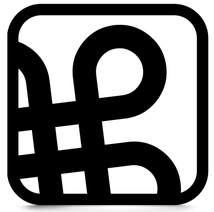


Macoun

ProjectWizards

Objekte Syncen?
Ich kündige!

Kai Brüning & Frank Illenberger

Ablauf

- Vorstellung des Problems
- Bekannte Lösungsansätze
- Unsere Lösung

Das Syncing-Problem

Das Syncing-Problem

- Benutzer wollen mit mehreren Geräten auf geteilten Daten arbeiten
  - mehrere Geräte pro Benutzer
  - mehrere Benutzer

Arbeiten auf geteilten Daten

Gewünschte Eigenschaften

- Ohne Blockieren
  - Freies Arbeiten ohne durch andere blockiert zu werden
  - Auch wenn man auf den gleichen Datensätzen arbeitet

Arbeiten auf geteilten Daten

Gewünschte Eigenschaften

- Mergen
  - Ohne zu Blockieren ist Mergen gleichzeitiger Änderungen unvermeidlich
  - Benutzer sollte sich nicht modal entscheiden müssen

Modales Entscheiden

Arbeiten auf geteilten Daten

Konsistentes Mergen

- Konvergenz
  - Gleiches Ergebnis für alle !!!
    - Sonst maximale Verwirrung der Benutzer
    - Es wird schnell schlimmer…

Arbeiten auf geteilten Daten

Konsistentes Mergen

- Intentionserhaltung
  - Konvergenz alleine garantiert noch keine zufriedenen Benutzer
  - Möglichst ursprünglichen Intentionen aller Benutzer erhalten
  - Nicht immer möglich

Arbeiten auf geteilten Daten

Erhalt der Intention

Chris

Thomas

Arbeiten auf geteilten Daten

Erhalt der Intention

Chris

Conference
kirschkuchen: 1 kg

Thomas

Conference
kirschkuchen: 1 kg

Arbeiten auf geteilten Daten

Erhalt der Intention

Chris

Conference
kirschkuchen: 5 kg

Thomas

Conference
kirschkuchen: 1 kg

Arbeiten auf geteilten Daten

Erhalt der Intention

Chris

Conference
kirschkuchen: 5 kg

Thomas

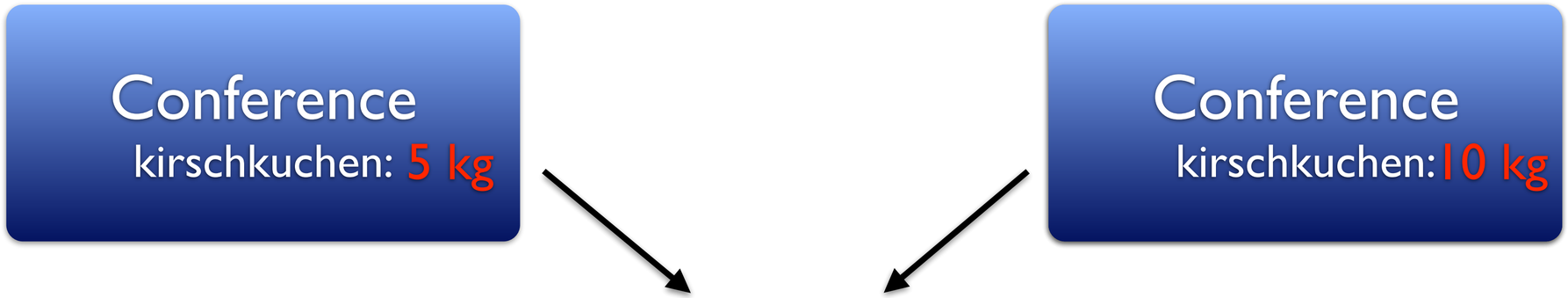
Conference
kirschkuchen: 10 kg

Arbeiten auf geteilten Daten

Erhalt der Intention

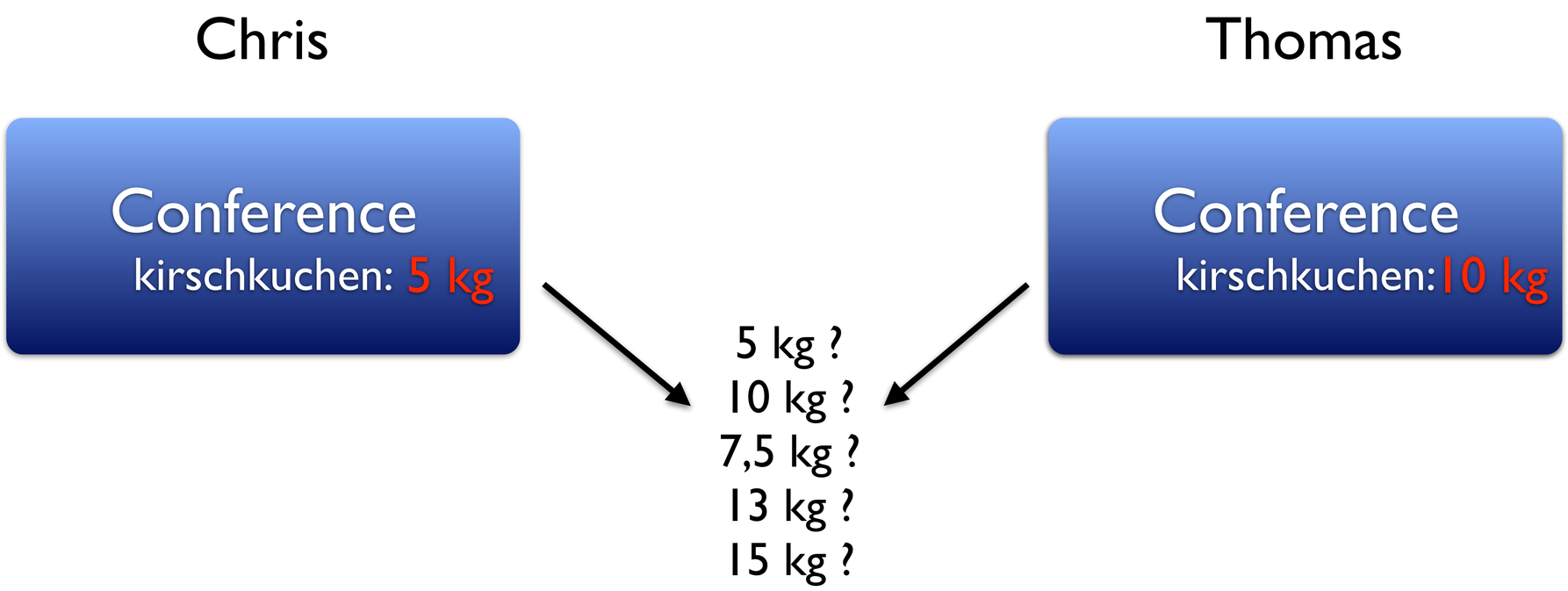
Chris

Thomas



Arbeiten auf geteilten Daten

Erhalt der Intention



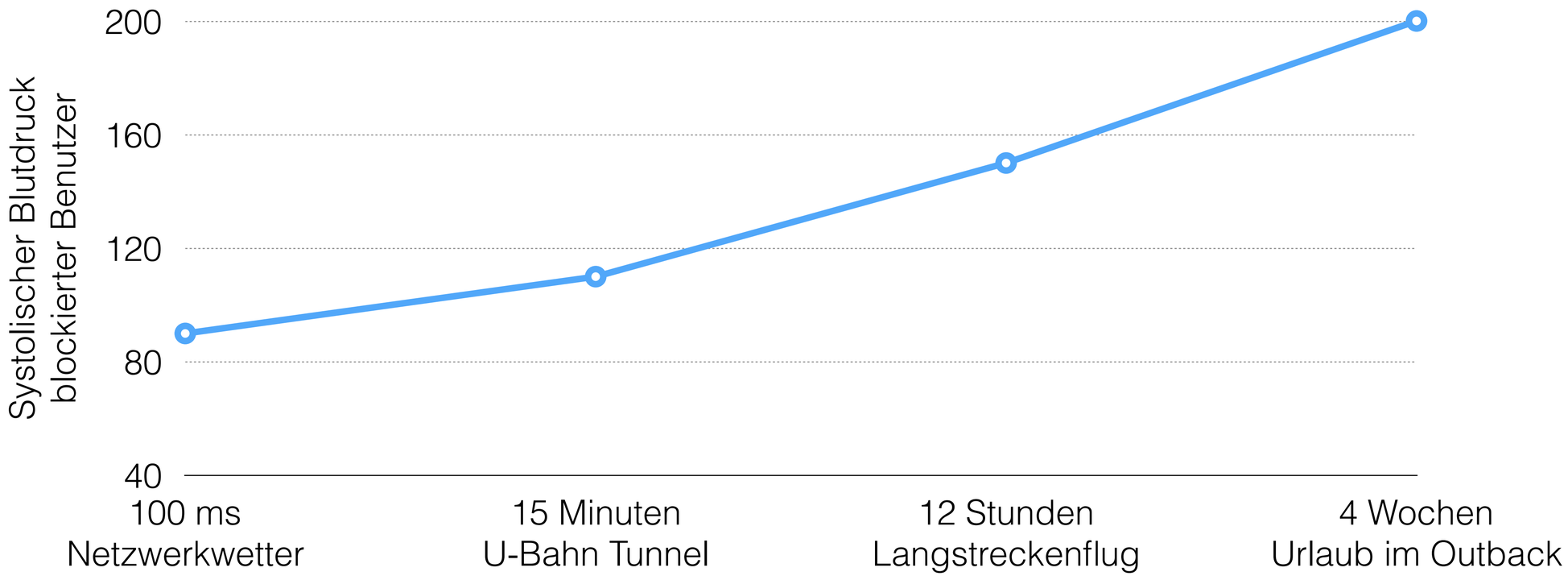
Arbeiten auf geteilten Daten

Gewünschte Eigenschaften

- Offline-Kollaboration
  - Verschiedene Gleichzeitigkeits-Intervalle
    - Netzwerklatenz
      - Latenzen erzwingen Gleichzeitigkeit auch bei permanenter Online-Verbindung
  - Permanente Online-Verbindung ist unrealistisch

Offline Kollaboration

Verschiedene Offline-Intervalle



Arbeiten auf geteilten Daten

Gewünschte Eigenschaften

- Widerrufen / Wiederherstellen
  - Benutzer müssen Fehler korrigieren können
  - Benutzer müssen komplexe Funktionen ausprobieren können

  → Damit sich Anwendungen robust anfühlen, muss es perfekt funktionieren.

Arbeiten auf geteilten Daten

Widerrufen / Wiederherstellen

- Besonderheiten bei Kollaboration
  - Apfel-Z-Modell
    - Was erwartet ein Benutzer, wenn zwischenzeitlich Merges stattfinden?
      - Nur seine eigenen Änderungen werden widerrufen?
      - Auch die Änderungen der anderen?

Arbeiten auf geteilten Daten

Widerrufen und Syncen

Änderungen an Chris' Objekt

Conference
title: Macoun
price: 99
date: 24.10.2015

title = **Macoun**

price = **99€** ←Merge von Thomas

date = **24.10.2015**

Cmd-Z

Arbeiten auf geteilten Daten

Widerrufen / Wiederherstellen

- Besonderheiten bei Kollaboration
  - Apfel-Z-Modell
    - Was erwartet ein Benutzer, wenn zwischenzeitlich Merges stattfinden?
      - Nur seine eigenen Änderungen werden widerrufen?
      - Auch die Änderungen der anderen?
        - Apfel-Z benötigt eine Reihenfolge, die es bei Gleichzeitigkeit nicht gibt.
        - Braucht nicht-lineares Widerrufen

Arbeiten auf geteilten Daten

Widerrufen / Wiederherstellen

- Besonderheiten bei Kollaboration
  - Historienmodell
    - Sichtbare Liste von Änderungen
    - Können einzeln nicht-linear widerrufen / wiederhergestellt werden

Arbeiten auf geteilten Daten

Gewünschte Eigenschaften

- Automatisches Sichern und Verteilen
  - Explizites Sichern ist Vergangenheit.
    - Änderungen sollten automatisch gesichert und auf alle Geräte eines Benutzers verteilt werden.
    - Auch an andere Nutzer?

Arbeiten auf geteilten Daten

Gewünschte Eigenschaften

- Automatisches Sichern und Verteilen
  - Auch an andere Nutzer?
    - Veröffentlichen muss unabhängig vom Sichern sein können

    → Branching, Git …

Arbeiten auf geteilten Daten

- Unsere Domäne:
  - Anwendungen, die ihre Daten in Objektgraphen abbilden
  - Kein Text! (Nur als atomare Werte)

Bekannte Lösungen

und ihre Probleme

Bekannte Lösungen

- Es gibt viele
- Keine (uns bekannte) erfüllt alle genannten Anforderungen

Bekannte Lösungen

Dateibasierte Synchronisation

- iCloudDrive, Dropbox und Konsorten
  - Granularität auf Dateiebene
  - Liefern Konfliktversionen bei gleichzeitigen Änderungen an einer Datei
  - Kennt die Semantik der Daten nicht.

    → Mergen wird der Anwendungsebene überlassen.

Bekannte Lösungen

Zentraler Online-Server
mit pessimistischem Locking

- Traditionelle Methode für Enterprise Anwendungen
- Bearbeiteter Datensatz wird für andere Benutzer gesperrt
  - Wie unabhängig sind Datensätze bei komplexer Business Logik?

☑ Konsistentes Mergen

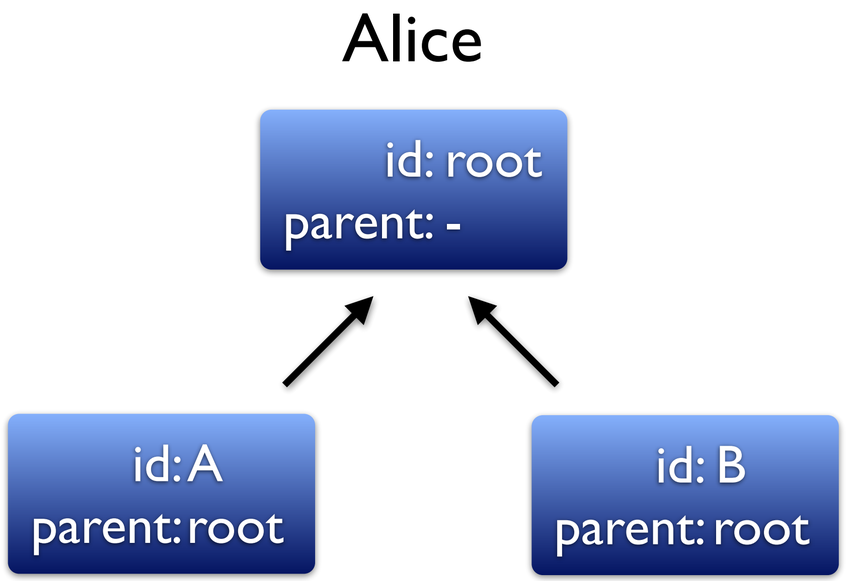
Bekannte Lösungen

Zentraler Online-Server
mit optimistischem Locking und Merging

- Mehrere Benutzer können den gleichen Datensatz bearbeiten.
  - Beim Versuch zu sichern wird dies detektiert.
    - Verwerfen einer der Seiten
    - Mergen, meist Feld-für-Feld
      - Komplexere Verfahren übernimmt die Anwendungslogik
    - Genutzt von objektrelationalen Mappern, z.B. CoreData

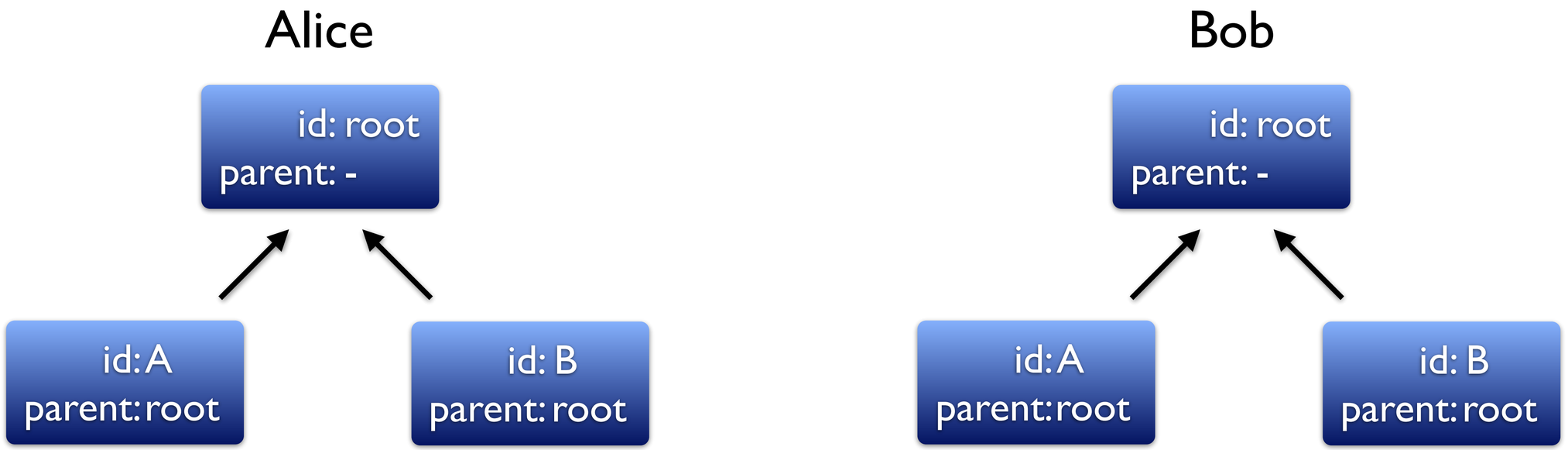
Bekannte Lösungen

Invariantenverletzung durch Mergen



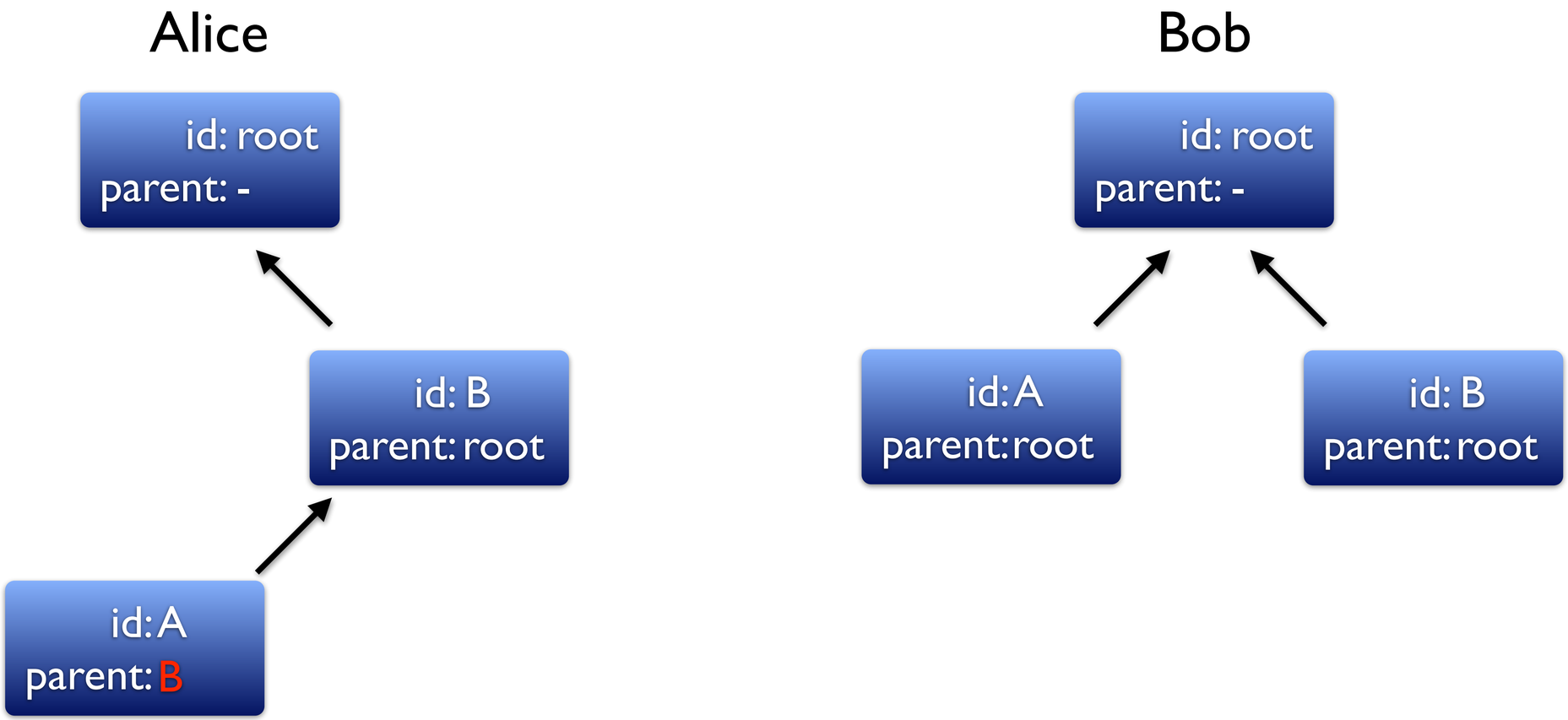
Bekannte Lösungen

Invariantenverletzung durch Mergen



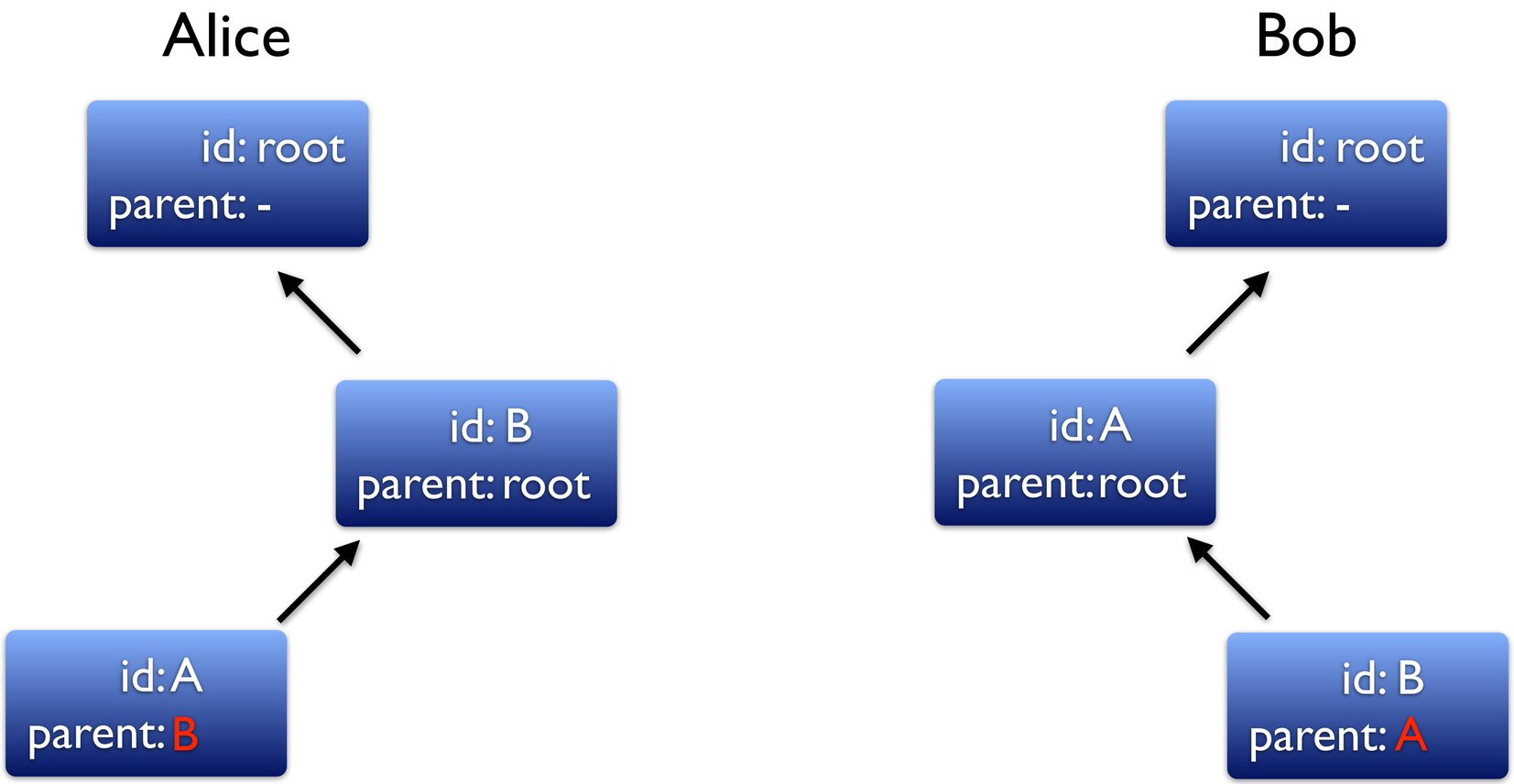
Bekannte Lösungen

Invariantenverletzung durch Mergen



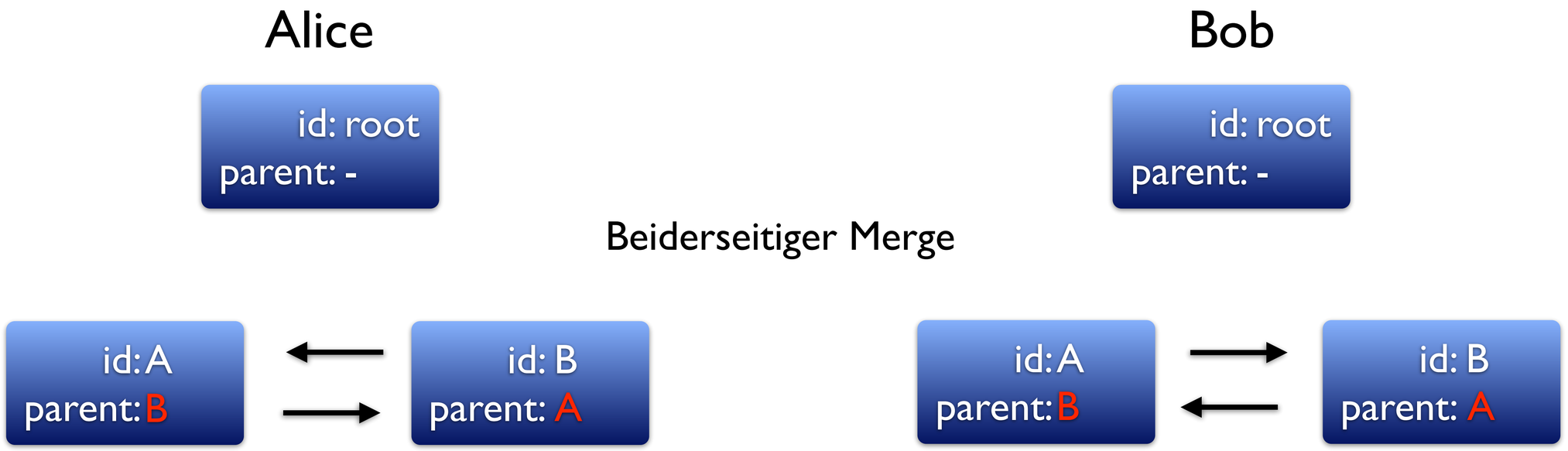
Bekannte Lösungen

Invariantenverletzung durch Mergen



Bekannte Lösungen

Invariantenverletzung durch Mergen



Bekannte Lösungen

Zentraler Online-Server
mit optimistischem Locking und Merging

- Mehrere Benutzer können den gleichen Datensatz bearbeiten.
  - Beim Versuch zu sichern wird dies detektiert.
    - Verwerfen einer der Seiten
    - Mergen, meist Feld-für-Feld
      - Komplexere Verfahren übernimmt die Anwendungslogik
    - Genutzt von Objektrelationalen Mappern, z.B. CoreData

☑ Ohne Blockieren

Bekannte Lösungen

Key-Value-Datenbanken (NoSQL)

- Gleiche Probleme wie optimistisches Locking
- Mergen in Anwendungsebene
- CloudKit

☑ Ohne Blockieren

Bekannte Lösungen

System Prevalence / History Log

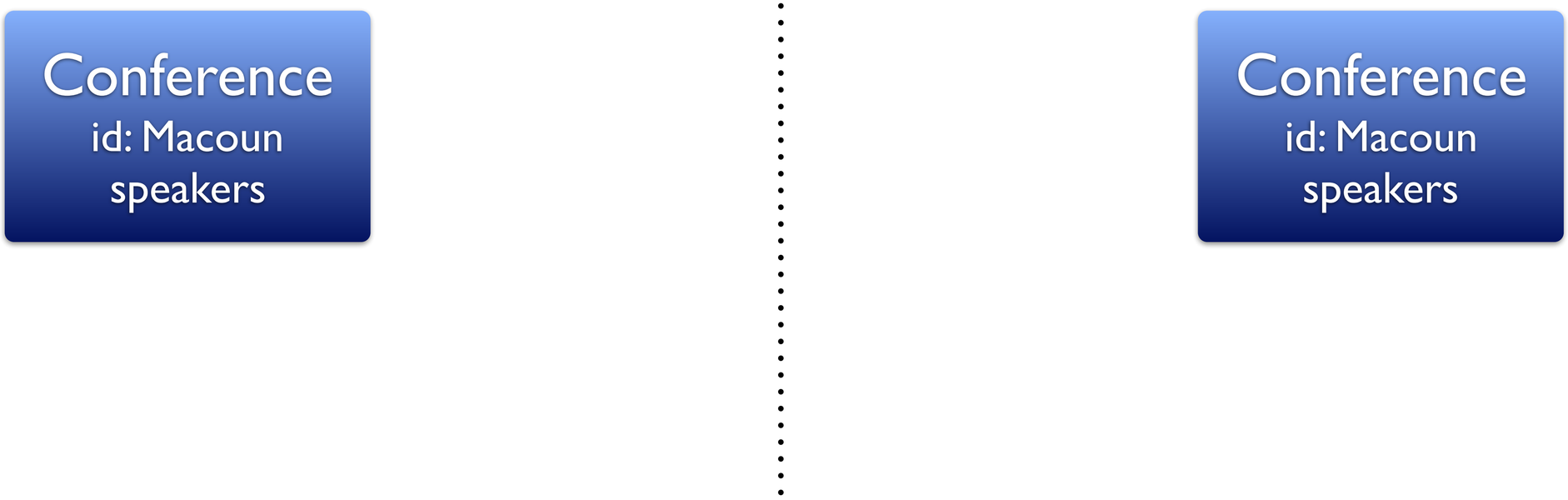
- Erhalten der gesamten Historie von Benutzeraktionen, die zum Zustand eines Objektgraphen führten
  - Log oder Journal genannt
  - Viele verschiedene Ansätze, um Benutzeraktionen aufzuzeichnen
    - in sogenannten *Operationen*
    - Wahl der Abstraktionsebene ist sehr wichtig → später

Bekannte Lösungen

To-Many Relationship ohne History Log

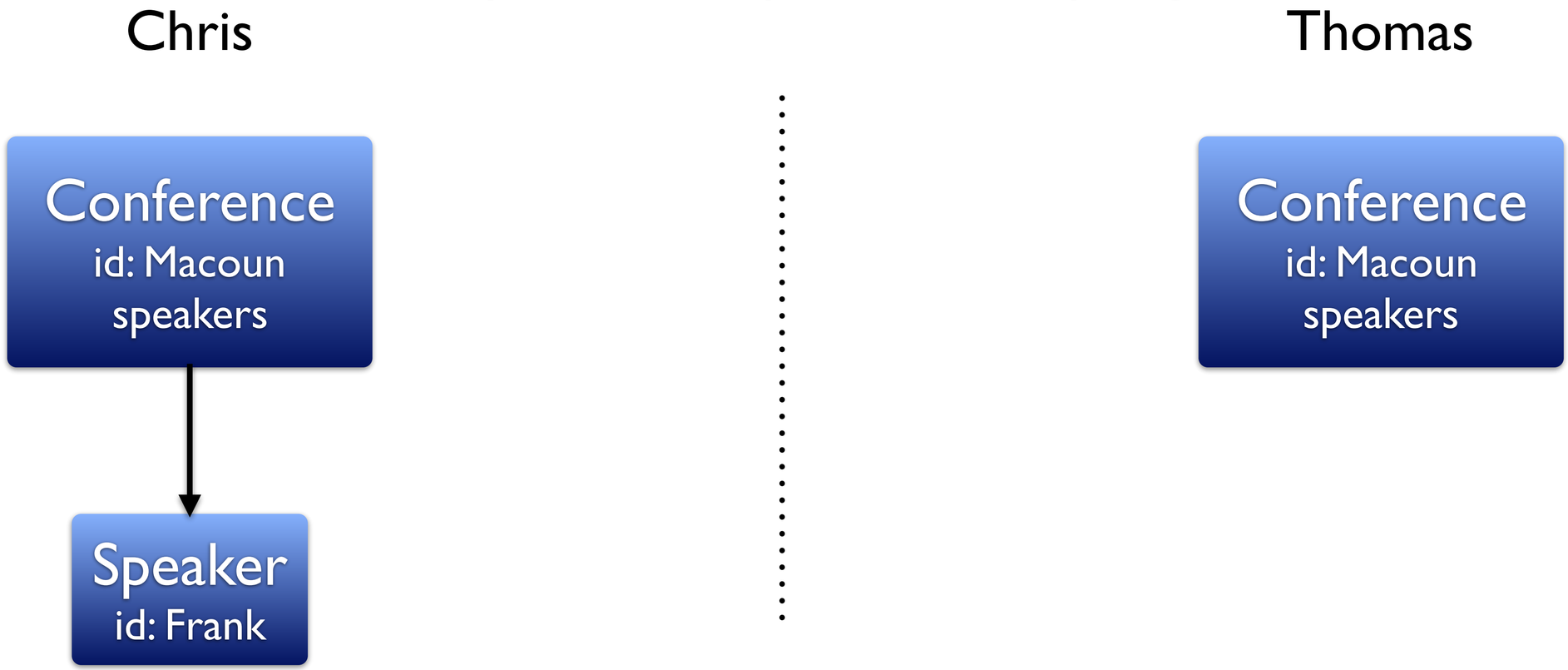
Chris

Thomas



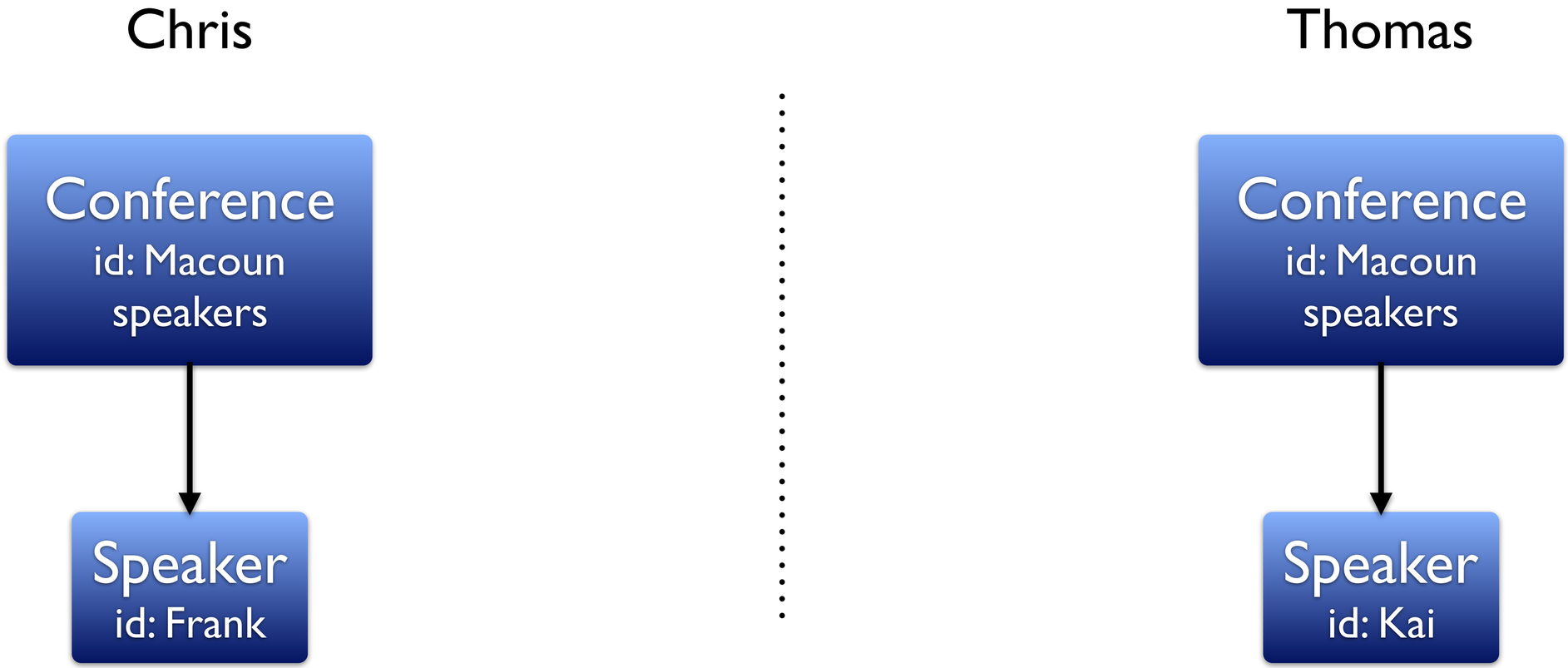
Bekannte Lösungen

To-Many Relationship ohne History Log



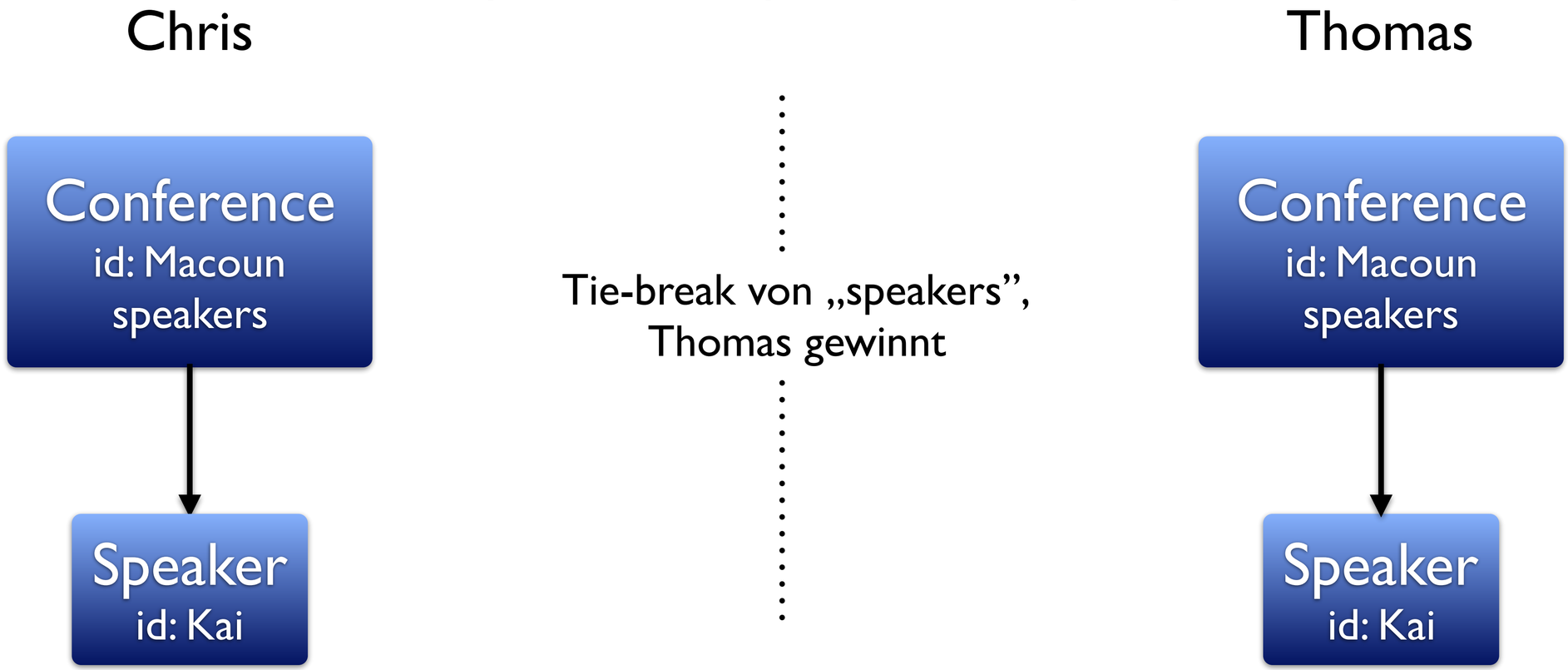
Bekannte Lösungen

To-Many Relationship ohne History Log



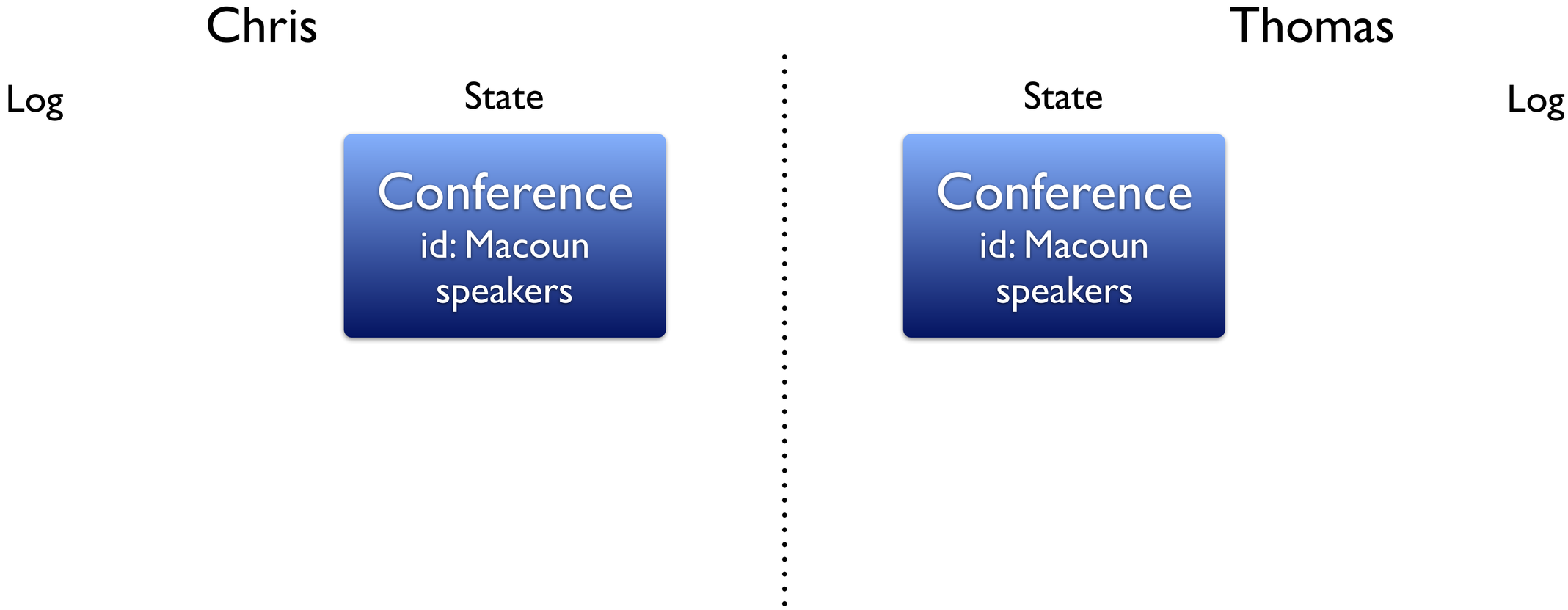
Bekannte Lösungen

To-Many Relationship ohne History Log



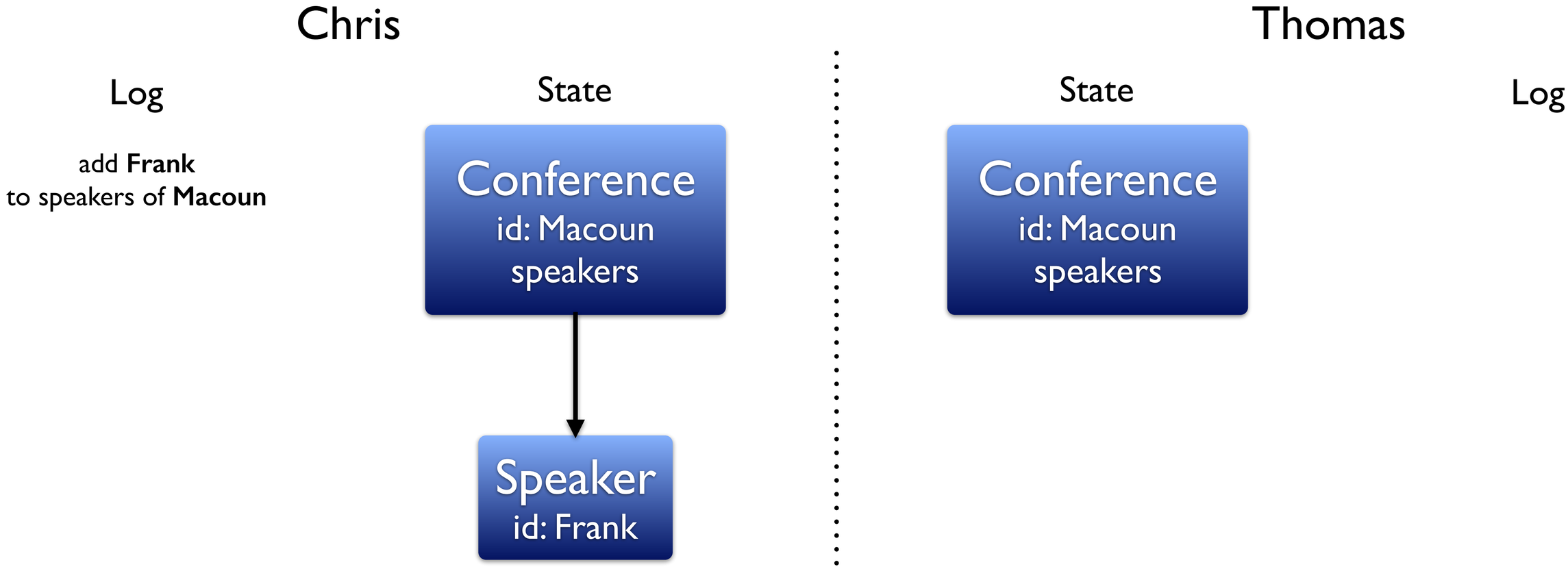
Bekannte Lösungen

To-Many Relationship mit History Log



Bekannte Lösungen

To-Many Relationship mit History Log



Bekannte Lösungen

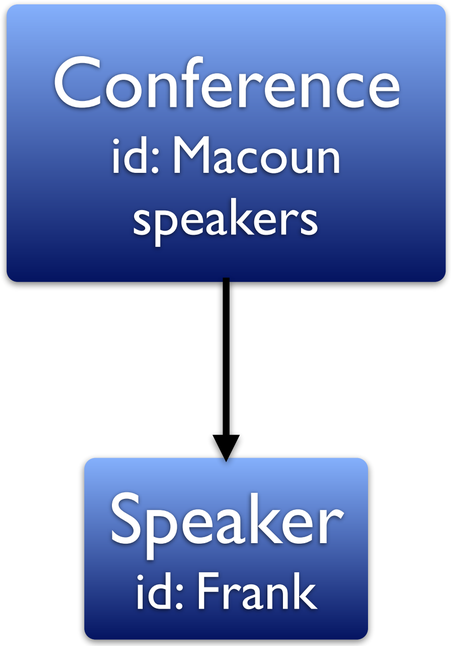
To-Many Relationship mit History Log

Chris

Log

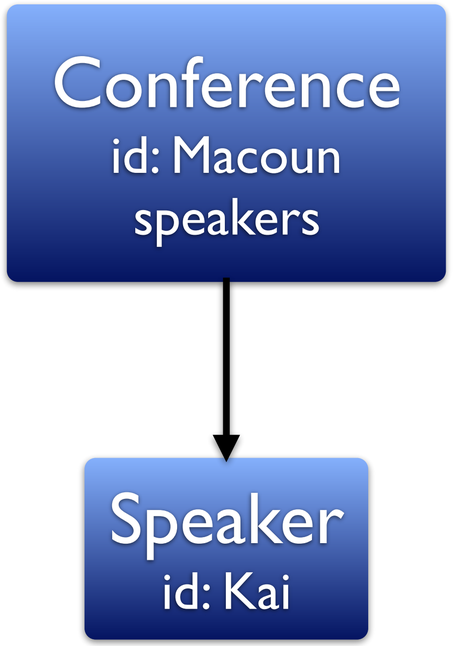
add **Frank**
to speakers of **Macoun**

State



Thomas

State



Log

add **Kai**
to speakers of **Macoun**

Bekannte Lösungen

To-Many Relationship mit History Log

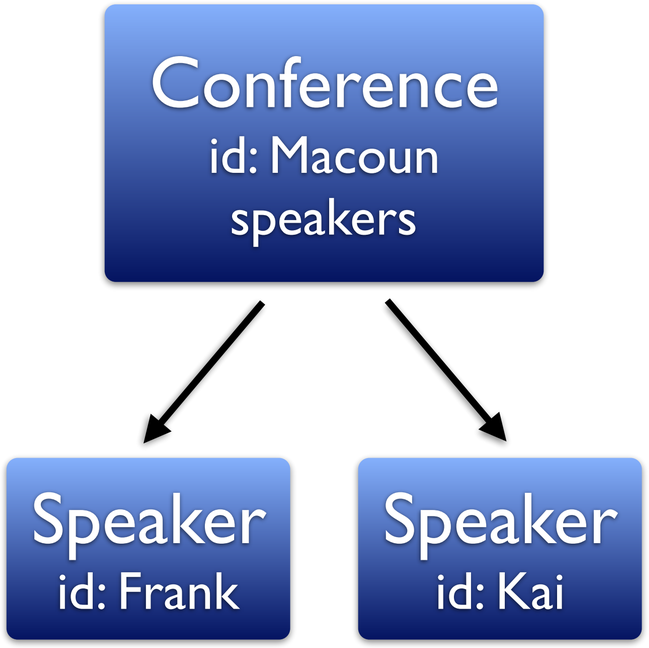
Chris

Log

add **Frank**
to speakers of **Macoun**

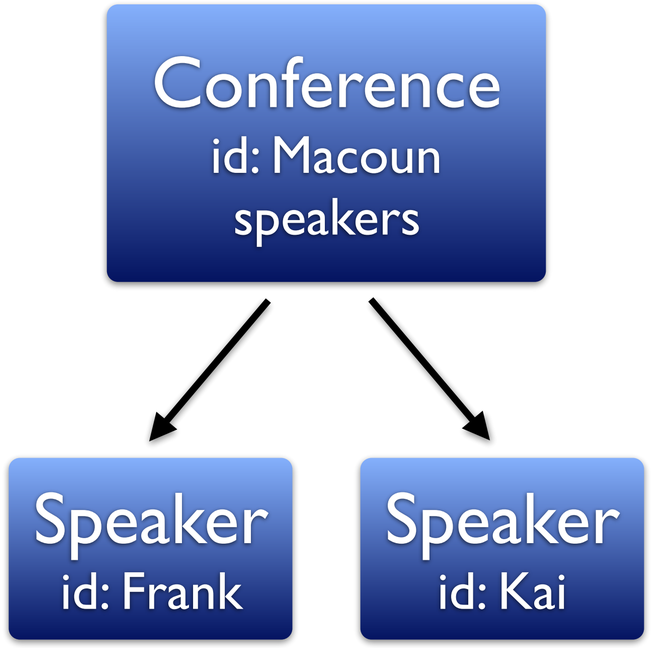
add **Kai**
to speakers of **Macoun**

State



Thomas

State



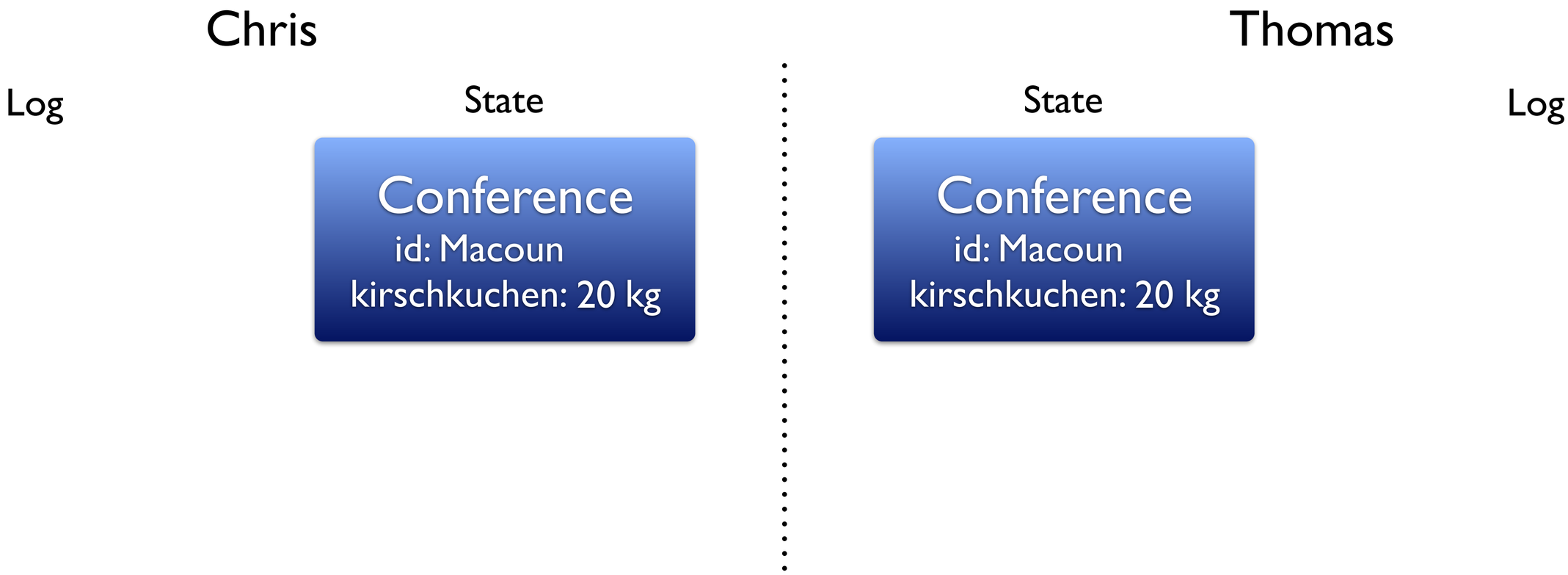
Log

add **Kai**
to speakers of **Macoun**

add **Frank**
to speakers of **Macoun**

Bekannte Lösungen

Ändern einer Objektadresse lässt Merge scheitern

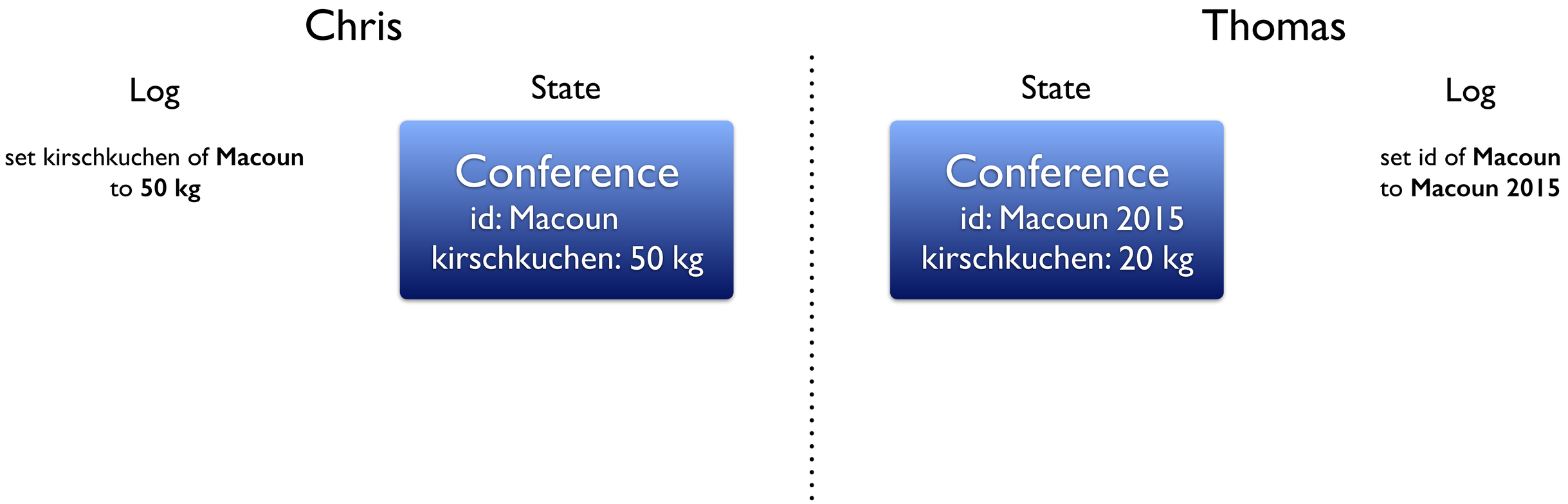


Bekannte Lösungen

Ändern einer Objektadresse lässt Merge scheitern

Bekannte Lösungen

Ändern einer Objektadresse lässt Merge scheitern



Bekannte Lösungen

Ändern einer Objektadresse lässt Merge scheitern

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

set id of **Macoun**
to **Macoun 2015**

State

Conference
id: Macoun 2015
kirschkuchen: 50 kg

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

Bekannte Lösungen

Ändern einer Objektadresse lässt Merge scheitern

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

set id of **Macoun**
to **Macoun 2015**

State

Conference
id: Macoun 2015
kirschkuchen: 50 kg

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

set kirschkuchen of **Macoun**
to **50 kg**

Bekannte Lösungen

Ändern einer Objektadresse lässt Merge scheitern

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

set id of **Macoun**
to **Macoun 2015**

State

Conference
id: Macoun 2015
kirschkuchen: 50 kg

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

set kirschkuchen of **Macoun**
to **50 kg**

Object not found!

Bekannte Lösungen

System Prevalence / History Log

- Erhalten der gesamten Historie von Benutzeraktionen, die zum Zustand eines Objektgraphen führten
  - Log oder Journal genannt
  - Viele verschiedene Ansätze, um Benutzeraktionen aufzuzeichnen
    - in sogenannten *Operationen*
    - Wahl der Abstraktionsebene ist sehr wichtig → später mehr

Bekannte Lösungen

Operational Transformation (OT)

- Erweiterung des System Prevalence-Ansatz
- Operationen eines anderen Benutzers werden nicht direkt angewendet.
- Zuvor werden sie gegeneinander transformiert
- Dies passt sie an den Zustand des lokalen Objektgraphen an

Bekannte Lösungen

Ändern einer Objektadresse mit OT

Bekannte Lösungen

Ändern einer Objektadresse mit OT

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

State

Conference
id: Macoun
kirschkuchen: 50 kg

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

Bekannte Lösungen

Ändern einer Objektadresse mit OT

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

State

Conference
id: Macoun
kirschkuchen: 50 kg

set kirschkuchen of **Macoun**
to **50 kg**

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

set id of **Macoun**
to **Macoun 2015**

Bekannte Lösungen

Ändern einer Objektadresse mit OT

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

State

Conference
id: Macoun
kirschkuchen: 50 kg

set id of **Macoun**
to **Macoun 2015**

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

set kirschkuchen of **Macoun 2015**
to **50 kg**

Bekannte Lösungen

Ändern einer Objektadresse mit OT

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

set id of **Macoun**
to **Macoun 2015**

State

Conference
id: Macoun 2015
kirschkuchen: 50 kg

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 20 kg

Log

set id of **Macoun**
to **Macoun 2015**

set kirschkuchen of **Macoun** 2015
to **50 kg**

Bekannte Lösungen

Ändern einer Objektadresse mit OT

Chris

Log

set kirschkuchen of **Macoun**
to **50 kg**

set id of **Macoun**
to **Macoun 2015**

State

Conference
id: Macoun 2015
kirschkuchen: 50 kg

Thomas

State

Conference
id: Macoun 2015
kirschkuchen: 50 kg

Log

set id of **Macoun**
to **Macoun 2015**

set kirschkuchen of **Macoun 2015**
to **50 kg**

Bekannte Lösungen

Operational Transformation

☑ Alles außer automatisches Sichern & Verteilen

- Erweiterung des System Prevalence-Ansatz
- Operationen eines anderen Benutzers werden nicht direkt angewendet.
- Zuvor werden sie gegeneinander transformiert
- Dies passt sie an den Zustand des lokalen Objektgraphen an

Operational Transformation

- Unsere Wahl für unsere Geschäftsanwendungen
- Wurde ursprünglich für Texte erfunden
  - Führt zu False-Tie Problem (vielleicht später)
  - wird inzwischen in vielen Domänen angewendet
    - Apache Wave
    - Google Docs

Operational Tranformation

Worauf lässt man sich ein?

- Finden eines Satzes von Operationen
  - Aufzeichnen aller möglichen Benutzeraktionen
  - und der Information, um sie widerrufen zu können

Operational Tranformation

Worauf lässt man sich ein?

- Definieren der Transformationen zwischen allen möglichen Operationspaaren
  - die Konvergenz garantieren und Intentionen möglichst erhalten
  - Korrektheit zu zeigen ist extrem schwierig.

Operational Tranformation

Worauf lässt man sich ein?

- Definieren eines Integrationsalgorithmus
  - welche Transformationen braucht man, um eine Operation auf den gewünschten Dokumentzustand anwenden zu können?
  - Muss garantieren, dass nur von der Businesslogik erlaubte Zustände des Objektgraphen entstehen.

Operational Tranformation

Worauf lässt man sich ein?

- Praktische Schwierigkeiten
  - Transformationsmatrix wächst quadratisch mit der Zahl der Operationen
    - Zahl der Operationen muss gering bleiben
      - deutlich kleiner 10

      → Operationen auf Ebene der Benutzeraktionen zu definieren ist nicht zielführend

      → wenige, atomare Operationen zur Manipulation des Objektgraphen

Operational Tranformation

Worauf lässt man sich ein?

- Praktische Schwierigkeiten
  - Implementation ist nicht fehlertolerant.
    - Konzeptionelle Transformationsfehler potenzieren sich und zerstören Konvergenz gründlich.
    - Selbst kleine Beispiele werden schnell unübersichtlich.

Operational Tranformation

„Basic Example" aus „Invention for Syncing Object Graphs"

Alice

Original State

Bob

Note: for simplicity the example uses integers for both object and operation ids. In reality these ids are built using universally unique identifiers to ensure uniqueness across different users.

Employee/ID1
name: Smith

Employee/ID2
name: Jones

Project/ID3
name: New Website

Concurrent Changes

Alice inserts a new Project object. This is recorded as an object operation for object class "Project", which changes the existence of this object from **F**alse to **T**rue and includes a snapshot with the state of the new object.

```
OOp(Project,ID4,F,T,{name:Patent})
```

Alice makes Smith the lead for the website project. The resulting assignment object uses the dependent id ID1-ID3. This id implies the relationships of the assignment to employee ID1 and project ID3.

```
OOp(Assignment,ID1-ID3,F,T,{role:lead})
```

Bob changes the name of project ID3. The resulting attribute operation records the id of the target object, the property name and the old and new values of the property.

```
AOp(ID3,name,New Website,Website)
```

Employee/ID1
name: Smith

Assignment
role: lead
work:

Project/ID3
name: New Website

Employee/ID2
name: Jones

Project/ID4
name: Patent

Employee/ID1
name: Smith

Employee/ID2
name: Jones

Project/ID3
name: **Website**

Operational Tranformation

Worauf lässt man sich ein?

- Praktische Schwierigkeiten
  - Implementation ist nicht fehlertolerant.
    - Konzeptionelle Transformationsfehler potenzieren sich und zerstören Konvergenz gründlich.
    - Selbst kleine Beispiele werden schnell unübersichtlich.
      - Kombinatorische Explosion
      - …Unit-Test-Permutation 3352 schlägt fehl…

Operational Tranformation

Worauf lässt man sich ein?

„Unfortunately, implementing OT sucks. There's a million algorithms with different tradeoffs, mostly trapped in academic papers. The algorithms are really hard and time consuming to implement correctly. …

Wave took 2 years to write and if we rewrote it today, it would take almost as long to write a second time.“

Joseph Gentle, Google Wave Engineer

Quelle: https://en.wikipedia.org/wiki/Operational_transformation

Unsere Lösung

mehr als 2 Mannjahre Entwicklung …

Unsere Lösung

Operationensatz

Name	Funktion	Werte
Object	Erzeugen/Löschen von Objekten	ja/nein
Identifier	Ändern von Objekt-Identifiern	spezielle Objekt-Identifier
Attribute	Setzen von Attributen	CoreData Attributewerte
Relationship	Manipulieren von Beziehungen	Objekt-Identifier
Tree	Spezialfall für Eltern-Beziehung in Bäumen	Objekt-Identifier-Pfade

Unsere Lösung

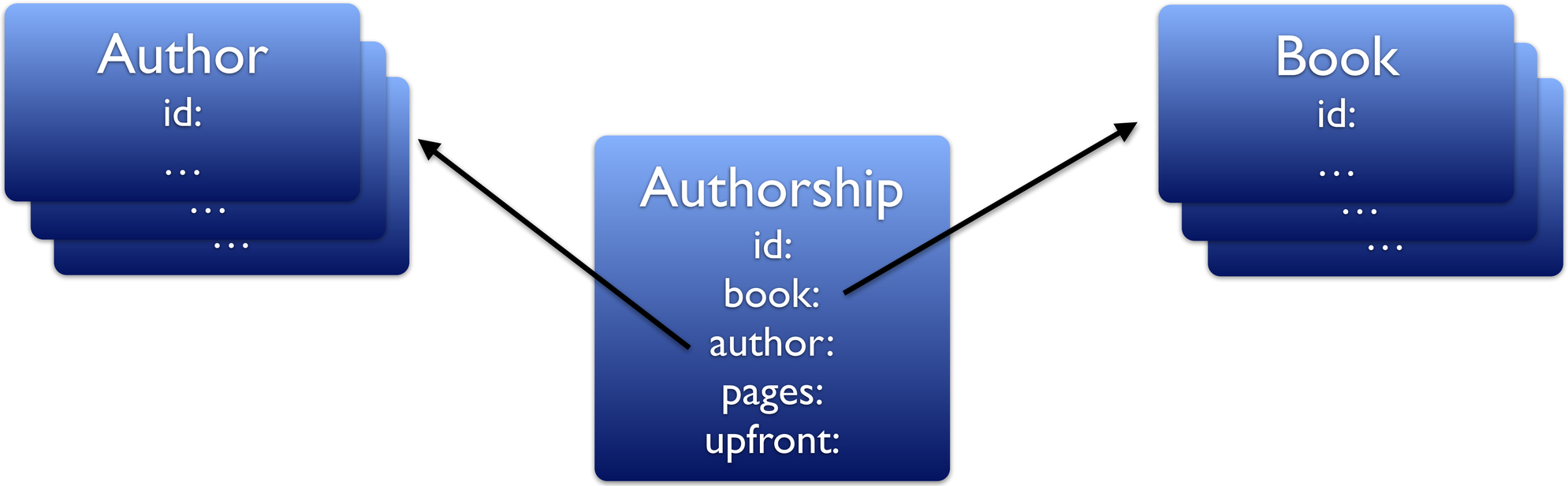
Identifizierung von Objekten

- In der Regel per UUID, erstellt beim Erzeugen der Objekte
  - Indexpfade durch den Objektgraphen führen zu False-Tie-Problem

  → wenig Änderung bei Transformationen
- Ausnahme: Join-Objekte (Many-to-many Beziehungen)

Unsere Lösung

Join-Objekte



Es kann nur einen geben!

Unsere Lösung

Join-Objekte

Alice

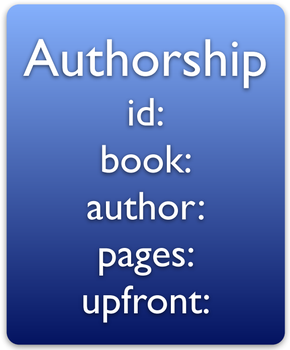
create Authorship with id **abc**

set book of **abc** to Steve Jobs

set author of **abc** to **Fritz Müller**

set pages of **abc** to **300**

create Authorship with id **xyz**



Bob

create Authorship with id **xyz**

set book of **xyz** to Steve Jobs

set author of **xyz** to **Fritz Müller**

set upfront of **xyz** to **100k$**

create Authorship with id **abc**

→ Zwei unabhängige Authorship Objekte

Unsere Lösung

Join-Objekte mit Join-Identifier

Alice

create Authorship with
id **Steve Jobs/Fritz Müller**

set pages of **Steve Jobs/Fritz Müller**
to **300**

create Authorship with
id **Steve Jobs/Fritz Müller**

set id of **Steve Jobs/Fritz Müller**
to **Steve Jobs/Walter Isaacson**

Bob

create Authorship with
id **Steve Jobs/Fritz Müller**

set upfront of **Steve Jobs/Fritz Müller**
to **100k$**

create Authorship with
id **Steve Jobs/Fritz Müller**

Unsere Lösung

Identifizierung von Objekten

- In der Regel per UUID, erstellt beim Erzeugen der Objekte
  - Indexpfade durch den Objektgraphen führen zu False-Tie-Problem

  → wenig Änderung bei Transformationen
- Ausnahme: Join-Objekte (Many-to-many Beziehungen)
  - Identifier ist die Kombination der Adressen der zusammengeführten Objekte

    → braucht Identifier Operation

Unsere Lösung

Operationensatz

Name	Funktion	Werte
Object	Erzeugen/Löschen von Objekten	ja/nein
Identifier	Ändern von Objekt-Identifiern	spezielle Objekt-Identifier
Attribute	Setzen von Attributen	CoreData Attributewerte
Relationship	Manipulieren von Beziehungen	Objekt-Identifier
Tree	Spezialfall für Eltern-Beziehung in Bäumen	Objekt-Identifier-Pfade

Unsere Lösung

- Weitere Konsistenzanforderungen der Businesslogik können kaum unterstützt werden.

→ Als Objektgraphzustand zulassen und im User Interface mit nicht-modalen Warnungen markieren.

Unsere Lösung

Widerrufen

Erster Hauptsatz des nicht-linearen Widerrufens

```
Nachdem eine Operation widerrufen wurde,
muss der Zustand des Objektgraphen so sein
als hätte es die Operation nie gegeben.
```

Klingt trivial, ist es aber nicht.

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Thomas

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Speaker
name: Kai
salary: 1000€

Thomas

Speaker
name: Kai
salary: 1000€

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Thomas

löscht Kai

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Thomas

Kai verzichtet auf Bezahlung

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Thomas

Merge von Chris

?

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Thomas

Widerruft das Löschen

Unsere Lösung

Widerrufen: Löschen gegen Setzen

Chris

Thomas

Widerruft das Löschen

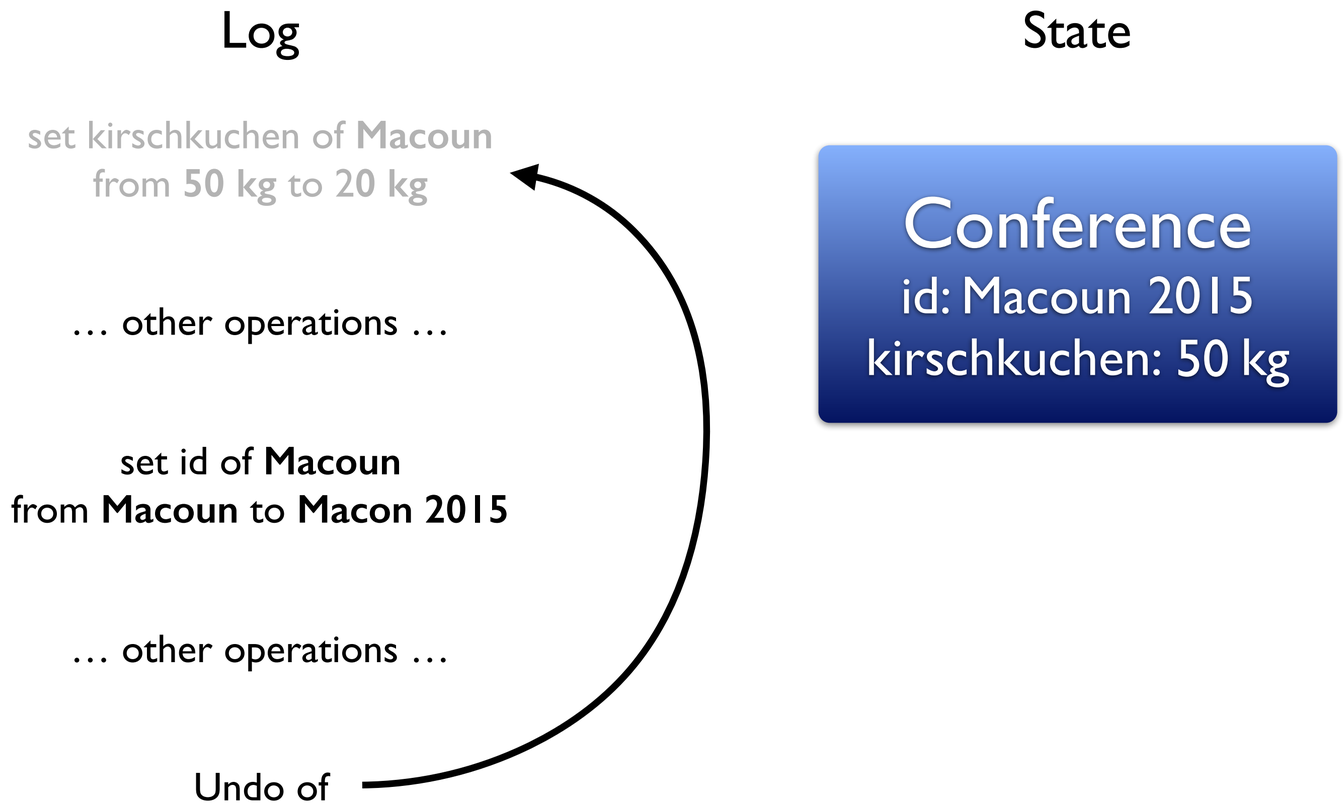
Unsere Lösung

Widerrufen

- Operationen müssen invertiert werden können
  - z.B. Die Lösch-Operation muss den gesamten Zustand des Objekts einfrieren, um es beim Widerrufen wiederherstellen zu können.

Unsere Lösung

Transformation bei Widerrufen



Unsere Lösung

Widerrufen

- Operationen müssen invertiert werden können
  - z.B. Die Lösch-Operation muss den gesamten Zustand des Objekts einfrieren, um es beim Widerrufen wiederherstellen zu können.
- In echt fassen wir alle Operations einer Benutzeraktion in *Events* zusammen, die die kleinste widerrufbare Einheit bilden.

Unsere Lösung

Transformationskonflikte

- Bei einigen Transformationen können nicht alle Intentionen erhalten werden

Unsere Lösung

Tie Break

Chris

Conference
kirschkuchen: 1 kg

Thomas

Conference
kirschkuchen: 1 kg

Unsere Lösung

Tie Break

Chris

Conference
kirschkuchen: 5 kg

Thomas

Conference
kirschkuchen: 1 kg

Unsere Lösung

Tie Break

Chris

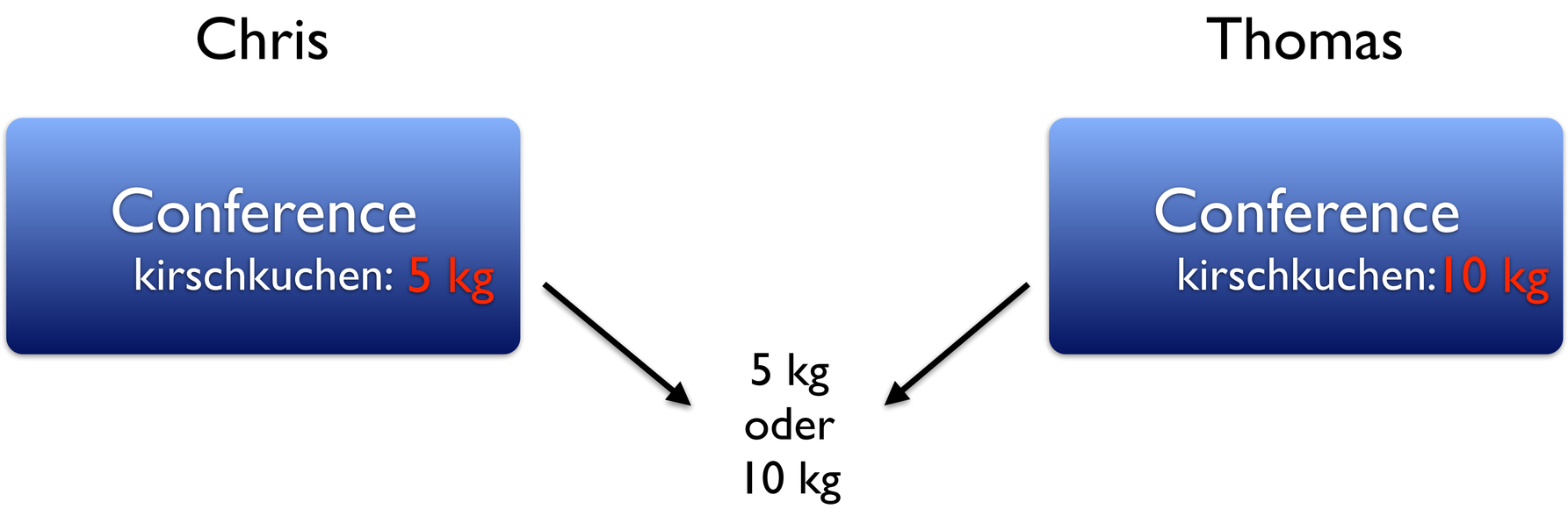
Conference
kirschkuchen: 5 kg

Thomas

Conference
kirschkuchen: 10 kg

Unsere Lösung

Tie Break



Unsere Lösung

Transformationskonflikte

- Bei einigen Transformationen können nicht alle Intentionen erhalten werden
- Wir lassen die jüngere der beiden Operationen gewinnen (Willkür)
- Man darf aber nicht einfach die andere Operation fallen lassen, wegen…

Erster Hauptsatz des nicht-linearen Widerrufens

Nachdem eine Operation widerrufen wurde,
muss der Zustand des Objektgraphen so sein
als hätte es die Operation nie gegeben.

Zweiter Hauptsatz des nicht-linearen Widerrufens

Beim Transformieren darf nie
Information verloren gehen.

Unsere Lösung

Transformationskonflikte

- Wir zeichnen auf, welche Operationen mit welchen in Konflikt ist.
  - Der Integrationsalgorithmus sorgt dafür, dass immer nur eine davon im ausgeführten Zustand ist.
  - Dies ist rekursiv.
- Objektbäume bleiben schleifenfrei.

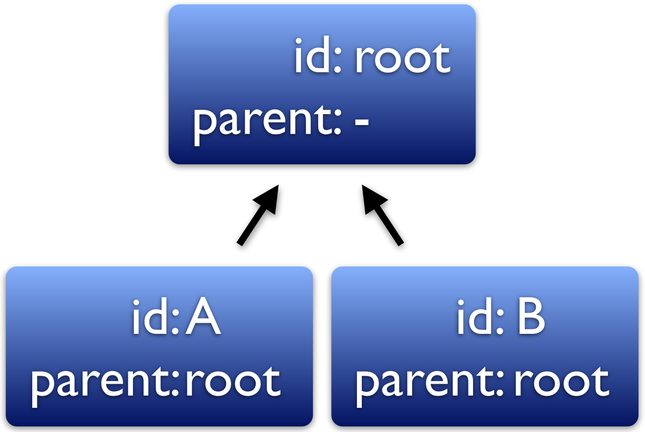
Unsere Lösung

Transformationskonflikte verhindern Schleifen

Alice

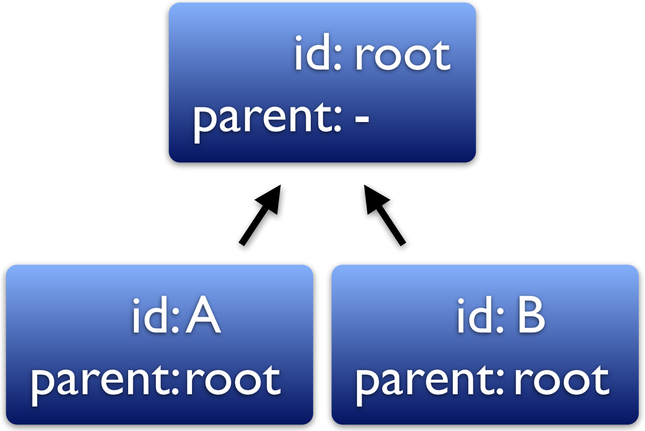
Log

State



Bob

State



Log

Unsere Lösung

Transformationskonflikte verhindern Schleifen

Alice

Bob

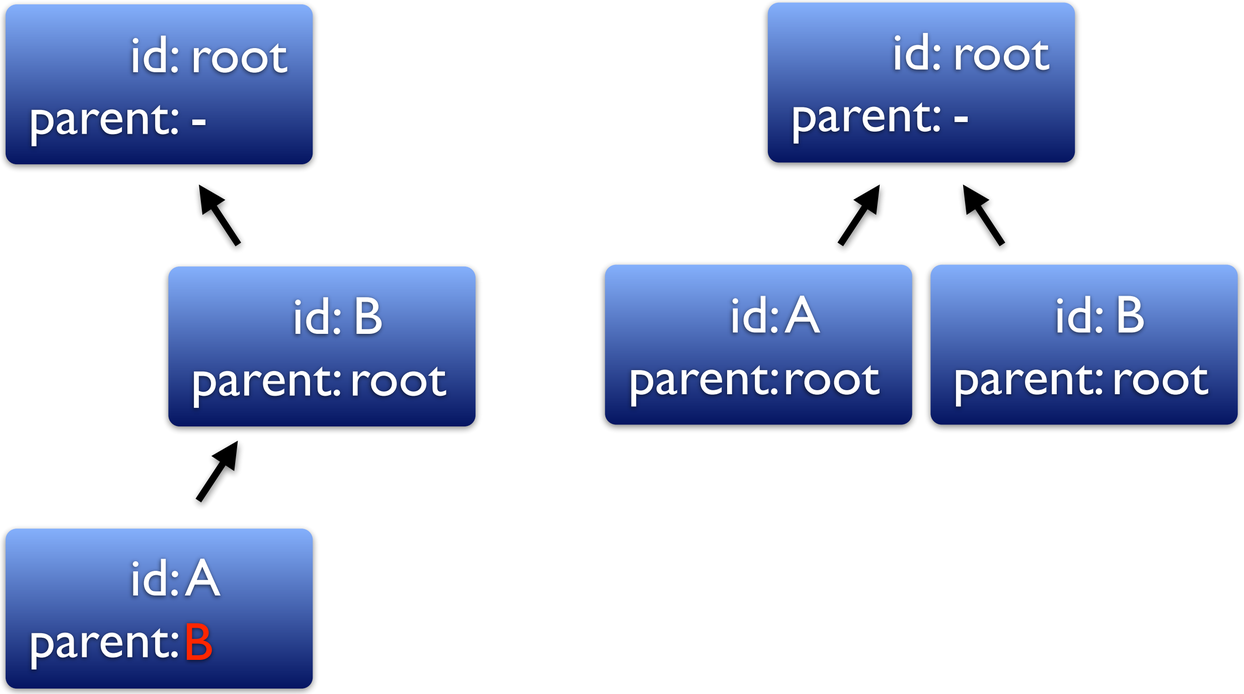
Log

State

State

Log

set parent of **A** to **B**,**root**



Unsere Lösung

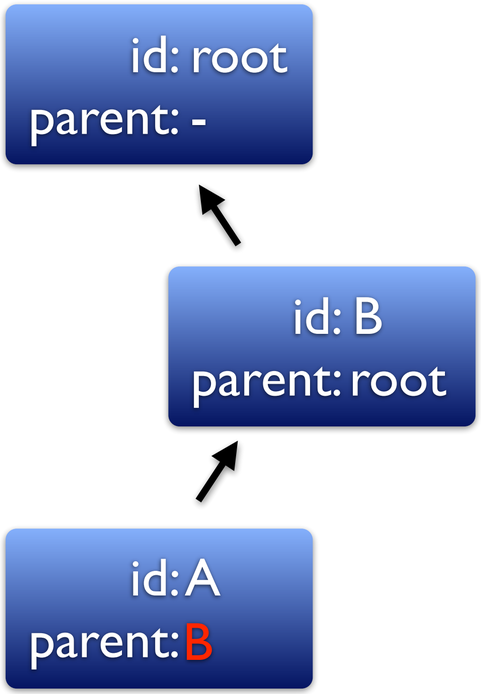
Transformationskonflikte verhindern Schleifen

Alice

Log

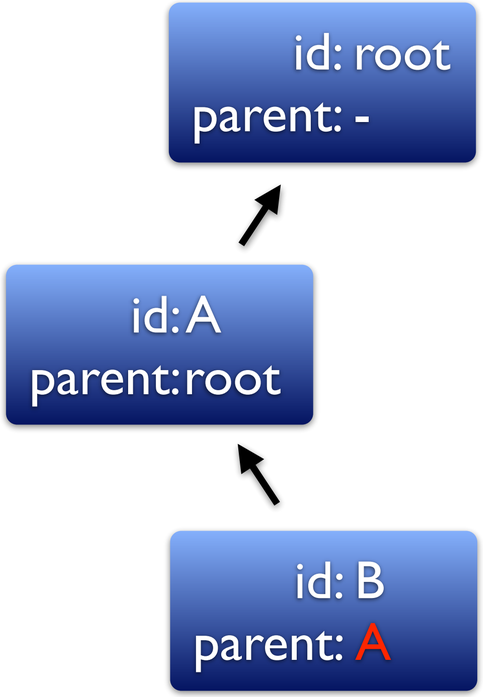
set parent of **A** to **B,root**

State



Bob

State



Log

set parent of **B** to **A,root**

Unsere Lösung

Transformationskonflikte verhindern Schleifen

Alice

Bob

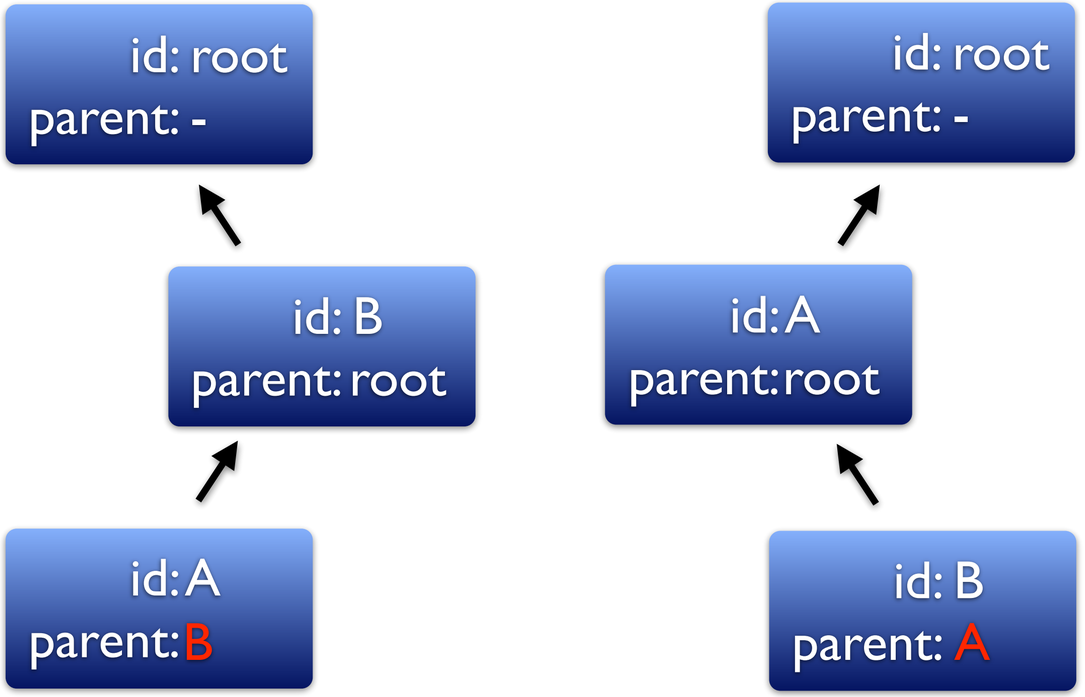
Log

State

State

Log

set parent of **A** to **B,root**



set parent of **B** to **A,root**

set parent of **A** to **B,root**
set parent of **B** to **A,root**

Unsere Lösung

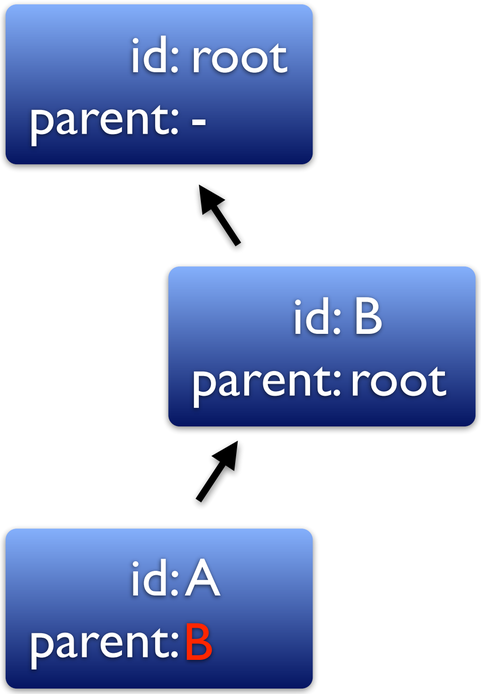
Transformationskonflikte verhindern Schleifen

Alice

Log

set parent of **A** to **B,root**

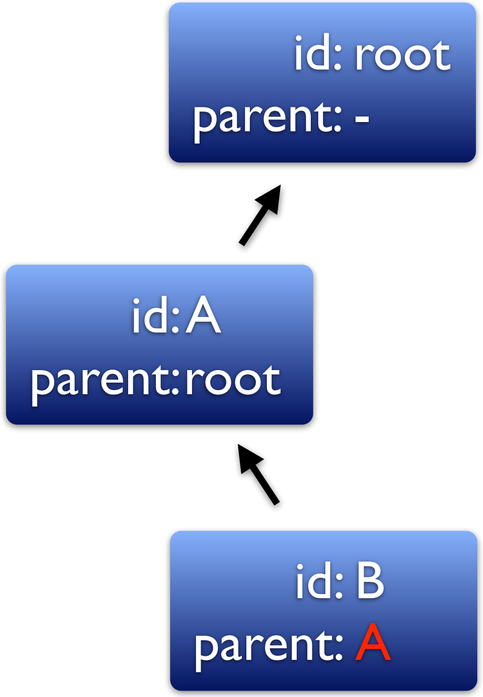
State



set parent of **B** to **A,B,root**

Bob

State



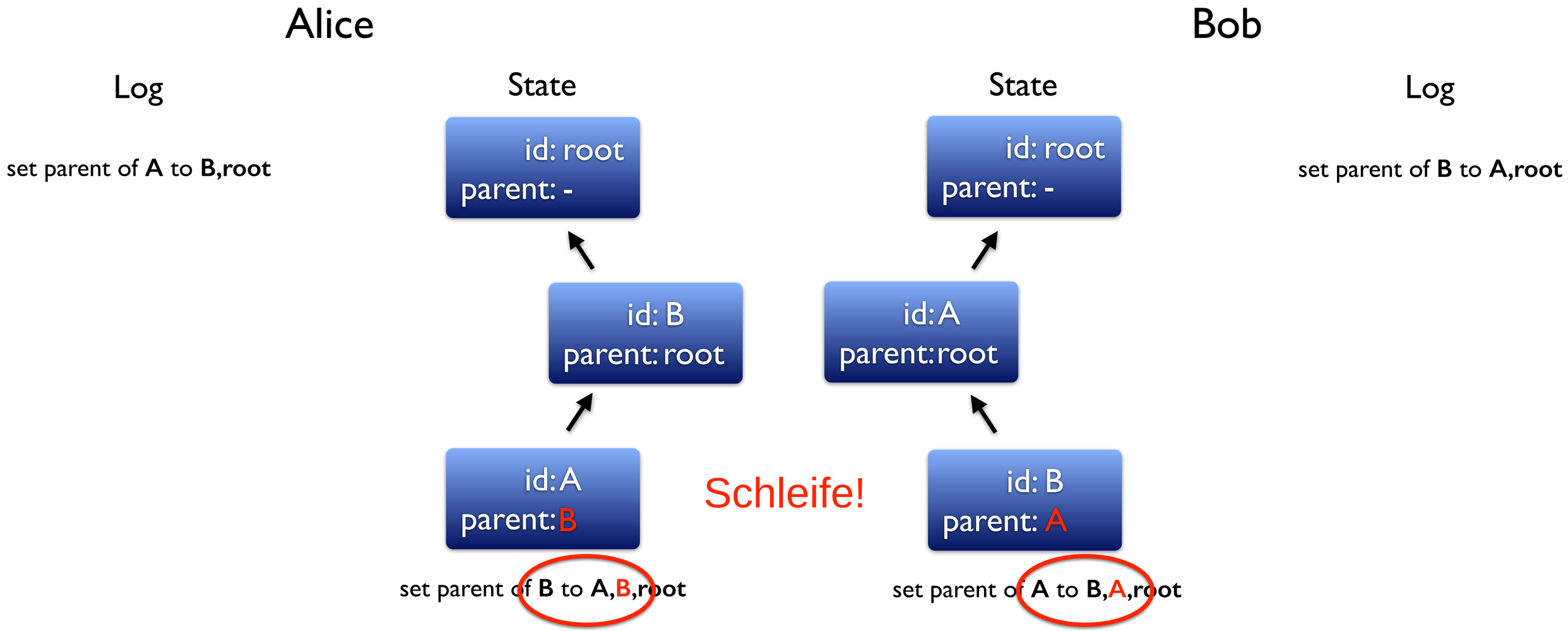
set parent of **A** to **B,A,root**

Log

set parent of **B** to **A,root**

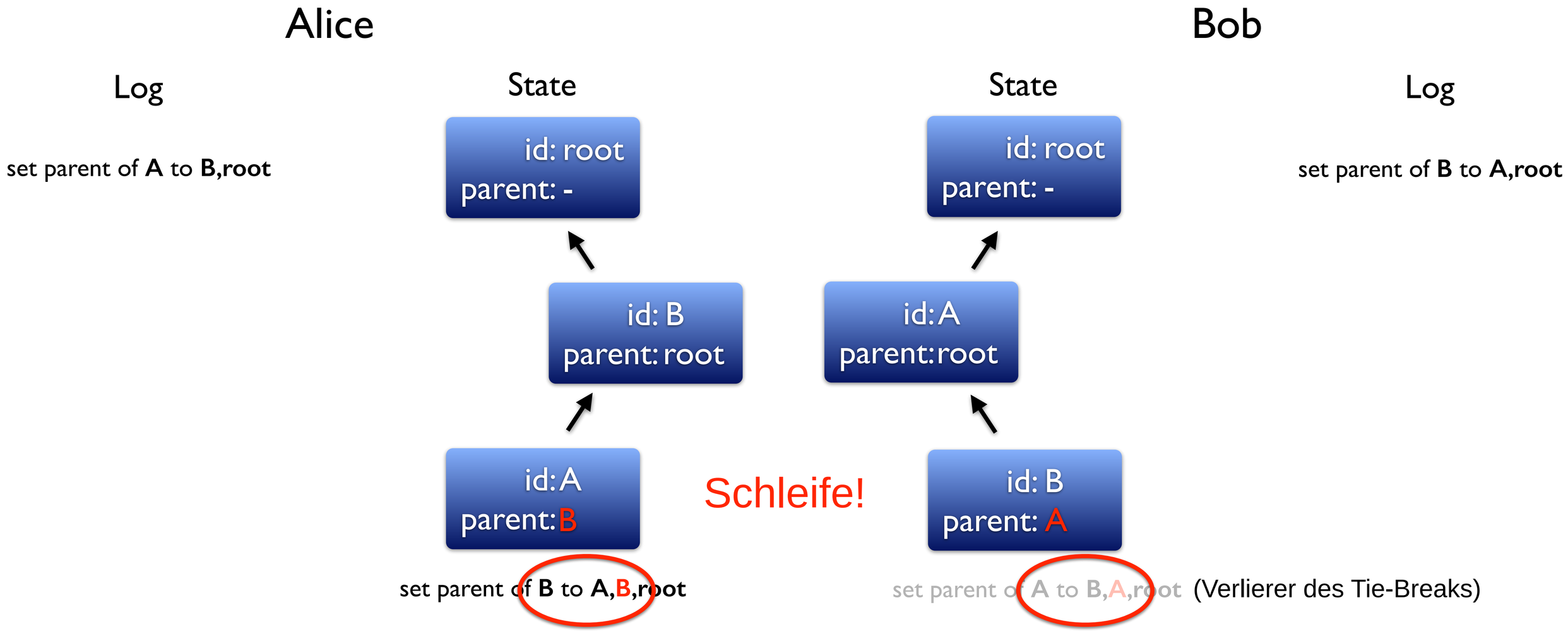
Unsere Lösung

Transformationskonflikte verhindern Schleifen



Unsere Lösung

Transformationskonflikte verhindern Schleifen



Unsere Lösung

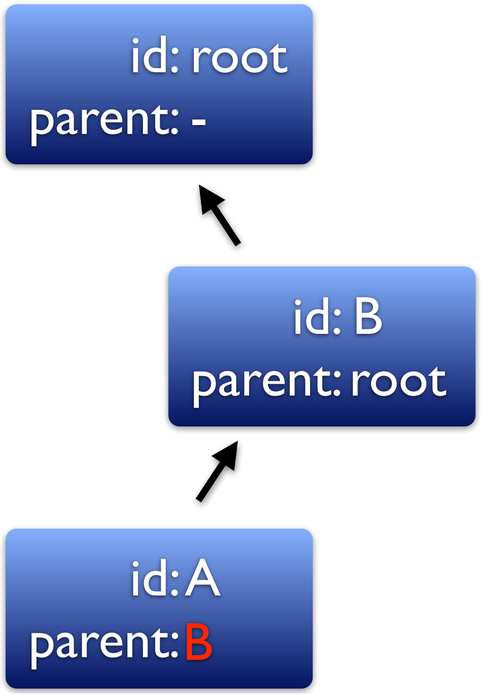
Transformationskonflikte verhindern Schleifen

Alice

Log

set parent of **A** to **B,root**

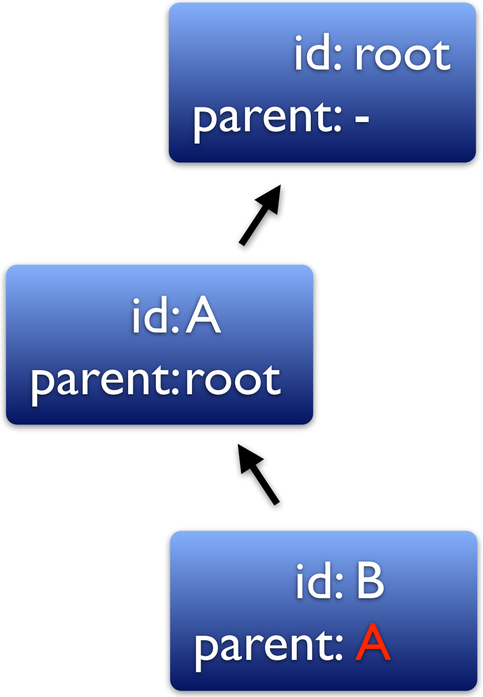
State



set parent of **B** to **A,root**

Bob

State



set parent of **A** to **B,A,root** (Verlierer des Tie-Breaks)

Log

set parent of **B** to **A,root**

Unsere Lösung

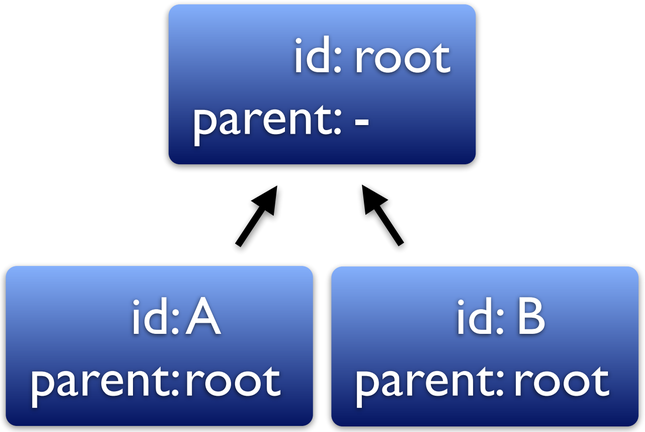
Transformationskonflikte verhindern Schleifen

Alice

Log

set parent of A to B,root

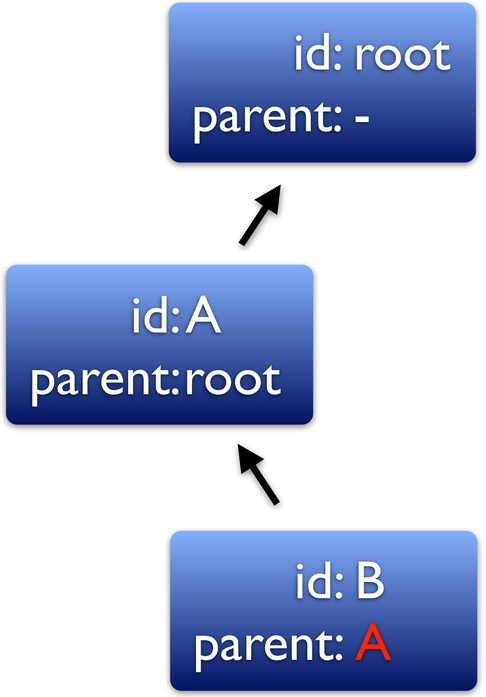
State



set parent of **B** to **A,root**

Bob

State



set parent of A to B,A,root

Log

set parent of **B** to **A,root**

Unsere Lösung

Transformationskonflikte verhindern Schleifen

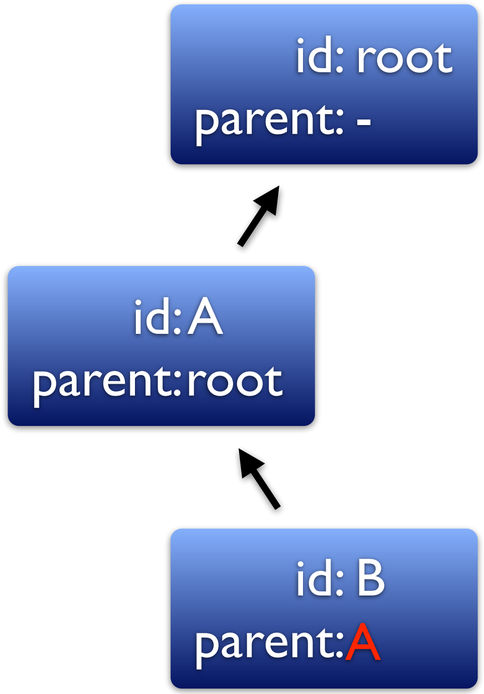
Alice

Log

set parent of A to B,root

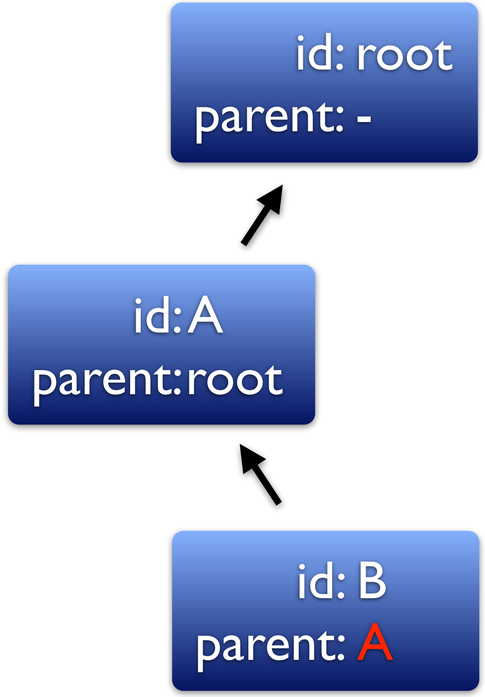
set parent of **B** to **A,root**

State



Bob

State



set parent of A to B,A,root

Log

set parent of **B** to **A,root**

Unsere Lösung

Transformationskonflikte verhindern Schleifen

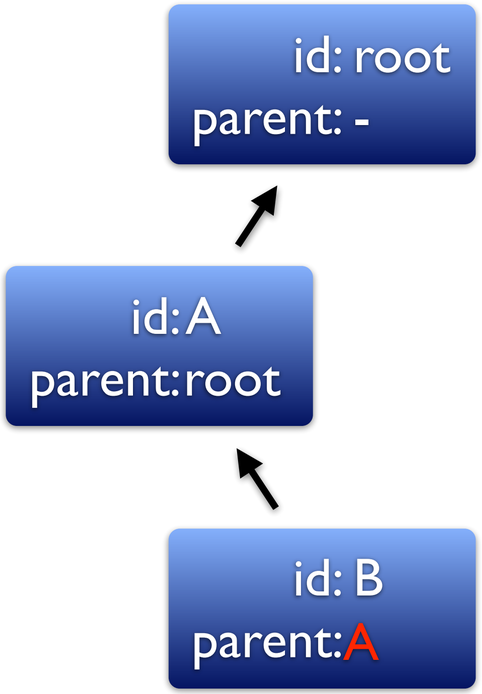
Alice

Log

set parent of A to B,root

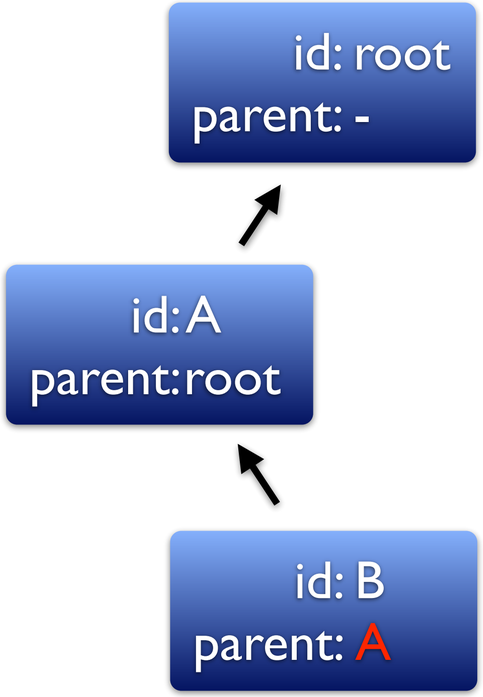
set parent of **B** to **A,root**

State



Bob

State



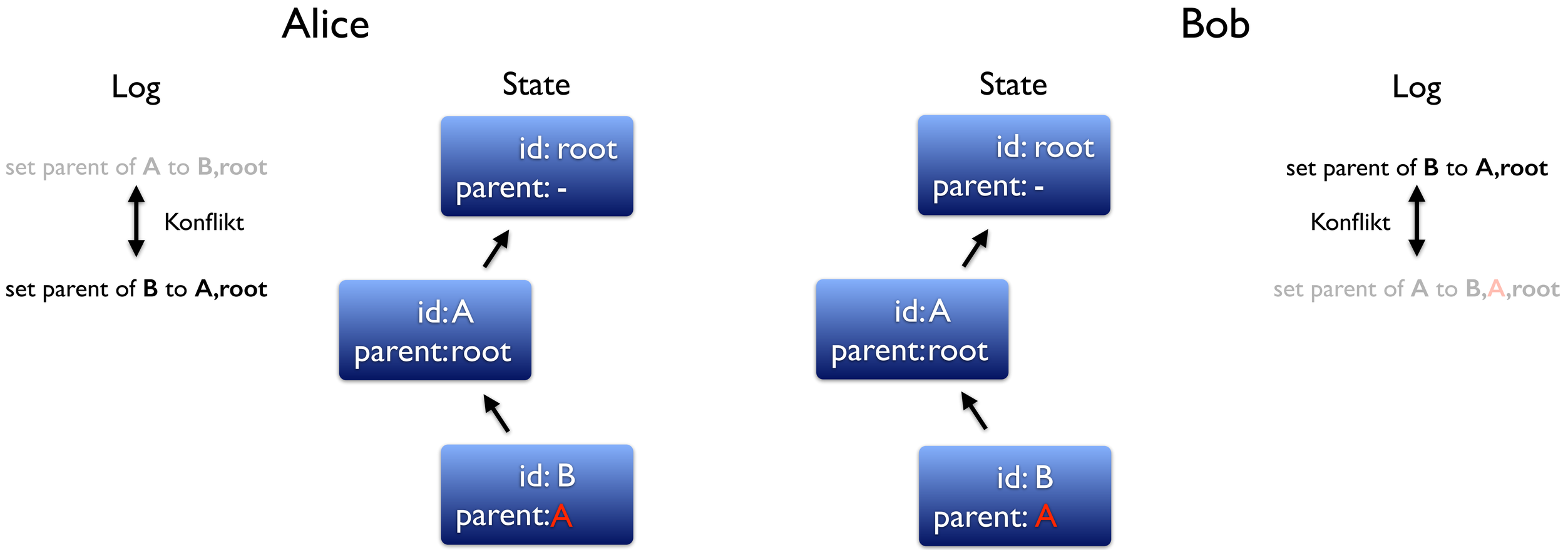
Log

set parent of **B** to **A,root**

set parent of A to B,A,root

Unsere Lösung

Transformationskonflikte verhindern Schleifen

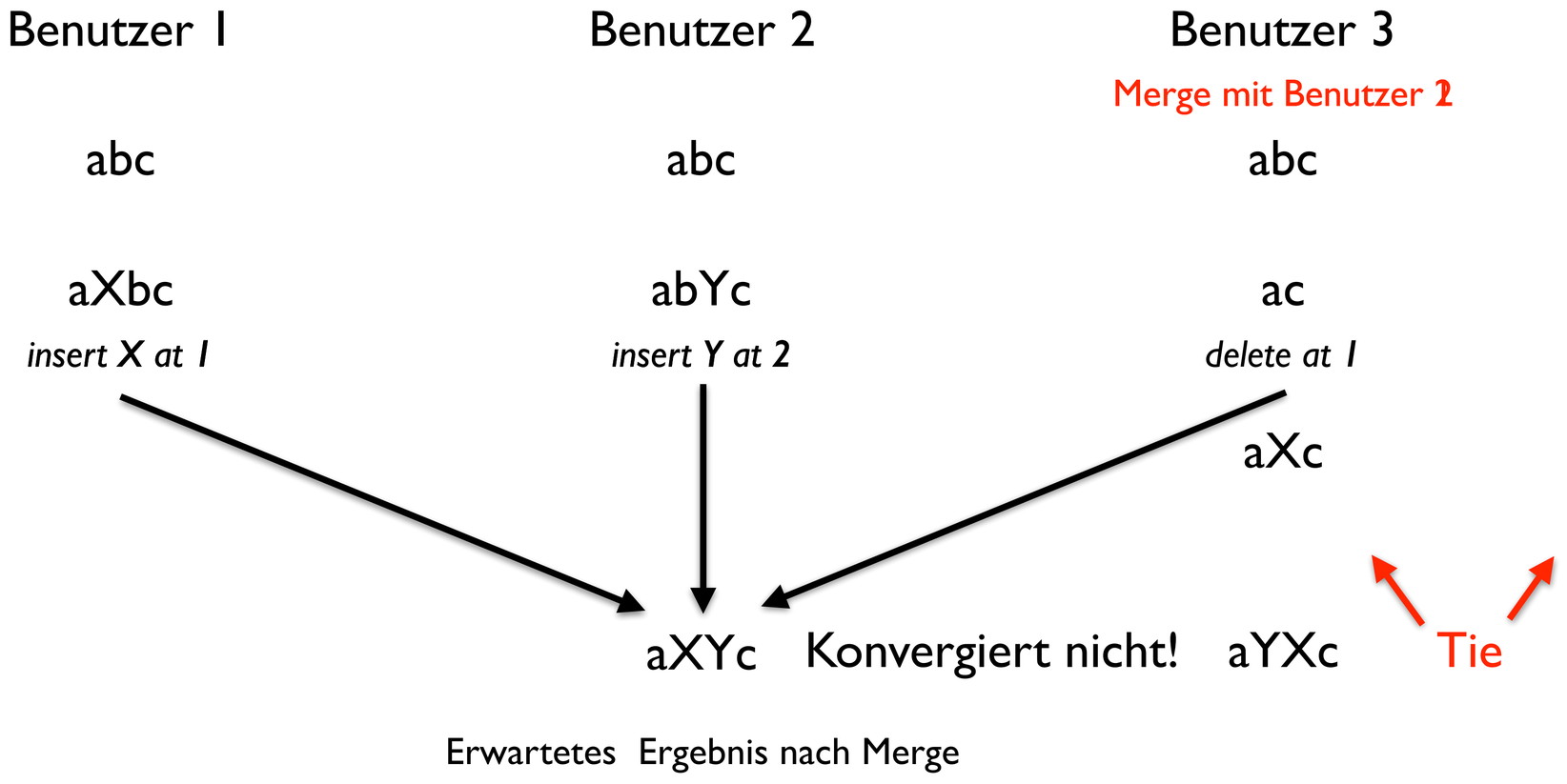


False Ties

- Problem tritt auf, sobald man zum Adressieren Indizes benötigt
  - Geordnete to-many Relationships
  - Text!

False Ties

Beispiel



False Ties

Lösungen

- Grabsteine
  - im Objektgraphen darf nicht gelöscht werden
- Sanftes numerisches Ordnungskriterium als Attribut
  - mit sekundärem Kriterium für Konvergenz
    - z.B. Objekt ID

Zusammenfassung

Zusammenfassung

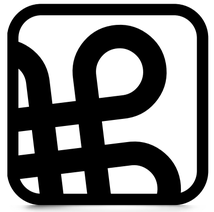
- Es gibt keine allgemeine Syncing-Lösung
- Wenn ihr auf Syncing Probleme stoßt:
  - Wenn möglich, sucht nach pragmatischen Abkürzungen
  - Rechnet damit, eure Business-Logik anpassen zu müssen.
    - Uniqueness constraints
    - Cross-property constraints
    - Wie schwer das ist, hängt stark von der Domäne der Anwendung ab. Plant früh!

Zusammenfassung

- Unterschätzt den Aufwand nicht!
  - „*Wir sind mit unserer App bald fertig, es fehlt nur noch das Syncing…*“
- oder: Sucht euch einen anderen Job!

Fragen?

Vielen Dank



Macoun