Bloom

(a Software Visualization System)

Bloom, Autoren

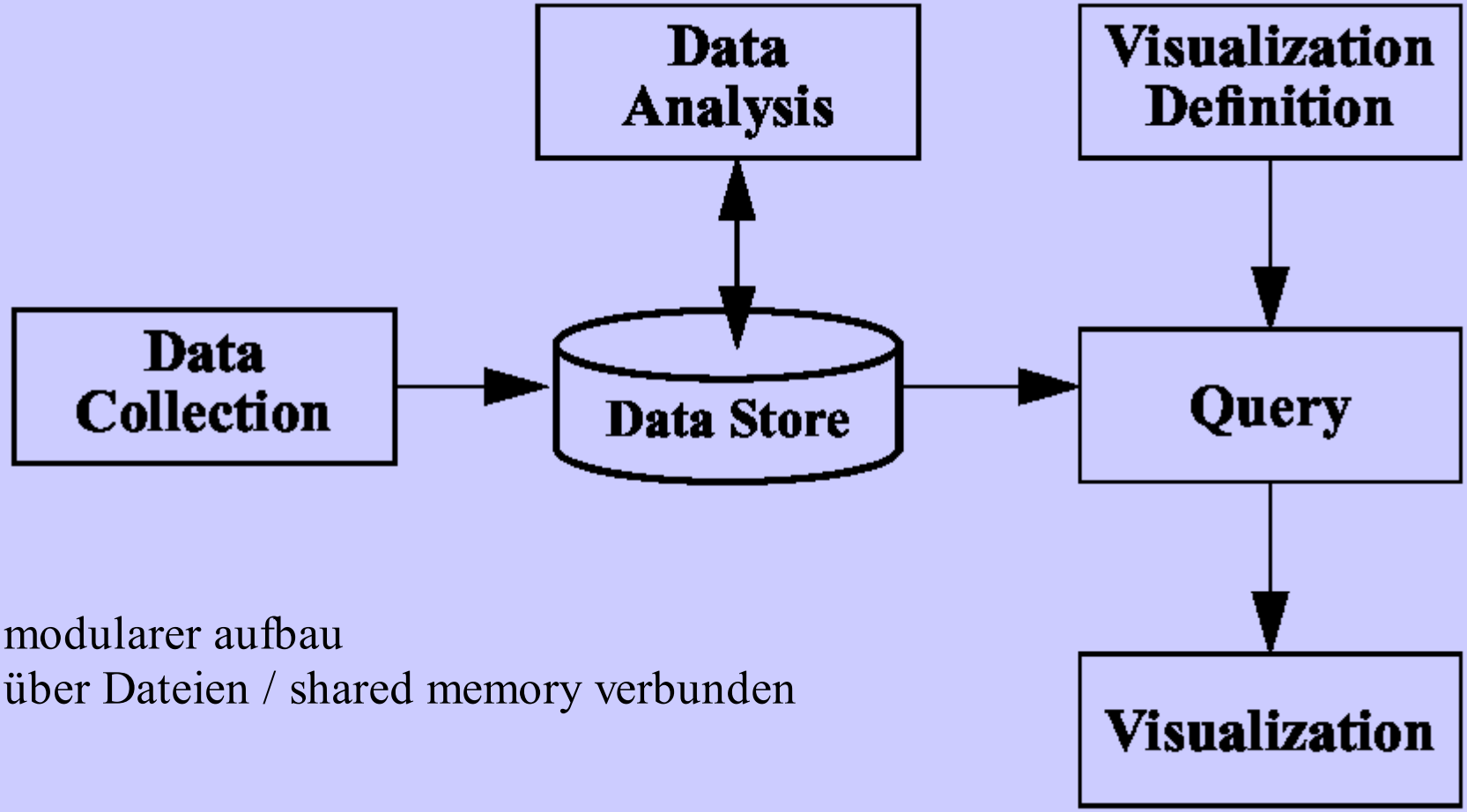
Steven P. Reiss
Department of Computer Science
Brown University, Providence

Manos Renieris

Ziele von Bloom

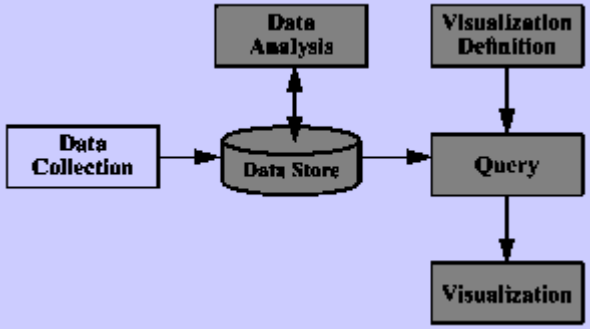
- möglichst viele Daten aufzeichnen
- einfaches Interface um weitere Datenquellen hinzuzufügen
- einheitliche und graphische Schnittstelle aller Datenquellen
- möglichst viele Darstellungsmöglichkeiten für die Daten
- einfaches Interface um weitere Darstellungsmöglichkeiten hinzuzufügen
- der Benutzer soll selbst festlegen welche Daten wie angezeigt werden (keine vorgegebene Visualisierung)

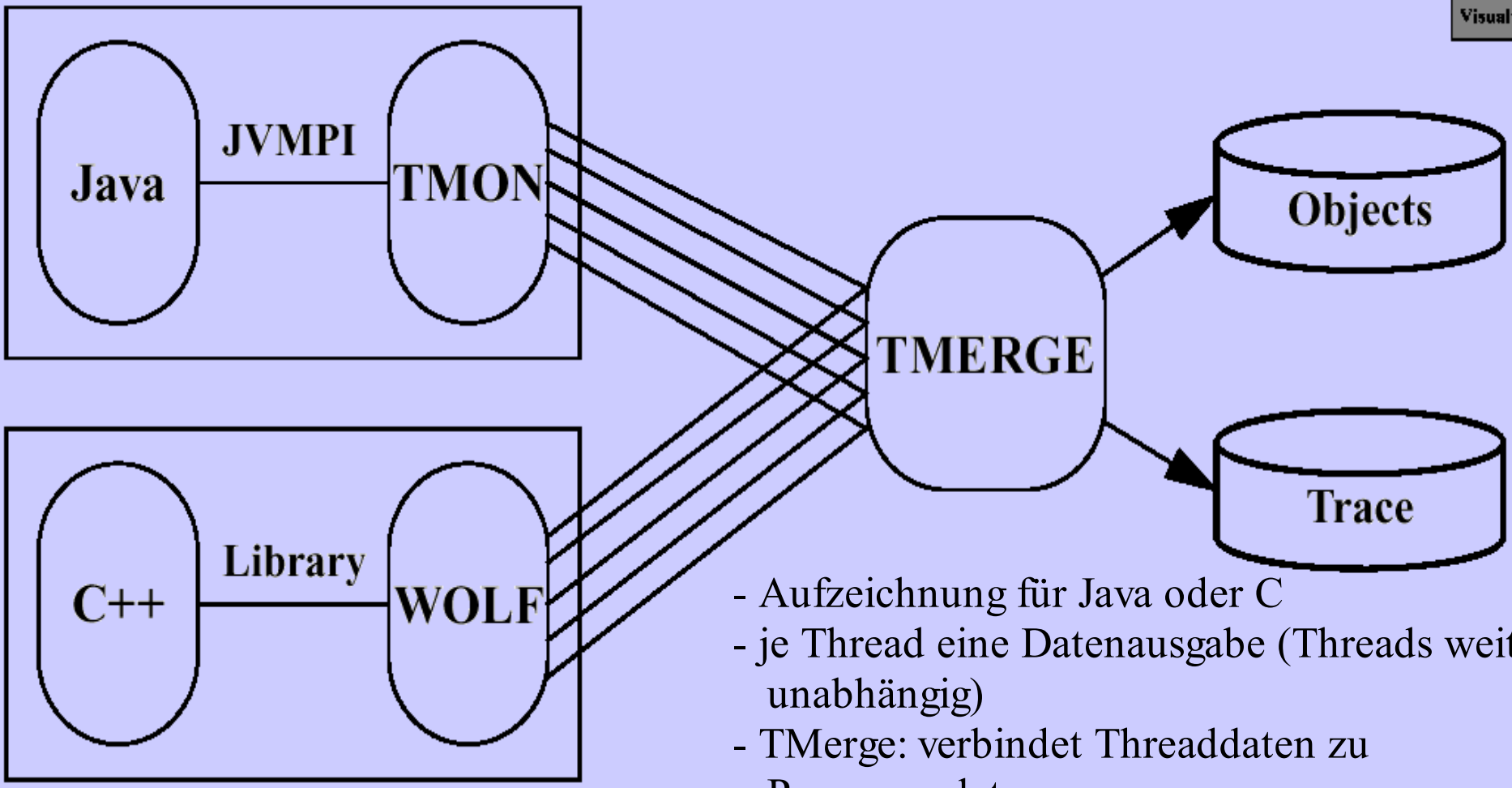
Bloom



- modularer aufbau
- über Dateien / shared memory verbunden

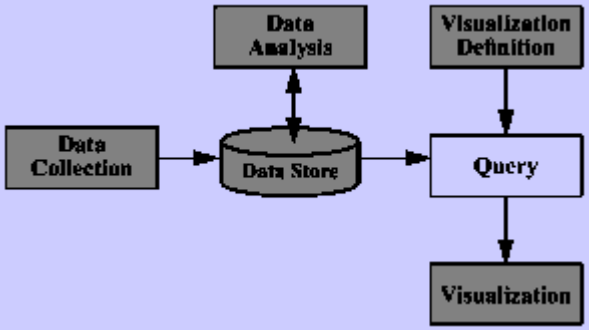
Bloom

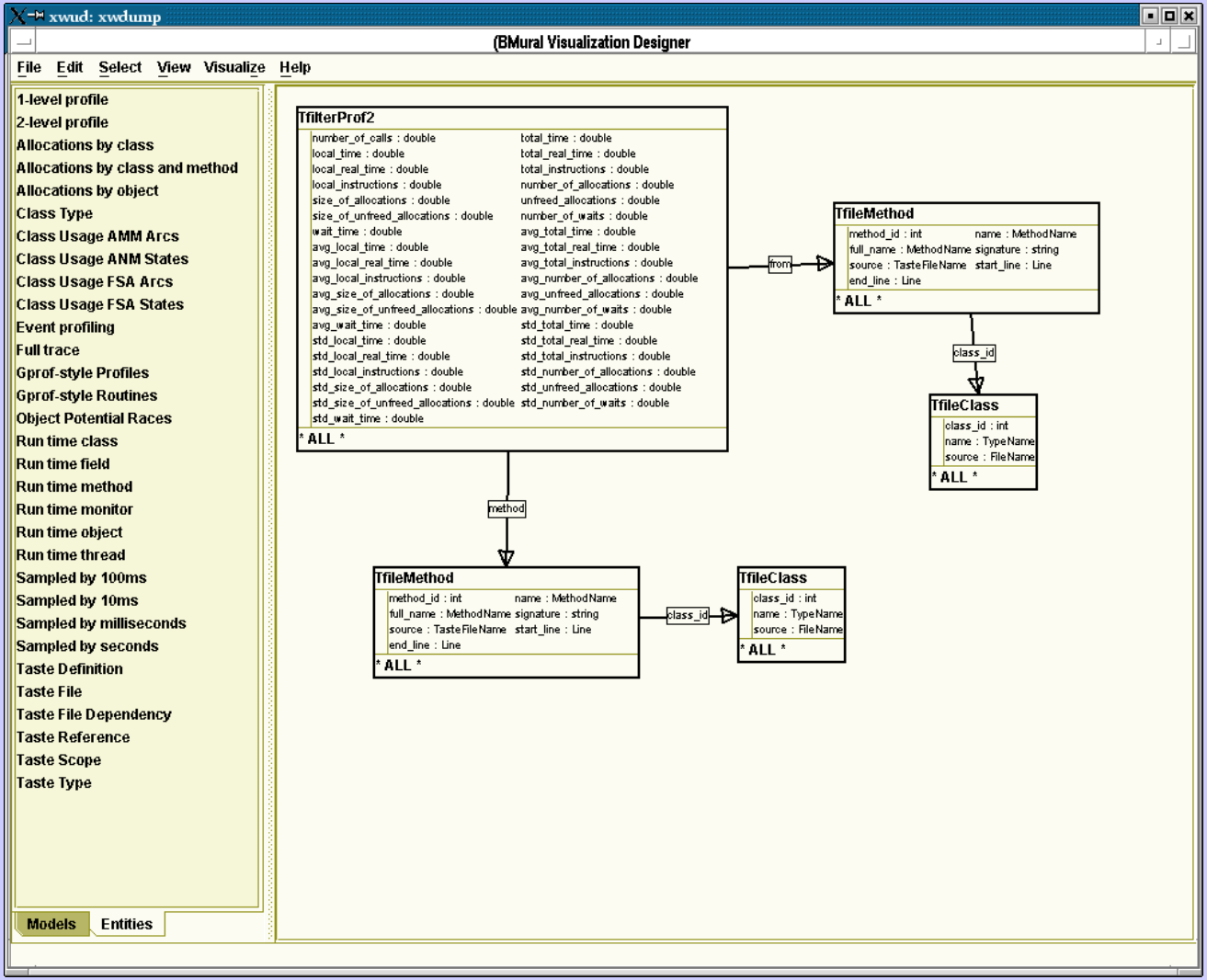




- Aufzeichnung für Java oder C
- je Thread eine Datenausgabe (Threads weiter unabhängig)
- TMerge: verbindet Threaddaten zu Programmdaten
- Ergebniss bildet eine (Art) relationale Datenbank

Bloom

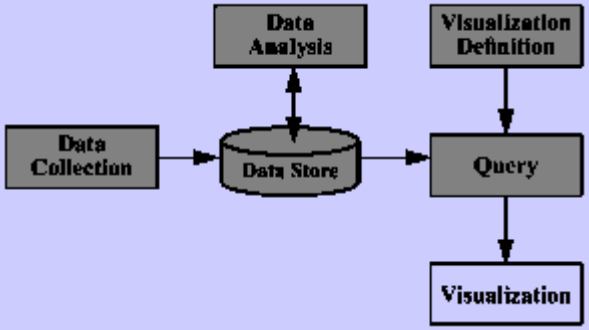




Mural

- Darstellungen werden als Anfragen an DB spezifiziert
- Benutzer kann beliebige (typgleiche) Felder in Beziehung setzen
- typische DB Operationen unterstützt:

Bloom



Framework für Visualisierung.

Basisbibliothek von Visualisierungen

erzeugt von Mural

Konfiguriert alles, basierend auf Desc.

HIVE

COMB

Description

BEE

GUI-Framework für Konf. Dlgs

APIS

USER

BUD filtert Daten, gesteuert von BEE (query-System)

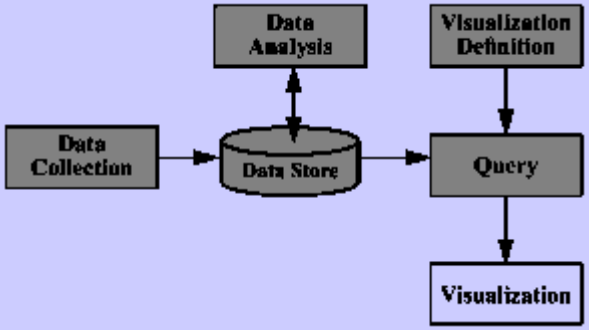
BUD

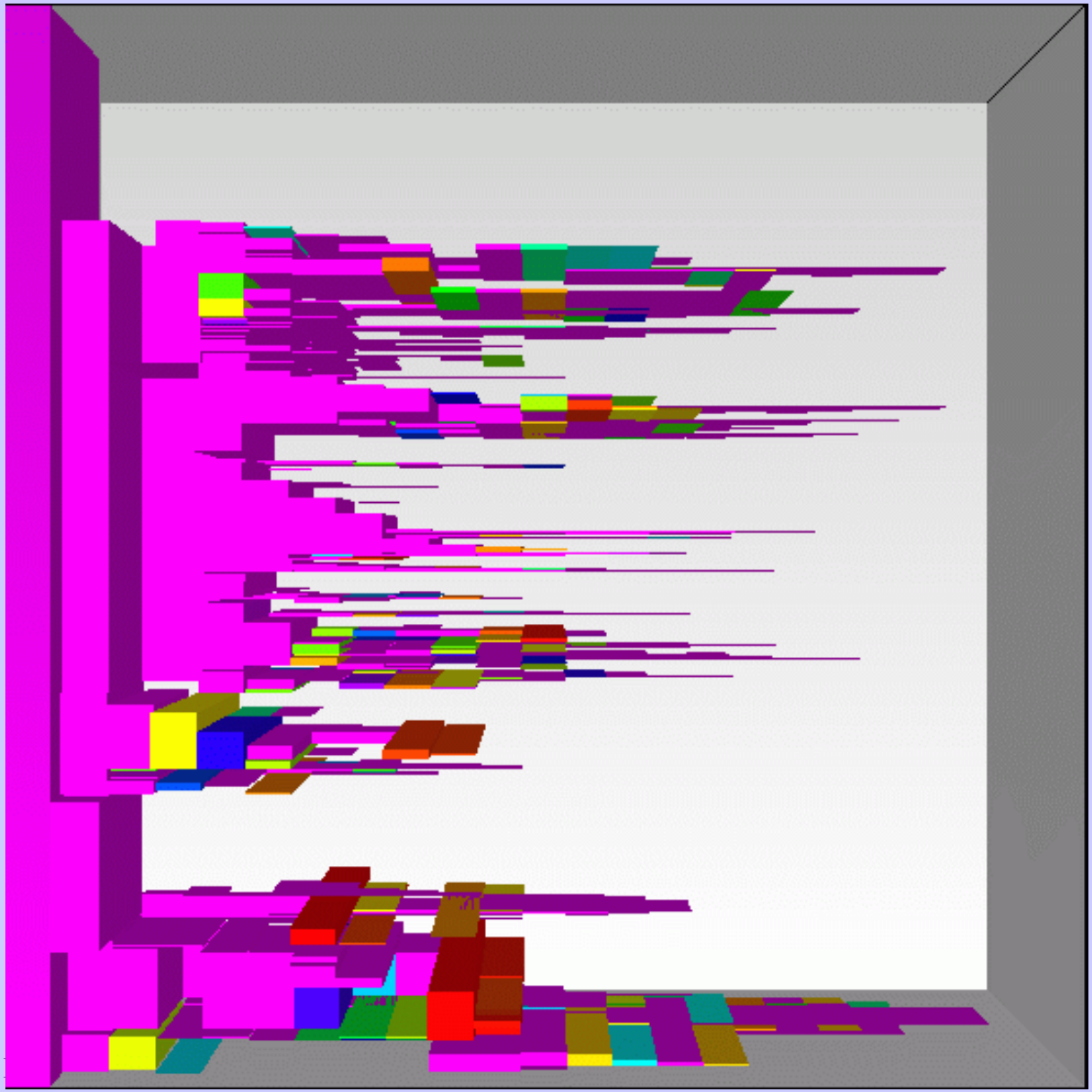
Trace Data

Analysis Data

Semantic Data

Bloom

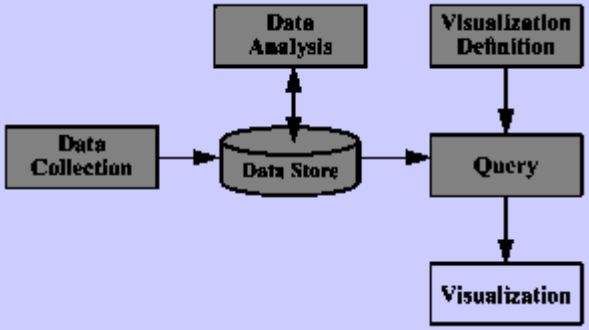


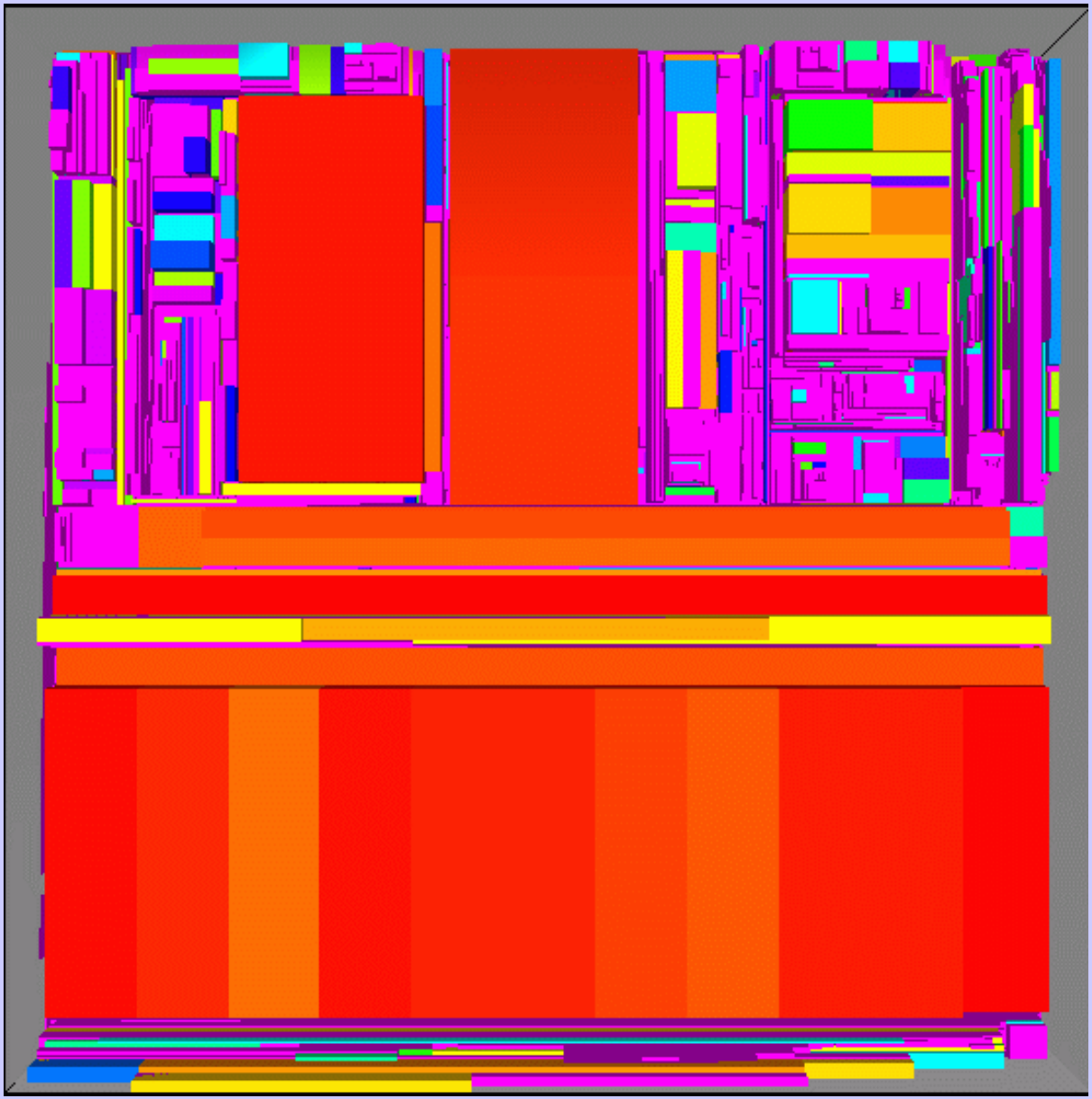


Beispiel
3D box tree
Aufrufgraph

- jede box ist eine Methode
- x-Achse: Aufruftiefe
- y-Achse: Dauer (incl. Unterauffrufe)
- z-Achse: Dauer (ohne Unterauffrufe)
- Farbe: Markierung wichtiger Methoden

Bloom

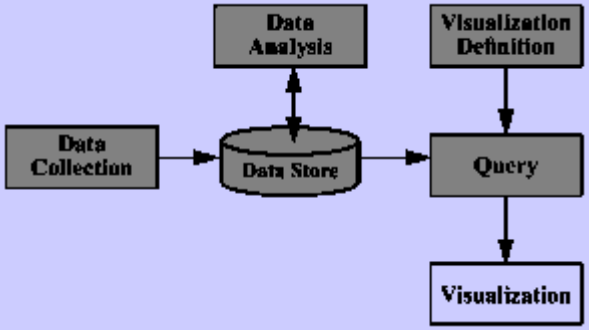


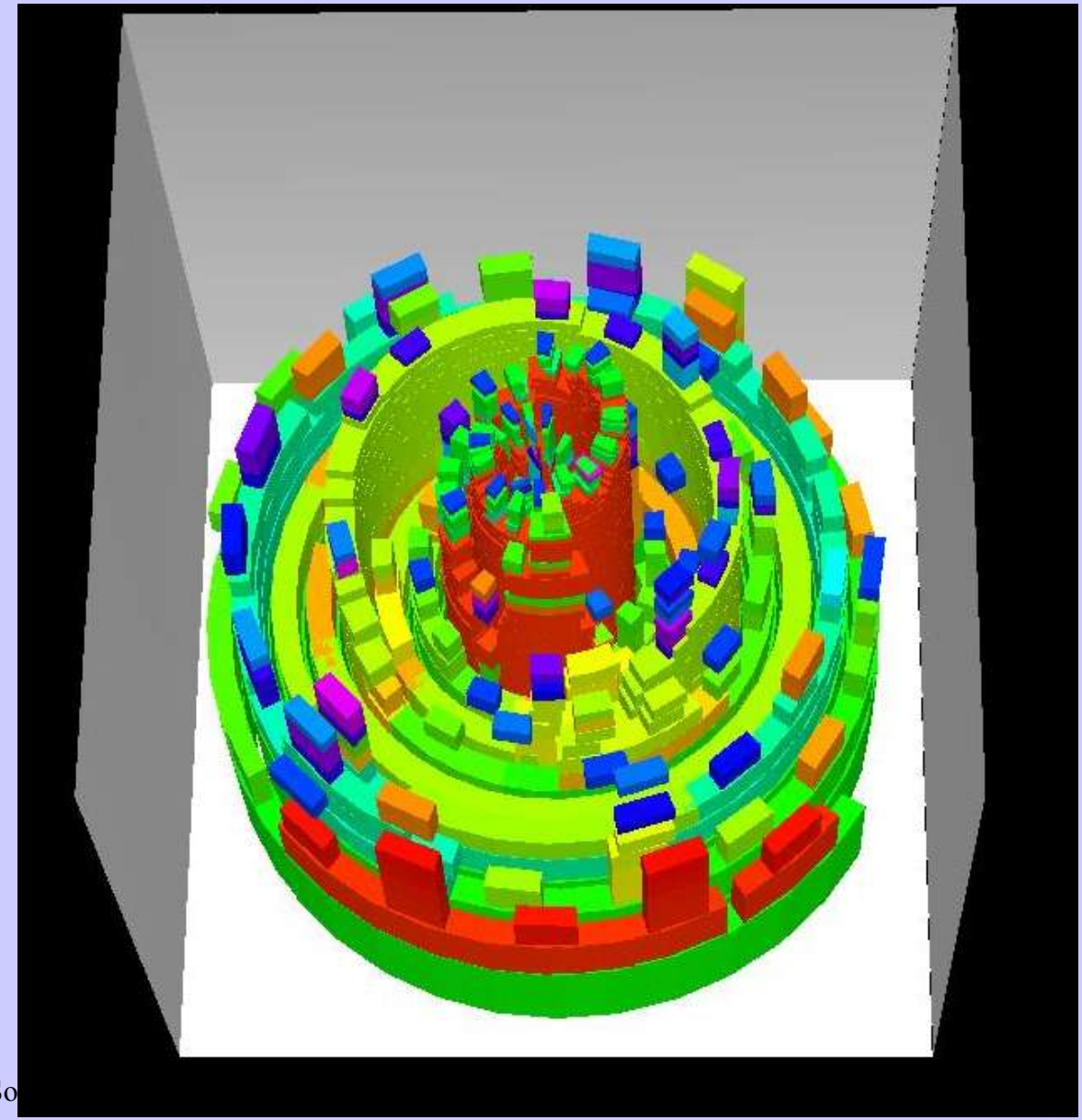


Beispiel
3D tree map
Aufrufgraph

- jede box ist eine Methode, Aufgerufene Methoden liegen oben auf dem Aufrufer
- dx*dy: Dauer (incl. Unterauffrufe)
- dz: Anzahl der Aufrufe
- Farbe: Markierung wichtiger Methoden

Bloom

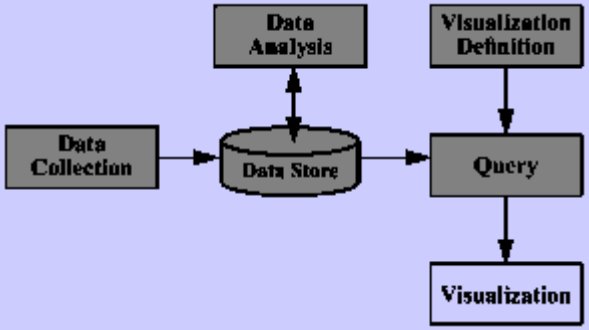


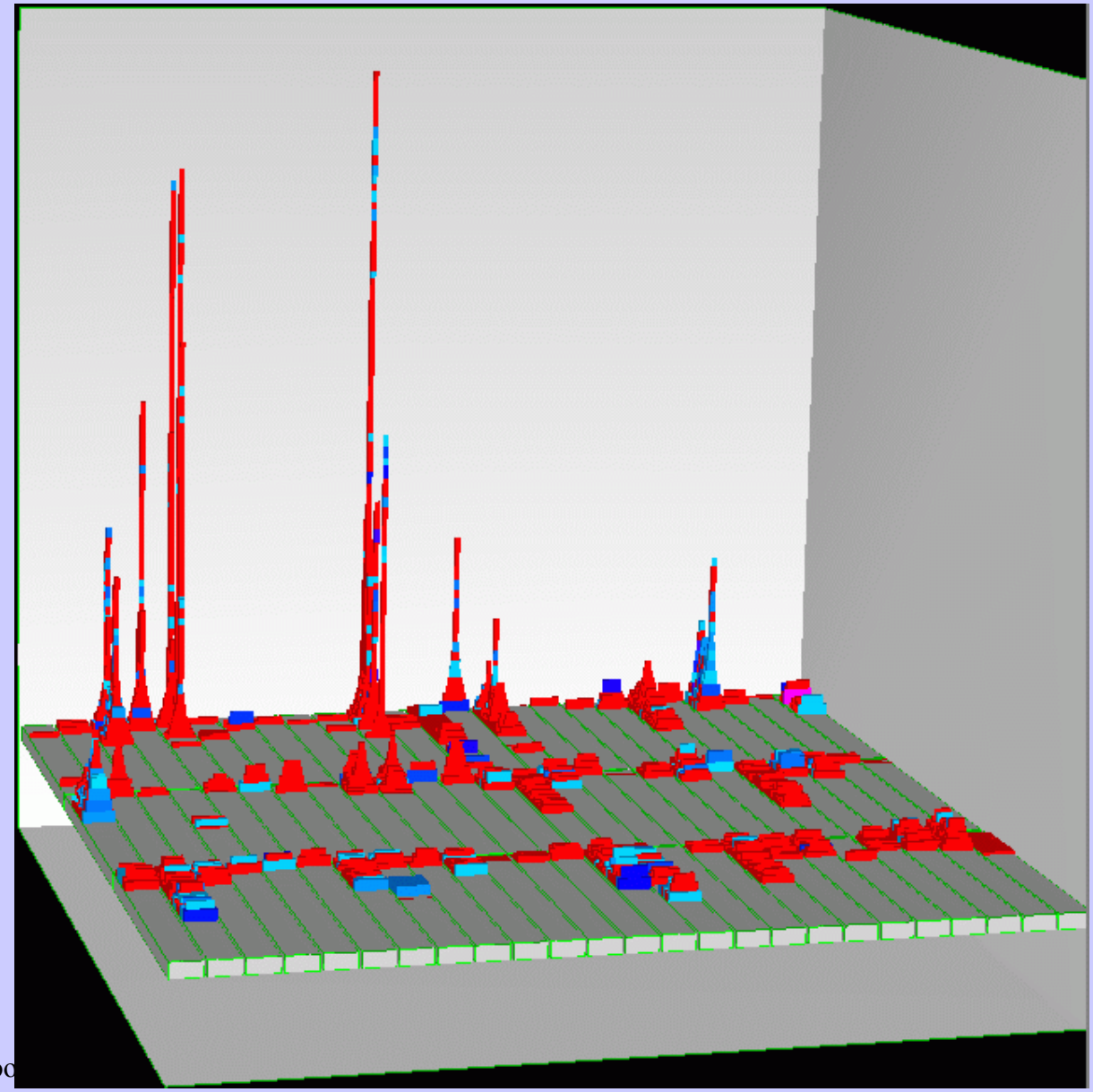


Beispiel
3D Spiral Visualisierung
Stack

- Farbe: Methoden (Gruppe)
- Höhe: Stackhöhe
- Breite (Winkel): Dauer

Bloom





Beispiel
Filemap
jedes Feld repräsentiert eine Gruppe von Zeilen in einer von mehreren Quelldateien

•Höhe: Ausführungszeit in diesem Block

Bloom
Pro&Contra

+ bietet starkes und flexiebles Framework für Darstellung von Trace-Daten
+ schon viele verschiedene Darstellungsformen implementiert
+ einfache GUI ermöglicht die Erstellung komplexer Relationen
+ bietet Analyse für mögliche Race Conditions und Deadlocks
- produziert sehr viele Rohdaten (Java: 1GB/10 sek, C: 1GB/sek)
- GUI noch nicht ganz ausgereift
- keine Dokumentation/Hilfe
- enthaltene Scripte für Übersetzung/Start funktionieren nicht immer.