

[Frankfurt University of Applied Sciences]
[Engineering Business Information Systems]
[Strukturierte Datenspeicher]
[Semester 3]

Verwaltungsapplikation mit Neo4J
&
Chat-Anwendung mit Redis

Vorgelegt von:

Umesh Dahal(1349090)
Akriti Akriti(1310786)
Anne Ateba(1349443)

Gutachter:

Prof. Andreas Berndt

INHALT

Seite Nr.

Teil 1

1. Verwaltungsapplikation mit Neo4J

| | | |
|---------------|--------|-----|
| 1.1. Konzept | -----→ | 1 |
| 1.2. Beispiel | -----→ | 2-4 |

Teil 2

1. Chat-Anwendung mit Redis

| | | |
|---------------|--------|---|
| 1.1. Konzept | -----→ | 5 |
| 1.2. Beispiel | -----→ | 6 |

Teil 1

1. Verwaltungsapplikation mit Neo4J

1.1. Konzept:

Die Verwaltung Applikation ist für die Zuordnung zwischen Organisationsstruktur und Mitarbeiter realisiert. In unserem Projekt haben wir die Organisation als "Tessa Motor" genannt. Die Organisation enthält 3 verschiedene Ressort(Beschaffung, Finanz, Produktion) & unter jeder Ressort befindet sich weitere drei Abteilungen mit zwei weiteren Gruppen pro Abteilung. Jeder Ressort/Abteilung/Gruppe ist durch einem Leiter geführt. Der GruppeLeiter arbeitet zusammen mit Mitarbeiter in einer Gruppe. Der Mitarbeiter/Leiter hat folgende Informationen: Vorname, Nachname, Adresse(Straße, PLZ & Ort). Je Gruppe besteht aus 2 Mitarbeitern und einem Leiter. Mitarbeiter und Gruppeleiter aus verschiedenen Gruppen arbeiten gruppenübergreifend zusammen.

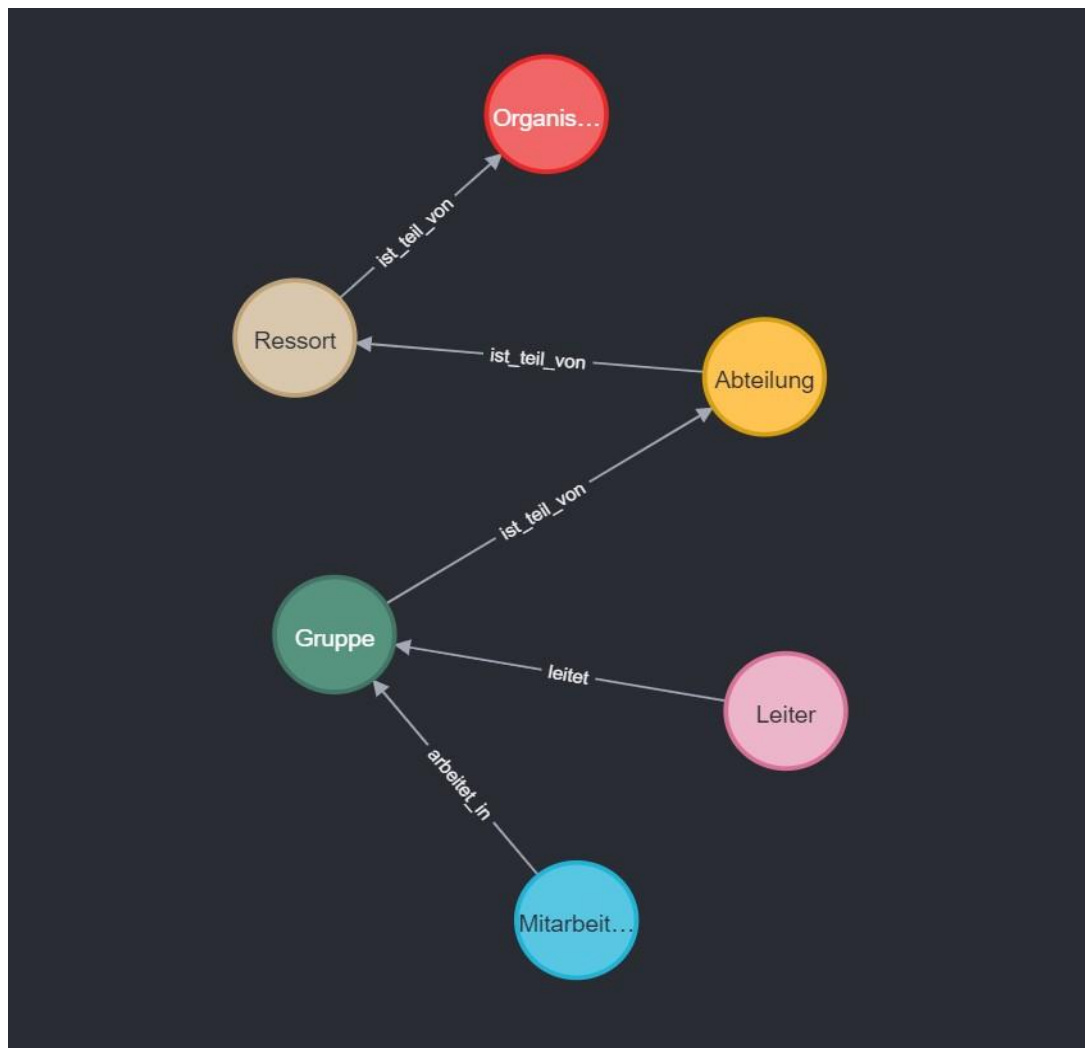


Fig 1: Aufbau der Projektstruktur in Neo4j

1.2. Beispiel

1.2.1 Relation zwischen Ressort zur Abteilung zur Gruppe:





Es gibt drei Ressorts (Farbe ) & unterhalb der drei Ressort gibt es weitere drei Abteilungen pro Ressort und insgesamt 9 (Farbe ). Wieder hat jede Abteilung noch weitere 2 Gruppe & insgesamt 18 (Farbe )



Fig2: Relation zwischen Ressort zur Abteilung zur Gruppe.

1.2.2. Relation Zwischen Leiter und Organisationsstruktur:

Jede Ressort/Abteilung/Gruppe ist von einem Leiter geführt. Insgesamt gibt es 30 Leitern(Farbe ) . Jeder Leiter hat folgende Informationen(Vorname, Name, Adresse(Straße, PLZ, Ort)). Gruppe Leiter arbeiten zusammen mit Mitarbeiter und können gruppenübergreifend in anderer Gruppe zusammen arbeiten.

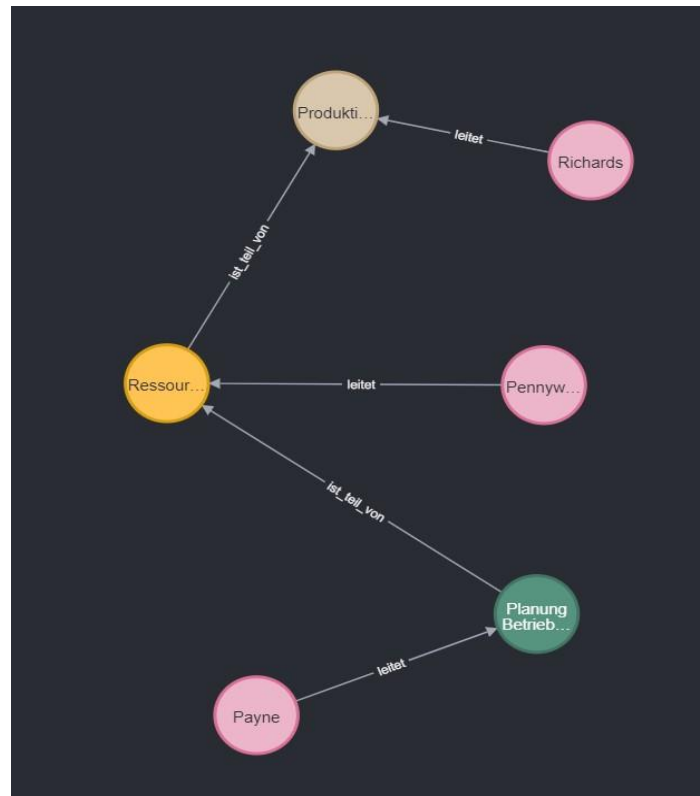


Fig3: Relation zwischen Leiter und Organisationsstruktur

1.2.3. Relation zwischen Mitarbeiter und Gruppe:

Alle Mitarbeiter arbeiten in Gruppe. Es sind insgesamt 36 Mitarbeiter(Farbe●) aus alle Gruppen. Mitarbeiter haben auch gleiche Daten wie Leiter(wie Vorname, Nachname, Adresse(Straße, PLZ, Ort)).

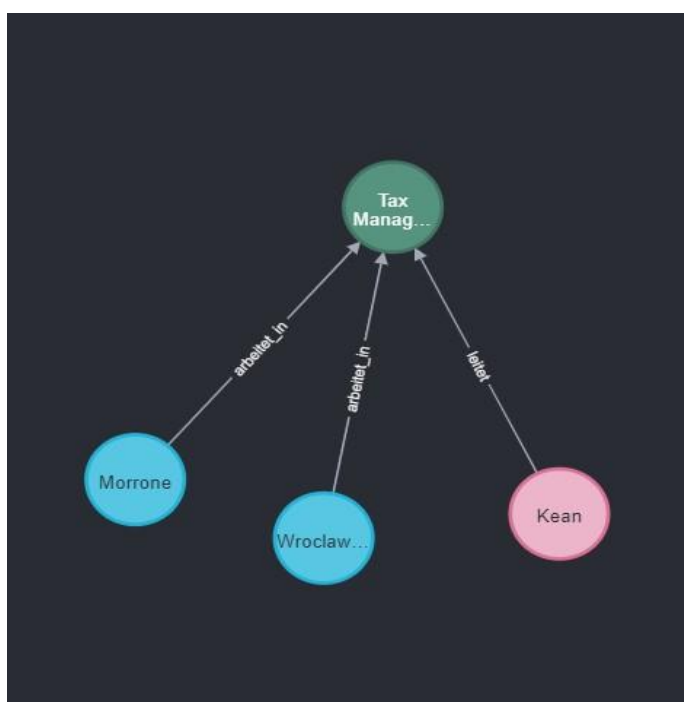


Fig4: Relation zwischen Mitarbeiter und Gruppe



| Mitarbeiter | |
|-------------|---------------|
| <id> | 221 |
| Nachname | Pike |
| Ort | Hattersheim |
| PLZ | 65795 |
| Straße | Am Markt 101a |
| Vorname | Joe |

Fig5: Daten der Mitarbeiter

1.2.4. Relation gruppenübergreifend Arbeit:

Mitarbeiter und Leiter unter einer Gruppe können in einem Projekt bei anderer Gruppe arbeiten. Wie in diesem Diagramm z.B. arbeitet Cabone in 2 weitere Gruppe (Planung Betriebsmittel & Produkt Verbesserung), damit arbeitet er unter der Gruppeleiter (Duncan & Payne) mit. Das gleiche ist bei Duncan & Payne, Sie arbeiten zusammen in anderen Gruppen mit & bilden einen Kreislauf zwischen 3 Gruppe innerhalb einer Abteilung.

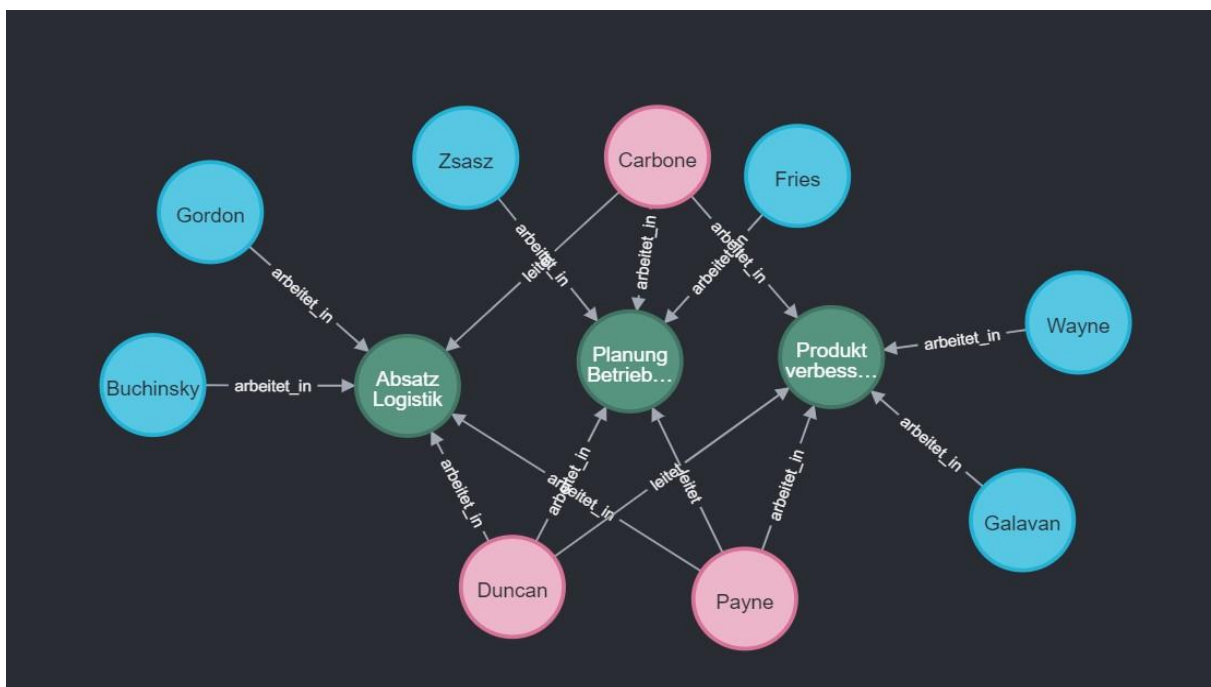


Fig6: Relation gruppenübergreifend Arbeit

Teil 2

1. Chat-Anwendung mit Redis

1.1. Konzept:

Der Konzept ist zur Aufbau Chat-Anwendung, die die Austausch der Nachrichten echtzeit zwischen der Nutzern ermöglicht. Dafür ist Redis Server benötigt und ist in C# codiert.

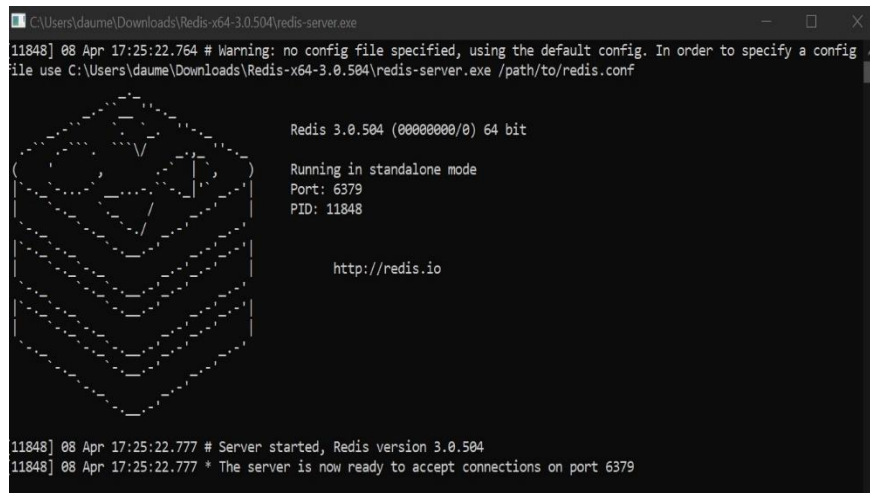


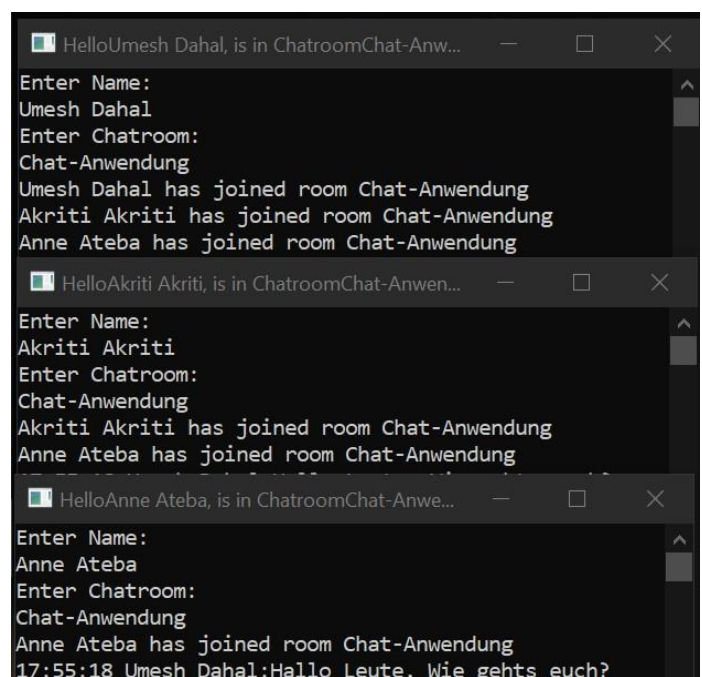
Fig 1: Redis in Hintergrund

1.2. Broadcasting:

Für die Broadcasting wird Package 'StackExchange.Redis' benötigt. Diese ermöglicht die Verbindung von mehreren Threads. Die Pub/Sub-Modell ermöglicht das asynchrone Broadcasten von Nachrichten über mehrere Bereiche der Anwendungen. Die Kernkomponente, die diese Funktionalität ermöglicht, ist ein sogenanntes "Topic/Room". Der Publisher schickt Nachrichten an Topic , und das Topic schickt die Nachricht sofort an alle Subscriber.

```
[using StackExchange.Redis;  
pubSub.Subscribe();  
pubSub.Publish();]
```

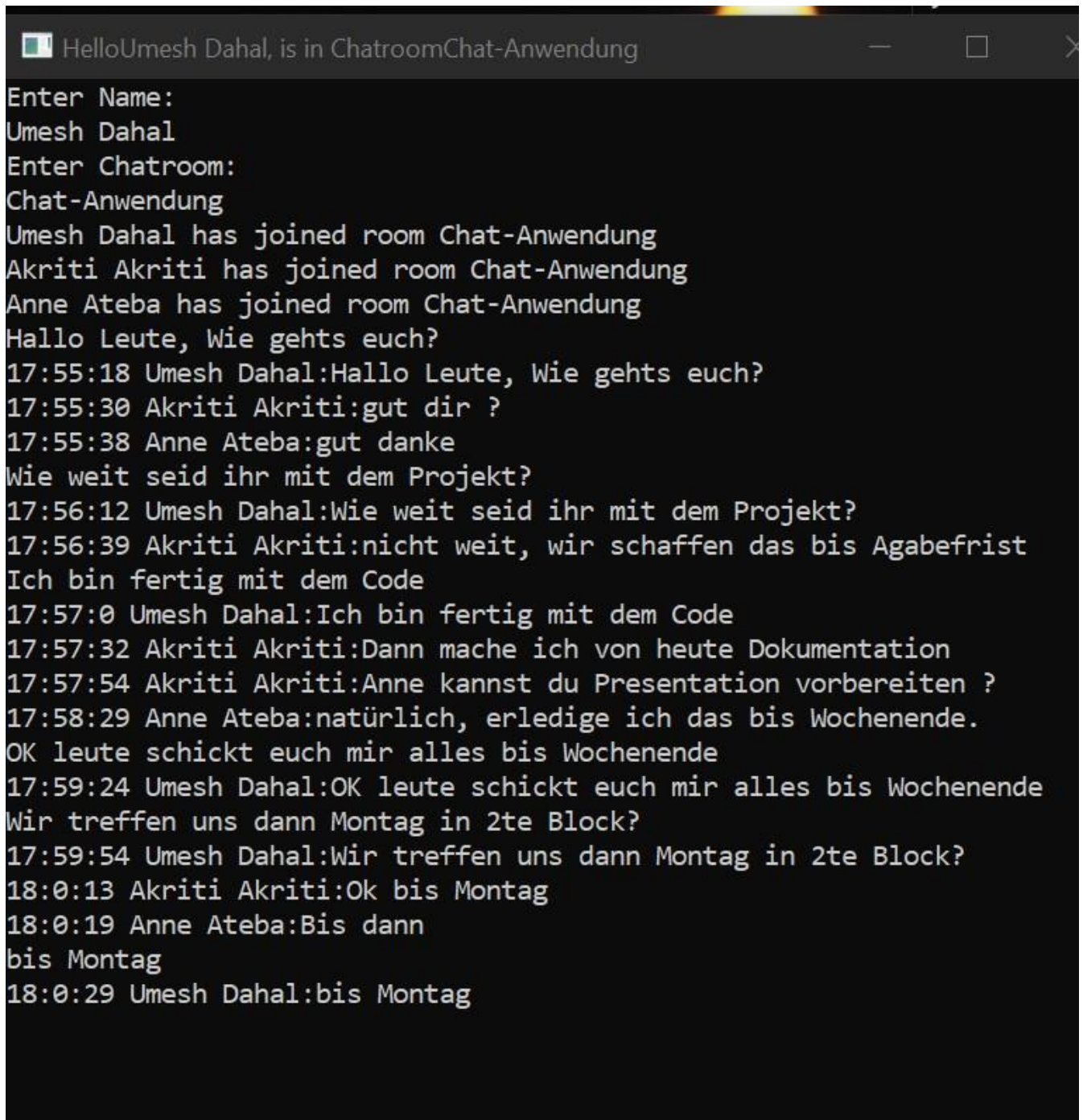
Fig 2: Broadcsting der Nachrichten
Zwischen 3 Nutzern



1.3. Echtzeit Chat:

Die Nachrichten werden echtzeitig in einem Raum geschickt. Das ist durch 'ConnectionMultiplexer' ermöglicht. Der 'ConnectionMultiplexer' zugreift in Redis Database und nutzt das Pub/Sub Funktion und verwaltet die Zugriffe.

```
[private static ConnectionMultiplexer _connectionMulplexer = ConnectionMulti-  
plexer.Connect(_connectionString);]
```



```
Enter Name:  
Umesh Dahal  
Enter Chatroom:  
Chat-Anwendung  
Umesh Dahal has joined room Chat-Anwendung  
Akriti Akriti has joined room Chat-Anwendung  
Anne Ateba has joined room Chat-Anwendung  
Hallo Leute, Wie gehts euch?  
17:55:18 Umesh Dahal:Hallo Leute, Wie gehts euch?  
17:55:30 Akriti Akriti:gut dir ?  
17:55:38 Anne Ateba:gut danke  
Wie weit seid ihr mit dem Projekt?  
17:56:12 Umesh Dahal:Wie weit seid ihr mit dem Projekt?  
17:56:39 Akriti Akriti:nicht weit, wir schaffen das bis Agabefrist  
Ich bin fertig mit dem Code  
17:57:0 Umesh Dahal:Ich bin fertig mit dem Code  
17:57:32 Akriti Akriti:Dann mache ich von heute Dokumentation  
17:57:54 Akriti Akriti:Anne kannst du Presentation vorbereiten ?  
17:58:29 Anne Ateba:natürlich, erledige ich das bis Wochenende.  
OK leute schickt euch mir alles bis Wochenende  
17:59:24 Umesh Dahal:OK leute schickt euch mir alles bis Wochenende  
Wir treffen uns dann Montag in 2te Block?  
17:59:54 Umesh Dahal:Wir treffen uns dann Montag in 2te Block?  
18:0:13 Akriti Akriti:Ok bis Montag  
18:0:19 Anne Ateba:Bis dann  
bis Montag  
18:0:29 Umesh Dahal:bis Montag
```

Fig 3: Chat in einem Raum