



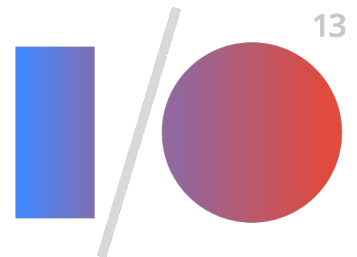
Google
Developers



The Secrets of the Drive Realtime API

A deep dive

Brian Cairns, Cheryl Simon
Software Engineers



Agenda

1. Introducing the Realtime API
2. Secrets of Realtime Collaboration
3. Designing a Good Data Model
4. The Future





Introducing the Realtime API

Realtime collaboration and storage for your apps

Realtime collaboration made easy

Your application

Google Docs-style instant collaboration

Conflict resolution

All Javascript, no messy server

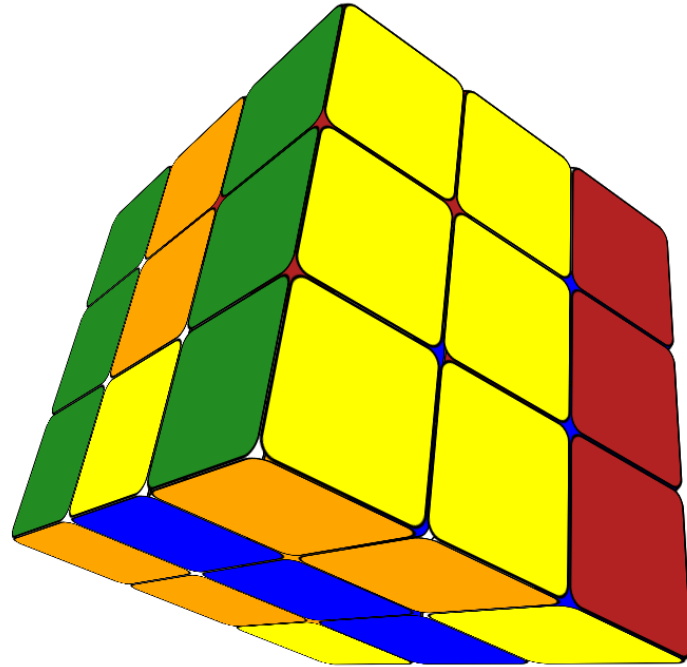
Click by click updates

Collaborator presence

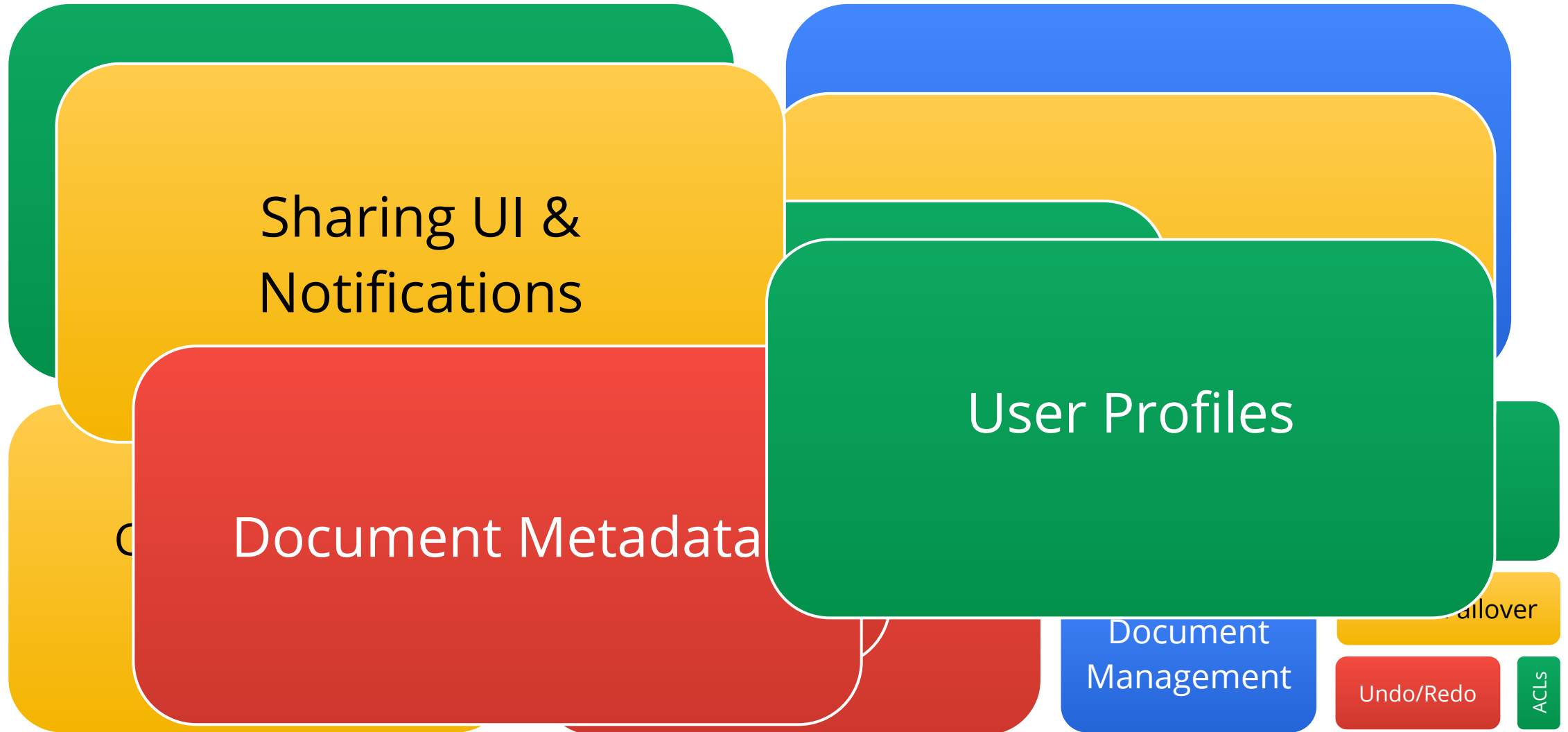
Backed by Google Drive



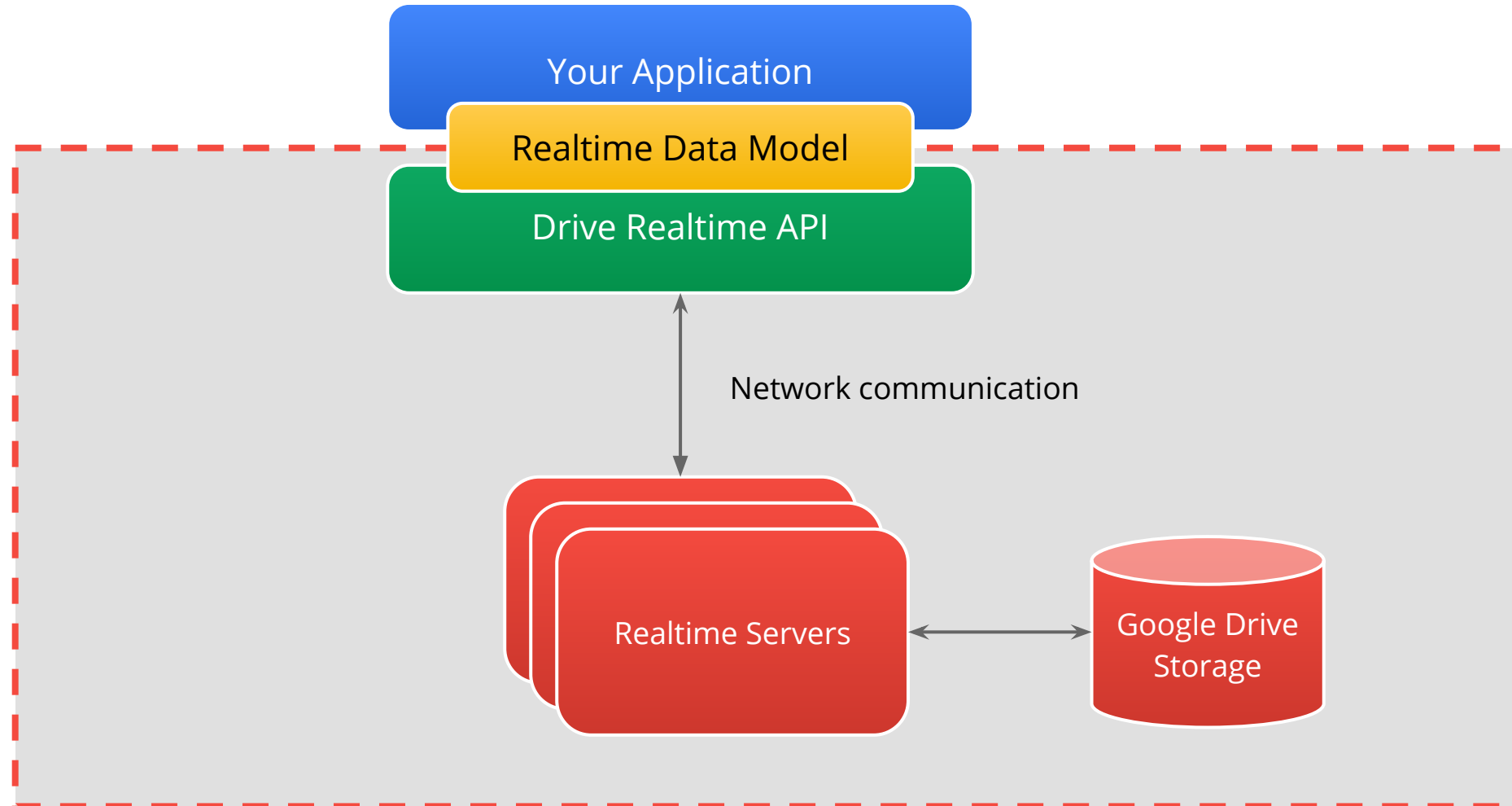
Demo: Realtime Cube



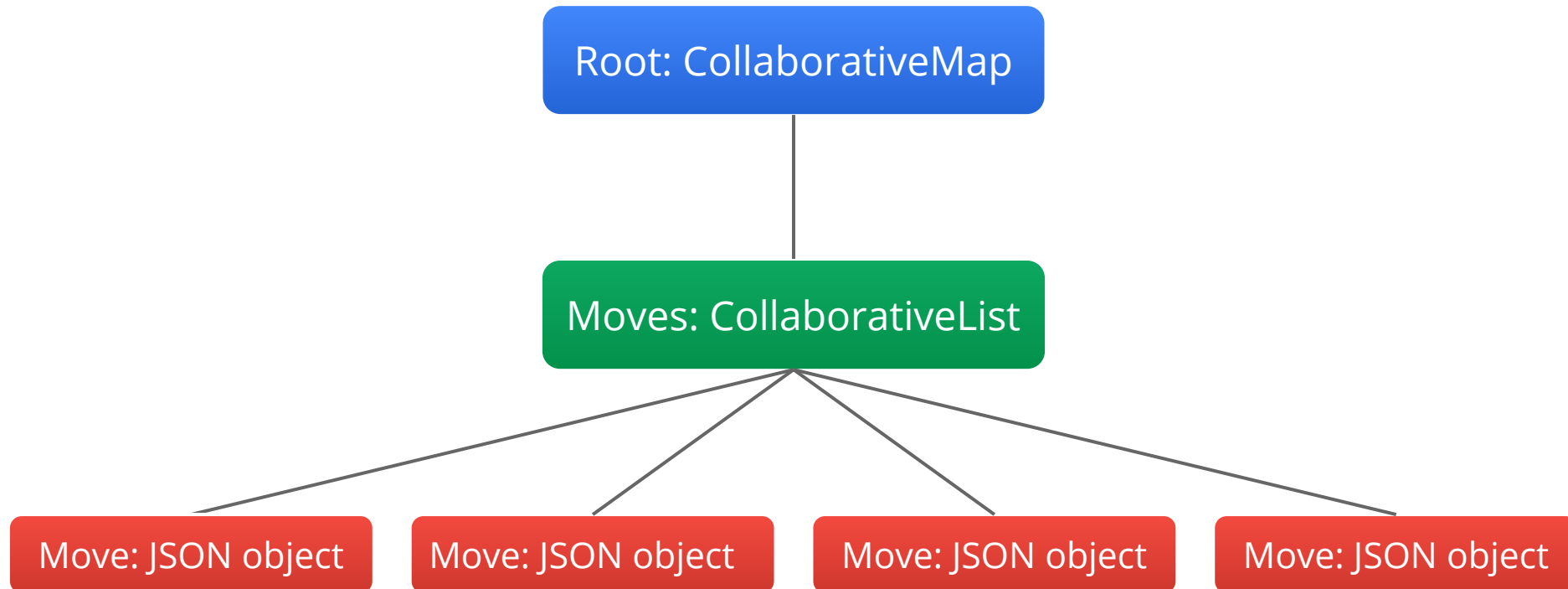
Building a realtime collaboration system



Or we can do it for you



Realtime Cube's data model



Working with the data model

Javascript

```
gapi.drive.realtime.load(fileId, onFileLoaded, initModel);
initModel = function(model) {
    var moves = model.createList();
    model.getRoot().set('moves', moves);
};

onFileLoaded = function(doc) {
    model_ = doc.getModel();
    model_.getRoot().get('moves').addEventListener(
        gapi.drive.realtime.EventType.VALUES_ADDED, updateDisplay);
};

addMove = function(side, start, direction) {
    var newMove = new cube.Move(side, start, direction);
    model_.getRoot().get('moves').push(newMove);
};
```



Apps

Realtime Cube



Multi
Sudoku



gantter 
FREE project scheduling

CollabQuest

Neutron Drive
Code Editor



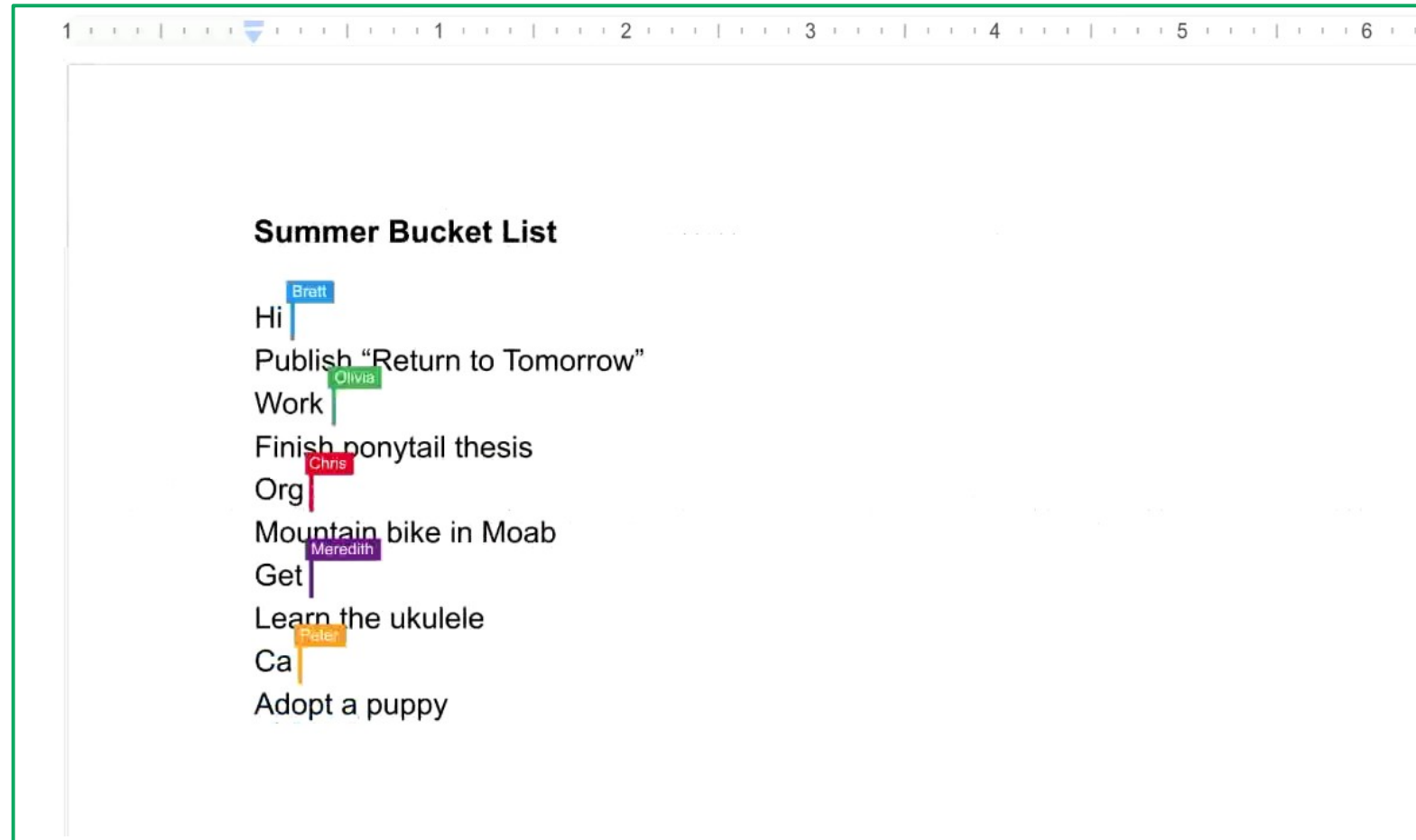
draw.io 





Secrets of Realtime Collaboration

Our goal



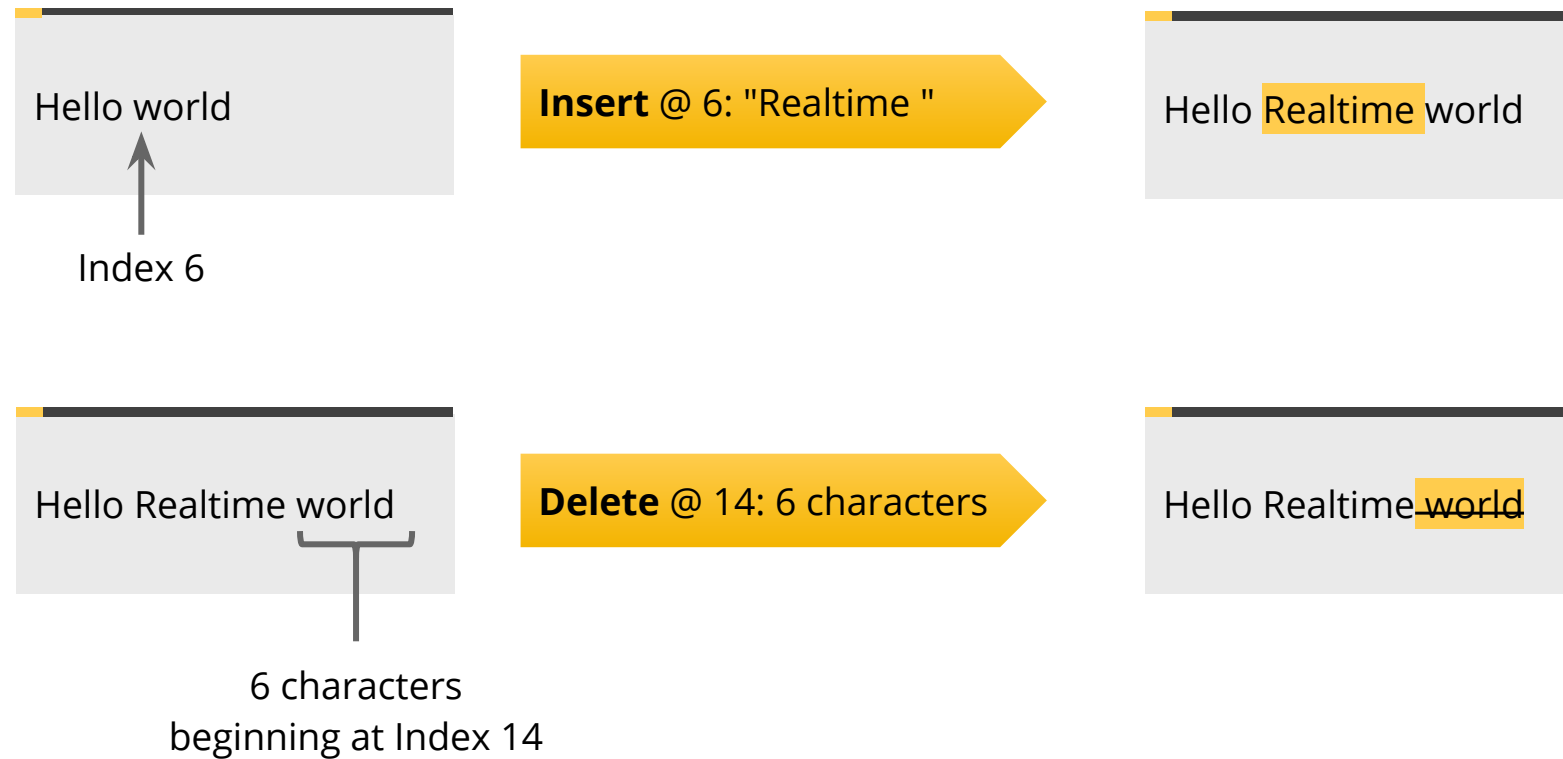
1 2 3 4 5 6

Summer Bucket List

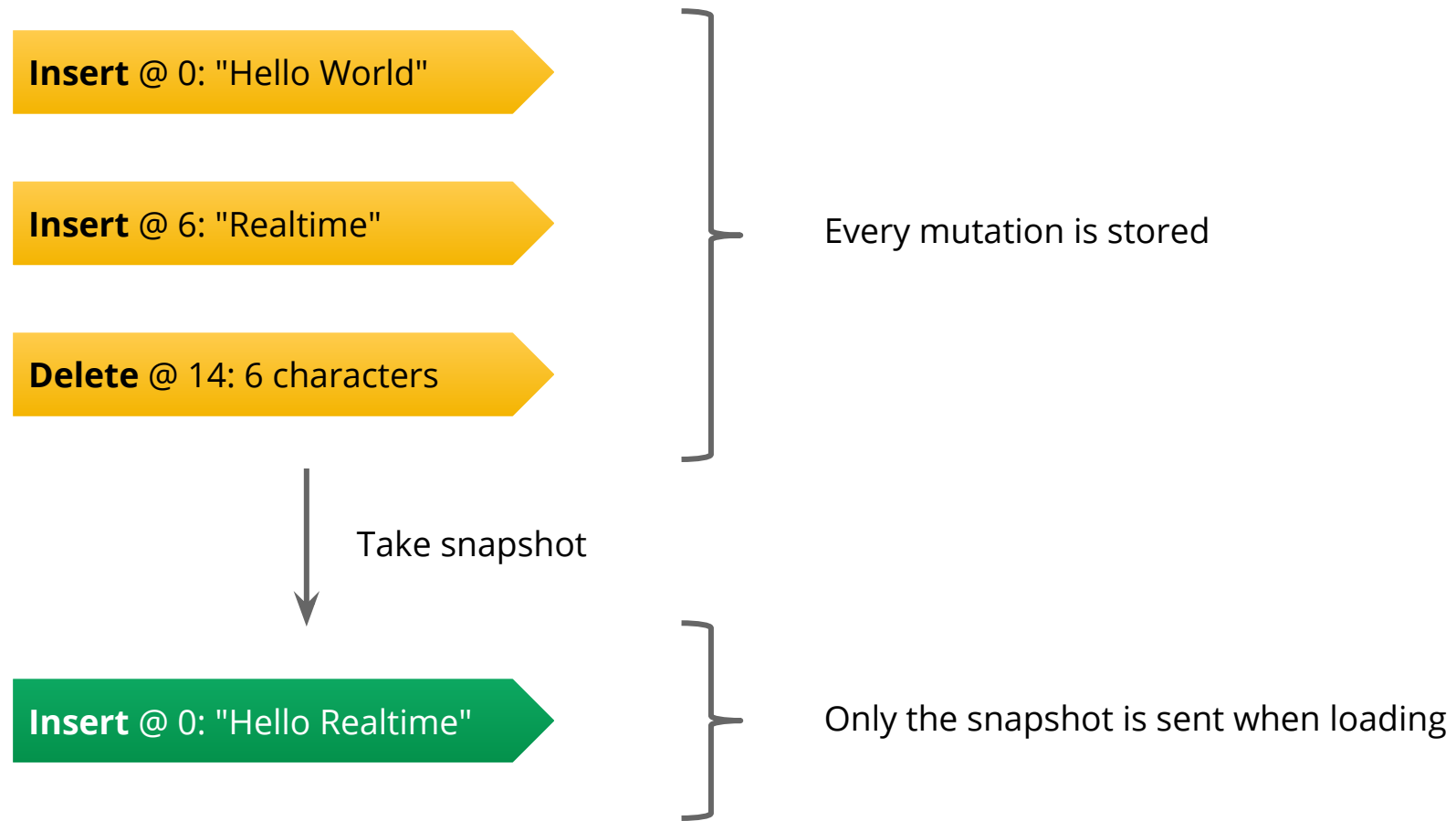
- Hi Bratt
- Publish "Return to Tomorrow" Olivia
- Work Chris
- Finish ponytail thesis Chris
- Org Chris
- Mountain bike in Moab Meredith
- Get Meredith
- Learn the ukulele Pete
- Ca Pete
- Adopt a puppy



Every change to the data model is a mutation



We store the entire mutation history



Local mutations are applied immediately

Alice

The day dawned.

Insert @ 4: "rainy "

The **rainy** day dawned.

Bob

The day dawned.

Insert @ 14: " brightly"

The day dawned **brightly**.



Challenge: Mutations get out of date

The day dawned.

Insert @ 4: "rainy "

Insert @ 14: " brightly"

The rainy day d brightlyawned.



If she applies the mutations as-is, Alice gets the **wrong result**

Example:
Alice's Document

The day dawned.

Insert @ 4: "rainy "

Insert @ 14: " brightly"



Insert @ 20: " brightly"

The rainy day dawned brightly.



Alice needs to apply a **modified version** of Bob's mutation



Solution: Transformation

The server *transforms* Bob's mutation so that it accounts for Alice's mutation

Alice

The day dawned.

Insert @ 4: "rainy "

Insert @ 20: " brightly"

The **rainy** day dawned **brightly**.



Bob

The day dawned.

Insert @ 14: " brightly"

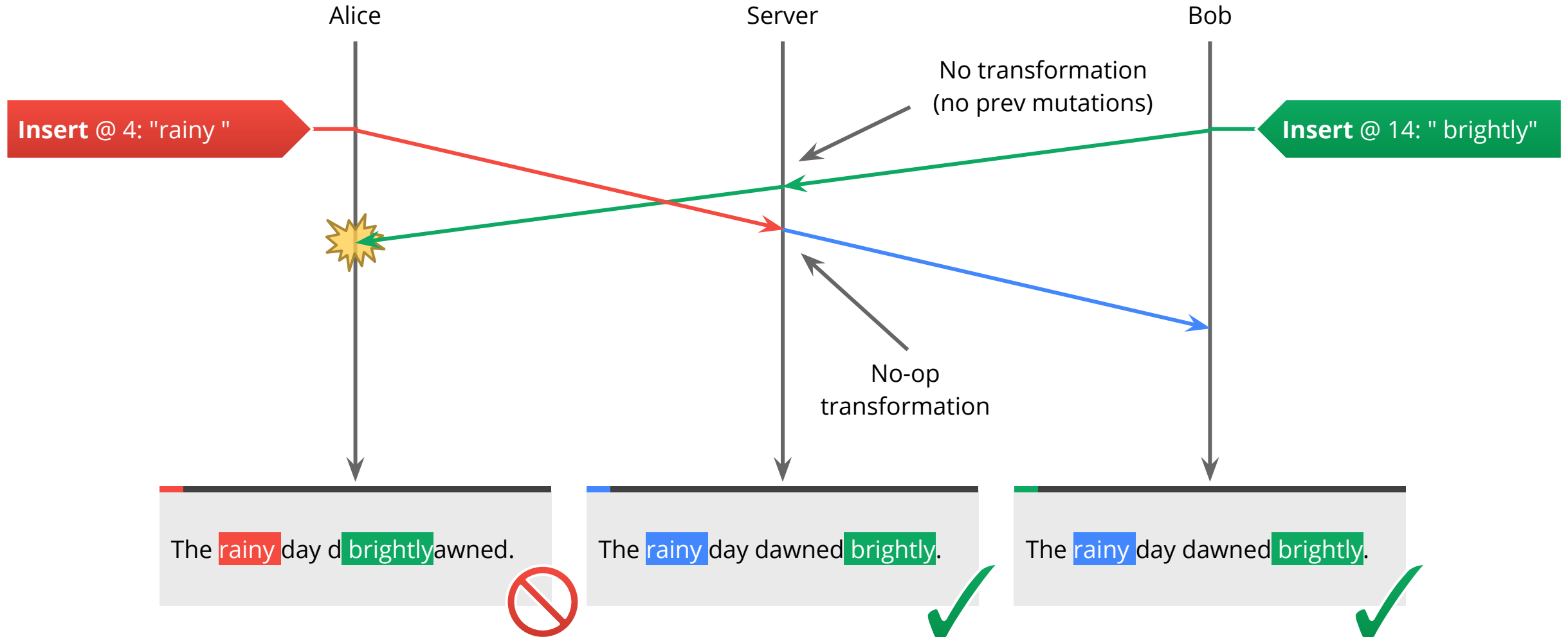
Insert @ 4: "rainy "

The **rainy** day dawned **brightly**.



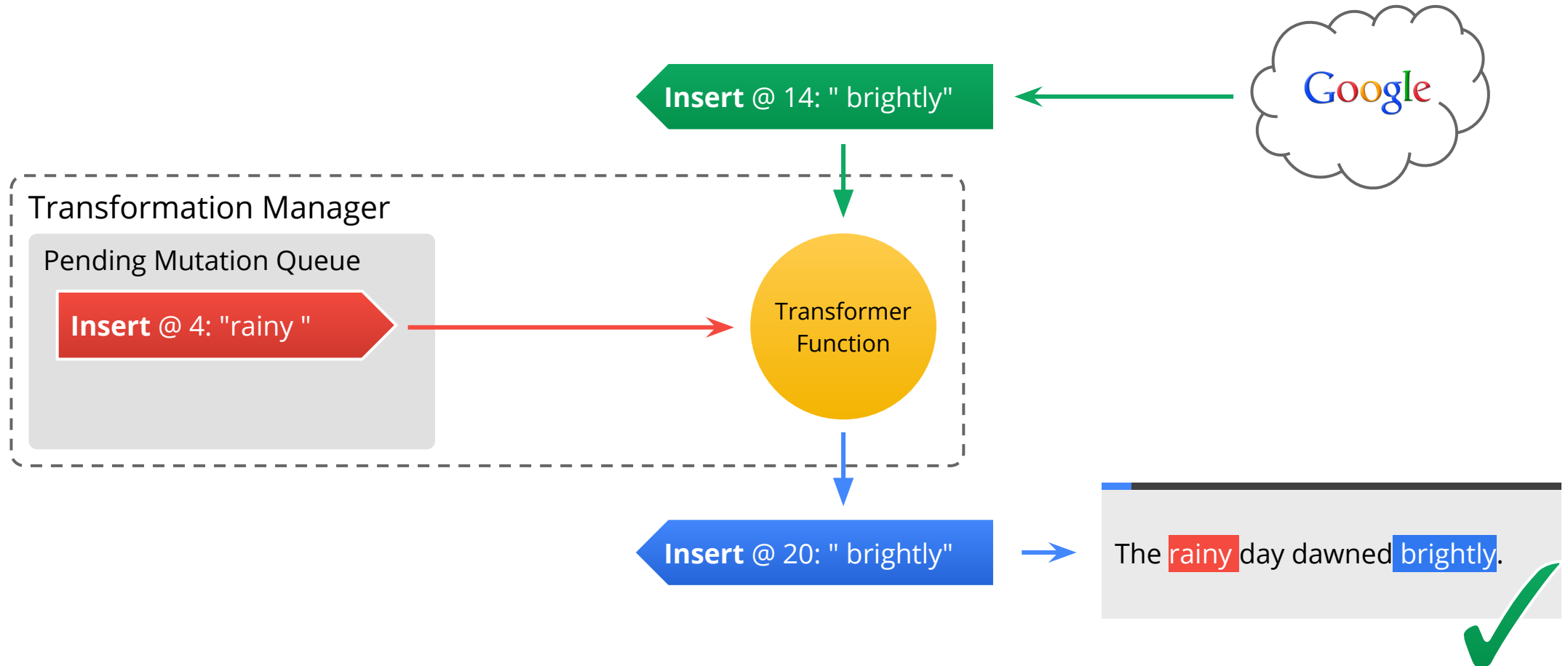
Challenge: The server can't transform every change

The server might not know about Alice's mutation until *after* it sends Bob's mutation to Alice



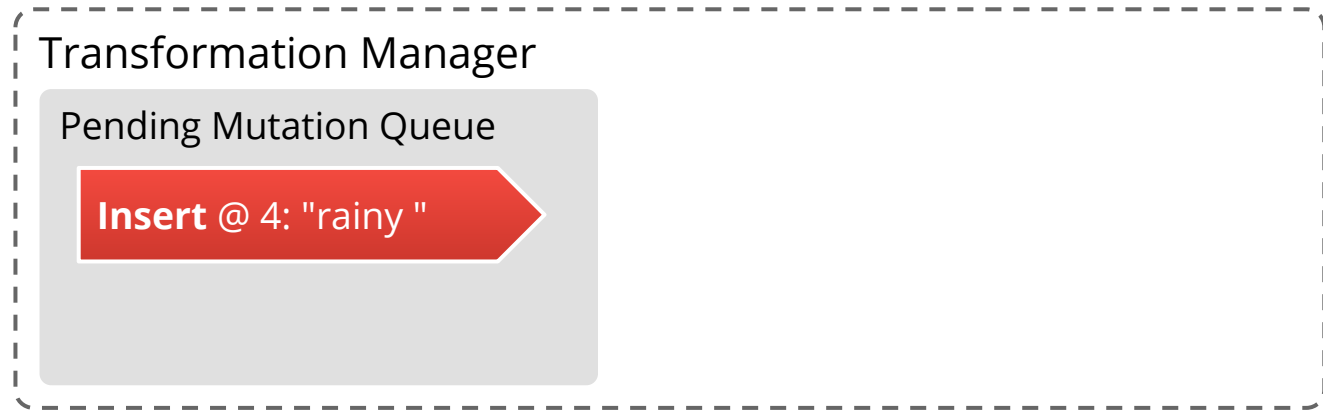
Solution: Transform on the client too!

Keep a client-side queue of sent mutations and transform incoming mutations against the queue



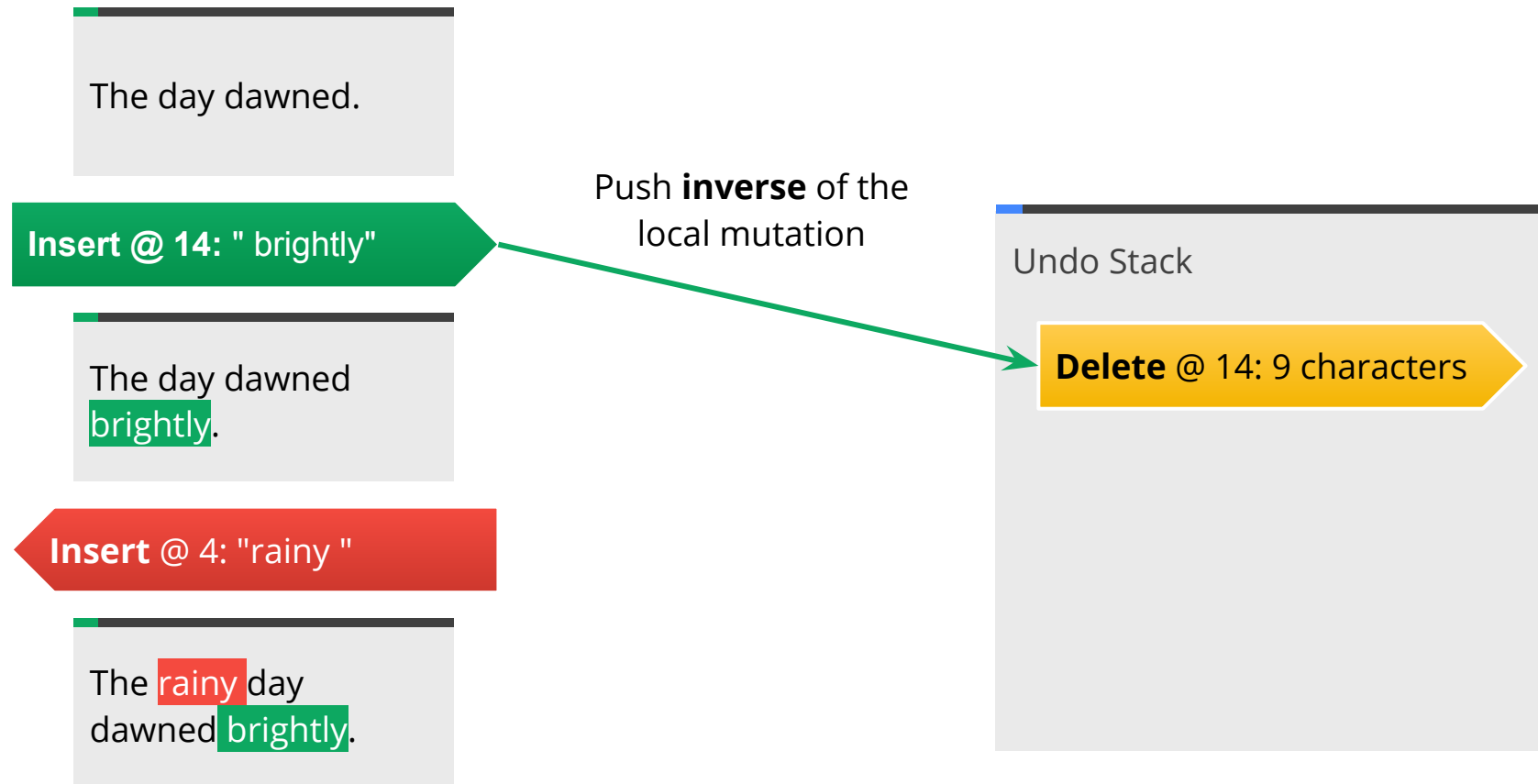
Solution: Acknowledge sent mutations

When the server acknowledges a mutation, it can be removed from the pending queue



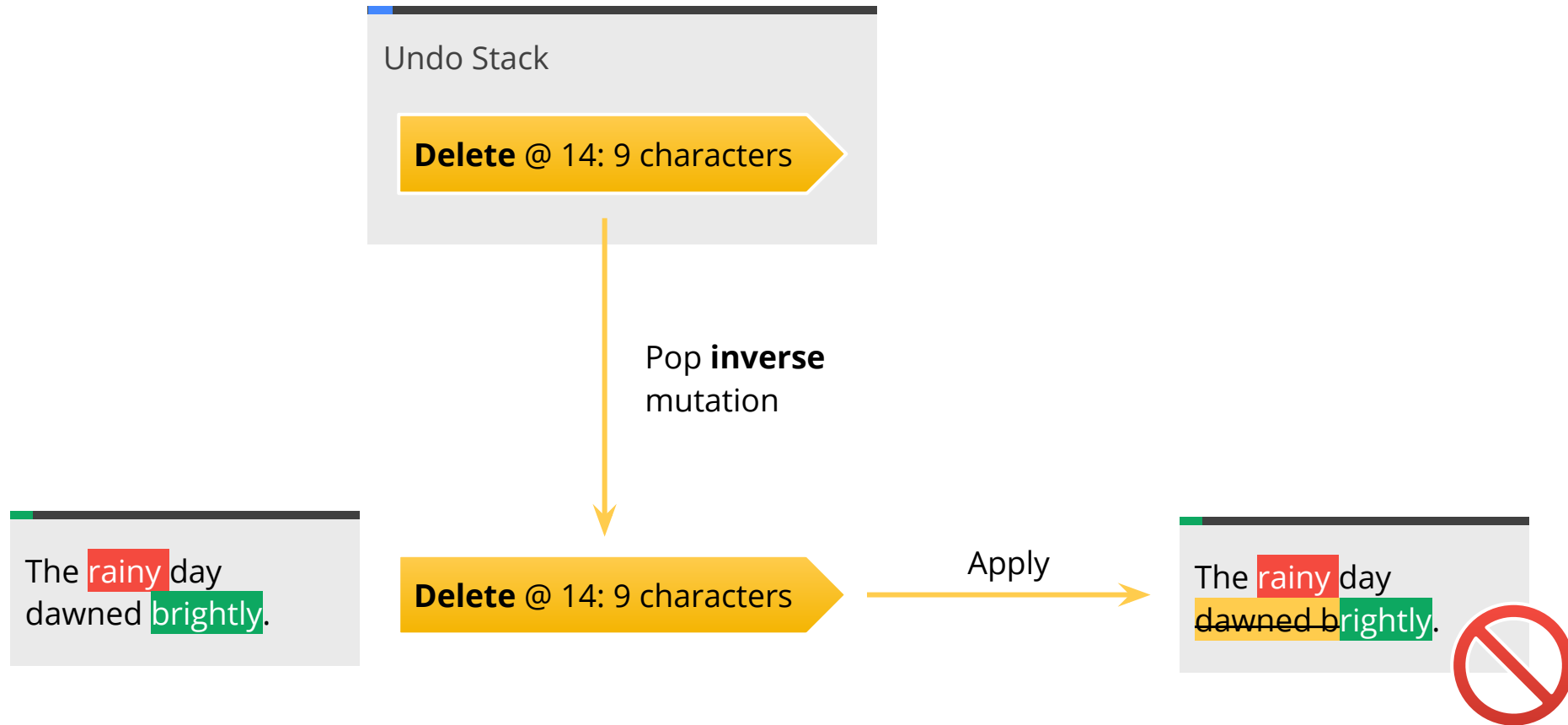
Use case: Undo the local user's last change

Keep an undo stack of inverted mutations

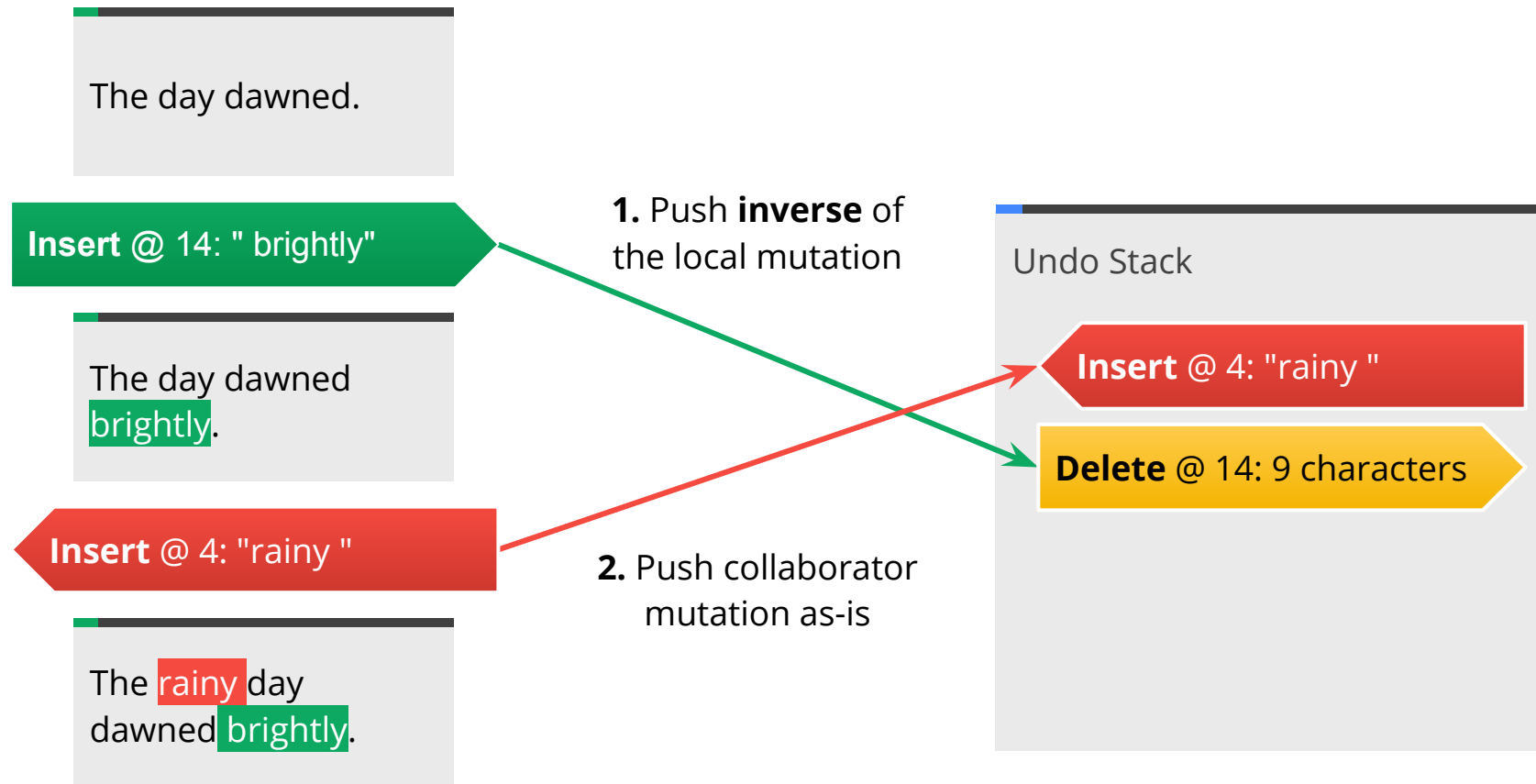


Challenge: Remote changes break undo

Applying the inverse of a previous mutation can give the wrong result

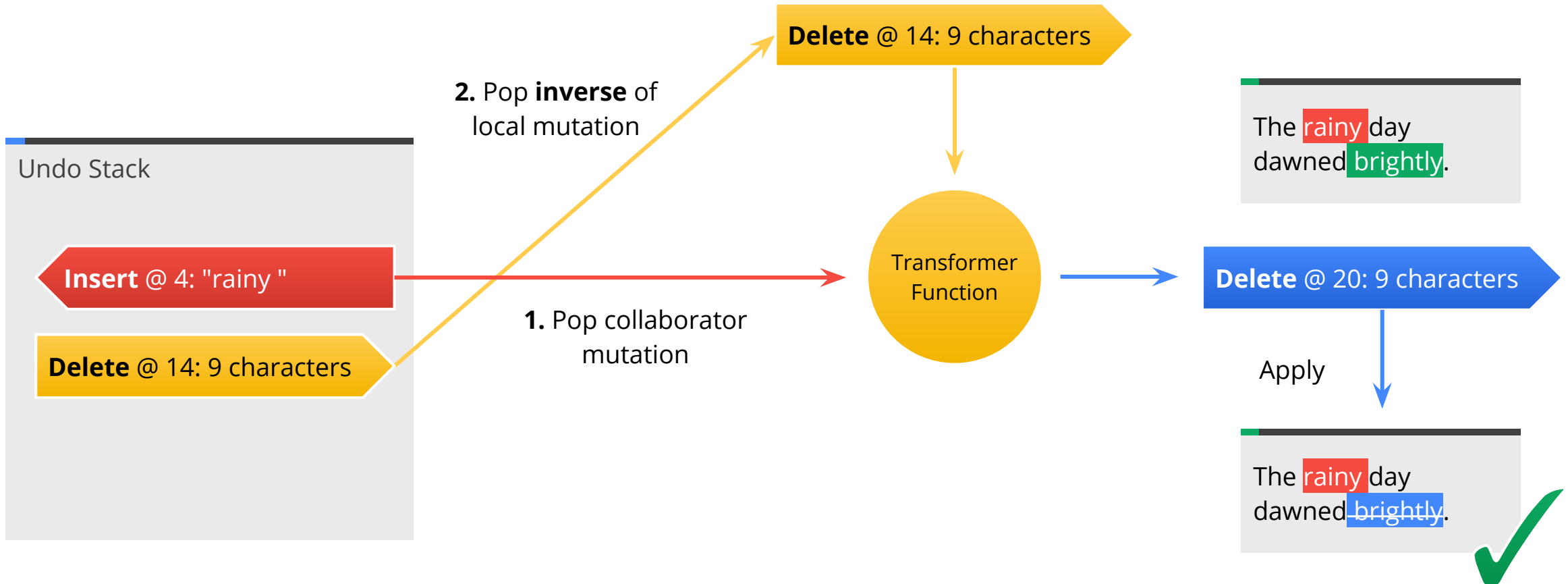


Solution: Store collaborator mutations



Solution: Transform undo mutations

Transform undo changes against remote mutations using the same transformer as remote changes





Designing a Good Data Model

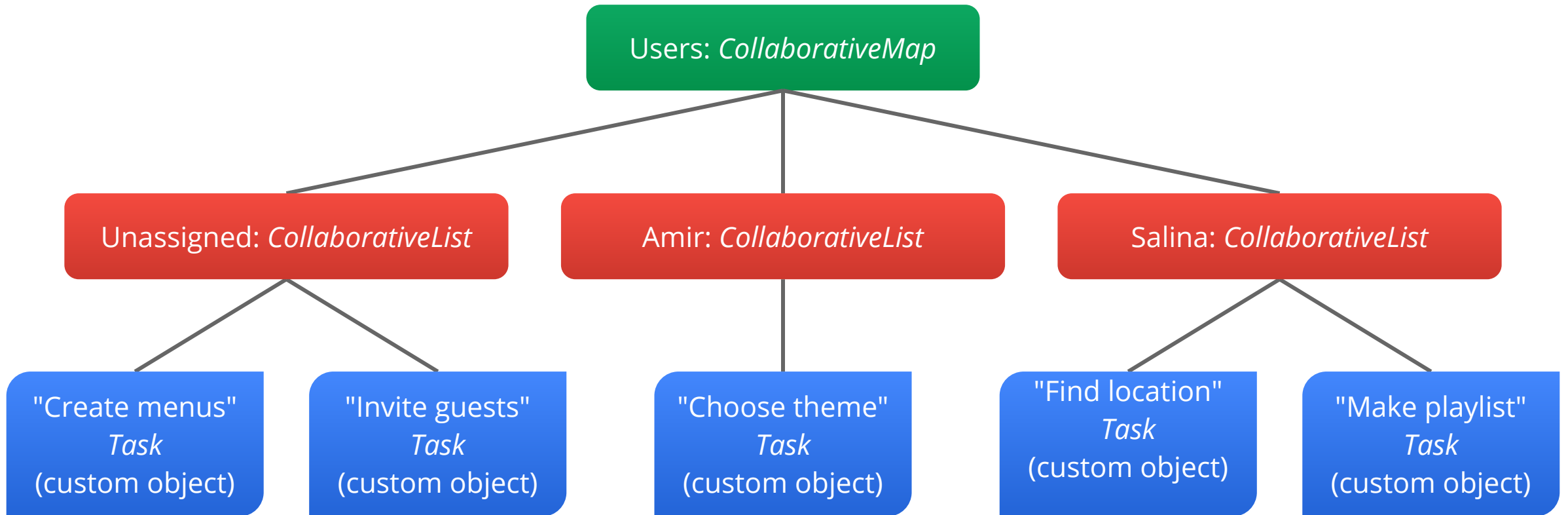
Or at least avoiding designing a broken one

Case study: Event planning system

Unassigned	Amir	Eric	Salina
Create menus Time: 2 Unit: days	Design decorations Time: 3 Unit: days	Find location Time: 3 Unit: hours	Make music playlist Time: 12 Unit: hours
Invite guests Time: 6 Unit: hours	Choose theme Time: 6 Unit: hours		
Buy ingredients Time: 5 Unit: hours			



Proposed data model



Use case: Updating multiple properties atomically

Unassigned	Amir	Eric	Salina
Create menus Time: 2 Unit: days			Make music playlist Time: 12 Unit: hours
Invite guests Time: 6 Unit: hours			
Buy ingredients Time: 5 Unit: hours			

Title:

Description:

Time:

Unit:



Problem: Changes can be interleaved

Depending on latency, Salina's changes might be applied before, after, or in-between Amir's changes

Amir

Salina

Set: "Unit" "days"

Set: "Time" "1"

Set: "Unit" "hours"

Set: "Time" "20"

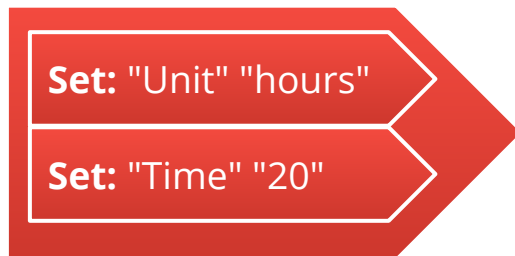
Create menu
Time: 20
Unit: days



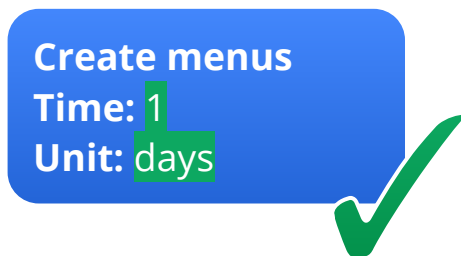
Solution: Use compound operations

Compound operations group the mutations so that they are processed together

Amir



Salina

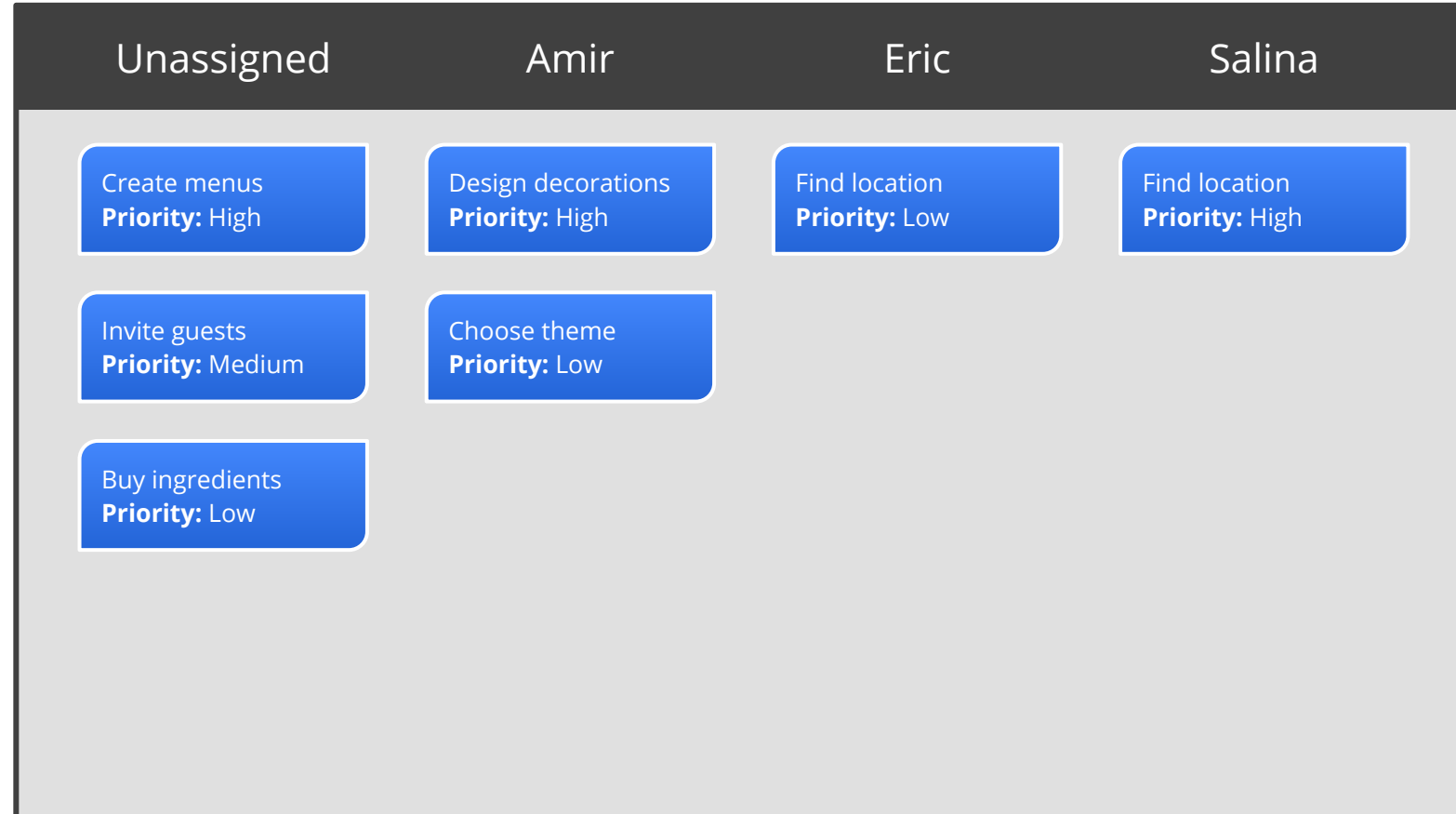


Javascript

```
Task.prototype.updateProperties =  
  function(newUnit, newTime) {  
    this.model_.beginCompoundOperation();  
    this.unit = newUnit;  
    this.time = newTime;  
    this.model_.endCompoundOperation();  
  }
```



Use case: Keep tasks in priority order

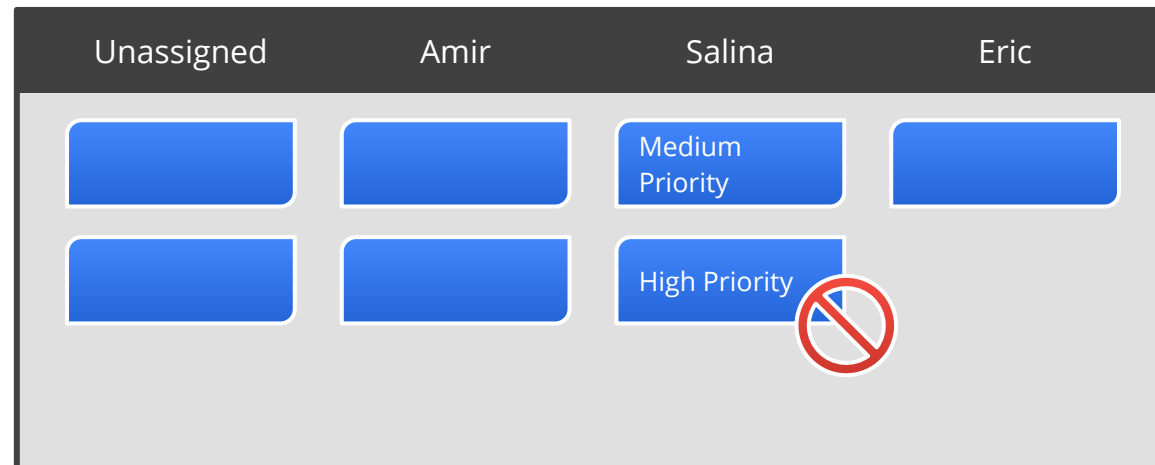


Problem: Sorted lists don't stay sorted

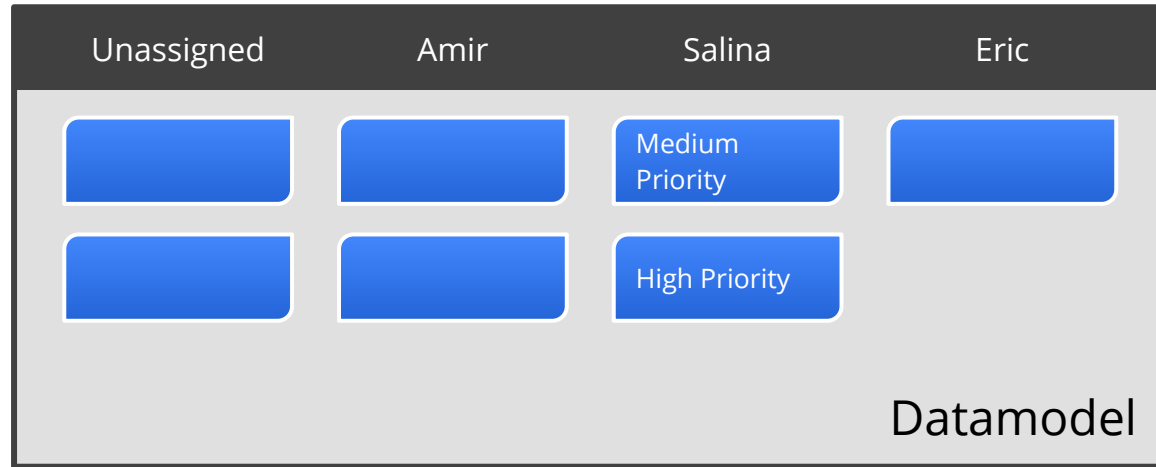
Sorts performed by different users can conflict, resulting in an unsorted list

Insert "Salina" @ 0: "High Priority"

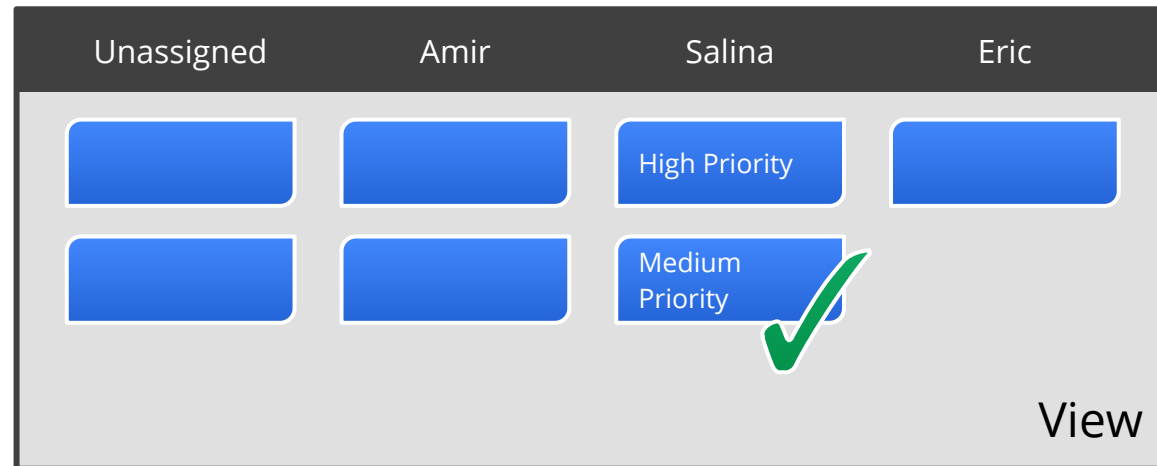
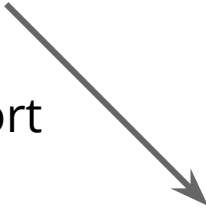
Insert "Salina" @ 0: "Medium Priority"



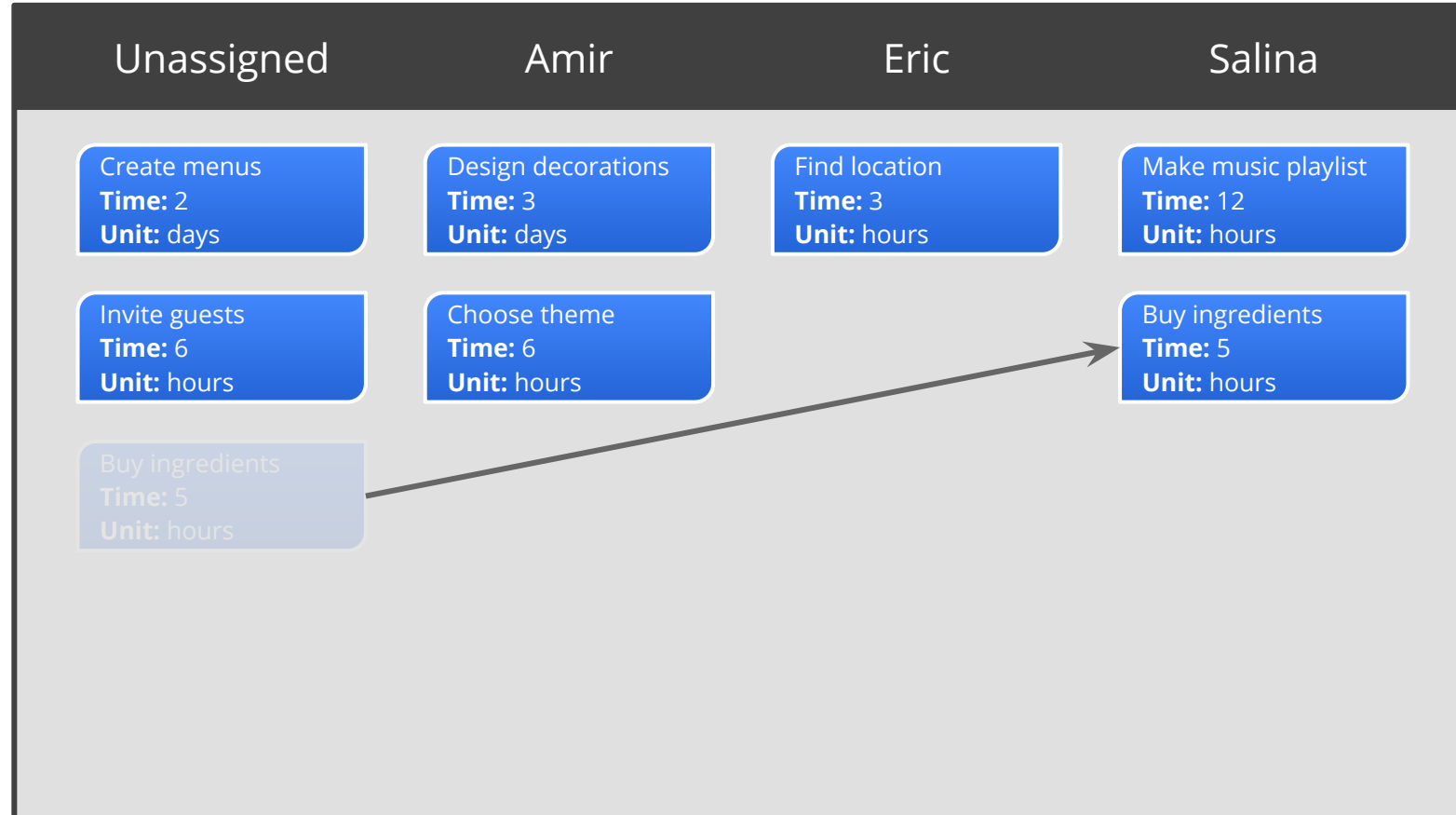
Solution: Unsorted data model, sorted view



Sort



Use case: Assigning tasks



Problem: List element moves aren't atomic

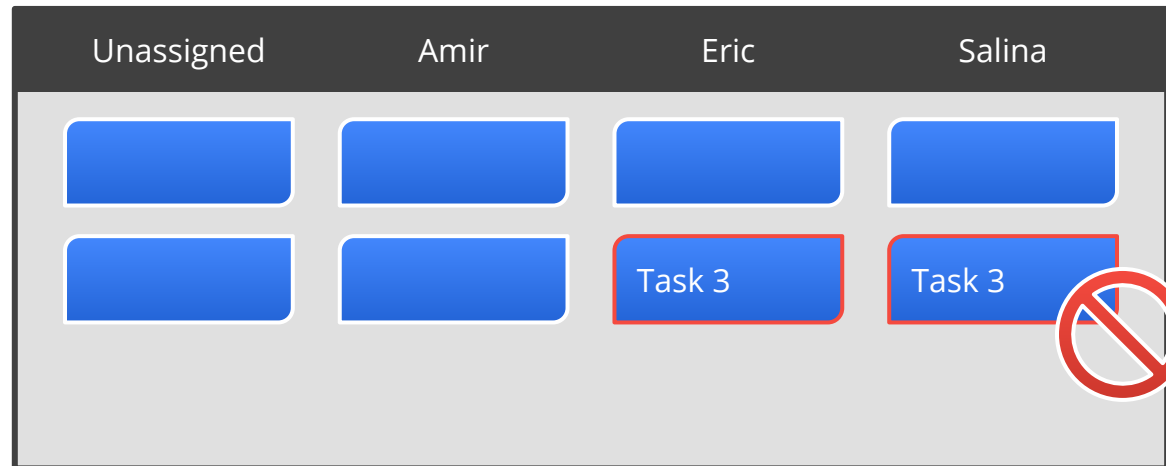
Simultaneous moves can lead to duplicate entries

Delete "Unassigned" @ 1: 1

Insert "Salina", @ 2: "Task 3"

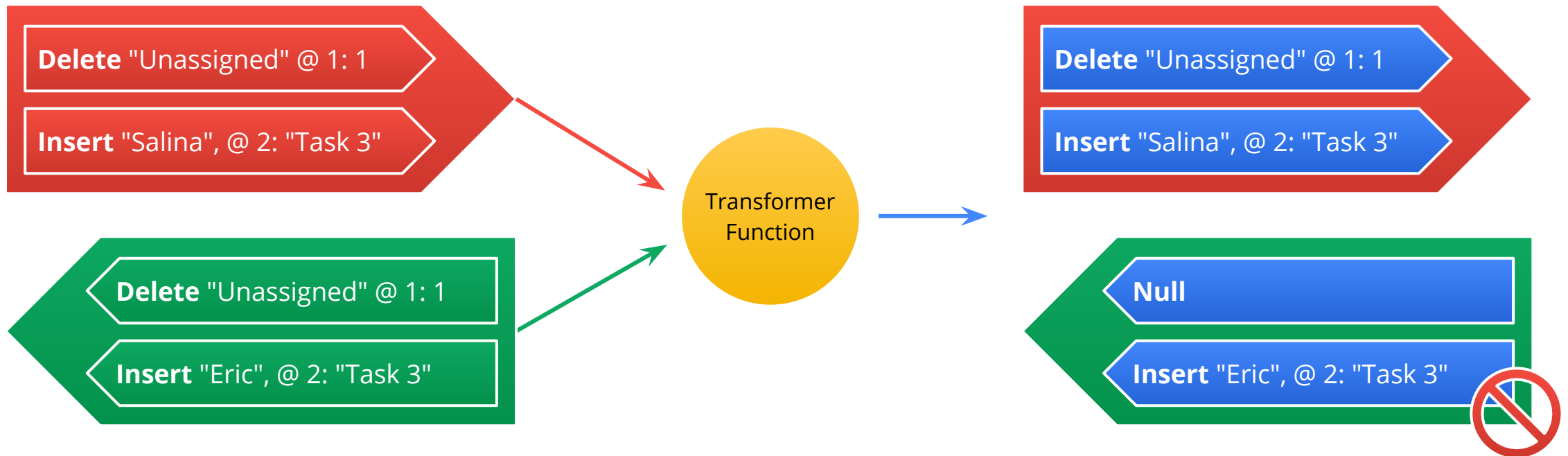
Delete "Unassigned" @ 1: 1

Insert "Eric" @ 2: "Task 3"



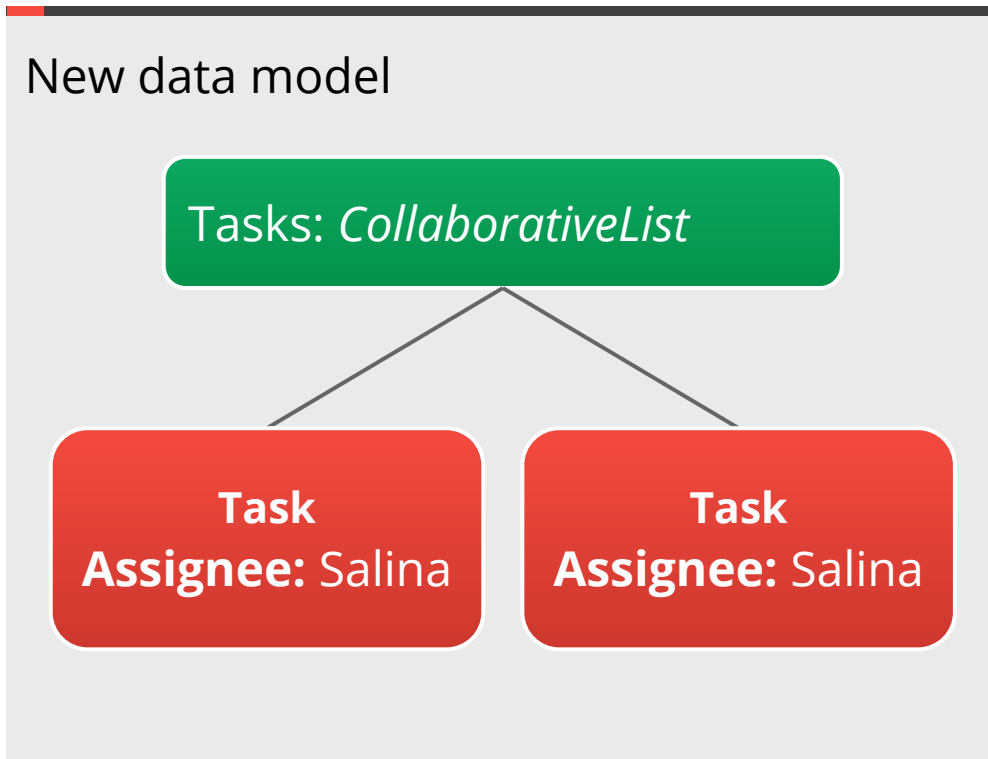
Incorrect solution: Compound operations

Compound operations affect the order that mutations are applied in, **not** how they are transformed



Solution: Invert the datamodel design

Reference assignees from tasks instead of tasks from assignees so moves are atomic



Set "Task 3": "Assignee", "Salina"

Set "Task 3": "Assignee", "Eric"

Unassigned	Amir	Eric	Salina

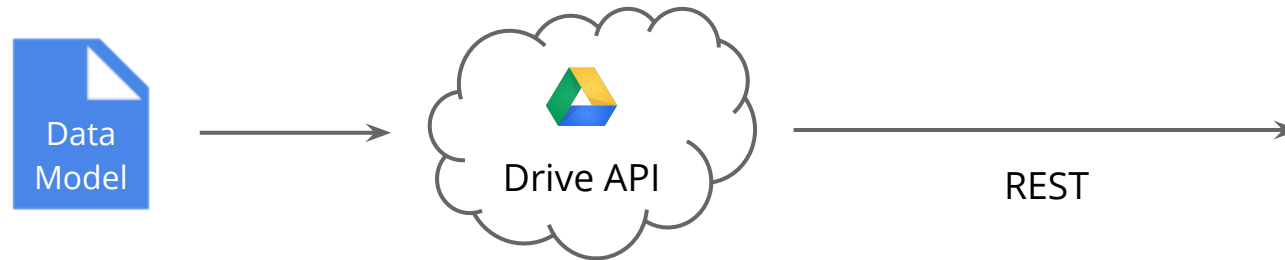




The Future

Things you wish you had now

Export your data model as JSON



or use libraries for:

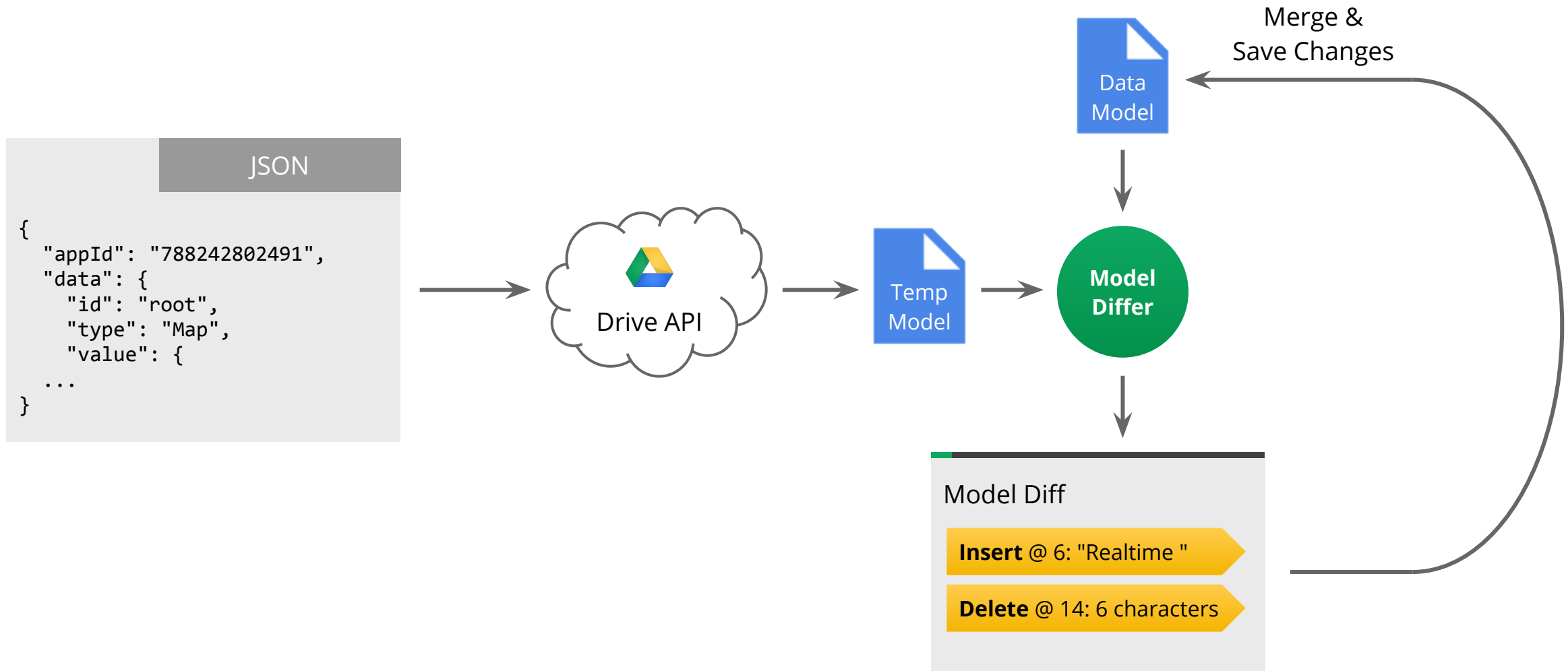
Java, Javascript, .NET,
Ruby, Python, PHP,
Objective C, Dart, Go

JSON

```
{
  "appId": "788242802491",
  "data": {
    "id": "root",
    "type": "Map",
    "value": {
      "key1": {
        "id": "gde2rir49hfrtjw1",
        "type": "List",
        "value": [
          {"json": 1},
          {"json": true},
          {"json": "hello"}
        ]
      },
      "key2": {
        "id": "gde2e545ahftu1aj4",
        "type": "EditableString",
        "value": "Hello, world!"
      }
    }
  }
}
```

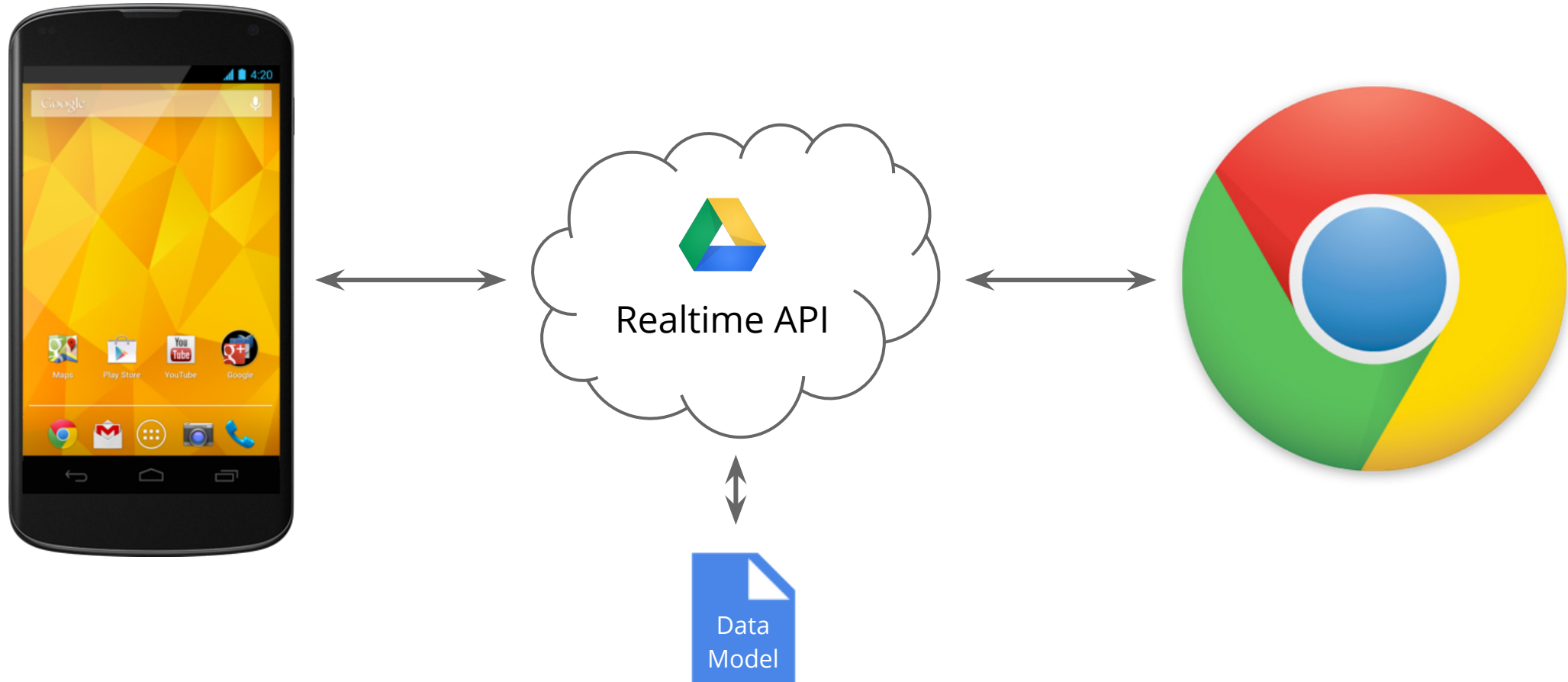


Convert JSON back to Realtime Models



Android / Java API

Real-time collaboration from any Android or Java application

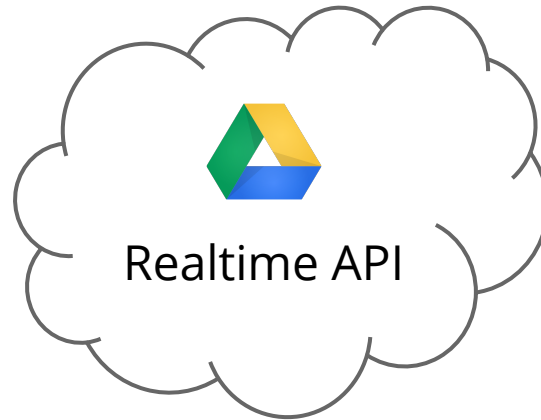


Cross-platform compatibility

Shared data models with the Javascript API

Java

```
class CollaborativeString  
implements CharSequence,  
Appendable
```



Javascript

```
gapi.drive.realtime.CollaborativeString =  
new function() { ... }
```

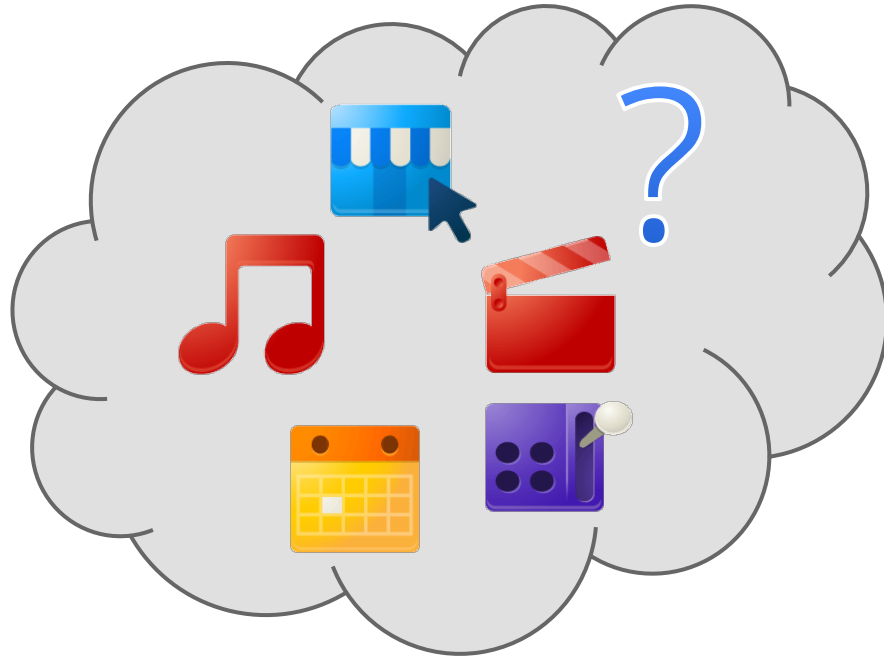


Summary

- The Realtime API makes collaboration **easy**!
- We use **mutations** and **transformation** to make real-time collaboration possible.
- Consider how **collaboration** works when designing your data model.
- Exciting new features like **Java/Android support** and **JSON import/export** are coming soon!



Now it's your turn!



Realtime API



Thank you!

Google Drive Developers on G+
google-drive-sdk on Stackoverflow

dory: <http://goo.gl/dqQum>





Google
Developers