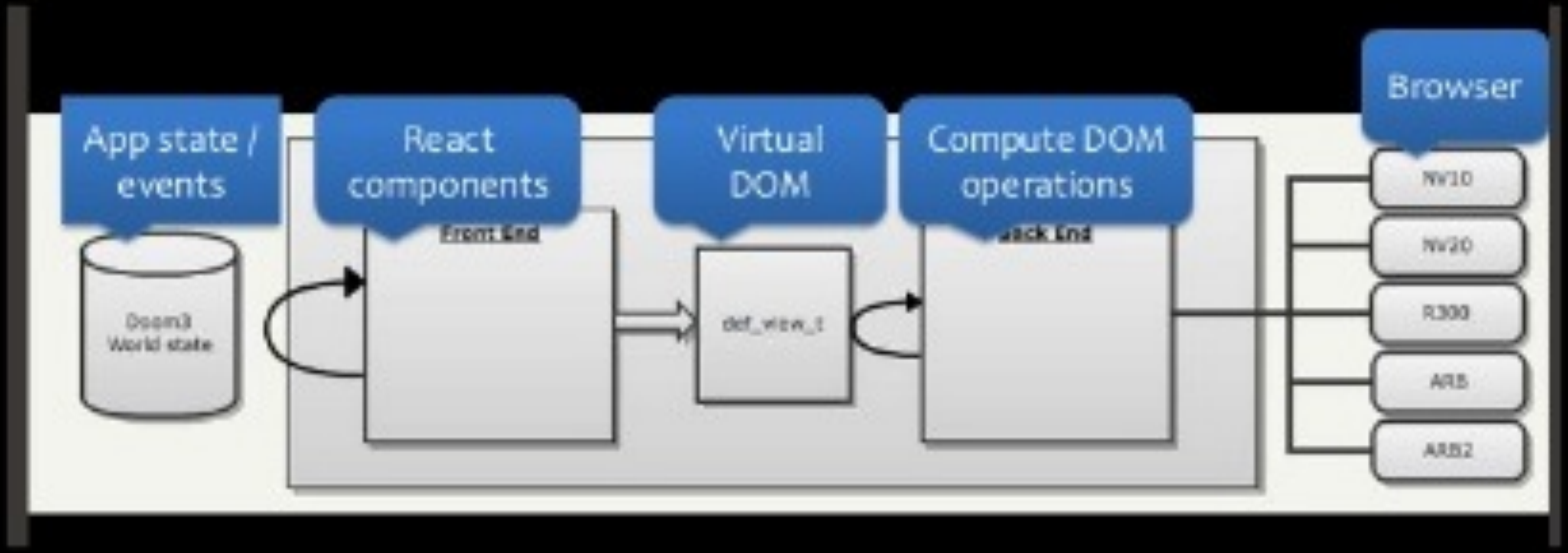
React + Flux(implements Command Query Separation):


query services update presentations from query model
query model reads from database
Query Model
Service Interfaces
UI
Command Model
user makes a change in the UI
command model updates database
application routes change information to command model
command model executes validations, and consequential logic

React + Flux(similar to game event loop):

Game event loop





sample code

http://jsfiddle.net/d7vget3y/4/

http://jsfiddle.net/d7vget3y/5/

http://jsfiddle.net/d7vget3y/7/

http://jsfiddle.net/d7vget3y/15/

http://jsfiddle.net/d7vget3y/17/

http://jsfiddle.net/d7vget3y/18/

http://jsfiddle.net/d7vget3y/9/

http://jsfiddle.net/d7vget3y/10/

http://jsfiddle.net/d7vget3y/11/

http://jsfiddle.net/d7vget3y/14/