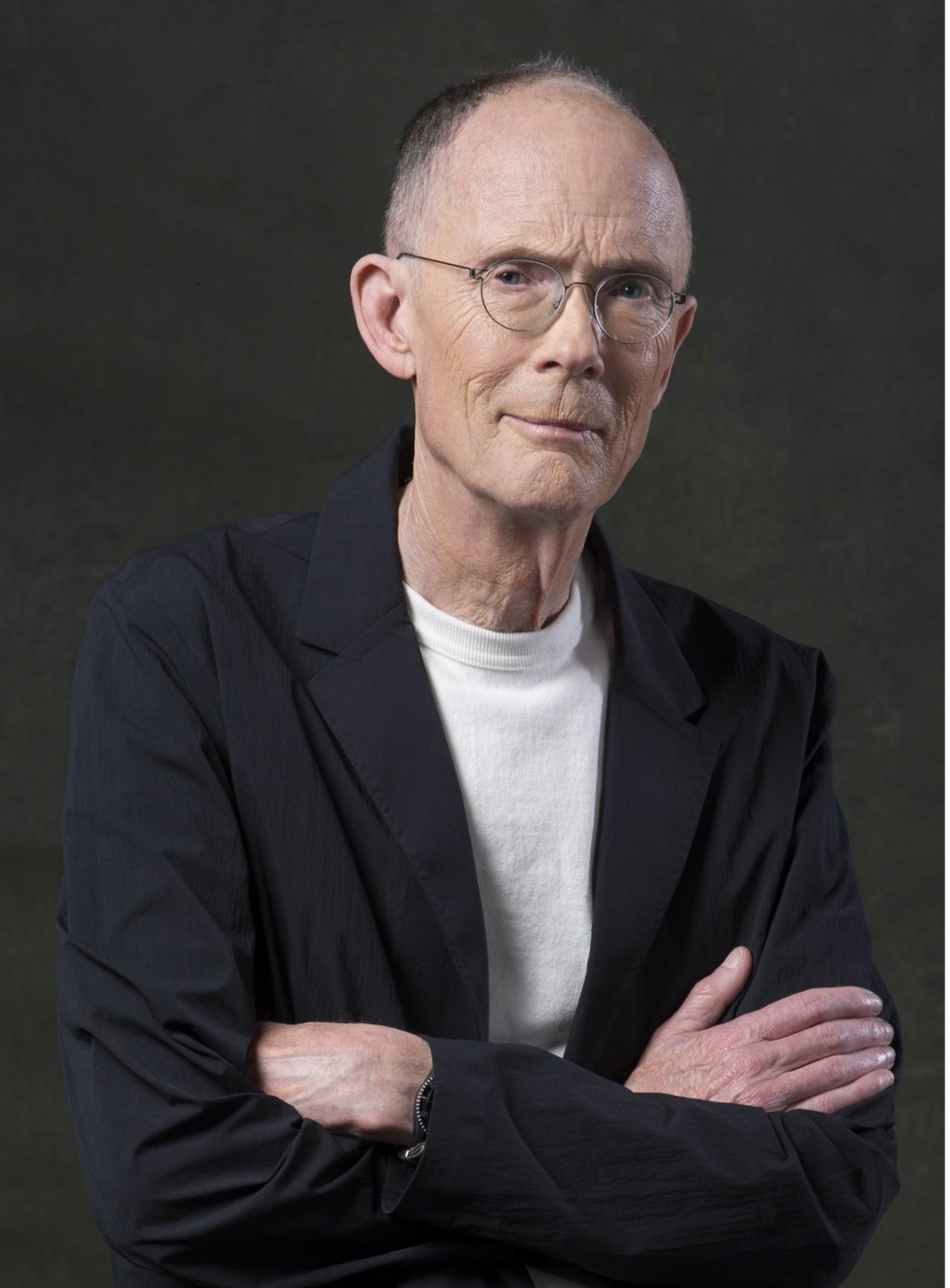
Effective React State

10 Years of Lessons Learned

Cory House
Founderreactjsconsulting.com
Author pluralsight.com
X @housecor



The future is already here. It’s just not evenly distributed.

William Gibson


User experience
Design

Fetching in React over 10 years

2014: Fetch in componentDidMount
2019: Fetch in useEffect
2020: Custom useFetch hook
2021: React query
2022: React query with useErrorBoundary
2023: React query with useSuspenseQuery
2024: React Server Components

Eight Ways to Handle State in React Apps

	When to use it
URL	Sharable app location
Web storage	Persist between sessions, one browser
Local state	Only one component needs the state
Lifted state	A few related components need the state
Derived state	State can be derived from existing state
Refs	DOM reference, state that isn't rendered
Context	Global or subtree state
Third party library	Global state, Remote state

30 Ways to Handle React State

Built in
useState
useReducer
useRef
useContext
useOptimistic
useSyncExternalStore
useActionState
React Server component

Web platform
URL
Cookie
localStorage
sessionStorage
indexDB

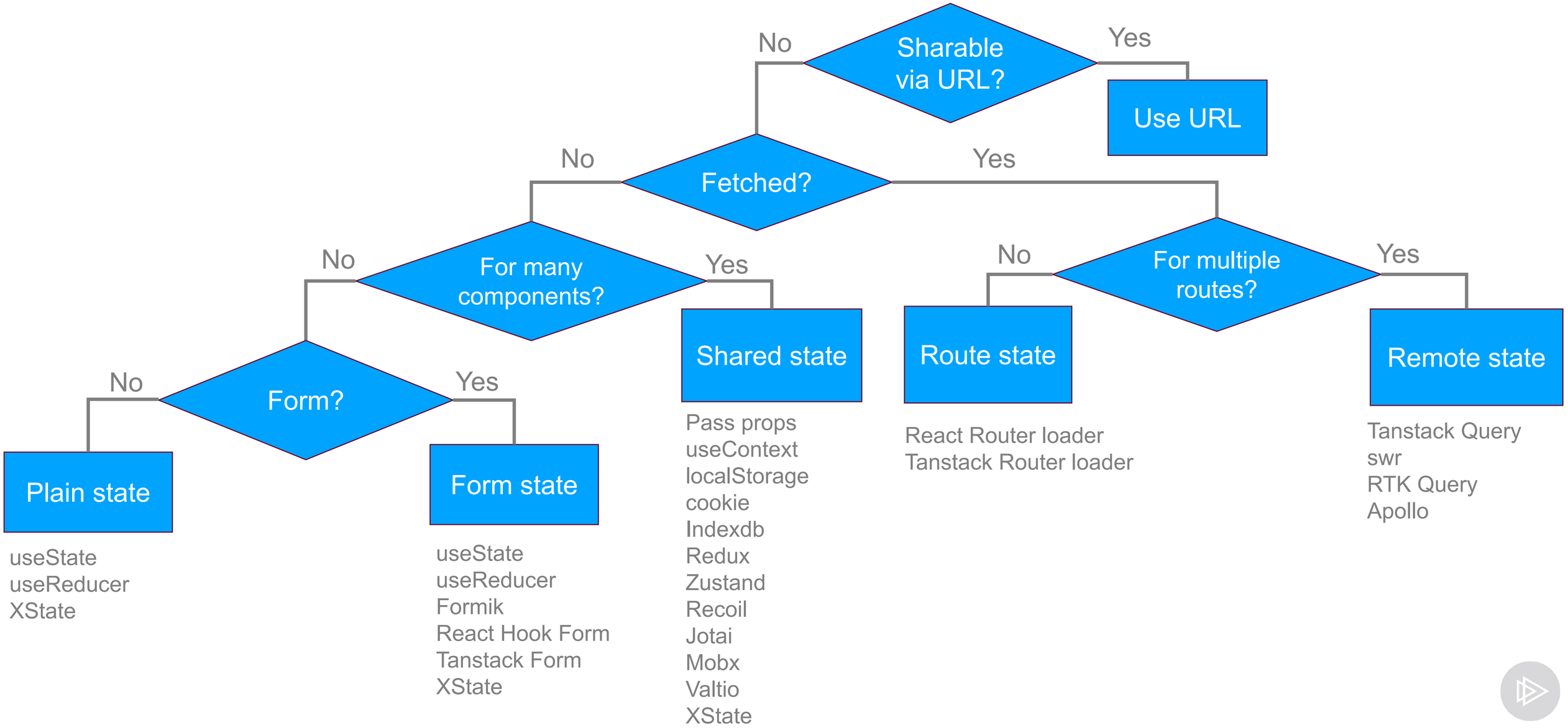
General state
Redux
Zustand
Jotai
Valtio
Mobx
Recoil
Xstate

Remote state
Tanstack query
swr
Apollo
RTK query

Route state
React Router loader
Remix loader
Tanstack router loader

Form state
Formik
React Hook Form
Tanstack Form

Picking a React State Approach



React State: 10 Years of Lessons Learned

@housecor

1. Know the 8 ways to handle state, and when to use each.
2. Optimize for state locality. Keep components small.
3. Start local, then lift. Prop drilling is over-dramatized. Global is a last resort.
4. Most state is remote. Use RSC, Tanstack Query, Apollo, etc.
5. Context is overused. Consider Zustand or Jotai instead.
6. Normalize state. Derive on render.
7. Embrace immutable JS features. You don't need LoDash, Underscore, etc.
8. You don't need a form library. You need a pattern.
9. State Enums are 🔥. You likely don't need XState.
10. Use TypeScript and validate runtime inputs via Zod.

Why use Third-party State?

State Hook Limitations

Can’t “protect” state, not global

Clunky DX, not global

Only accessible in React

```
const [loading, setLoading] = useState(false);

const [state, dispatch] = useReducer(myReducer, initialState);

const countRef = useRef(0);

const selectionContext = useContext(SelectionContext);
```

Doesn’t render

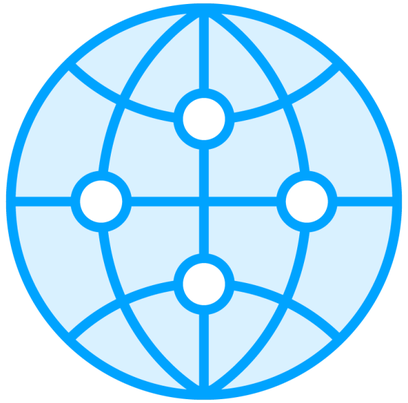
For data that changes *infrequently*

No granular render optimizations

Why Third-party State?

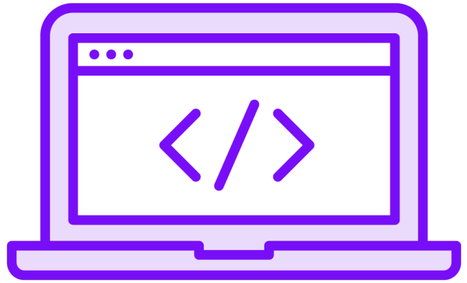
Performance

Render optimization
Caching



Flexibility

Access outside React
Global by default
“Protect” state

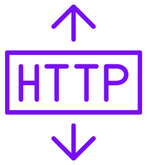


DX

Less code
Devtools

When Is Third-Party State Helpful?

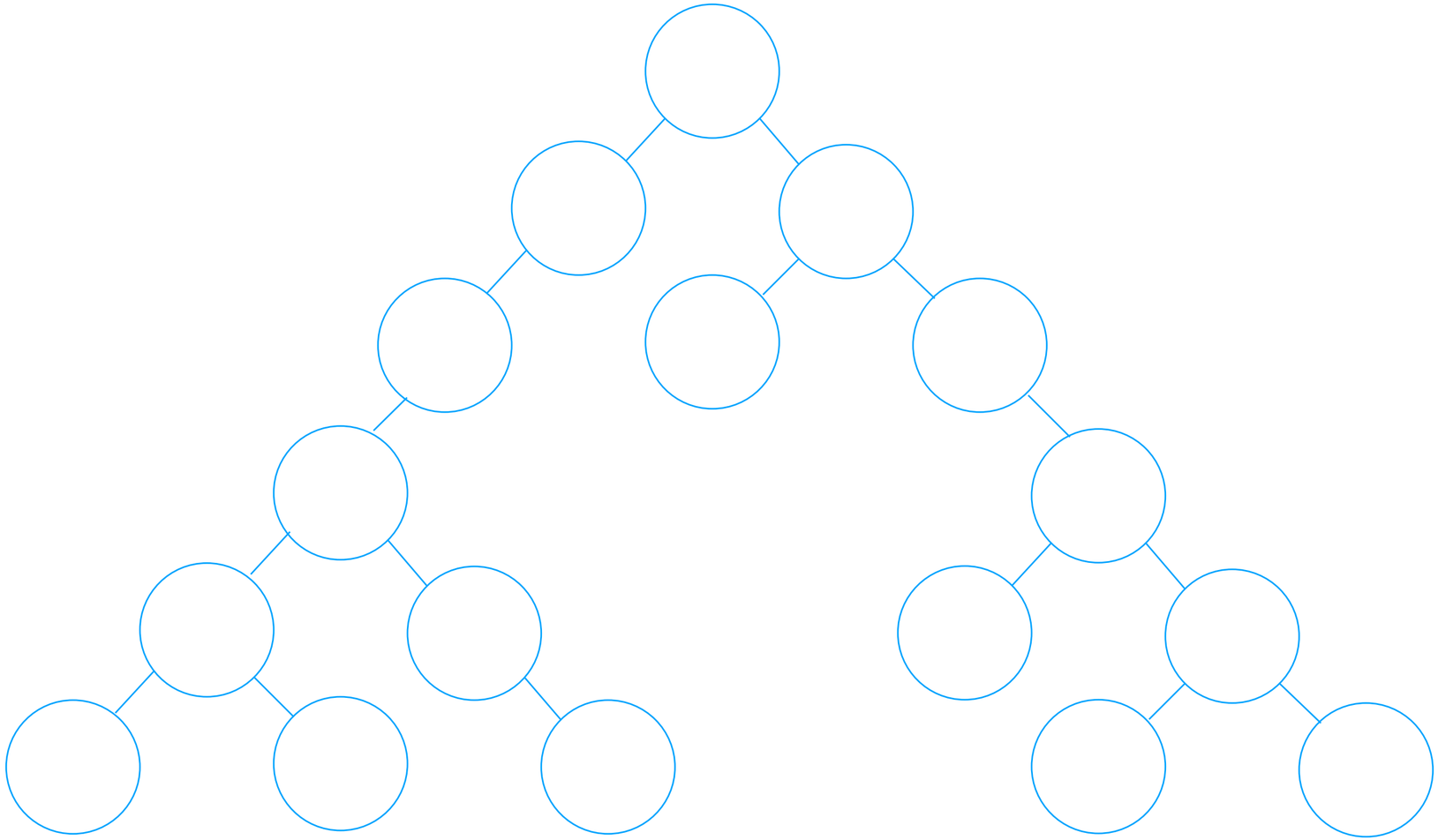
Same data or actions used in many places



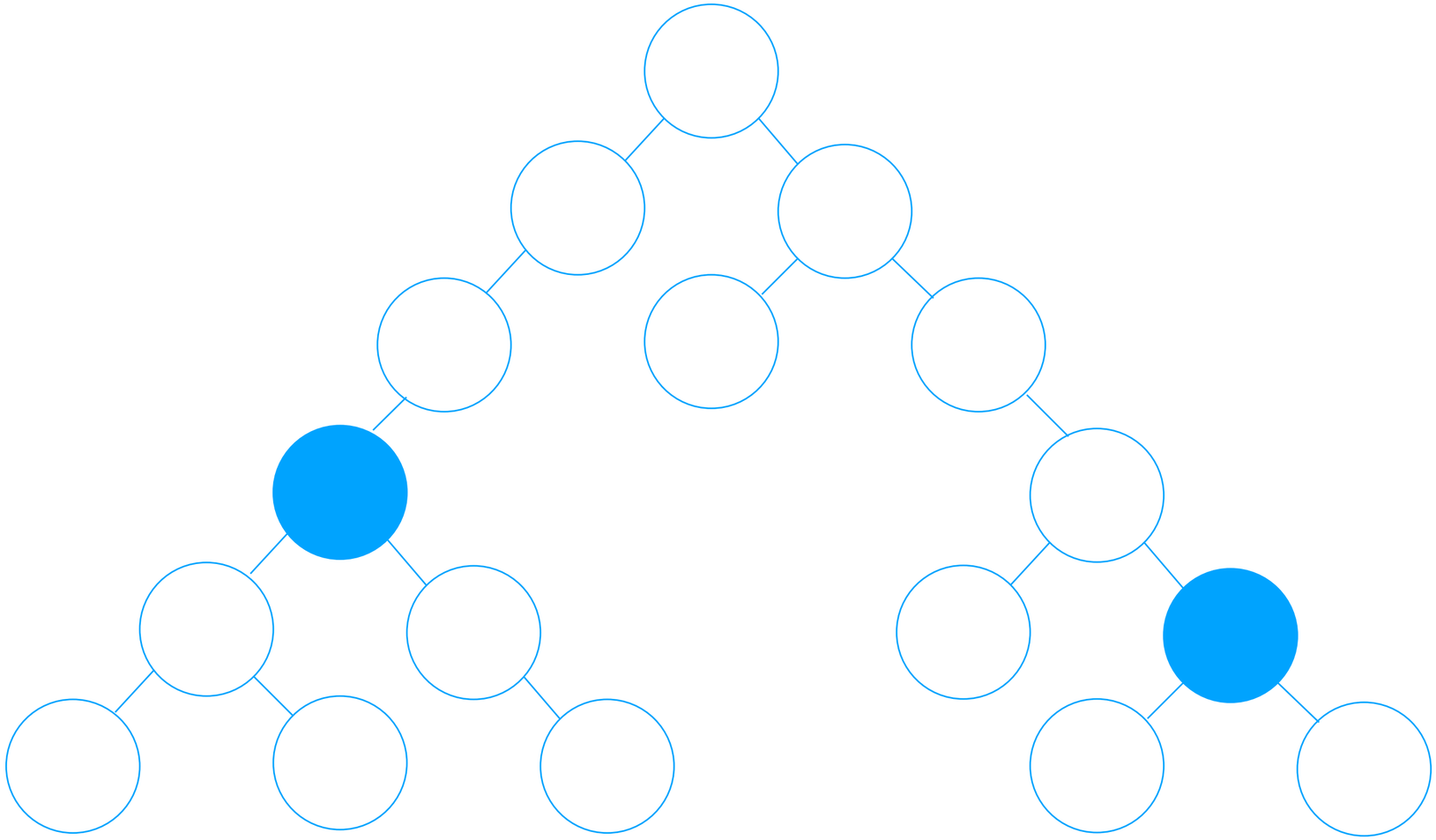
Many fetch calls

Complex multi-step forms

When to Consider Global State

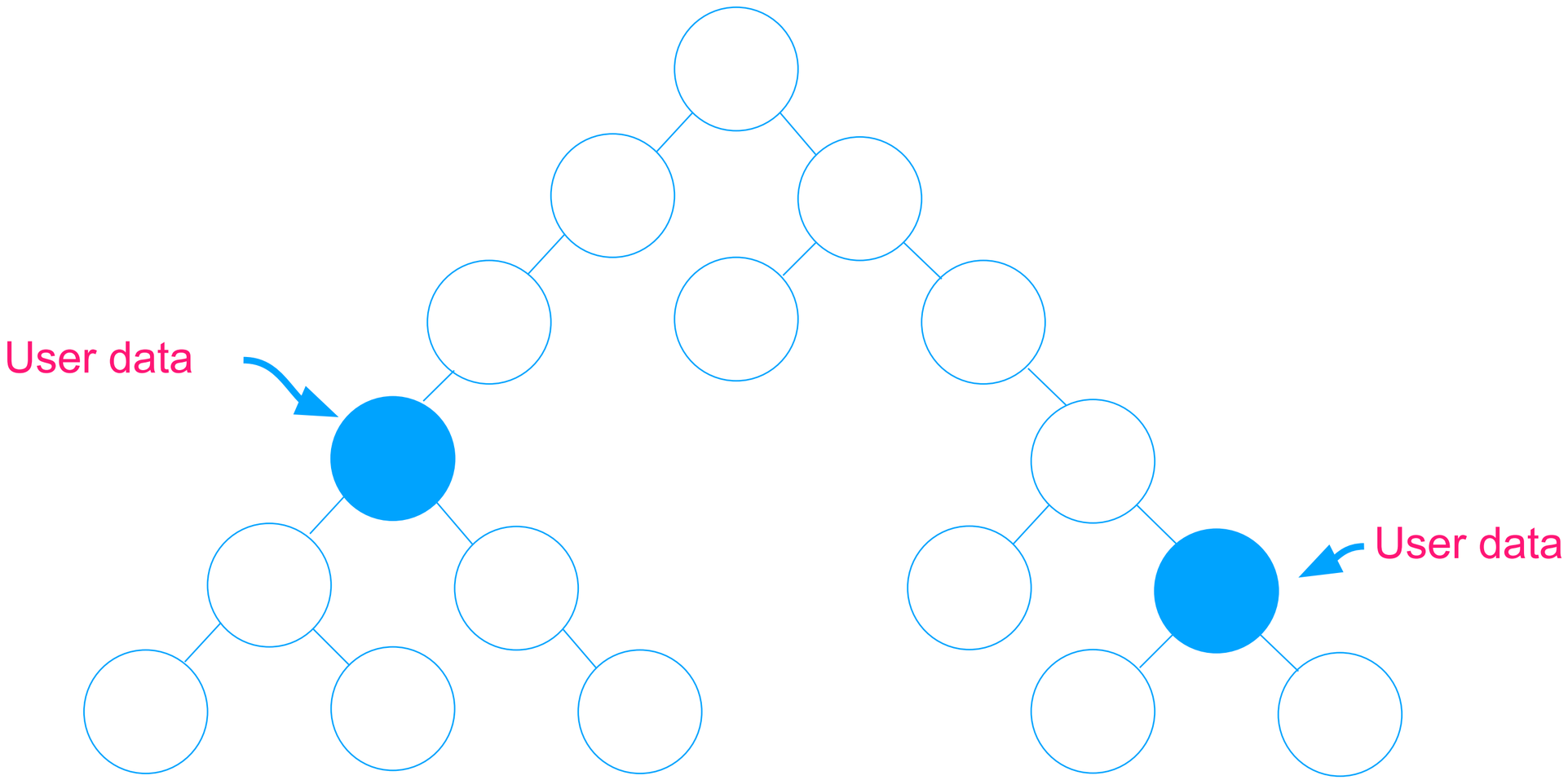


What if distant components need the same data?



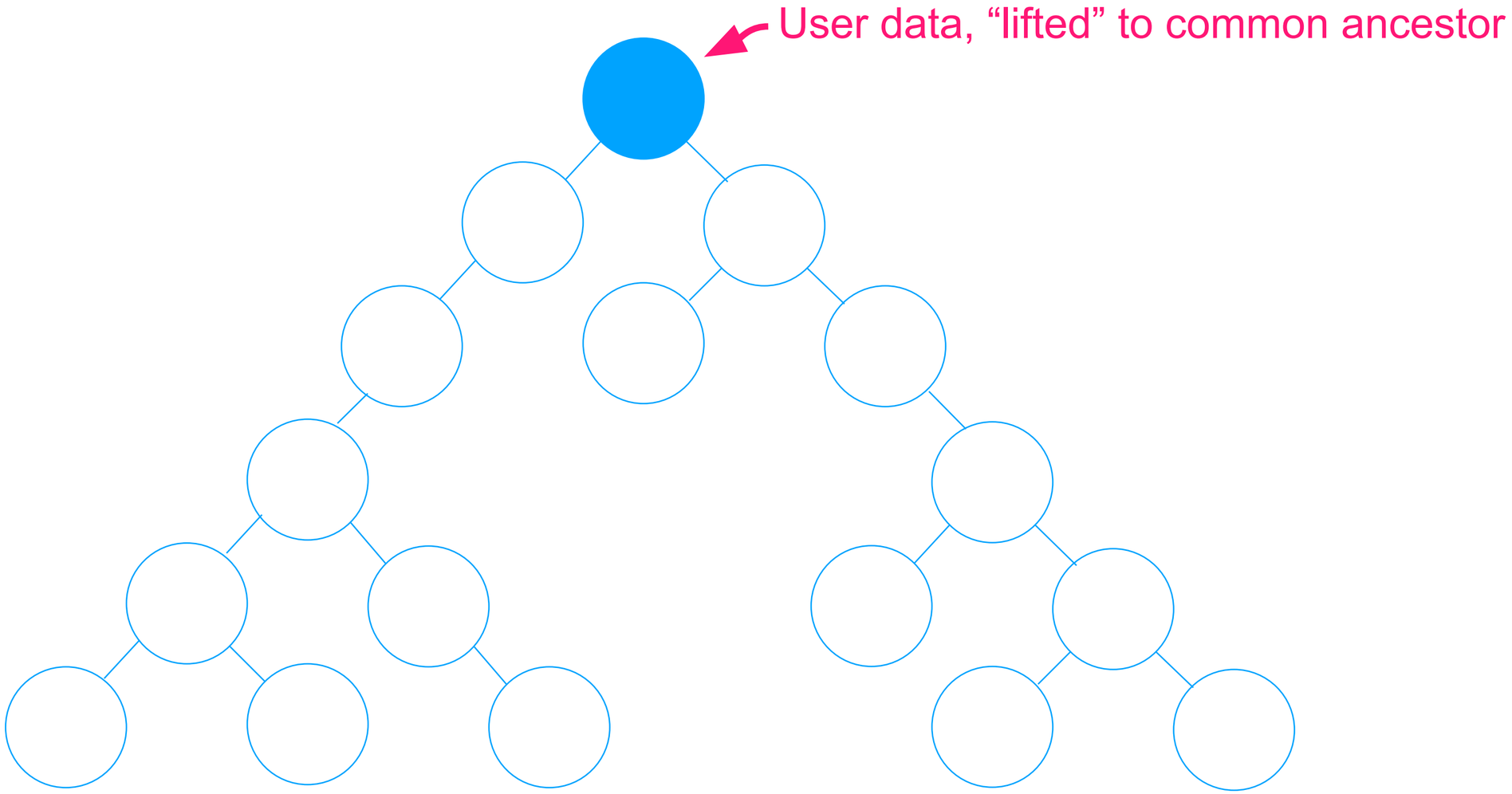
3 Solutions

1. Lift State



3 Solutions

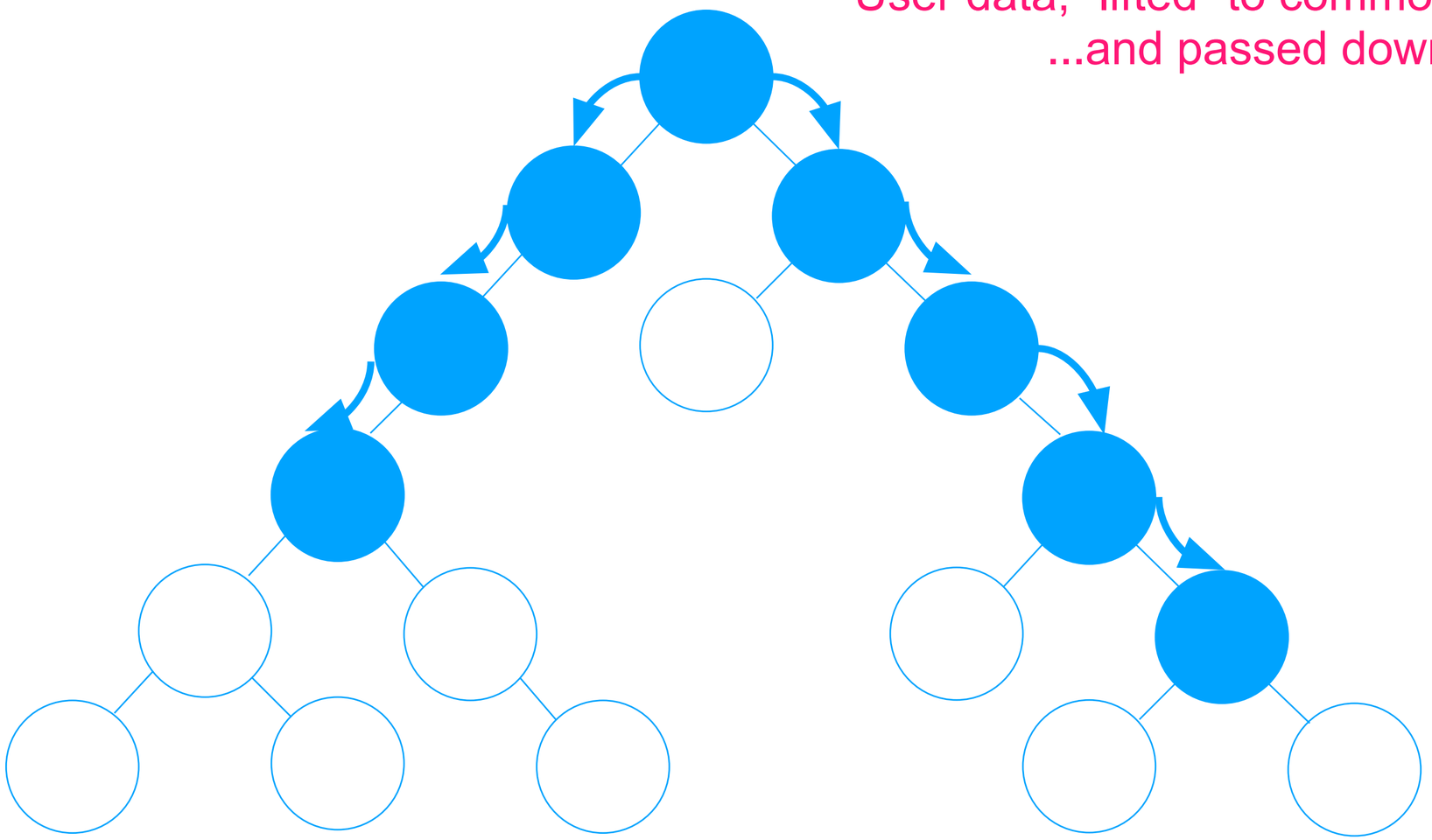
1. Lift State



3 Solutions

1. Lift State

User data, “lifted” to common ancestor
...and passed down to children

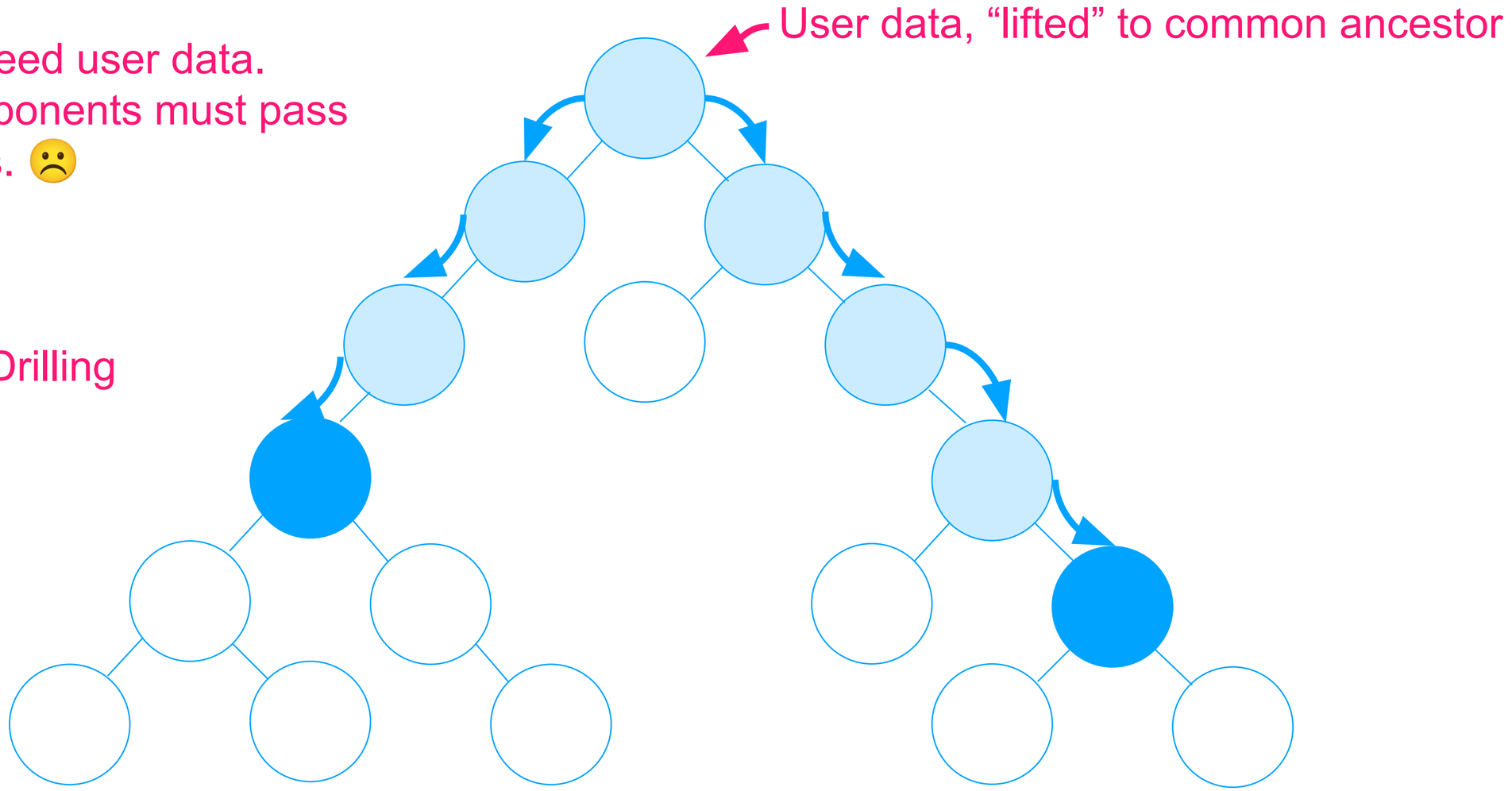


3 Solutions

1. Lift State

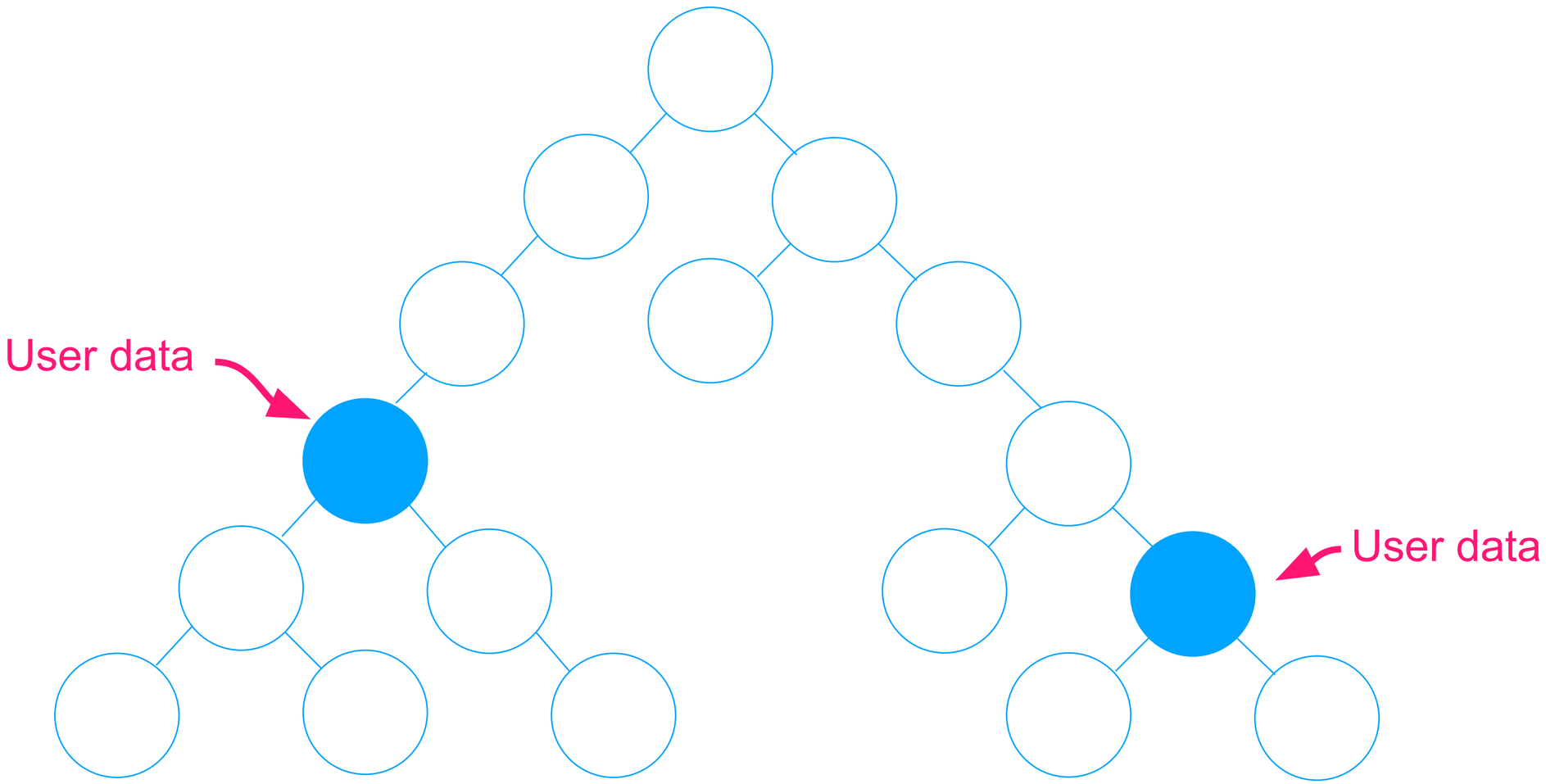
2 components need user data.
But 6 other components must pass it down on props. ☹️

Problem: Prop Drilling



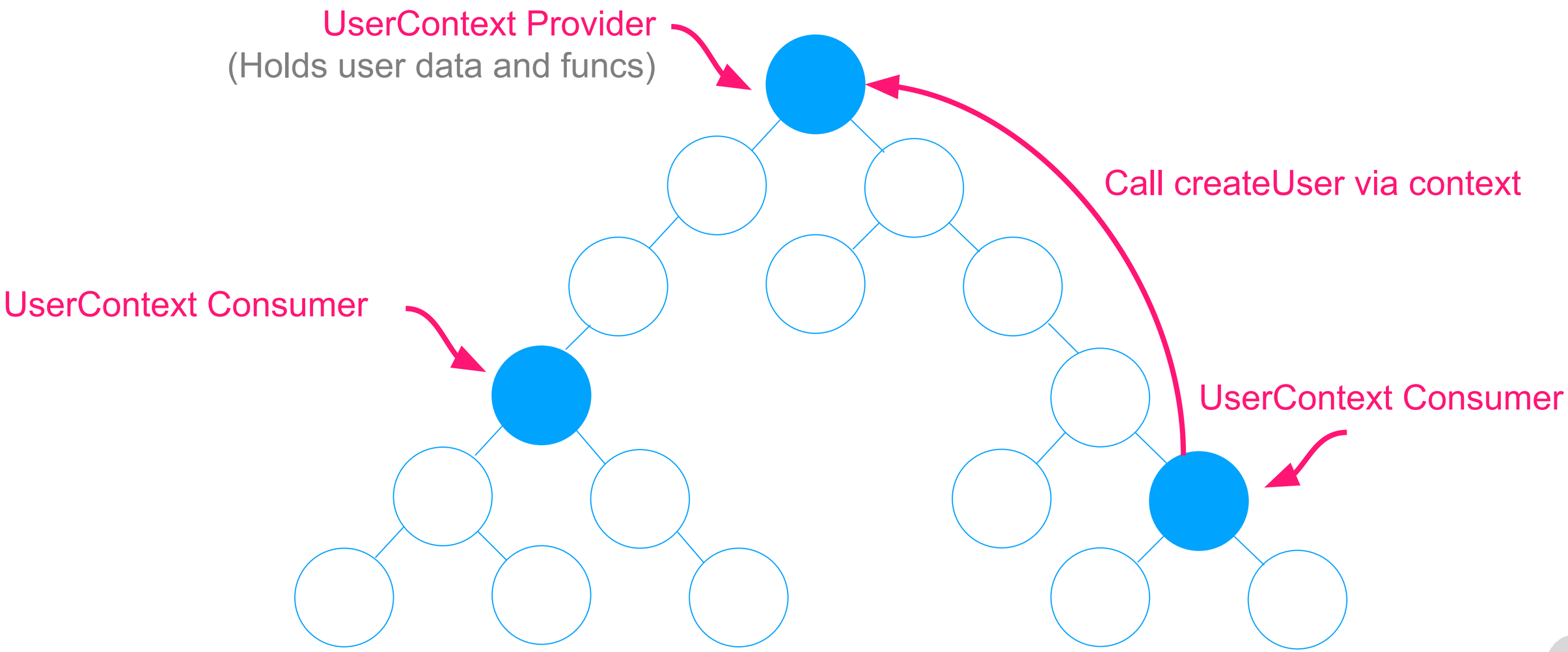
3 Solutions

1. Lift State
2. React context



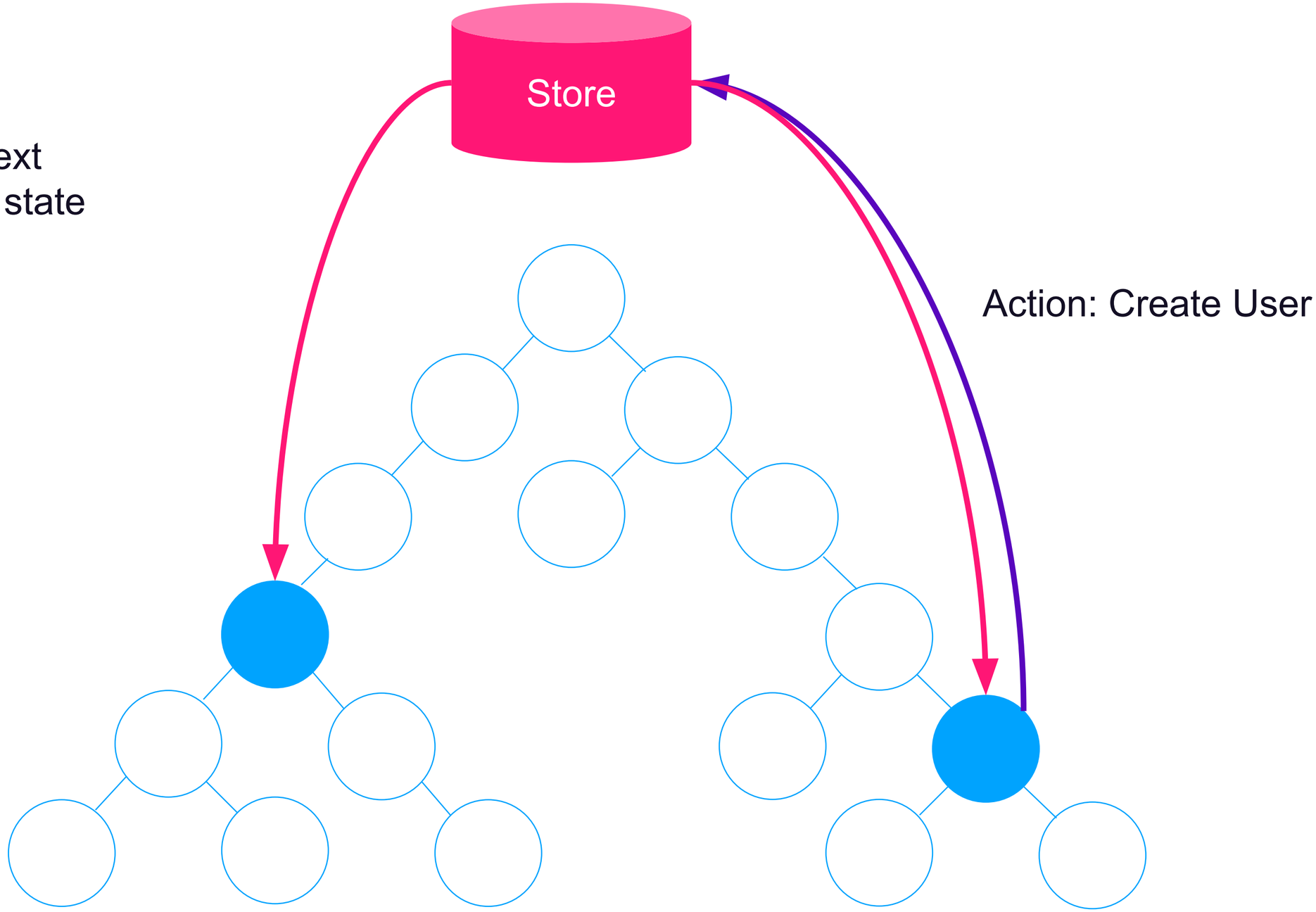
3 Solutions

1. Lift State
2. React context



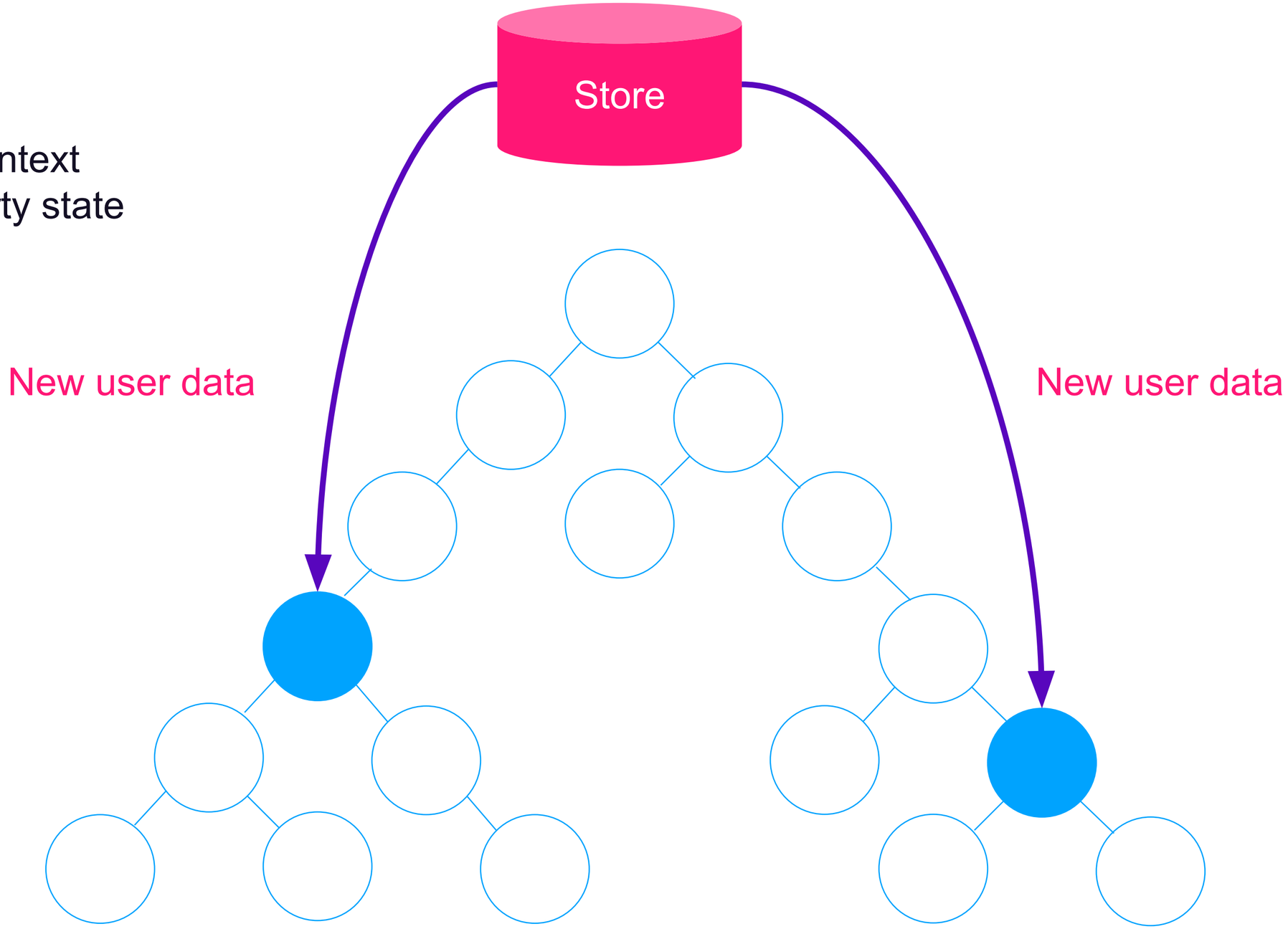
3 Solutions

1. Lift State
2. React context
3. Third-party state



3 Solutions

1. Lift State
2. React context
3. Third-party state



Third-party State Options

State Libraries by Category

Redux

Unidirectional

recoil

Jōtai

Atomic

Mobx

valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

State Libraries by Category

Redux

Unidirectional

recoil

Jōtai

Atomic

Mobx

valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

Redux

Unidirectional

Store state outside of React
Protect changes to the state
You create the state change API
Redux: one store, Zustand: multiple

Recoil

Jōtai

Atomic

Store small pieces of state
State is unprotected
Granular state updates

Store small pieces of state
Wrap state in a proxy object
State is mutable
Automatically optimizes performance

Mobx valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

Store data fetched from a server
Handle loading and error state
Cache and dedupe requests
Automatically refetch when needed.

Third-party State: Key Differences

General vs. Specific

State Libraries by Category

General

Redux

Unidirectional

recoil

Jōtai

Atomic

Mobx

valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

Fetched Data

Third-party State: Key Differences

General vs. Specific

Mutable vs. Immutable

Mutable vs Immutable State

Immutable

Redux

Unidirectional

recoil

Jōtai

Atomic

Mutable

Mobx

valtio

Proxy

TanStack Query

APOLLO

SWR

RTK Query

Remote

Varies

Immutable vs. Mutable State

Zustand

```
const useStore = create((set) => ({
  count: 1,
  inc: () => set((state) => (
    { count: state.count + 1 }
  )),
}));
```

Valtio

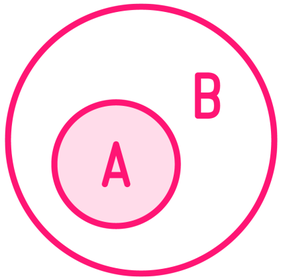
```
const state = proxy({
  count: 1
});

const incrementCount = () => {++state.count};
```

Third-party State: Key Differences

General vs. Specific

Mutable vs. Immutable



External vs. Internal

Internal vs External

External

Redux

Unidirectional

Internal

Recoil

Jōtai

Atomic

External

External

Mobx valtio

Proxy

Varies

TanStack Query

APOLLO

SWR

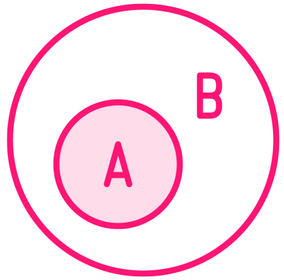
RTK Query

Remote

Third-party State: Key Differences

General vs. Specific

Mutable vs. Immutable



External vs. Internal

Auto vs. Manual

Render Optimization

Manual

Redux

Unidirectional

recoil

Jōtai

Atomic

Automatic

Mobx valtio

Proxy

Automatic

TanStack Query

APOLLO

SWR

RTK Query

Remote

Manual vs. Automatic Render Optimization

Zustand – Manual selector

```
const Component = () => {
  const count = useStore(state => state.count)
  return <p>{count}</p>
}
```

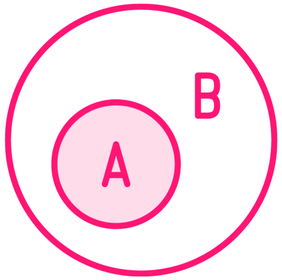
Valtio - Automatic

```
const Component = () => {
  const snap = useSnapshot(store)
  return <p>{snap.count}</p>;
}
```

Third-party State: Key Differences

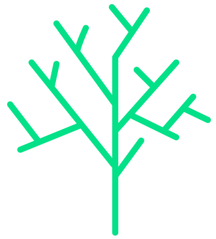
General vs. Specific

Mutable vs. Immutable



External vs. Internal

Auto vs. Manual



One Store vs. Multiple

Number of Stores

One

Redux

Many

recoil

Jōtai

Mobx

valtio

One

TanStack Query

APOLLO

SWR

RTK Query

Top-down vs Bottom-up

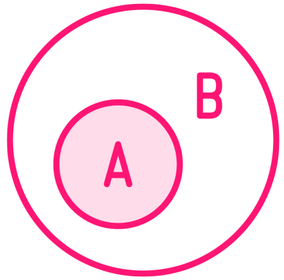
Top-down
Start with the overview,
then add details.

Bottom-up
Start with small pieces of state, then
compose them.

Third-party State: Key Differences

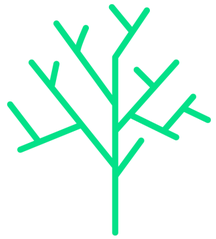
General vs. Specific

Mutable vs. Immutable

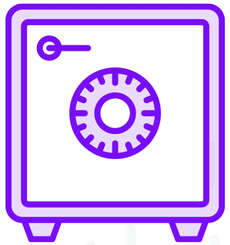


External vs. Internal

Auto vs. Manual



One Store vs. Multiple



Protected vs. Unprotected

“Protected” State

Protected

Redux

Unidirectional

Direct access

recoil

Jōtai

Atomic

Mobx

valtio

Proxy

Varies

TanStack Query

APOLLO

SWR

RTK Query

Remote

Protected vs. Unprotected State

Zustand

```
// userStore.js
export const useStore = create((set) => ({
  user: {
    id: 1,
    name: "Cory House",
  },
  logout: () => set({ user: null }),
}));

const { user, logout } = useStore();
```

Jotai

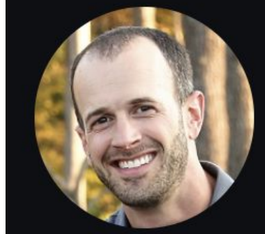
```
// userAtom.js
export const userAtom = atom({
  id: 1,
  name: "Cory House",
});

const [user, setUser] = useAtom(userAtom);
```

Courses at Pluralsight.com

@housecor

Workshops, consulting



Cory House

Pluralsight Author

Following

5610 Followers

Cory is the principal consultant at reactjsconsulting.com, where he has helped dozens of companies transition to React. Cory has trained over 10,000 software developers at events and businesses worldwide. He is a seven time Microsoft MVP, and speaks regularly at conferences around the world. Cory...

Show more...

www.bitnative.com
Twitter
LinkedIn

CONTENT AUTHORED	TOPICS AUTHORED	TOTAL RATINGS	AVG CONTENT RATING
12 All time	React	10,418	4.7

Content authored

Courses

- Building Applications with React and Redux NEW! — Course · Intermediate · 6 hr 38m · Jul 18, 2021 · ★★★★★ (1,811)
- Building a JavaScript Development Environment NEW! — Course · Beginner · 4 hr 55m · Mar 15, 2021 · ★★★★★ (705)
- Managing React State — Course · Intermediate · 5 hr 6m · Aug 19, 2020 · ★★★★★ (143)
- React: The Big Picture — Course · Beginner · 1 hr 10m · May 10, 2020 · ★★★★½ (771)
- Clean Coding Principles in C# — Course · Beginner · 3 hr 19m · Jan 1, 2020 · ★★★★★ (292)
- Building Applications with React and Flux — Course · Intermediate · 5 hr 11m · Jun 18, 2019 · ★★★★★ (1,595)
- Securing React Apps with Auth0 — Course · Intermediate · 3 hr 18m · Nov 29, 2018 · ★★★★★ (149)
- Creating Reusable React Components — Course · Intermediate · 6 hr 20m · Jun 4, 2017 · ★★★★★ (127)