

Operational Transformation

Oder: Wie funktioniert Etherpad?

Etherpad

The screenshot displays the Etherpad web interface. At the top, a blue header bar contains the text "EtherPad". Below this, a navigation bar includes a "Public Pad" button, a "Pad Options" button, an "Import/Export" button, a "Saved revisions" button, and a "Time Slider" button. The main editing area features a toolbar with various text formatting icons (bold, italic, underline, strikethrough, list, link, unlink, image, undo, redo) and a "Share" button. The document content is as follows:

27 Arbeit, z.B. das gemeinsame Erstellen von Konzepten, Protokollen und Berichten.

28 In einer fünfteiligen Artikelreihe stellt pb21.de die wichtigsten Werkzeuge für kollaboratives Schreiben vor. Dabei schauen wir nicht nur auf die Funktionen und technischen Voraussetzungen, sondern auch auf mögliche Einsatzszenarien im Bildungszusammenhang.

29 • Teil I: Überblick

30 • Teil II: Etherpad

31 • Teil III: Wikis

32 • Teil IV: Google Docs, Office Web Apps etc.

33 • Teil V: Fazit - Welche Werkzeuge für welche Aufgaben?

34

35 **KOLLABORATIVES SCHREIBEN I: ÜBERBLICK**

36 **Warum und zu welcher Gelegenheit ist kollaboratives Schreiben interessant?**

37 Das gemeinsame Schreiben kann motivierend wirken, Menschen können ihre Kenntnisse einbringen, sich als ein wichtiger Teil einer Gemeinschaft wahrnehmen, die zusammen Inhaltliches erarbeitet. Wenn das ein gewünschter Effekt ist, sollte mit Namen gearbeitet werden, so dass die AutorInnen identifizierbar sind. Andersherum muss man sich vorher überlegen, wie man damit umgeht, dass ein Text von vielen, nicht identifizierbaren AutorInnen stammt, wenn ohne Namen gearbeitet wird.

38 Innerhalb der Organisation oder Institution eingesetzt, kann kollaboratives Schreiben Offenheit und Partizipation fördern - beides wird natürlich auch bei einem Einsatz nach außen signalisiert.

39

40 **Kooperation oder Kollaboration?**

41 Mit dem Begriff *Kollaboration* wird im Deutschen zurückhaltend umgegangen. Wie unterscheiden sich die Bedeutungen von *Kollaboration* und *Kooperation*?

42

43 *Kollaboration* ist z.B.: einen Kuchen zusammen backen. Dem Endprodukt kann man nicht mehr ansehen, wer genau was beigetragen hat. Die *Zusammenarbeit* ist also *zusammen* im Sinne von "gemeinsam" (*to share*).

44 *Kooperation* ist z.B.: ein Staffellauf. Das Ergebnis stammt von mehreren Teammitgliedern gemeinsam, aber es ist deutlich, wer welche Teilleistung beigetragen hat und wo die Schnittstellen untereinander lagen. Die *Zusammenarbeit* ist also *zusammen* im

On the right side, there is a sidebar. At the top, it shows the name "Jöran" next to a yellow square. Below this is a section labeled "Jöran meta". A "Share this pad" button is present. The chat history is as follows:

October 10, 2010

Jöran: ein Anfang ... 16:52

October 11, 2010

unnamed: ich habe ergänzt, wie sollen wir weitermachen? 10:42

Guido: Soll ich Was ist ein Etherpad und Einsatzszenarien übernehmen? 10:44

Jöran: Ich habe aus "meinem" Forum hierher 22:52
verlinkt. Vielleicht bekommen wir ja noch Anregungen oder sogar Co-Autoren.

October 13, 2010

Jöran: Ich habe auch im Forum der Vernetzer 15:55
nochmal hierher verlinkt

ute: so, jetzt bin ich auch dabei 17:14

ute: gute Idee, Danke! 17:14

October 14, 2010

Jöran: linnie: hallo l!te! :-)) 23:20

Chat: [input field]

At the bottom left, a "Zoom:" dropdown menu is set to "100%". At the bottom right, there are two buttons: "Sidebar" and "Full Window".

Warum?

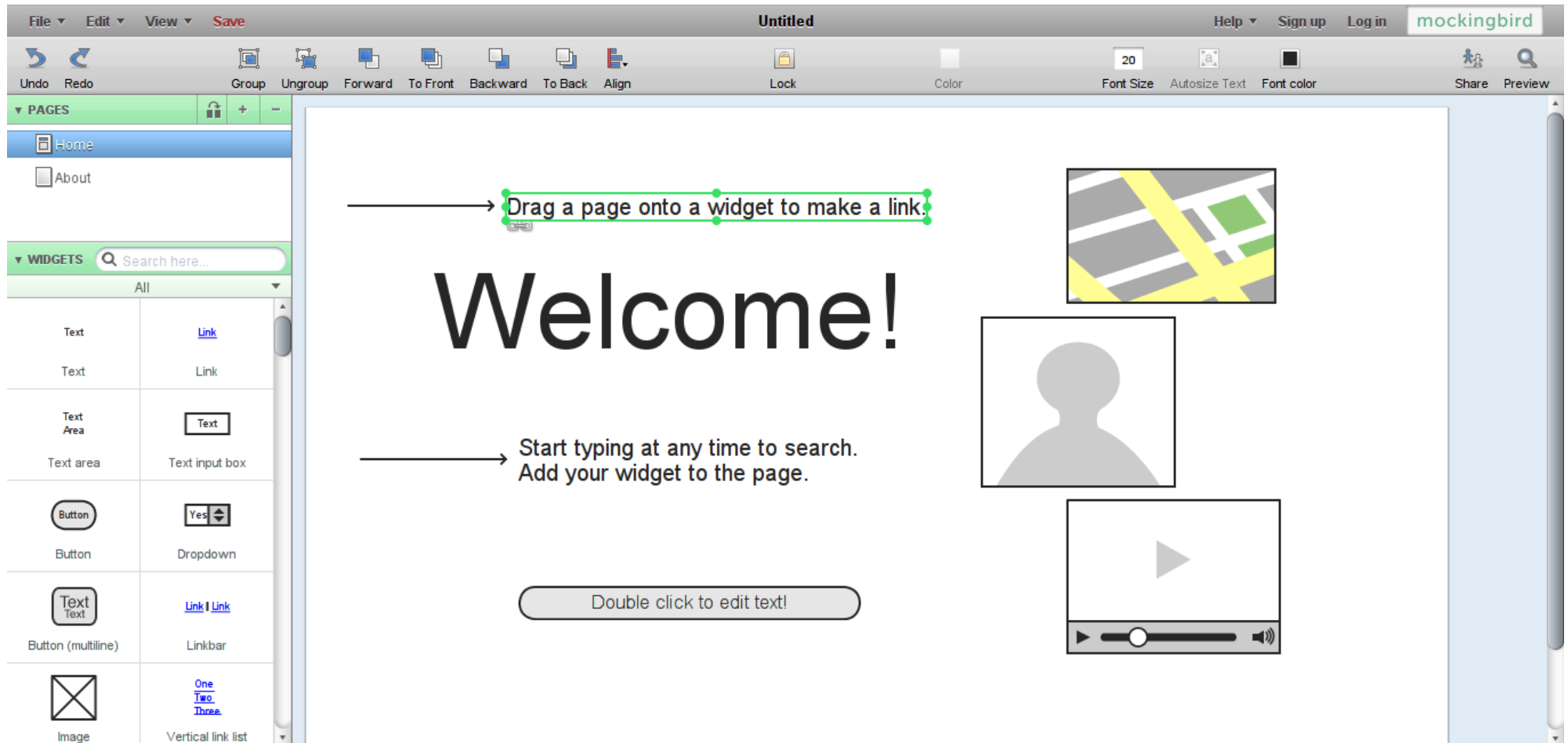
- Jeder ist immer auf dem gleichen Stand
- Kein Chaos mit verschiedenen Versionen einer Datei, die per E-Mail verteilt werden
- Kein versehentliches Überschreiben Änderungen anderer mit eigenen Änderungen
- Schnelles Feedback zu eigenen Änderungen
- Gleichzeitiges, verteiltes Arbeiten möglich

Weitere Echtzeit-Kollaborations-Systeme

- Google Wave
- Google Docs
- ShareLaTeX (LaTeX-Editor)
- SubEthaEdit (Texteditor)
- Mockingbird (Wireframing-Tool für Webdesigner)

Mockingbird

Webpage Screenshot



<https://gomockingbird.com/mockingbird/>

Naive Umsetzungen

- Nach jeder Änderung eines Benutzers wird das Dokument an alle anderen Benutzer geschickt
 - Nachteil: kein gleichzeitiges Tippen möglich, Änderungen werden überschrieben
- Bevor ein Benutzer eine Änderung macht, muss er erst einen Lock anfordern
 - Nachteil: kein gleichzeitiges Tippen möglich

Operational Transformation

- Am häufigsten eingesetzte Technologie für Echtzeitsysteme
- Ursprung: “The Jupiter collaboration system”, entwickelt am Xerox PARC und beschrieben in “High-latency, low-bandwidth windowing in the Jupiter collaboration system” (1995)

Überblick über Operational Transformation

Allgemeiner Algorithmus zum Integrieren
der Operationen

Applikationsspezifische
Transformations-Funktion

Operationen

- Operationen repräsentieren Änderungen an einem Dokument
- Können auf ein Dokument in einem bestimmten Zustand angewendet werden, wodurch sich ein neuer Zustand ergibt
- “Command Pattern”: Entwurfsmuster, in dem Änderungen des Benutzers als Objekte repräsentiert und in der Undo-Historie gespeichert werden

Operationen an Textdokumenten

- Füge den Buchstaben 'A' an Position 33 ein:
`Insert(33, 'A')`
- Lösche den Buchstaben an Position 45:
`Delete(45)`
- Ähnlichkeit zu Diffs

Beispiel

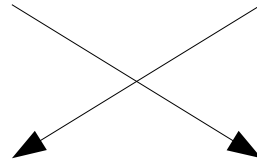
Anfangsdokument: `Haallo Wlt!`

Alice

Delete(1) \Rightarrow
`Hallo Wlt!`

Bob

Insert(8, 'e') \Rightarrow
`Haallo Welt!`

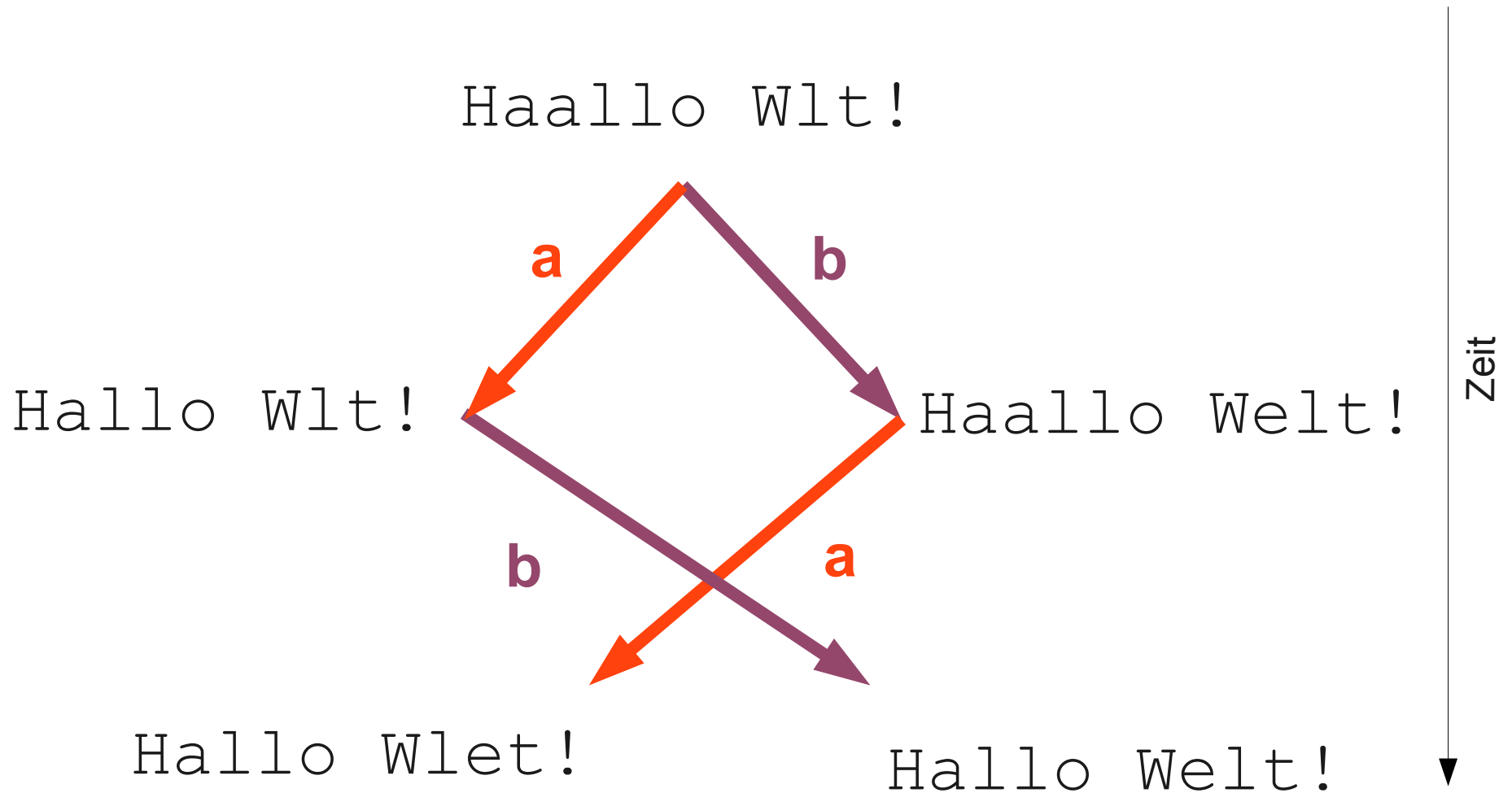


Gegenseitiger Austausch und Anwenden der Operationen

`Hallo Wlet!` \neq `Hallo Welt!`

Problem!

Zustandsdiagramm

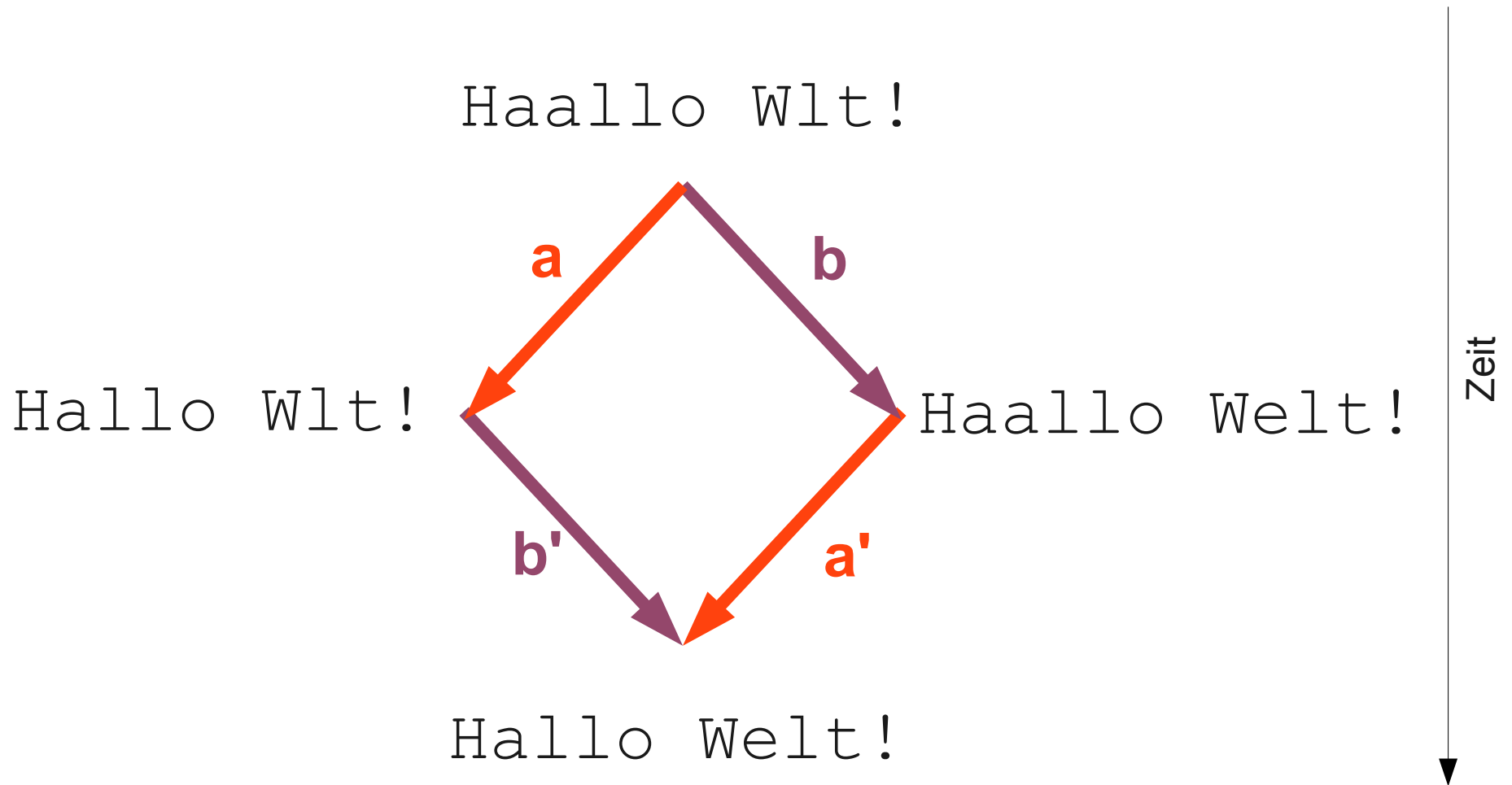


Lösung: Transformations-Funktion

Die Transformationsfunktion nimmt als Argumente zwei Operationen **a** und **b**, die zeitgleich erzeugt wurden, und gibt zwei transformierte Operationen **a'** und **b'** zurück, sodass, wenn **b'** nach **a** und **a'** nach **b** angewendet wird, der Endzustand des Dokumentes der gleiche ist. Das bedeutet, dass für alle Paare von Operationen, die an dem gleichen Zustand des Dokuments doc angewendet wurden, gelten muss:

```
(a', b') := transform(a, b)
apply(apply(doc, a), b') = apply(apply(doc, b), a')
```

Zustandsdiagramm



Transformations-Funktion für Textdokumente

- Erinnerung: Die Operationen sind `insert(int, char)` und `delete(int)`
- Wenn eine der beiden Operationen an einer früheren Position im Dokument als die andere anfängt, dann wird die andere Operation folgendermaßen geändert:
 - Wenn die frühere Operation eine `insert`-Operation ist, dann wird die Position der späteren Operation um eins erhöht
 - Andernfalls ist die frühere Operation eine `delete`-Operation und die Position der späteren wird um eins verringert

Transformations-Funktion für Textdokumente (Forts.)

- Andernfalls werden die beiden Operationen an der gleichen Position im Dokument angewendet.
 - Wenn beides delete-Operationen sind, gib zwei leere Operationen zurück.
 - Wenn es sich um eine insert- und eine delete-Operation handelt, erhöhe die Position der delete-Operation um eins.
 - Wenn beides insert-Operationen sind, erhöhe die Position der Operation, die als zweites Argument übergeben wurde, um eins.

Zusammenfügen von Operationen

- Bisher: jede Operation fügt einen Buchstaben ein oder löscht einen Buchstaben.
- Neue Definition von Operationen: eine Operation besteht aus einer Liste von diesen Anweisungen:
 - `retain(n)`: Überspringe die nächsten `n` Buchstaben
 - `insert(string)`: Füge den String an der aktuellen Cursorposition ein
 - `delete(string)`: Lösche die angegebenen Buchstaben an der aktuellen Cursorposition

Zusammenfügen von Operationen (Forts.)

- Dadurch können nacheinander stattfindende Operationen zu einer einzigen zusammengefasst werden.
- Beispiel:

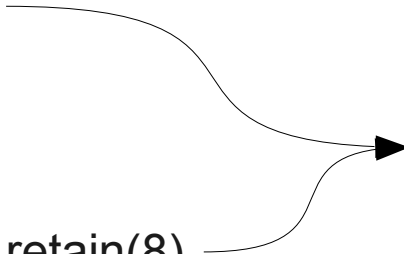
Hallo

retain(5); insert(" Welt")

Hallo Welt

retain(1); delete(1); insert("e"); retain(8)

Hello Welt



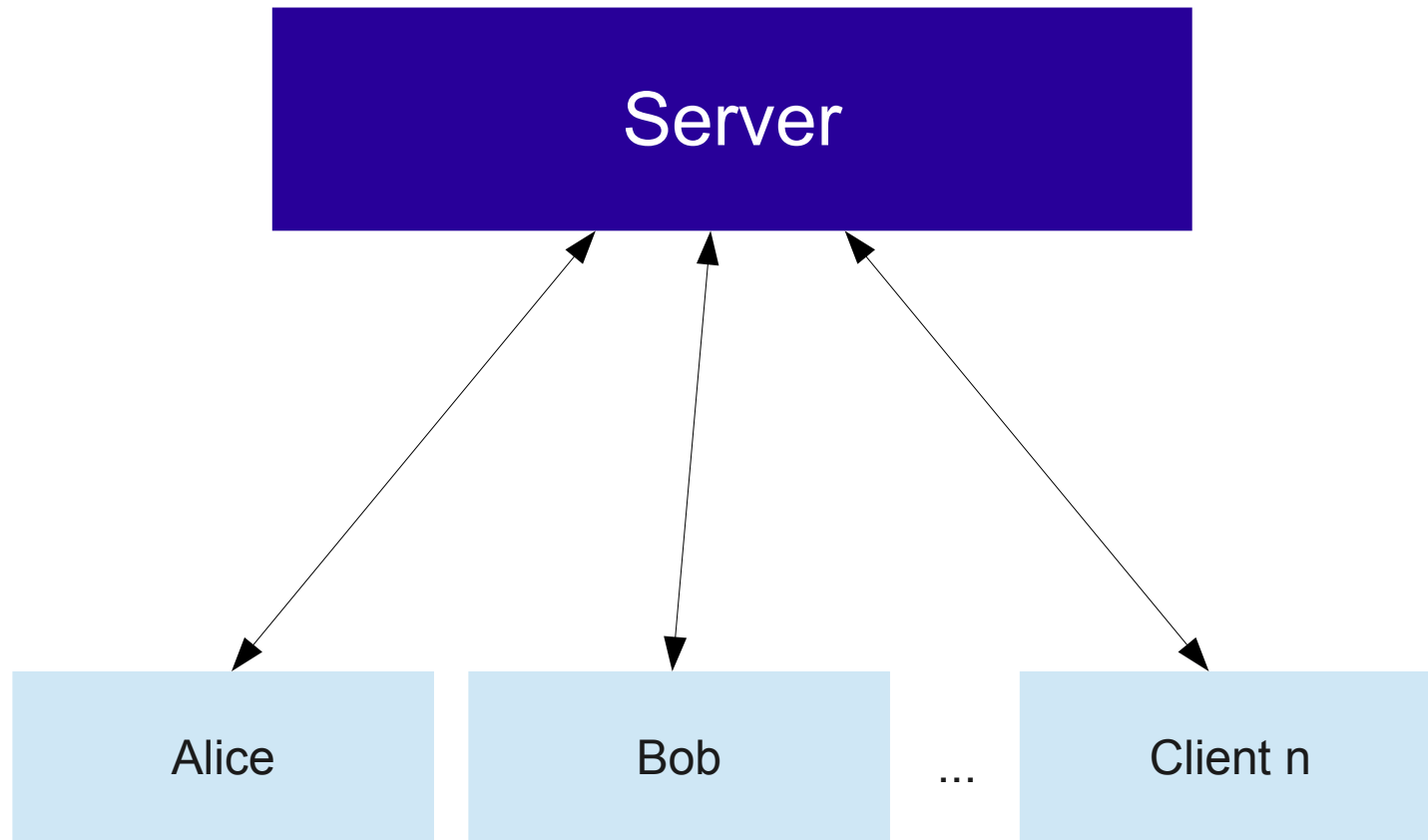
retain(1); delete(1); insert("e");
retain(3); insert(" Welt")

Überblick über Operational Transformation

Allgemeiner Algorithmus zum Integrieren
der Operationen

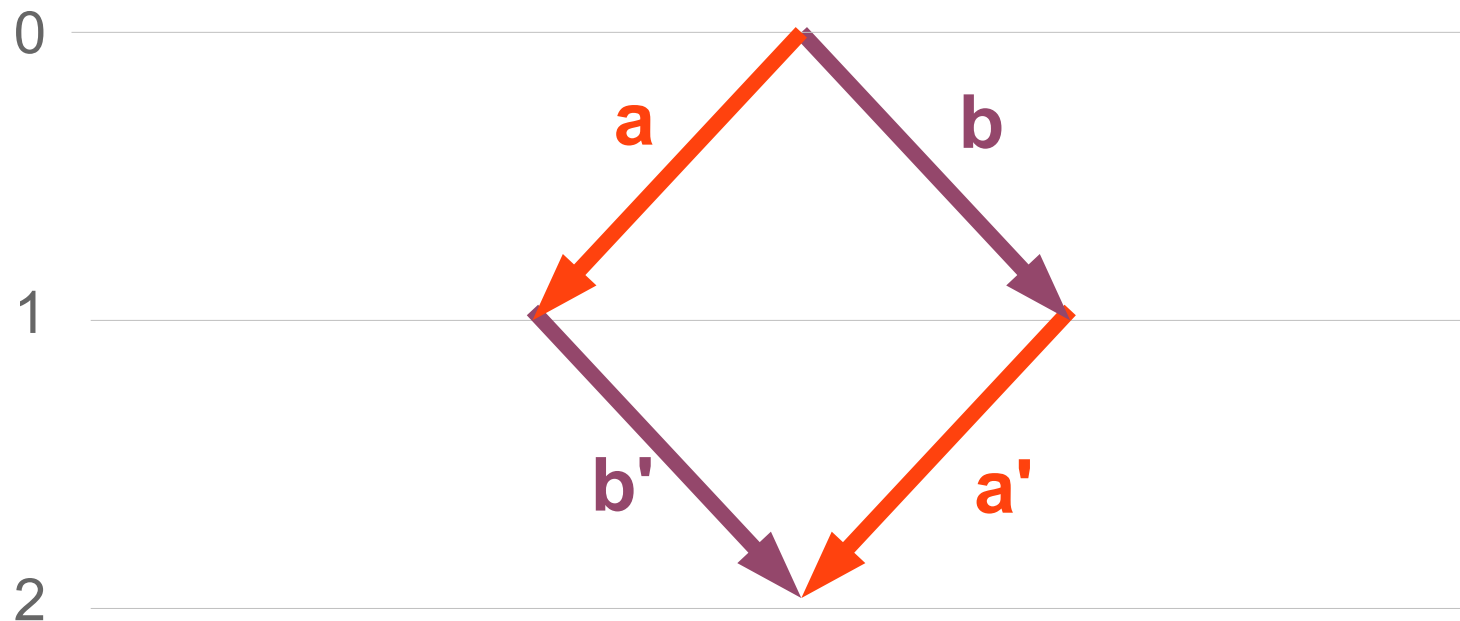
Applikationsspezifische
Transformations-Funktion

Architektur



Revisionsnummern

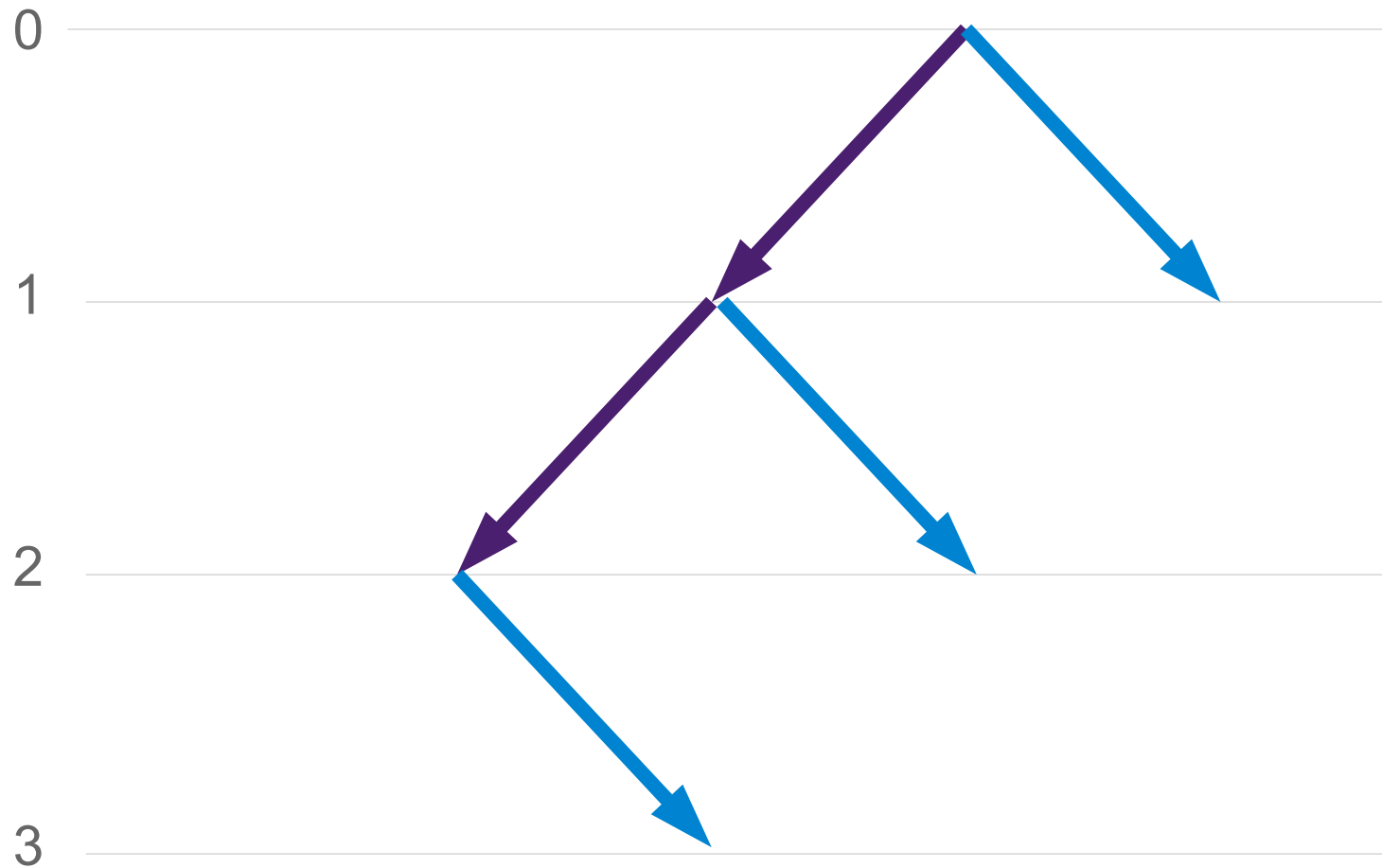
Fortlaufende Nummern, beginnend bei 0.



Aufgaben des Servers

- Wenn sich ein Client verbindet, schickt der Server ihm das aktuelle Dokument und die aktuelle Revisionsnummer.
- Wenn ein Client dem Server eine Operation mit Revisionsnummer des Ausgangszustands schickt, transformiere die Operation gegen alle Operationen, die seit dieser Revisionsnummer geschehen sind, speichere die so erhaltene Operation in der Operationshistorie (und in einer Datenbank) und schicke sie an alle verbundenen Clients zusammen mit dem Namen/ID des Urhebers (auch den Urheber der Operation)

Aufgaben des Servers



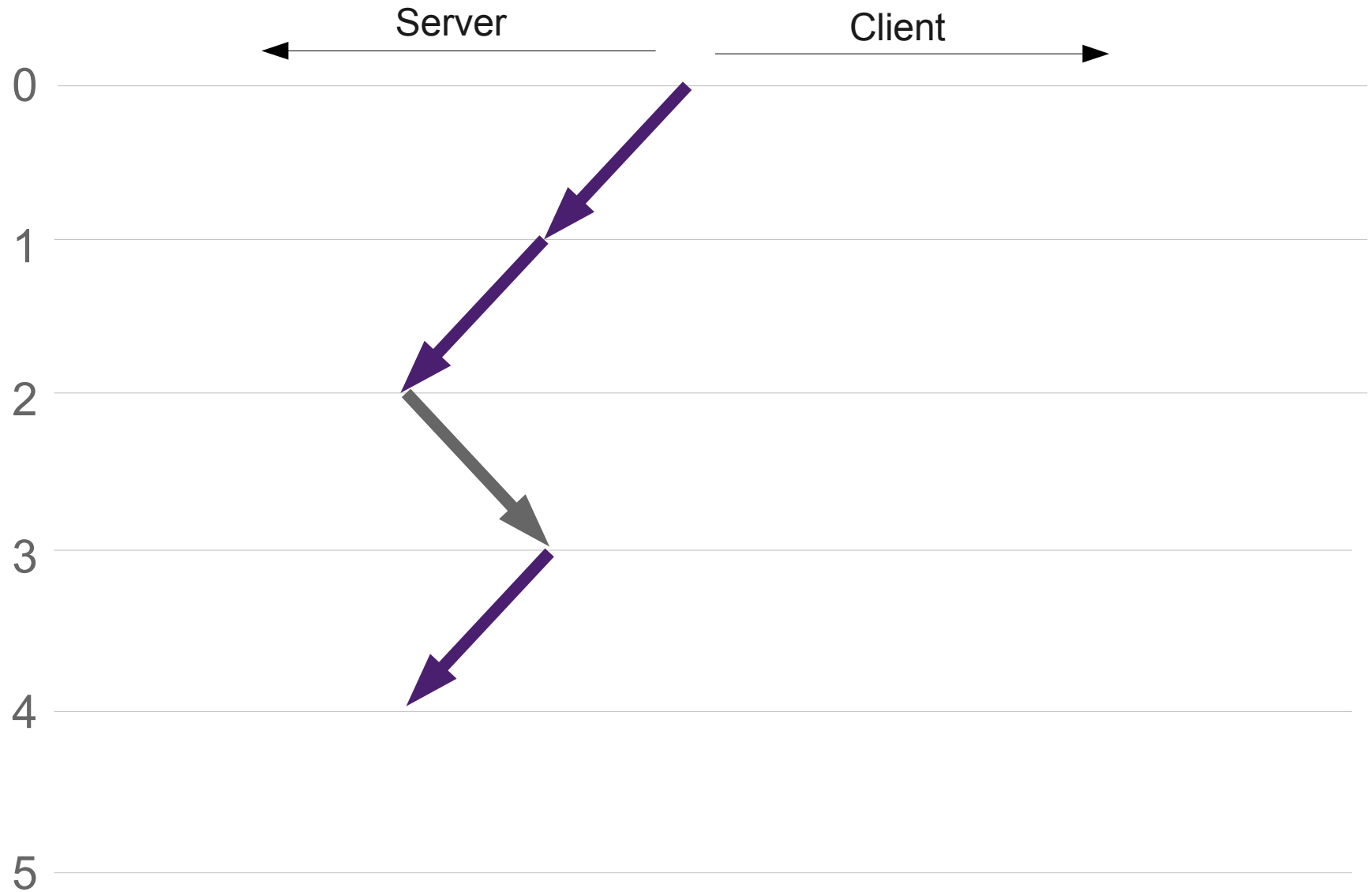
Aufgaben des Clients

- Wenn der Client eine Operation eines anderen Benutzers empfängt, so transformiert er sie so, dass sie am aktuellen Zustand angewendet werden kann.
- Die eigenen Operationen werden so transformiert, dass sie an einem Dokumentenzustand des Servers angewendet werden können.

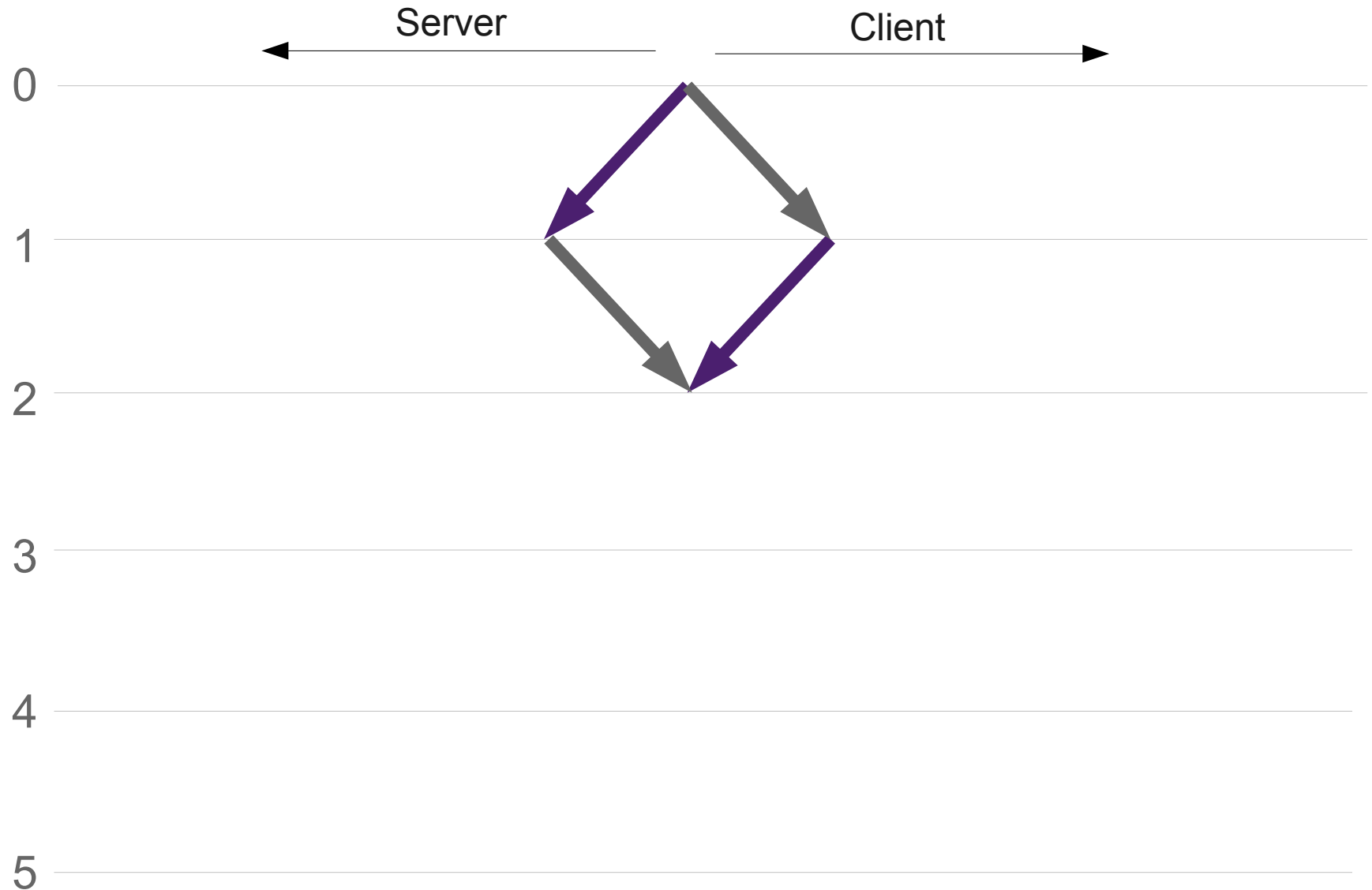
Zustände des Clients

- **Synchronisiert:** der Client hat keine eigenen, offenen Operationen
- **Wartend:** der Client hat eine Operation an den Server gesendet und sie noch nicht zurück erhalten
- **Wartend mit Buffer:** seitdem der Client eine Operation an den Server gesendet und noch nicht zurück erhalten hat, hat der Benutzer weitere Änderungen vorgenommen. Diese Änderungen werden in einem Buffer zusammengefügt.

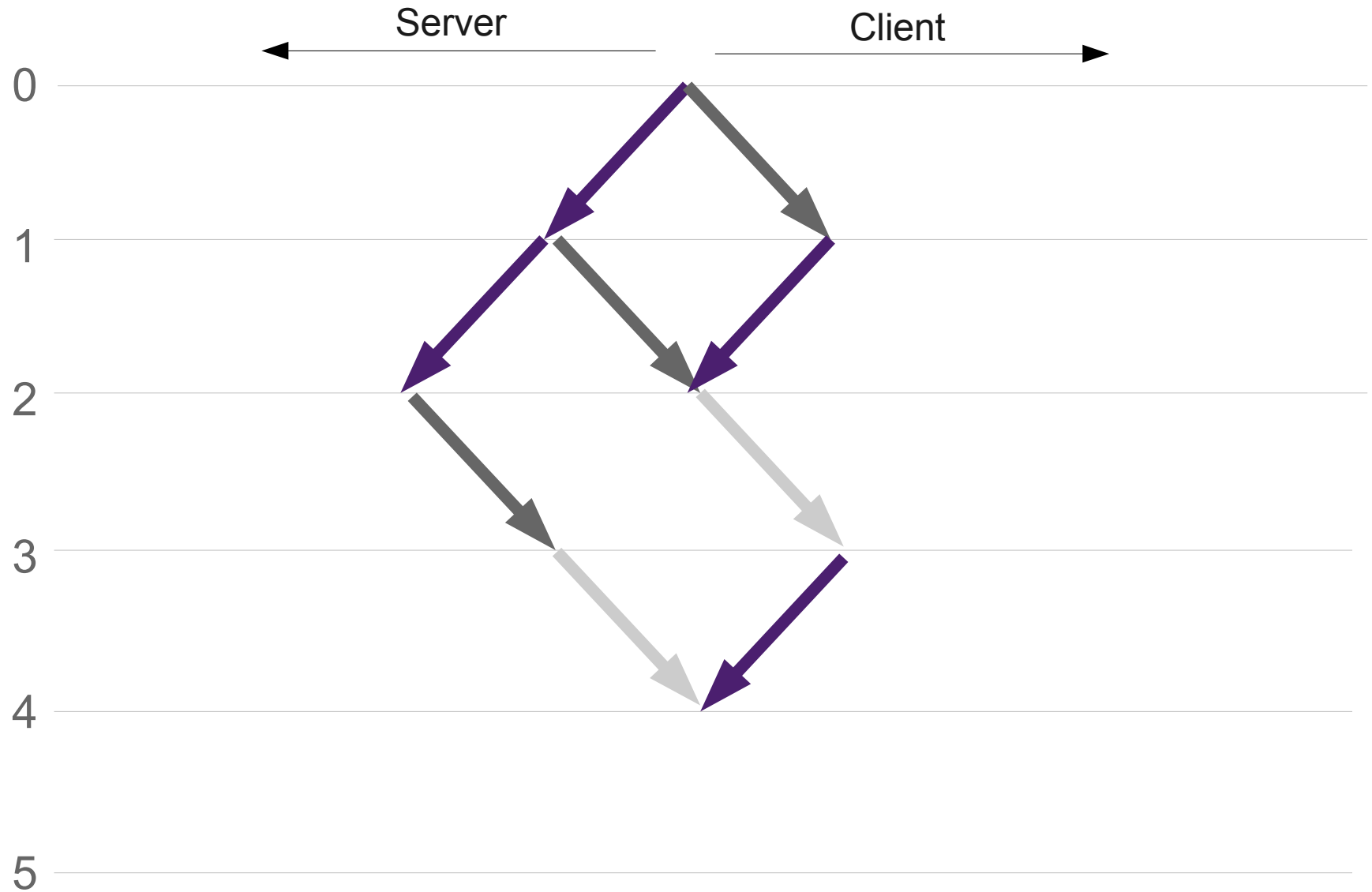
Beispiel 1



Beispiel 2



Beispiel 3



One more thing ...

Demo

Links

- Diese Präsentation: bit.ly/KckmBK
- Daniel Spiewak, “Understanding and Applying Operational Transformation”, bit.ly/bIJAng
(sehr ausführliche Beschreibung der Aufgaben des Clients)
- Meine Implementation: bit.ly/LqHfyF