THREAD ARCS: An Email Thread Visualization

Eine Technik zur Visualisierung der Email Threads

Wladimir Emdin
Seminar „Visualisierung verteilter Systeme“

Gliederung

1. Einführung: Email Threads und Ziele deren Visualisierung
2. Anforderungen an eine Thread – Visualisierung
3. Visualisierung mit Thread Arcs
4. Thread Arcs vs. andere Visualisierungsarten
5. Einsatz in einem eMail-Client
6. Zusammenfassung und Fazit

1.1 Email Thread – Definition & Begriffe

<u>Email Thread:</u> Sammlung einzelner Emails, die durch die „Reply-To“-Funktion des Email Clients kausal abhängig sind.

- Erste Nachricht ist die **Wurzel (root)**
- Nachricht, auf welche geantwortet wird, heißt **Elternnachricht (parent)**.
- Die Antwortnachrichten sind **Kinder (children)** dieser Elternnachricht.
- **Generationstiefe** einer Nachricht gibt die Anzahl der „Reply-To“-Schritte an, die die Nachricht von der Wurzel entfernt ist.

1.2 Ziele der Visualisierung

- Darstellung soll die Möglichkeit geben, schnell die relevanten Gesprächsteile (Email-Nachrichten) zu finden.

- gleichzeitige Darstellung der Chronologie und der „Reply-To“- Relationen einzelner Emails in Threads

- Darstellung soll gut lesbar und kompakt sein

1.3 Der Entwickler

Bernard Kerr, IBM Research Division

Vorstellung auf IEEE INFOVIS 2003

2. Anforderungen an eine Thread Visualisierung

1. Chronologie: Nachrichtenanzeige in der Reihenfolge ihres Empfangzeitpunktes bzw. Empfangzeitpunkt später nachvollziehbar

2. Nachrichtenrelation: Anzeige aller Beziehungen zwischen den Nachrichten eines Threads

3. Stabilität: Jede Nachricht (Knoten) hat einen festen Platz in der visuellen Darstellung

4. Kompaktheit: So wenig Platz wie möglich einnehmen

5. Attribute Highlighting: Möglichkeit zum Hervorheben der Nachrichten mit ausgewählten Attributen

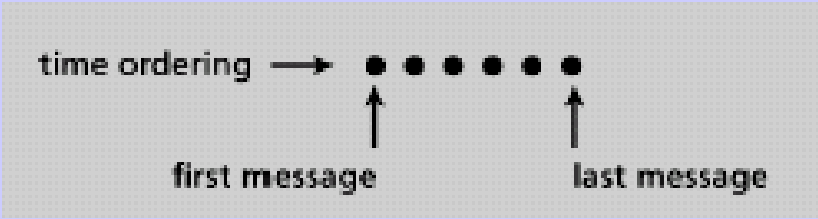
6. Scalability: Die Darstellung soll für kleine wie für relativ große Threads gut funktionieren

7. Mustererkennung: Bestimmte Thread – Muster (z.B. Dialog) sollen gut erkennbar sein

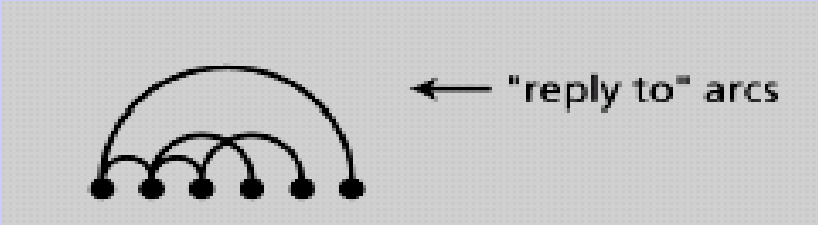
3. Visualisierung mit Thread Arcs

3.1 Erfüllung der Kriterien aus (2):

Chronologie



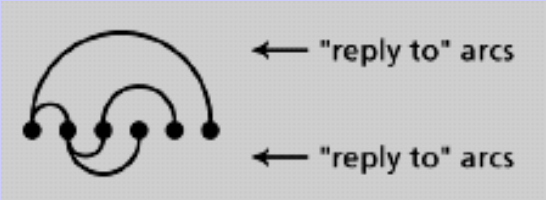
Relationen



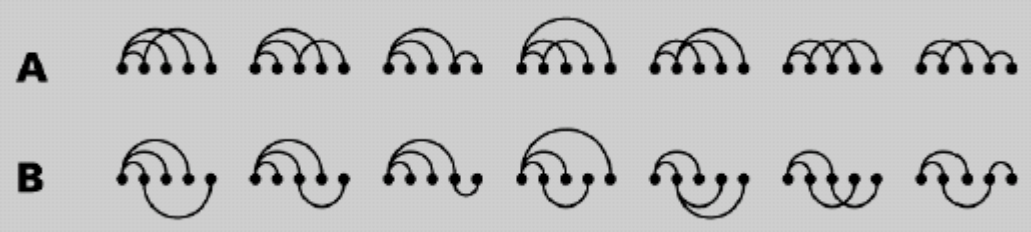
Stabilität ist durch die Wahl dieser Darstellung gewährleistet

Erste Verbesserung:

Die Verbindungen zwischen den Knoten (Arcs) werden über und unter den Knoten gezeichnet

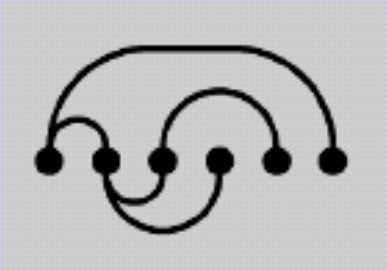


→Die Beziehungen sind besser lesbar



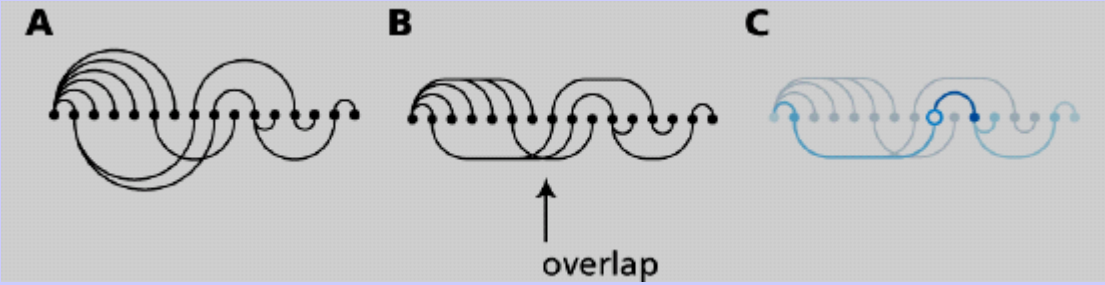
<u>Zweite Verbesserung:</u> die Höhe der Arcs wird eingedämmt, um die Darstellung auch für große Threads kompakt zu machen.

Nachteil: Arcs können sich vermischen und so die Lesbarkeit mindern



Lösung: Highlighting

Beispiel der Visualisierung eines Threads mit 16 Emails:

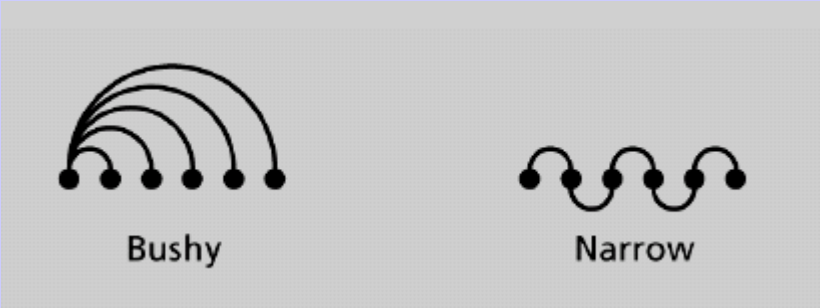


3.2 Thread-Arten (Muster der Konversation)

Man unterscheidet 2 Threadarten:

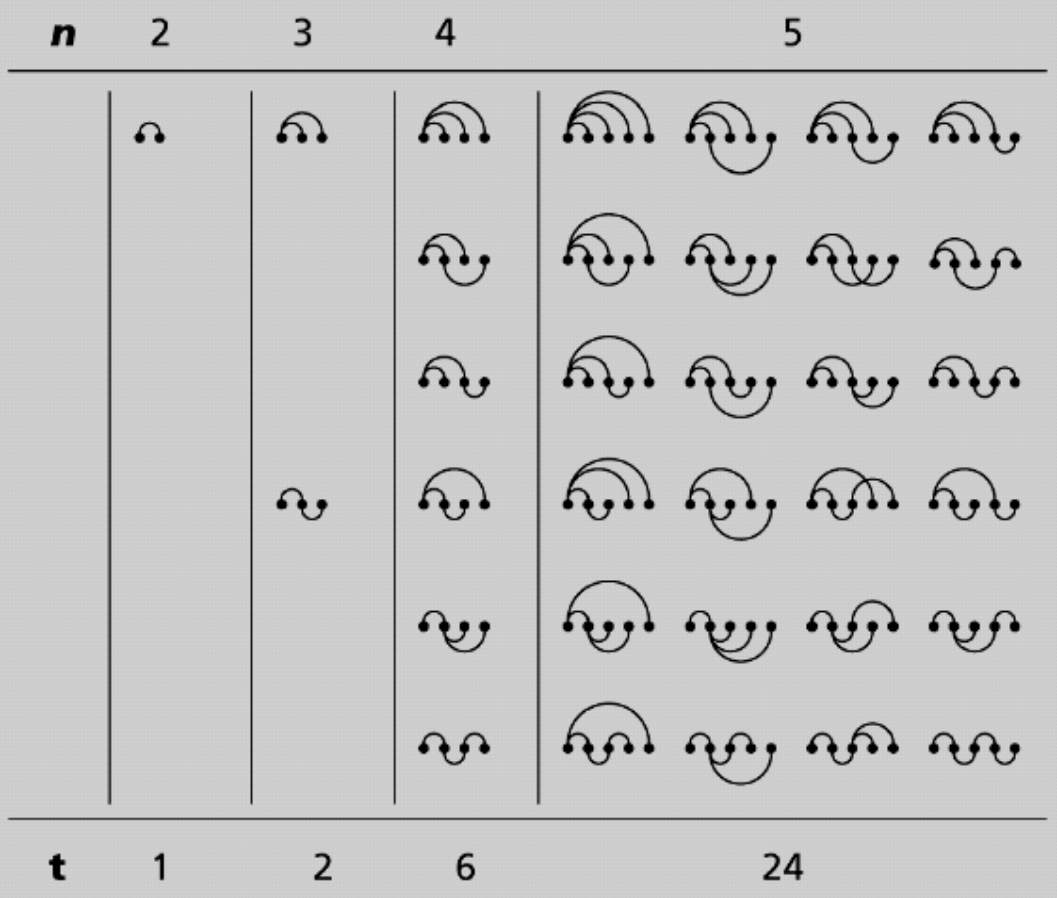
Bushy (verzweigt) – Thread enthält Emails, die 2 oder mehr Antworten besitzen

Narrow (persönlich) – Thread enthält Emails, die nur eine Antwort pro Mail besitzen



3.3 Kombinatorische Zahl möglicher Thread Arcs

Für n Emails gibt es (n-1)! mögliche Arcs - Kombinationen



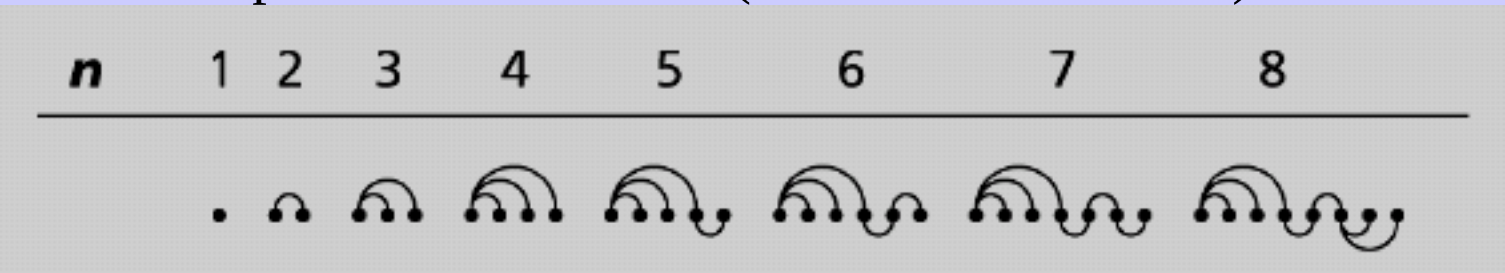
Thread Arcs kann gut mit Bushy & Narrow Threads bis 20 Emails arbeiten

3.4 Pseudo Code für Thread Arcs

```
Sortiere alle Nachrichten chronologisch
Bestimme die Generationstiefe jeder Nachricht

FOR each message
{
    IF ( ROOT_message )
  Setze die Nachricht auf die Startposition (links)
    ELSE
    {
      Setze die Nachricht rechts vom letzten Knoten
      IF ( Generationstiefe ungerade )
            Zeichne die Verbindung zum Elternknoten oben
      ELSE
            Zeichne die Verbindung zum Elternknoten unten
     }
}
```

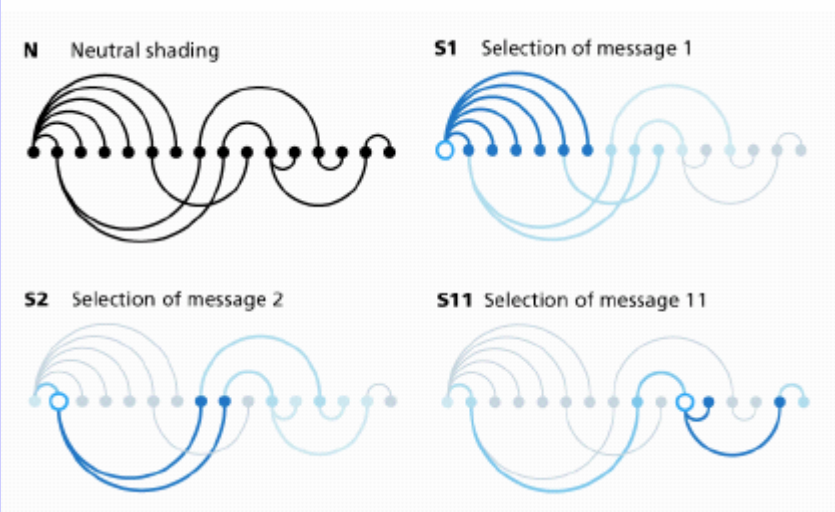
Beispiel für den Ablauf: (Thread mit 8 Emails)



3.5 Interaktion

verschiedene interaktive Methoden vorhanden :

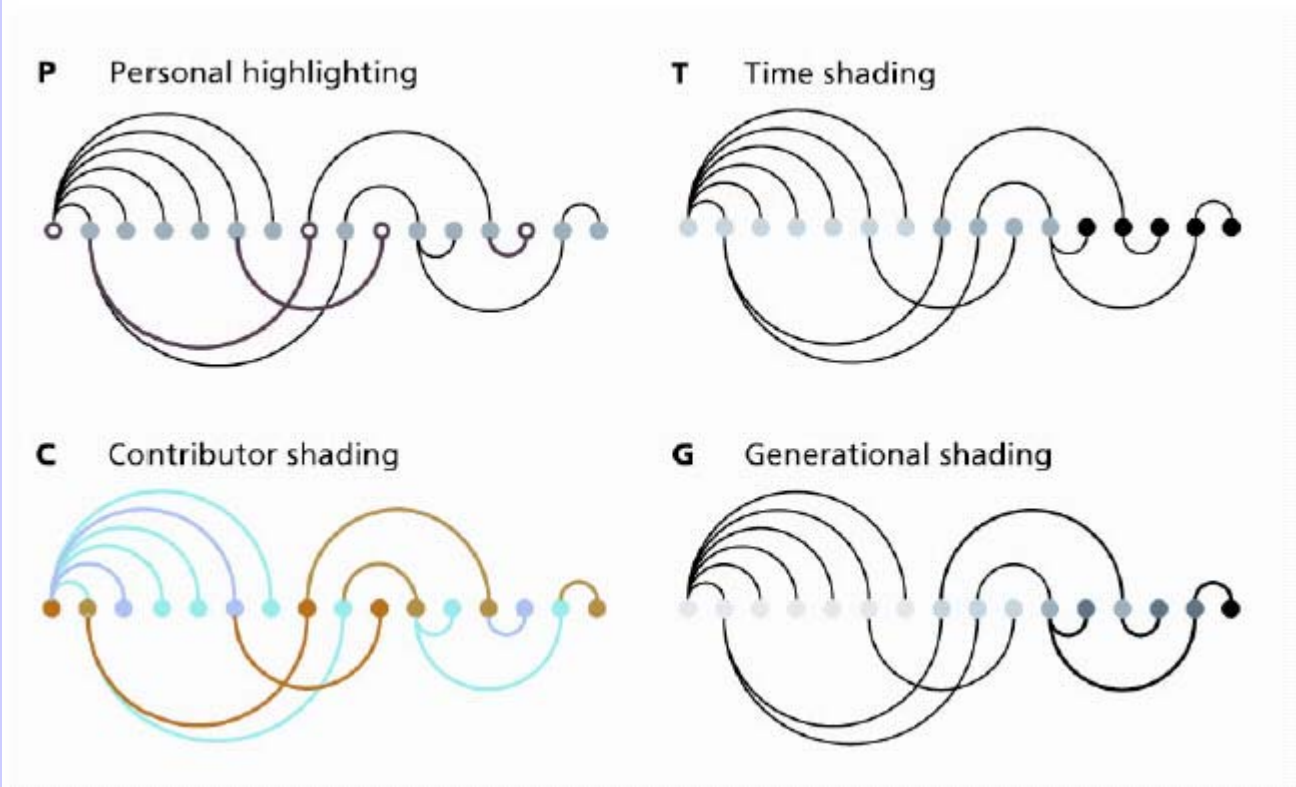
- Selection Highlighting



-People Highlighting

- Attribute Shading

Beispiel für verschiedene Highlighting Schemas eines Threads

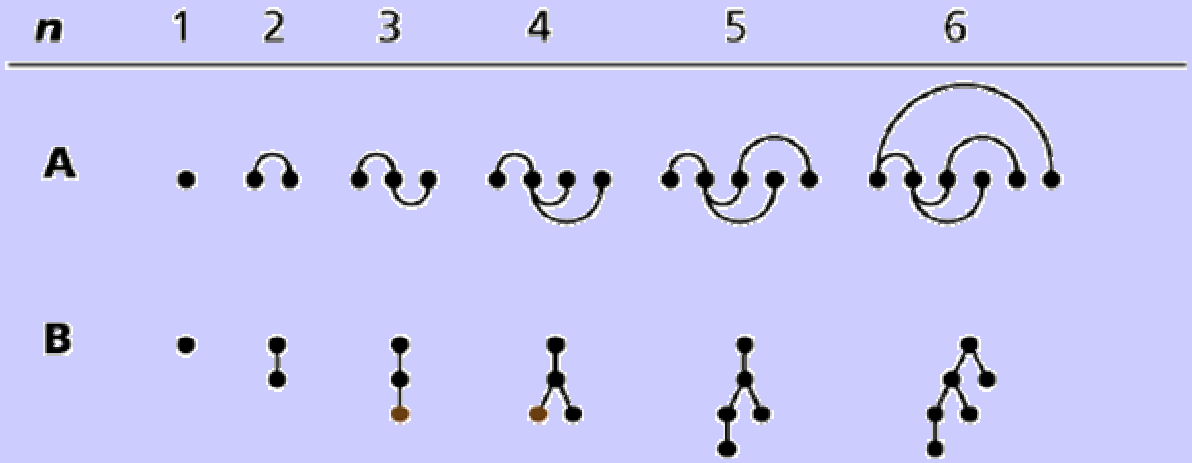


4. Thread Arcs vs. andere Visualisierungsarten

Vergleich mit 3 anderen Visualisierungstechniken:

- Baumdiagramme (Tree Diagrams)
- Tree Tables
- Compact Chronological Tree Tables (CCTT)

4.1 Tree Diagrams

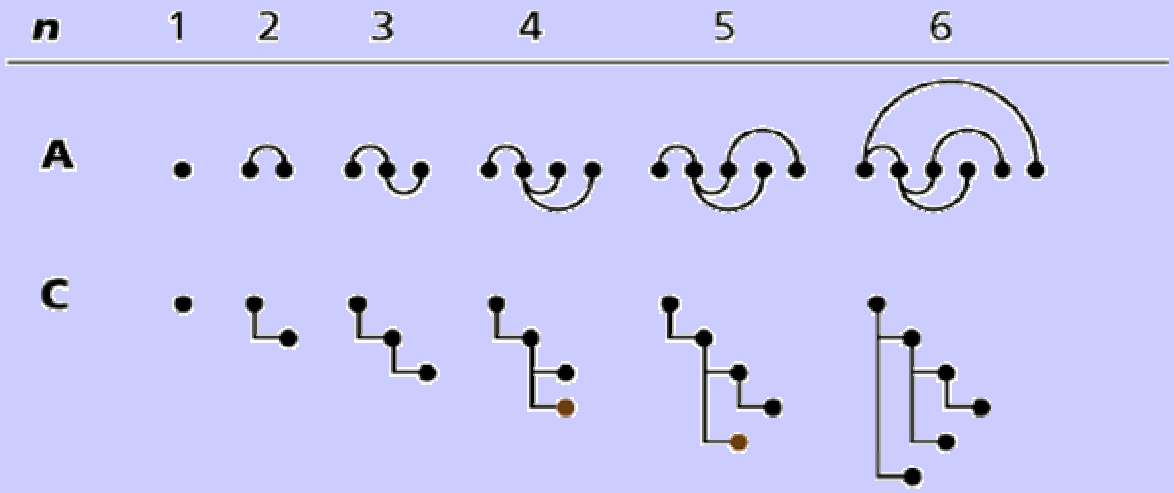


<u>Vorteile:</u> verständlichere Darstellung der Beziehungen zwischen den Nachrichten.

- jede Reihe repräsentiert eine Generationstiefe
- jeder Baumast repräsentiert einen „Unterthread“

<u>Nachteile:</u> - keine Chronologie, keine Stabilität

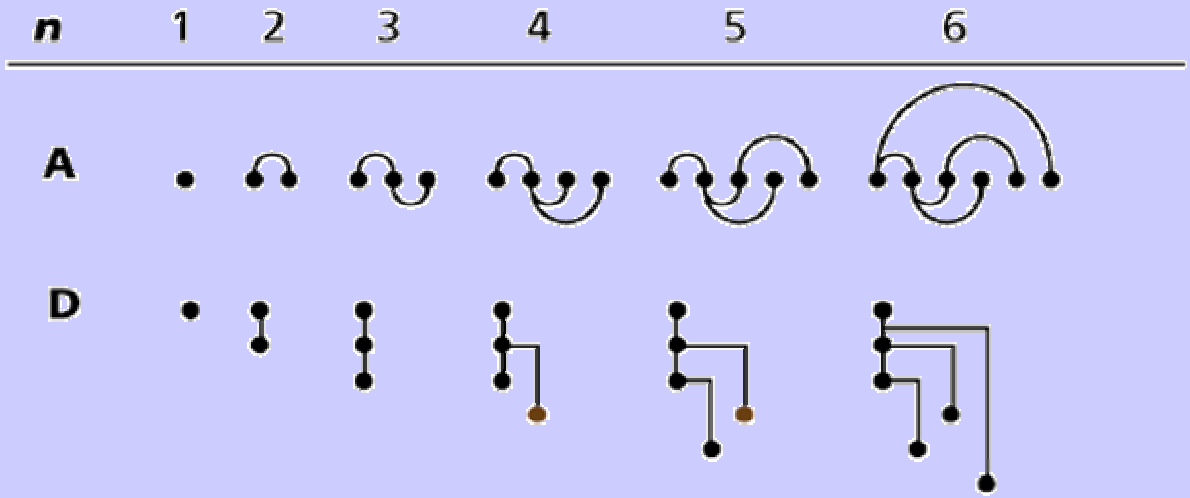
4.2 Tree Tables



Gleiche Vor- und Nachteile wie bei Tree Diagrams

Baum „wächst“ nicht nach unten, sondern nach rechts

4.3 Compact Chronological Tree Table



Vorteile: - Chronologie

Nachteile: - Struktur der Beziehungen schlecht sichtbar
- keine Stabilität

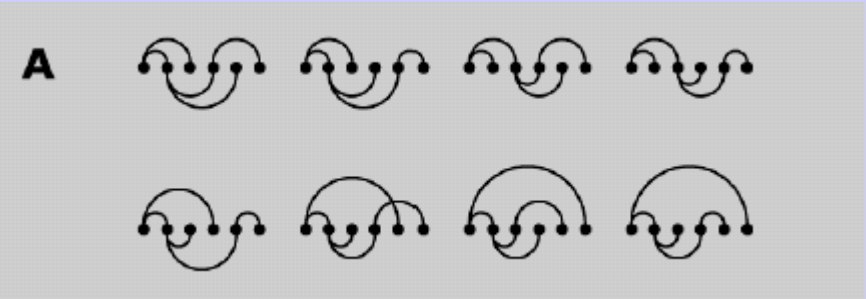
4.4 Thread Arcs als Spezifizierung von Tree Diagrams und Tree Tables

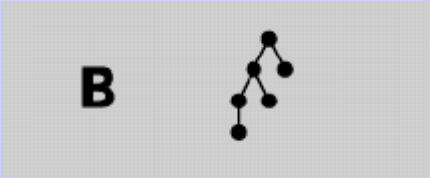
- Tree Diagrams und Tree Tables stellen keine Chronologie dar
→ sie sind allgemeiner als Thread Arcs

→Ein Tree Diagram oder Table hat mehrere äquivalente Darstellungen mit Thread Arcs

Im folgenden Beispiel gibt es für eine Darstellung mit Tree Diagrams (B) insgesamt 15 äquivalente Thread Arcs – Darstellungen (A).

[Nur 8 von 15 zu sehen]





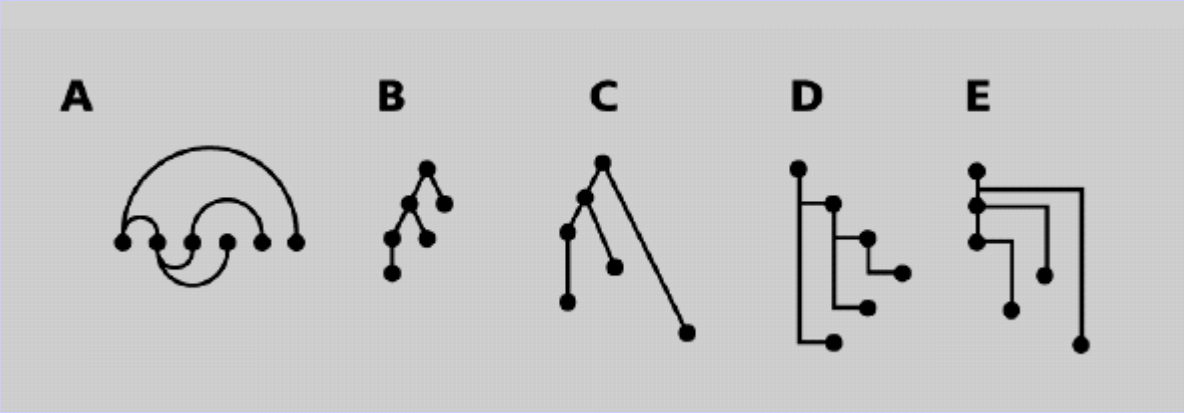
→ Thread Arcs können eine Konversation präziser darstellen

4.5 Auswertung der 4 Visualisierungstechniken

	Thread Arcs	Tree Diagrams	Tree Tables	CCTT
Chronologie	+	-	-	+
Relation	+	+	+	+
Stabilität	+	-	-	-
Kompaktheit / Scalability	+	-	-	-
Highlighting	+	+	+	+
Mustererkennung	+	+	+	-

+ ja/vorhanden; - nein/nicht vorhanden;

Anmerkung: Modifikation von Tree Diagram und Tree Table für die Darstellung der Chronologie



A – Thread Arcs, B – Tree Diagram, C – Tree Diagram (modifiziert)
D – Tree Table, E – Tree Table (modifiziert) = CCTT

→ Gewinnung der Chronologiedarstellung auf Kosten der Lesbarkeit und Kompaktheit

5. Einsatz im Mail-Client

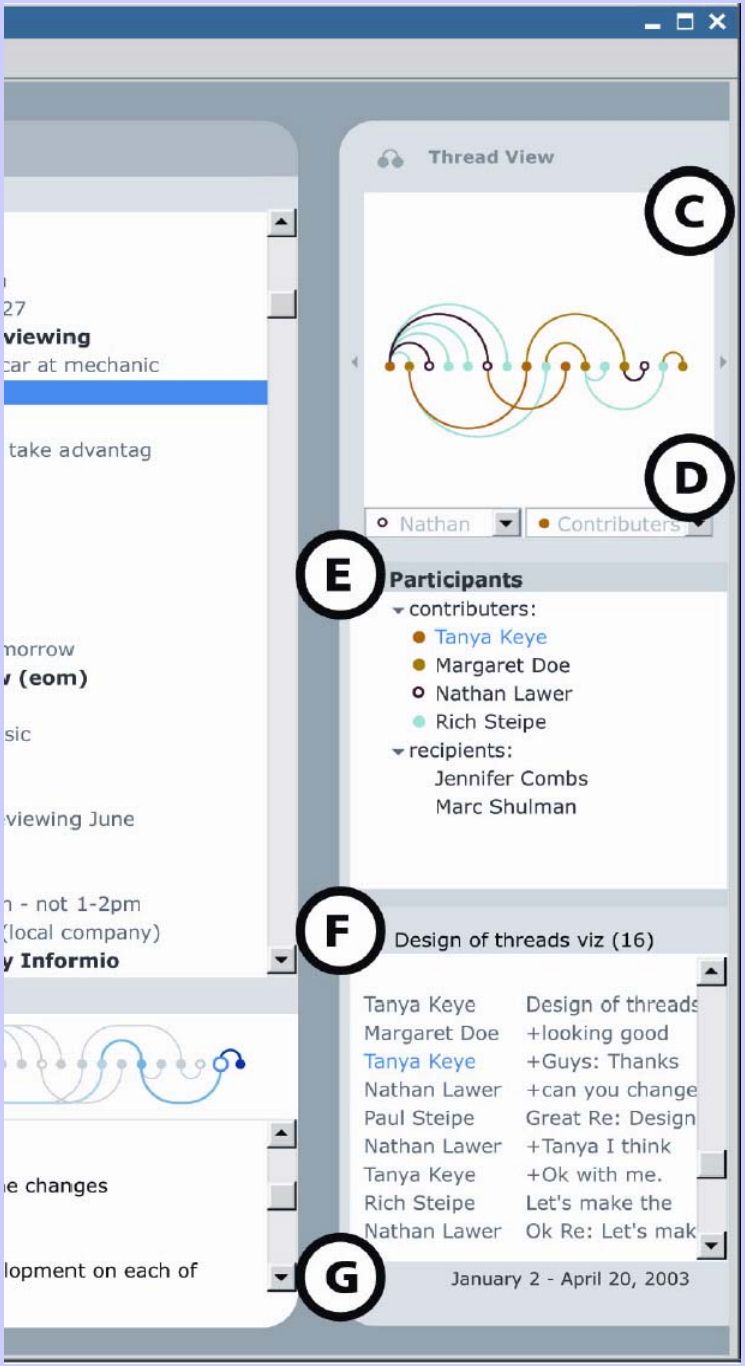
5.1 Bildung eines Threads

- 1. Durchgang durch alle Mails: finde zusammen gehörende Nachrichten anhand der ID-Referenzen im Email-Header

- 2. Durchgang durch alle im ersten Durchgang gefundenen Mails: fasse die Nachrichten anhand ihres Betreffs zu Threads zusammen

5.2 Eine Studie mit ReMail

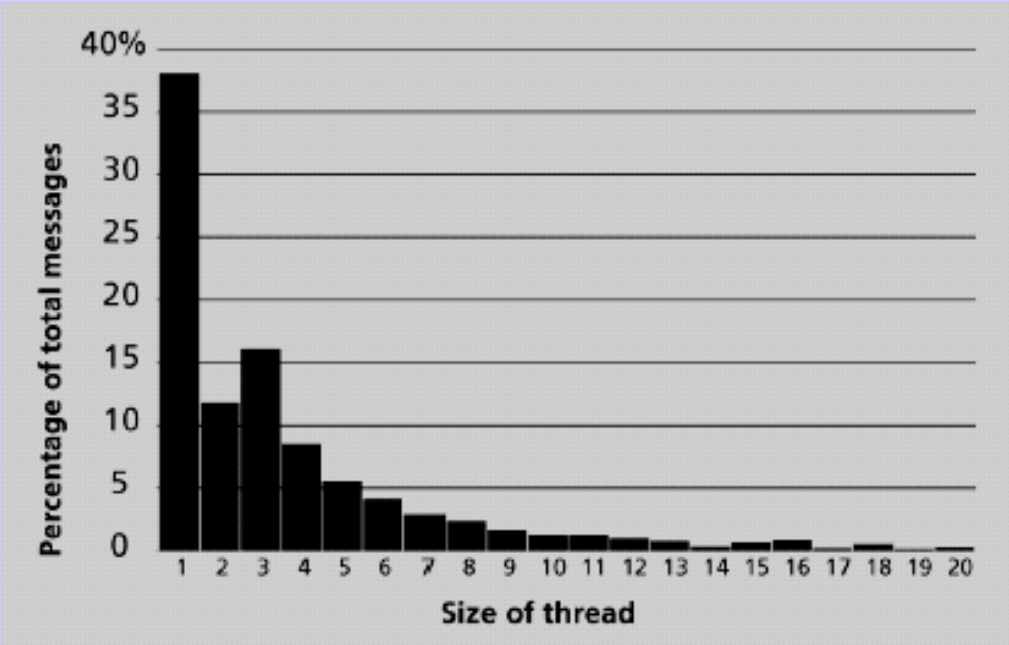
A – Preview Fenster; B – Thread View Fenster



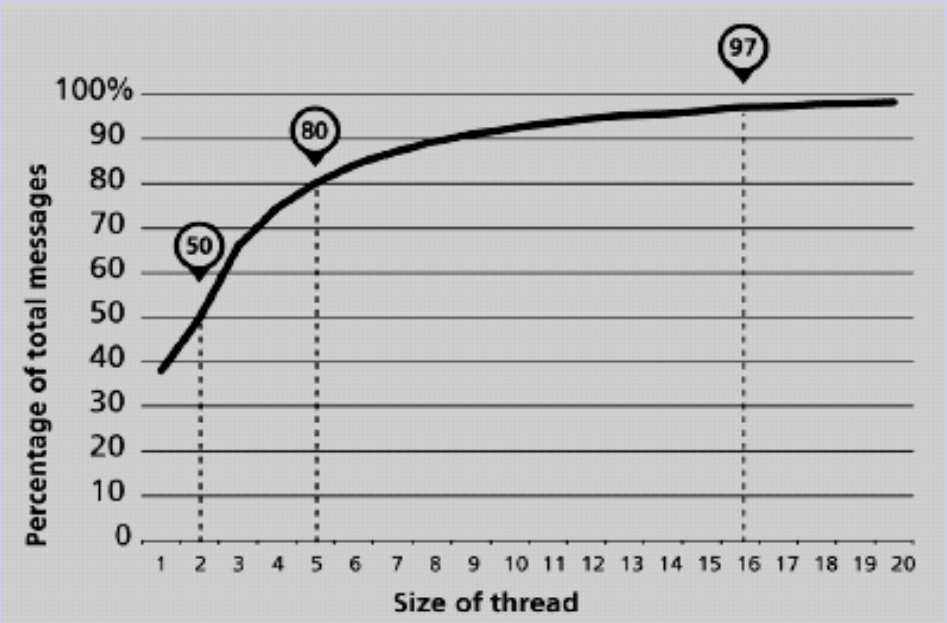
C – Thread View Fenster
D – Auswahl der Highlighting Schemas
E – Legende der Teilnehmer
F – Liste der Beiträge
G – Zeitspanne des Threads

5.3 Ergebnisse der Studie

Insgesamt wurden von 8 Personen ca. 42.000 Emails geprüft

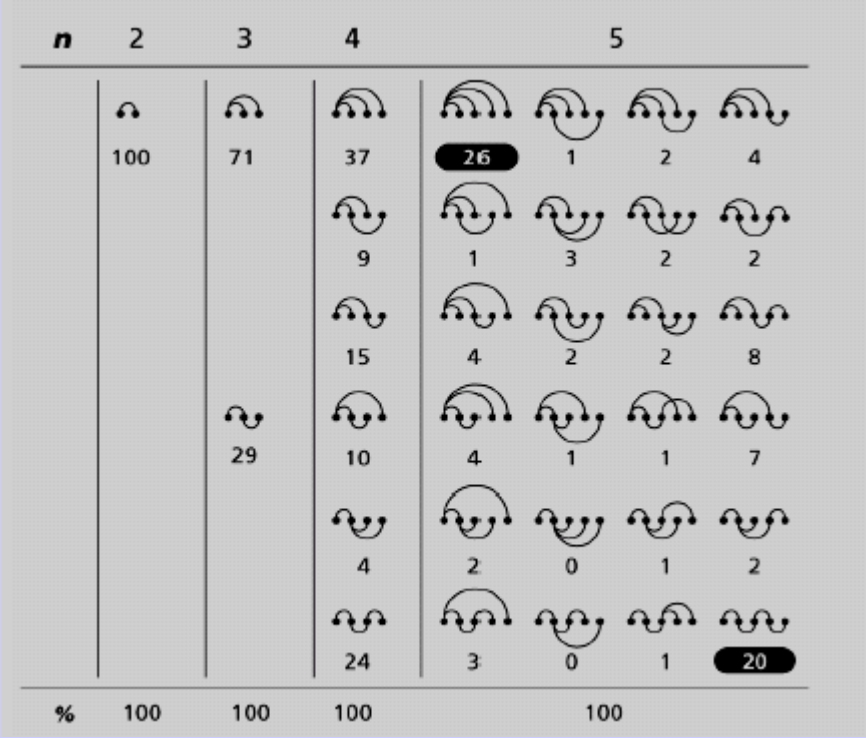


38% Einzelnachrichten, 11% Threads der Länge 2,
16% Threads der Länge 3 etc.



50% aller Emails in Threads der Länge <=2 enthalten,
80% - in Threads der Länge <= 5

Prozentuelle Verteilung einzelner Thread Strukturen in den Längen von 2 bis 5



Fazit: - Vorrang ausgewählter Gesprächstrukturen, nämlich Narrow und Bushy

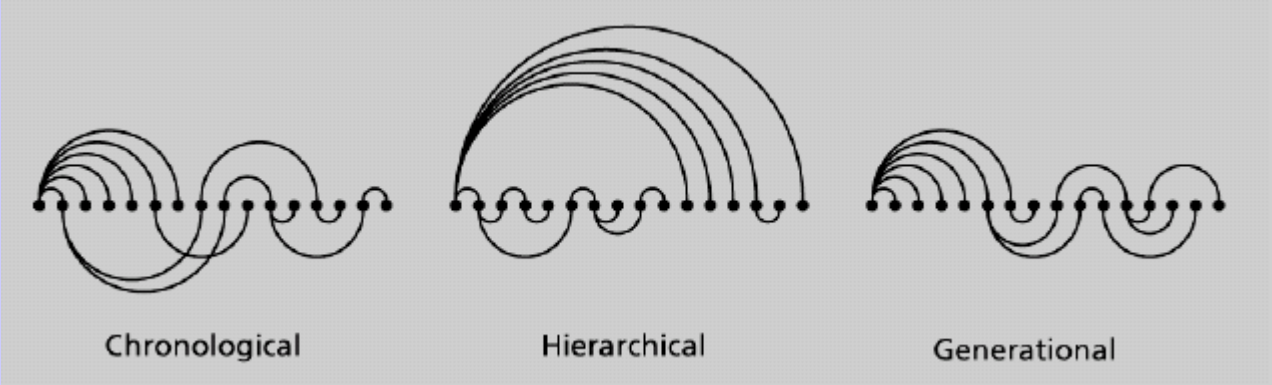
- Antworttendenz auf erste Nachricht (40% aller Mails in n=5) und auf letzte Nachricht (43% aller Mails in n=5)

Rating-Vergleich von Arcs, Diagram & Table

Key Qualities	Arc	Diagram	Table
Relationships	1.0	2.0	**2.5**
Chronology	**3.0**	1.5	1.5
Sense/Scanability	1.3	**2.0**	**2.0**
Scalability	1.0	**2.5**	2.0
Stability	**3.0**	1.5	1.3
Compactness	**3.0**	2.0	1.3

Figure 27. Median of ratings for each visualization (1-3 scale, 3 is best).

→Ansatz für potenzielle Verbesserung in Kritikpunkten der Arcs: Umschaltung zwischen verschiedenen Sortierarten der Nachrichten



6. Zusammenfassung und Fazit

-Visualisierung zeigt Chronologie der Nachrichten und Beziehungen zwischen ihnen. Akzent liegt auf der Visualisierung der Chronologie

→Thread Arcs sind gut „gerüstet“ für die Größen und Typen der Konversationen mittels Email (Studie)

<u>Fazit:</u> - Einsatz in Firmen möglich, weil teilweise viele interne Firmenangelegenheiten über Email abgewickelt werden, die sich dann über mehrere Emails erstrecken können

- Privateinsatz eher unnötig, weil Threads zu kurz oder gar nicht vorhanden sind

Quellen:

Bernard Kerr
IBM Research Report: Thread Arcs, An Email Thread Visualization, July 22, 2003

Bernard Kerr
IBM Research Report: Thread Arcs, An Email Thread Visualization, October 30, 2003