Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3483

Interaktive Analyse von Supercomputer-Leistungsdaten

Jessica Hackländer

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dipl. Inf. Alexandros Panagiotidis, Dr. Guido Reina
Beginn am:	8. Mai 2013
Beendet am:	7. November 2013
CR-Nummer:	C.4, C.5.1, D.4.5, D.4.8

Kurzfassung

Mit den stetig wachsenden Datenmengen wachsen auch heutige High Performance Computing (HPC)-Umgebungen in ihrer Größe und Komplexität. Gleichzeitig wächst aber auch ihre Fehleranfälligkeit, wodurch es zunehmend wichtig ist, HPC-Systeme fehlertoleranter zu machen. In dieser Arbeit werden Visualisierungstechniken vorgestellt, die für die Visualisierung und Analyse von HPC-Leistungsdaten geeignet scheinen und gezeigt, dass keine dieser Visualisierungen alleine die Komplexität dieser Daten vollständig erfassen kann. Bereits bestehende Ansätze im Bereich der HPC-Leistungsanalyse werden vorgestellt und gezeigt, dass diese entweder nur Anwendungsdaten betrachten oder nur einfache Visualisierungs- und Interaktionstechniken bieten. Im Hauptteil dieser Arbeit wird die Entwicklung eines Prototyps beschrieben, der auf diesen Kenntnissen basierend verschiedene Visualisierungen implementiert, die unterschiedliche Aspekte der Daten betrachten und anwendungsbasierte mit infrastrukturellen Daten kombinieren. Durch die Implementierung verschiedener Interaktionstechniken, wie Multiple Coordinated Views, Selektion und Filtermöglichkeiten, bietet der Prototyp unterschiedliche Möglichkeiten, um die Daten zu explorieren und analysieren. In einer anschließenden beschriebenen Benutzerstudie wird für einzelne umgesetzte Visualisierungs- und Interaktionstechniken gezeigt, dass diese sinnvoll und hilfreich für die Analyse von HPC-Leistungsdaten sind. Der Prototyp stellt damit einen sinnvollen Ansatz für die Analyse von HPC-Leistungsdaten dar und kann in Zukunft durch weitere Visualisierungs- und Interaktionstechniken erweitert werden.

Abstract

As today's high-performance computing (HPC) systems grow continously in their size and complexity, so does their error-proneness and fault tolerance in these systems has become a major issue. This thesis introduces several visualization techniques that seem appropriate for the visualization and analysis of HPC performance data and shows that one single visualization can not cover the complexity of the data. Existing approaches, in the scope of HPC performance analysis, consider only application data or implement simple visualization and interaction techniques. The main part of this thesis describes the concept and development of a prototype that, based on the insights above, implements several visualizations that cover different aspects of the data and combines application data and infrastructural data of HPC systems. The implementation of appropriate interaction techniques, like Multiple Coordinated Views, selection and filters, allows the exploration and analysis of the dataset. A case study shows for selected visualization and interaction techniques that they are helpful for the analysis of HPC performance data. Therefore the prototype is a useful approach for the analysis of HPC performance data and can be extended by the implementation of more visualization and interaction techniques.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Dieses Kapitel beschreibt die Motivation und das Ziel, sowie die Gliederung für diese Arbeit.

Motivation

Supercomputer wachsen und werden immer leistungsstärker. Heutige HPC-Systeme können aus 100 000 oder mehr Knoten bestehen. Mit zunehmender Größe und Komplexität der eingebauten Komponenten steigt aber auch ihre Anfälligkeit für Fehler. Statistisch gesehen tritt bei jeder der eingebauten Komponenten früher oder später ein Fehler auf und unterbricht ihre Operationen. Als 2001 der ASCI (Accelerated Strategic Computing Initiative) White Supercomputer von Lawrence Livermore National Laboratory online ging, trat im Durchschnitt alle fünf Stunden ein Fehler auf. Diese Quote konnte zwar später auf ca. fünfundfünfzig Stunden erhöht werden, doch mit dem anhaltenden Wachstum von Supercomputern wächst auch dieses Problem stetig weiter und es wird zunehmend wichtiger, die Fehlertoleranz solcher Systeme zu gewährleisten. [COM12]
Gleichzeitig wächst die Menge der Daten, die es zu überwachen gilt. Ein großes Problem besteht darin, dass die Daten von Supercomputersystemen oftmals in vielen unterschiedlichen Formaten vorliegen, die sich nicht so einfach miteinander vergleichen lassen. Bestehende Frameworks, wie VAMPIR, Nagios und Munin, betrachten meist nur Applikationsdaten oder bieten sehr einfache Methoden zur Visualisierung und Interaktion. Gleichzeitig liegt für jede Komponente des Systems eine Vielzahl verschiedener Daten in unterschiedlichen Formaten und zu unterschiedlichen Zeitpunkten gemessen vor. Die bisherigen Methoden reichen deshalb nicht mehr aus, um die Komplexität solcher Systeme zu erfassen und gleichzeitig überschaubar darzustellen.

Ziel

Das Ziel dieser Arbeit ist es, durch ausgewählte Visualisierungen und Interaktionstechniken die Analyse von anwendungsbasierten Leistungsdaten und infrastrukturellen Daten aus Supercomputersystemen miteinander zu kombinieren. Dafür sollen die Daten, die oft in unterschiedlichen Formaten vorliegen, auf dieselbe Visualisierung abgebildet werden, um sie zu vergleichen und Zusammenhänge zwischen den Daten herstellen zu können. Die Menge an Daten, die in heutigen Supercomputersystemen vorliegt, soll durch die gewählten

Visualisierungen gleichzeitig überschaubar und skalierbar auf dem Bildschirm dargestellt werden. Durch die Kombination verschiedener Visualisierungen sollen unterschiedliche Aspekte der Daten dargestellt und miteinander verglichen werden können. Mithilfe ausgewählter Interaktionstechniken sollen diese Daten exploriert und Details analysiert werden können. Der Benutzer soll im Idealfall mithilfe eines einzigen Tools alle ihm vorliegenden Leistungs- und Infrastrukturdaten analysieren und vergleichen können und dadurch Möglichkeiten erkennen, die Fehlertoleranz des Systems zu verbessern.

Komponenten aus anderen Arbeiten

Der im Zuge dieser Arbeit entwickelte Prototyp enthält ein Dashboard, welches Kleinansichten zu geöffneten Fenstern enthält (siehe Kapitel 5.2.2 und Abbildung 5.9). Dabei handelt es sich um Buttons mit einem Bild und Label, welche das entsprechende Fenster in den Vordergrund holen. Die Idee zu dieser Art Übersicht mit Kleinansichten stammt aus einem Studienprojekt der Universität Stuttgart mit dem Titel „Holistische Anzeige von Softwaresystemen“ [WMR+11], welches von Oktober 2010 bis Oktober 2011 stattfand. Bei den Kleinansichten aus dem Studienprojekt handelt es sich um GUI-Steuerelemente, die während des Projekts entwickelt wurden und andere Interaktionsmöglichkeiten mit den Fenstern enthalten, als die in dieser Arbeit implementierten Kleinansichten. Es wurde lediglich die Idee eines Dashboards übernommen, vom Quellcode wurden keine Teile verwendet.

Gliederung

Kapitel 2 – Grundlagen und Definitionen beschreibt und erklärt zunächst Grundlagen und Begriffe, die für diese Arbeit von Bedeutung sind.

Kapitel 3 – Visualisierungsmethoden beschreibt und vergleicht verschiedene Visualisierungsmethoden, die für die Visualisierung zeitabhängiger und multivariater Daten in Frage kommen.

Kapitel 4 – Verwandte Arbeiten analysiert bestehende Arbeiten zum Thema „Visualisierung von Supercomputer-Leistungsdaten“ und stellt die Unterschiede zu dieser Arbeit heraus.

Kapitel 5 – Entwicklung eines Prototyps beschreibt den Hauptteil dieser Arbeit, das Konzept und die Umsetzung eines Prototyps zur Visualisierung multivariater, zeitabhängiger, großer Datenmengen.

Kapitel 6 – Benutzerstudie befasst sich mit dem Aufbau, der Durchführung und den Ergebnissen einer Benutzerstudie zur Evaluation des Prototyps.

Kapitel 7 – Zusammenfassung und Ausblick fasst die Ergebnisse dieser Arbeit zusammen und stellt Anknüpfungspunkte vor.

2. Grundlagen und Definitionen

Dieses Kapitel beschreibt die Grundlagen und Definitionen, die für diese Arbeit von Bedeutung sind.

2.1. Definitionen

Supercomputer/HPC

Der Begriff Supercomputer bezeichnet Hochleistungsrechner, deren Rechenleistung zum Zeitpunkt ihrer Einführung im obersten erreichbaren Bereich liegt. Diese Leistung wird durch Parallelisierung erreicht. Auszuführende Operationen werden auf mehrere Prozessoren verteilt, um die Arbeitsgeschwindigkeit zu erhöhen [DEC11]. HPC, oder High Performance Computing, bezeichnet die Verwendung von solchen Supercomputern zur Lösung komplexer mathematischer Probleme, für die die Leistung konventioneller Rechner nicht mehr ausreicht [How10]. Supercomputer werden oft für die Durchführung von Simulationen, welche immense Rechenleistungen benötigen, verwendet. Sie werden häufig in der Wissenschaft eingesetzt, zum Beispiel für die Simulation von Vulkanausbrüchen und Erdbeben [DEC11] oder in der Krebsbehandlung [Deu12]. Durch den hohen Grad an Parallelisierung auf verteilten Rechnern werden Verfügbarkeit und Zuverlässigkeit in Supercomputersystemen zu zentralen Themen.

Zeitabhängige Daten

Zeitabhängige Daten sind im Allgemeinen Attribute, die von der Dimension „Zeit“ abhängen. Nach Öttl [Ött08] und Daassi et al. [DFN02] enthalten zeitabhängige Daten sowohl Datenaspekte, als auch zeitliche Aspekte. Bei den Datenaspekten handelt es sich um nominale, ordinale oder numerische Attribute, die von den zeitlichen Aspekten abhängen und sich durch diese ändern können. Die zeitlichen Aspekte hängen wiederum davon ab, ob man von linear oder periodisch verlaufender Zeit ausgeht, und ob es sich bei den zugrundeliegenden Zeitprimitiven um Zeitpunkte oder Zeitintervalle handelt.
Bei der Analyse von Supercomputer-Leistungsdaten handelt es sich im Normalfall um einen linearen Zeitverlauf, während dem periodisch Daten aufgezeichnet werden, zum Beispiel die CPU- und Speicherauslastung der Rechner, die in bestimmten Intervallen aufgezeichnet werden.

Multivariate/mehrdimensionale Daten

Von multivariaten Daten spricht man, wenn für ein Objekt Daten zu mehreren Variablen vorliegen. Im Kontext von Supercomputer-Leistungsdaten liegen beispielsweise Daten für die CPU-Auslastung, Daten für die Speicherauslastung und Daten für die gesendeten Pakete pro Sekunde zu einem bestimmten Zeitpunkt und Rechner vor. Die Schwierigkeit bei multivariaten Daten liegt in der Visualisierung von vielen verschiedenen Variablen in einer einzigen zweidimensionalen Visualisierung, ohne dass diese für den Betrachter überladen oder unübersichtlich wird [Cha06].

Multiple Coordinated Views

Multiple Coordinated Views sind eine Visualisierungstechnik zur Exploration von Daten. Unterschiedliche Aspekte dieser Daten werden in mehreren Fenstern repräsentiert, wobei die Operationen auf den Repräsentationen koordiniert sind. Der Hauptgedanke hinter dieser Technik ist, dass die explorierende Person die zu explorierenden Daten besser versteht, wenn sie sie aus verschiedenen Perspektiven betrachten und mit ihnen interagieren kann [Rob07]. Multiple Coordinated Views nutzen daher oft Interaktionstechniken wie „Brushing und Linking“.

Brushing und Linking

Brushing und Linking beschreibt ein Interaktionskonzept, das verschiedene Visualisierungen miteinander kombiniert, um die Defizite einzelner Visualisierungen zu überwinden. Interaktionen in einer Visualisierung wirken sich automatisch auf alle anderen Visualisierungen aus, zum Beispiel indem Elemente, die in einer Visualisierung markiert werden, automatisch in allen anderen Visualisierungen hervorgehoben werden. Dadurch können Abhängigkeiten und Korrelationen sichtbar gemacht werden [Kei02].

Fokus + Kontext

Fokus + Kontext beschreibt ein Konzept, das von folgenden drei Voraussetzungen ausgeht: Der Benutzer benötigt eine Übersicht, den Kontext, und eine Detailansicht, den Fokus, zur selben Zeit. Die in der Übersicht benötigten Informationen können sich von den in der Detailansicht benötigten Informationen unterscheiden. Diese beiden Arten von Informationen können in einer einzigen Ansicht miteinander kombiniert werden, in etwa wie beim menschlichen Sehvermögen. [CMS99]

Visual Clutter

Visual Clutter tritt auf, wenn zu viele Daten auf einer zu kleinen Fläche dargestellt werden. Elemente der Visualisierung überdecken sich und sind schlecht voneinander unterscheidbar, wodurch das Analysieren der Daten erheblich erschwert wird [ED07]. Bei den Daten aus Supercomputersystemen sind Visualisierungs- und Interaktionstechniken, die auf diese Datenmengen ausgelegt sind, daher von hoher Bedeutung.

2.2. Datengrundlage dieser Diplomarbeit

Als Grundlage für diese Arbeit wurden verschiedene Daten eines Computerclusters gesammelt. Daraus ergeben sich unter anderem Daten für die Rechner des Supercomputers und Daten für das Netzwerk über welches die Rechner miteinander kommunizieren. Für die Rechner ergeben sich zum Beispiel Daten für die CPU-Kerne und den Speicher, wie die CPU-Auslastung der Kerne und die Speicherauslastung des Rechners. Für das Netzwerk ergeben sich Daten für die Kommunikation zwischen den Rechnern und CPU-Kernen, zum Beispiel die übertragenen Pakete pro Sekunde. So erhält man eine Menge von Daten, die sowohl Leistungsdaten als auch infrastrukturelle Daten des Systems beinhalten. Die Daten für diese Arbeit wurden mit Hilfe von Sysstat aus einem Computercluster an der Universität Stuttgart überwacht. Sysstat ist eine Sammlung von Programmen, mit welchem eine Vielzahl an verschiedenen Daten zur Leistung und Anwendungsaktivität von Linux- und vielen anderen Unix-basierten Systemen gesammelt und ausgegeben werden kann. Der Dienst sar zeichnet diese Daten periodisch aus dem Supercomputersystem auf und der Dienst sadf gibt sie in verschiedenen Formaten aus [God10]. Die Daten umfassen einen Zeitraum von einem Monat, wobei durchschnittlich alle zehn Minuten Daten aufgezeichnet wurden. Inhaltlich umfassen die Daten viele verschiedene Arten von Daten, die man mit Hilfe von Sysstat erfassen kann. Dazu gehören zum Beispiel die CPU-Auslastung der Rechnerkerne, die Speicherauslastung der Rechner und die empfangenen bzw. gesendeten Pakete pro Sekunde, jeweils in Kilobyte und Prozentangaben.

2.3. Arten von Daten

Leistungsdaten, die aus Supercomputersystemen gesammelt werden können, liegen oft in unterschiedlichen Formaten vor. Ludewig und Lichter [LL07] teilen Daten allgemein in folgende Kategorien ein:

- **Nominale Daten** sind Daten, die unterschieden werden können, die aber in keine logische Reihenfolge gebracht werden können. Sie können nur in „gleich“ und "nicht gleich“ unterschieden werden. Nominale Daten sind in diesem Kontext zum Beispiel der Name und die ID eines Rechners im Netzwerk.

- **Ordinale Daten** sind nominale Daten, die zusätzlich in eine logische Reihenfolge gebracht werden können. Es gibt jedoch keine definierten Differenzen zwischen den Werten. Von ordinalen Daten können der Median und andere Perzentilen bestimmt werden. In diesem Kontext kann beispielsweise die CPU-Auslastung eines Rechners in „hoch“, „mittel“ und „niedrig“ eingestuft werden. Dadurch kann man nur aussagen, dass ein Rechner eine höhere Auslastung hat, als ein anderer, aber nicht, wie viel höher diese Auslastung ist.
- **Intervalldaten** sind ordinale Daten, die zusätzlich immer die gleiche Differenz zwischen zwei Werten haben. Mit Intervalldaten können Berechnungen angestellt werden und zum Beispiel der Durchschnitt oder Differenzen zwischen zwei Werten berechnet werden. Intervalldaten sind beispielsweise die Zeitpunkte, zu denen die Speicherauslastung der Rechner im Netzwerk gemessen wurde, denn zwischen zwischen zwei Sekunden, Minuten oder Stunden liegt immer die gleiche Zeitspanne.
- **Rationale Daten** sind Intervalldaten, die zusätzlich einen natürlichen Nullpunkt haben. Dadurch können Verhältnisse zwischen zwei Werten berechnet werden. Rationale Daten sind in diesem Kontext zum Beispiel die gesendeten Pakete pro Sekunde oder Prozentangaben der CPU-Auslastung. Der Nullpunkt sind hier 0 gesendete Pakete pro Sekunde, bzw. 0% Auslastung. Man kann zum Beispiel aussagen, dass die Auslastung eines Rechners zu einem Zeitpunkt doppelt so hoch war, wie zu einem anderen Zeitpunkt.
- **Absolute Daten** sind rationale Daten, bei denen ein Zahlenwert selbst die gesuchte Größe ist und nicht nur im Verhältnis zur gesuchten Größe steht. Absolute Daten sind auf die natürlichen Zahlen beschränkt und beschreiben eine Größe, die durch Zählen ermittelt wird. Dadurch können sie unter anderem nicht dividiert werden, um den Mittelwert zu berechnen, aber es kann der Median berechnet werden. Absolute Daten sind in diesem Kontext die Anzahl der Rechner des Netzwerks oder die Anzahl der aufgezeichneten Zeitpunkte.
  Oft werden absolute Daten zu rationalen Daten erweitert, um dennoch Werte wie den Durchschnitt berechnen zu können. So kommt man zu Aussagen, wie „Durchschnittlich waren diese Woche 5,7 von 10 Rechnern aktiv.“

Diese unterschiedlichen Arten von Daten werden auf verschiedene Visualisierungen abgebildet. Die Schwierigkeit besteht darin, die unterschiedlichen Eigenschaften dieser Daten so in einer Visualisierung darzustellen, dass man sie gut miteinander vergleichen kann.

2.4. Begriffe in dieser Arbeit

Dieser Abschnitt beschreibt Begriffe, die in dieser Arbeit häufig verwendet werden.

Die Struktur eines HPC-Systems kann als Graph aufgefasst werden. In Anlehnung an die Graphentheorie werden deshalb in dieser Arbeit die Begriffe „Knoten“ und „Kanten“ verwendet. Sofern dies im Text nicht explizit anders angegeben ist, sind mit diesen Begriffen immer die

Knoten und Kanten eines Supercomputersystems gemeint, nicht die Knoten und Kanten einer Visualisierung.

Knoten Ein „Knoten“ kann einen Rechner des HPC-Systems oder auch einen CPU-Kern eines solchen Rechners beschreiben.

Kanten Eine „Kante“ beschreibt die Kommunikation zwischen oben genannten Knoten. So kann eine Kante beispielsweise eine Anfrage von einem Rechner an einen anderen Rechner des Netzwerks darstellen.

Auffälligkeiten in den Daten In dieser Arbeit wird mehrfach von „auffälligen Daten“ oder „Auffälligkeiten in den Daten“ gesprochen. Damit ist ungewöhnliches Verhalten von Knoten im Supercomputersystem gemeint, zum Beispiel Ausfälle von Rechnern oder Rechner bzw. CPU-Kerne, die keine Aufgaben haben, während andere voll ausgelastet sind und damit die Leistung eines Supercomputersystems beeinträchtigen. In Kapitel 5.2.2, in den Paragraphen „Pixelmap“ bis „Timeline Chart“, wird beispielhaft gezeigt, wie sich solche Auffälligkeiten in einer Visualisierung äußern.

3. Visualisierungsmethoden

Dieses Kapitel beschäftigt sich mit bestehenden Visualisierungsmethoden für die Visualisierung von großen, multivariaten, zeitabhängigen Daten. Dabei werden zunächst mögliche Visualisierungsmethoden vorgestellt und ihre Eignung für die Darstellung großer Datenmengen, multivariater Daten und zeitabhängiger Daten einzeln bewertet. Am Ende des Kapitels werden die Visualisierungsmethoden in einer Tabellenübersicht direkt miteinander verglichen, um die geeignetsten Methoden für die Ziele dieser Arbeit herauszustellen.

3.1. Kategorien

Nach Chan [Cha06] lassen sich Visualisierungen in vier verschiedene Kategorien einteilen:

Geometrische Projektionen versuchen einen multidimensionalen Datensatz durch informative Projektionen oder Transformationen darzustellen. Dazu gehört zum Beispiel die Projektion auf ein kartesisches Koordinatensystem, wie das bei Scatter Plots der Fall ist, aber auch die Projektion auf einen willkürlichen Raum, wie zum Beispiel bei Parallelen Koordinaten. In diesen Darstellungen lassen sich leicht Ausreißer finden und mit Hilfe entsprechender Interaktionstechniken auch große Datensätze behandeln. Dennoch können Visual Clutter (siehe Kapitel 2.1) oder sich überdeckende Datenelemente bei großen oder multivariaten Datensätzen zum Problem werden. Als Beispiele für geometrische Projektionen nennt Chan die Scatterplot Matrix, Prosection Matrix, HyperSlice, Hyperbox, Parallele Koordinaten, Radial Coordinate Visualization, Andrews Curve, Star Coordinates und Table Lens. Auch einfache Scatter Plots [DR11], Spiral Graphs [CK98] und der ThemeRiver [HHN00] können zu dieser Kategorie gezählt werden.

Pixelorientierte Techniken stellen den Wert einzelner Attribute durch einzelne Pixel mit Farben, basierend auf einer Farbskala, dar. Ein Datenobjekt wird in einem n-dimensionalen Raum daher durch n Pixel dargestellt. Als Beispiele für pixelorientierte Visualisierungen nennt Chan Space Filling Curves, Recursive Pattern, Spiral and Axes Techniques, Circle Segment und Pixel Bar Charts. Auch die Pixelmap [BDW05] und Table Lens [RC94] können zu dieser Kategorie gezählt werden.

Hierarchische Darstellungen unterteilen den gegebenen Datenraum in mehrere Teile und stellen diese Teilbereiche in einer hierarchischen Ordnung dar. Dadurch werden nicht alle Attribute gleich behandelt, da manche weiter oben und manche weiter unten in der Hierarchie stehen. Deshalb werden sie unterschiedlich abgebildet, beispielsweise indem Elemente weiter oben in der Hierarchie wichtiger sind oder als Aggregation der darunterliegenden Elemente dargestellt werden. Das führt zu verschiedenen Ergebnissen. In diesen Darstellungen werden meist Daten dargestellt, die von vornherein hierarchisch sind oder in denen manche Attribute wichtiger sind als andere. Als Beispiele für hierarchische Darstellungen nennt Chan Hierarchical Axis, Stacking, Worlds Within Worlds und Treemaps.

Ikonografien oder auch Icon-basierte Techniken bilden multidimensionale Datenobjekte auf Icons oder Glyphen ab, die jeweils die verschiedenen Attribute des Objekts enthalten. Es kann jedoch schnell passieren, dass der Benutzer nicht alle Attribute gleich wichtig wahrnimmt, da ihm bestimmte Teile des Icons mehr auffallen, als andere. Daher kann hier nicht sichergestellt werden, dass alle Attribute gleichwertig behandelt werden. Als Beispiele für ikonografische Visualisierungen nennt Chan Chernoff Faces, Star Glyphs, Stick Figures, Shape Coding, Color Icons und Texture.

3.2. Visualisierungen

Aus den oben genannten Kategorien sind diejenigen Visualisierungen interessant, die sich insbesondere für die Visualisierung vieler, multivariater und zeitabhängiger Daten eignen. Im folgenden Kapitel werden einige Visualisierungen, die dafür geeignet scheinen, genauer betrachtet.

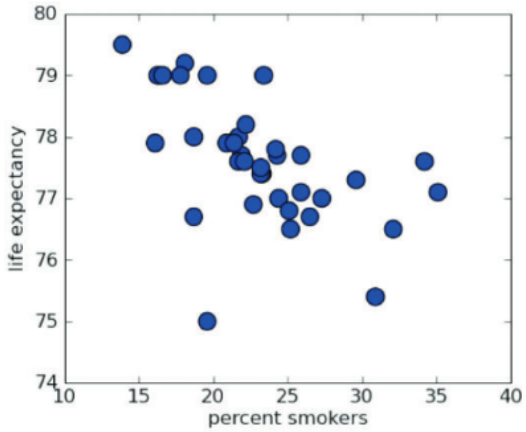

Abbildung 3.1.: Scatter Plot [DR11]

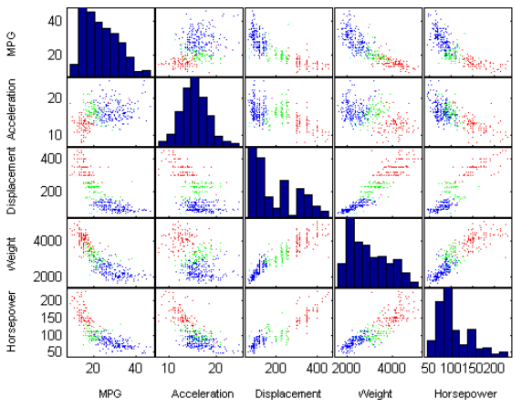

Abbildung 3.2.: Scatterplot Matrix [Mat]

3.2.1. Scatter Plots

Scatter Plots (siehe Abbildung 3.1) stellen einen Zusammenhang zwischen zwei Variablen dar. In einem zweidimensionalen Raum wird ein Objekt durch einen Punkt mit den entsprechenden Variablenwerten des Objekts dargestellt. Im Gegensatz zum Liniendiagramm werden die Punkte aber nicht miteinander verbunden. Durch verschiedene Muster in den Punkten können Rückschlüsse über Zusammenhänge gezogen werden, zum Beispiel wenn sich viele Punkte an einer Stelle häufen. Scatter Plots sind für die Darstellung großer Datenmengen gut geeignet, da erst durch sehr viele Punkte Muster erkannt werden können. Für die Darstellung von zeitlichen Daten kann die Zeit auf einer der Achsen abgetragen werden, jedoch sind die Verläufe der Werte nicht so gut erkennbar, wie in einem Liniendiagramm. Für die Darstellung von mehreren Objekten muss eine Möglichkeit gefunden werden, einen Punkt einem bestimmten Objekt zuzuordnen, zum Beispiel in dem alle Punkte eines Objekts die gleiche Farbe haben. Es kann aber passieren, dass zwei Objekte an der gleichen Stelle einen Punkt haben, und einer dann überdeckt wird. Scatter Plots sind nach Chan [Cha06] für die Darstellung bivariater Daten gemacht, multivariate Daten können nicht visualisiert werden.

3.2.2. Scatterplot Matrix

Die Scatterplot Matrix (siehe Abbildung 3.2) ist eine Abwandlung des normalen Scatter Plots für die Visualisierung multivariater Daten. Es handelt sich dabei um eine Matrix, die aus mehreren Scatter Plots für jedes mögliche Paar von Variablen besteht. Durch die Anordnung als Matrix ist es möglich, multivariate Daten mit beliebig vielen Variablen darzustellen. So können auch zeitabhängige Daten mit vielen weiteren Variablen in dieser Visualisierung dargestellt werden. Die Zeitvariable wird dabei wie jede andere Variable des Datensatzes behandelt. Man erhält also eine Reihe von Scatter Plots, in der jede Variable einmal in Abhängigkeit der Zeit dargestellt wird. Große Datenmengen können, wie in den einfachen Scatter Plots, auch hier sehr gut dargestellt werden und Muster erkennbar machen. Nach Chan [Cha06] kann es bei hoher Dimensionalität der Daten passieren, dass Muster in den Scatter Plots nur schwer erkannt werden können. Wenn die Anzahl der zu visualisierenden Datenobjekte zu groß wird, wird die Scatterplot Matrix außerdem chaotisch und unübersichtlich. Durch Brushing und Linking (siehe Kapitel 2.1) kann dem zumindest ein Stück weit entgegengewirkt werden.

3.2.3. Parallele Koordinaten

Parallele Koordinaten (siehe Abbildung 3.3) sind speziell für die Visualisierung multivariater Daten gedacht. Verschiedene Variablen werden auf parallelen Achsen abgetragen. Für ein dargestelltes Objekt wird auf jeder Achse der Wert der Variablen markiert und mit dem Wert der benachbarten Achse verbunden. Dadurch ergeben sich Muster, die auf bestimmte Abhängigkeiten, wie umgekehrte Proportionalität, zwischen den Achsenvariablen schließen lassen [HW12]. Für die Visualisierung multivariater Daten sind Parallele Koordinaten also gut geeignet, allerdings hängt es stark von der Anordnung der Achsen ab, zwischen welchen Variablen man direkte Zusammenhänge erkennen kann. Die Visualisierung zeitlicher Daten mittels

Paralleler Koordinaten ist schwierig. Man kann zwar die Zeit als Variable auf einer Achse abtragen, sie dann aber nur mit den direkt benachbarten Variablen gut vergleichen. Alternativ kann man eine Visualisierung pro Zeitpunkt erstellen und diese einzeln betrachten.

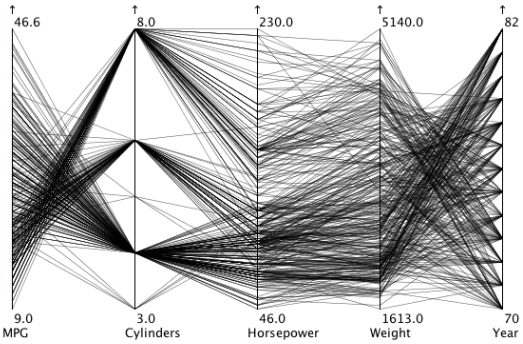

Abbildung 3.3.: Parallele Koordinaten [Kos]

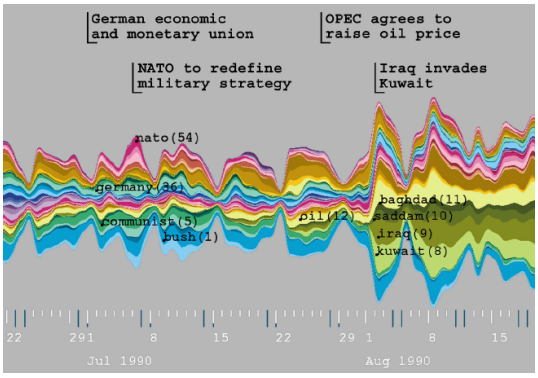

Abbildung 3.4.: Theme River [HHN00]

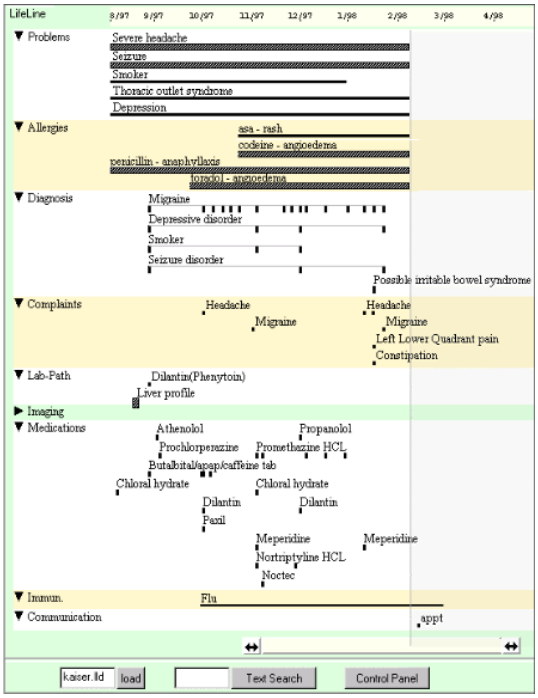

Abbildung 3.5.: LifeLines [PMS+98]

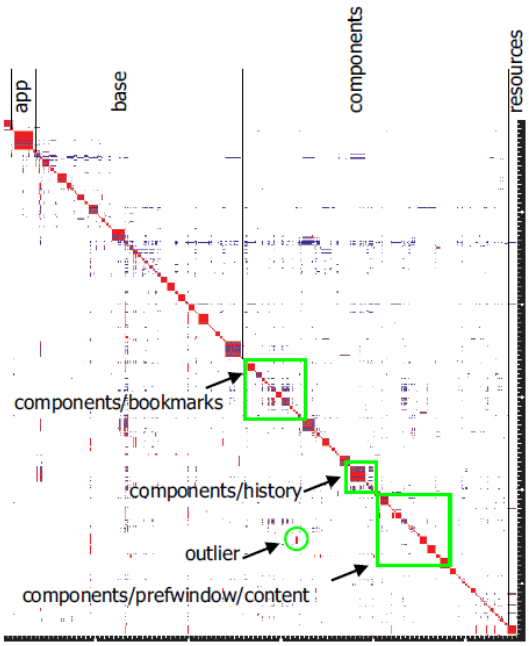

Abbildung 3.6.: Pixelmap [BDW05]

3.2.4. GANTT-Diagramme und Life Lines

Mit GANTT-Diagrammen [BTG+05] und Life Lines (siehe Abbildung 3.5) kann man Zeitspannen, zum Beispiel die Dauer von Aufgaben, visualisieren. In horizontaler Richtung wird die Zeit abgetragen, während in vertikaler Richtung die verschiedenen Aufgaben untereinander aufgelistet werden. Eine Zeitspanne wird als horizontaler Balken vom geplanten Start- bis zum geplanten Endzeitpunkt dargestellt. GANTT-Diagramme und Life Lines sind für die Darstellung von Zeitspannen gedacht, durch einen farblichen Verlauf in diesen Zeitspannen können auch zeitliche Verläufe dargestellt werden. Es kann aber neben der Zeitspanne nur eine einzige Variable dargestellt werden, für die Visualisierung multivariater Daten ist diese Darstellung also nicht geeignet. Bei sehr großen Datenmengen kann man schnell die Übersicht verlieren und muss eventuell viel Scrollen. Um das etwas zu erleichtern kann man die Aufgaben zum Beispiel nach ihrer Startzeit sortieren.

3.2.5. Theme River

Die Theme-River-Visualisierung wird verwendet, um die Veränderung von Mengen über die Zeit darzustellen. Abbildung 3.4 zeigt einen beispielhaften Verlauf verschiedener Werte über die Zeit. Dadurch kann man zum Beispiel erkennen, ob ein Wert immer ansteigt, wenn ein anderer auch ansteigt, oder ob ein Wert immer ansteigt, wenn ein anderer absinkt, und so auf Zusammenhänge zwischen den Werten schließen. Bei dieser Darstellung kann man gut sehen, ob die Summe aller Werte an einem bestimmten Zeitpunkt hoch oder niedrig ist. Um den Verlauf eines einzelnen Wertes zu betrachten, muss man genauer hinsehen, denn bei sehr vielen Daten kann es passieren, dass die einzelnen Flächen sehr dünn werden und gleichzeitig nicht auf einer waagerechten Linie verlaufen, sondern in großen Wellen, wie in Abbildung 3.4 zu sehen ist. Diese Darstellung ist gut für die Visualisierung von zeitlichen Daten geeignet, wobei die x-Achse die Zeit darstellt. Es können jedoch nur Mengen visualisiert werden, keine anderen Datenarten, auch keine negativen Werte. Für die Visualisierung multivariater Daten ist diese Darstellung nicht gut geeignet, denn es können zwar verschiedene Variablen dargestellt werden, indem jede Linie eine Variable repräsentiert, dann aber nur für ein Objekt. Bei großen Datenmengen werden die Schwankungen in den Linien größer und es hängt stark von der Gesamtsumme der Mengen zu einem Zeitpunkt ab, wie viele Werte auf den Bildschirm passen. Bei sehr vielen Werten werden die einzelnen Linien unter Umständen so dünn, dass man sie kaum noch unterscheiden kann. Nach Friendly [HHN00] hat sich der ThemeRiver in einer Benutzerstudie als einfach zu verstehen und nützlich zur Analyse von größeren Entwicklungstendenzen erwiesen. Für die Analyse von sehr kleinen Tendenzen erwies sich die Visualisierung als weniger hilfreich, da durch die Kurven sehr kleine Werte nicht genug hervorgehoben werden. Durch die durchgehenden Linien fällt es leicht, eine Tendenz zu verfolgen, es fehlt den Teilnehmern der Studie aber die Möglichkeit, genaue Werte herauszulesen, da aus der Visualisierung nicht hervorgeht, wie die Werte genau dargestellt werden, und die Werte zwischen zwei gemessenen Punkten interpoliert werden.

3.2.6. Pixelmap

In einer Pixelmap wird jeder Wert durch einen Pixel, bzw. ein Rechteck, repräsentiert. Im zweidimensionalen Raum gibt es eine x- und eine y-Achse, an denen die Werte eingetragen werden. Durch Farbe und/oder Sättigung können weitere Informationen in einem Pixel dargestellt werden. So kann man an der x-Achse die Zeit abtragen und an der y-Achse die zu visualisierenden Objekte untereinander. Dann erhält man zu jedem Objekt und Zeitpunkt einen Pixel, dessen Farbe einen bestimmten Wert repräsentiert. Alternativ könnte man an der y-Achse verschiedene Variablen für ein bestimmtes Objekt darstellen. Die Farbe kann zum Beispiel durch einen definierten Farbverlauf die Höhe des Werts angeben. Die Autoren Burch et. al [BDW05] verwenden eine Pixelmap, um die Abhängigkeiten von Dateien im CVS-Archiv des Mozilla-Projekts zu visualisieren (siehe Abbildung 3.6). Die Pixelmap nutzt den Bildschirmplatz sehr gut aus, da keine Hohlräume übrig bleiben. So können sehr viele Werte über- und untereinander dargestellt werden. Bei sehr großen Datenmengen können außerdem mehrere Werte in einem Pixel aggregiert werden, welcher dann beispielsweise den Durchschnitt oder das Maximum darstellt. Auch für die Visualisierung von zeitlichen Daten ist eine Pixelmap gut geeignet. An der x-Achse kann die Zeit abgetragen werden und durch mögliche Aggregationen von Pixeln können auch sehr große Zeiträume dargestellt werden. Die Visualisierung multivariater Daten ist schwieriger. Man könnte für ein Objekt alle Variablen untereinander als Zeilen von Pixeln darstellen. Dazu müssen die Variablen aber ähnliche Metriken haben, oder der Benutzer muss unterscheiden können, was in welcher Zeile dargestellt wird. Wenn man viele Variablen für viele Objekte darstellt, werden die Zeilen aber schnell so dünn, dass man sie nicht mehr beschriften kann, um die dargestellten Metriken zu beschreiben.

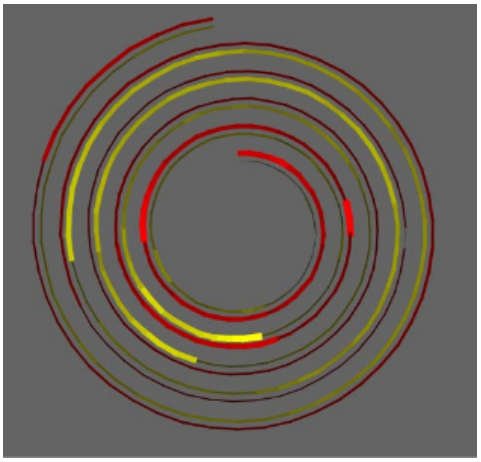

Abbildung 3.7.: Spiral Graph [WAM01]

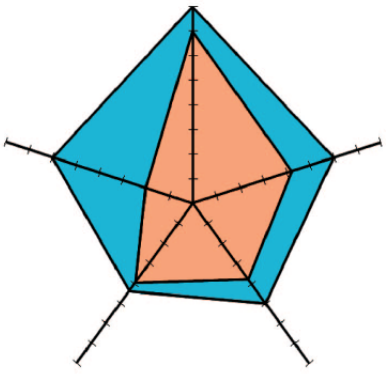

Abbildung 3.8.: Star Plot [DLR09]

3.2.7. Spiral Graphs

Spiral Graphs (siehe Abbildung 3.7) wurden für die Darstellung von zeitlichen Daten entwickelt. Um lange Zeitspannen darstellen zu können, verläuft die Zeit in einer Spirale von innen nach außen. Indem beispielsweise eine volle Umdrehung immer einen ganzen Tag darstellt, können auch Aussagen über periodisch auftretende Ereignisse getroffen werden, zum Beispiel "Ereignis X tritt immer um 12:00 Uhr auf". Innerhalb der Spiralen können die Daten unterschiedlich dargestellt werden, zum Beispiel als Liniendiagramm oder Farbverlauf. Zeitliche Daten können gut dargestellt werden, da die Spirale einen Zeitverlauf beschreibt. Große Datenmengen können nur bedingt gut dargestellt werden. Große Zeiträume lassen sich durch die Spirale zwar gut auf dem begrenzten Bildschirmplatz darstellen, aber die Dicke der Spirale ist begrenzt, wodurch viele Daten zu einem bestimmten Zeitpunkt schlecht darstellbar und unterscheidbar sind. Auch die Visualisierung multivariater Daten ist nicht einfach. Man könnte in der Spirale über mehrere Farbverläufe untereinander verschiedene Variablen darstellen. Um viele Objekte mit vielen Variablen darzustellen, könnte man sich überlagernde transparente Spiralen verwenden. Nach Carlis et al. [CK98] erwiesen sich Spiral Graphs in einer Nutzerstudie als schwierig zu verstehen und die Teilnehmer benötigten Hilfe von jemandem, der sich mit der Visualisierung bereits auskannte. Spiral Graphs geben dem Benutzer einfache visuelle Hinweise in periodischen und seriellen Aspekten von zeitlichen Daten. Die Teilnehmer der Studie fanden die Visualisierung hilfreich zur Extraktion saisonabhängiger und periodischer Daten, die aus anderen Ansichten der Daten nicht offensichtlich hervorgehen.

3.2.8. Star Glyphs

Star Glyphs oder Star Plots (siehe Abbildung 3.8) sind ähnlich wie die Parallelen Koordinaten aufgebaut. Auf mehreren kreisförmig angeordneten Achsen werden die Werte der Variablen eines Objekts abgetragen, wobei eine Achse eine Variable beschreibt. Die benachbarten Punkte eines Objekts werden durch Linien verbunden. Durch die kreisförmige Anordnung der Achsen lassen sich Werte verschiedener Achsen unter Umständen besser vergleichen, als in den Parallelen Koordinaten. Gleichzeitig wird durch die Kreisform aber der Bildschirmplatz nicht optimal genutzt. Star Plots sind gut für die Darstellung multivariater Daten geeignet, da auf jeder Achse eine Variable abgetragen werden kann. Für die Darstellung zeitlicher Daten sind Star Plots bedingt geeignet. Die Zeit kann auf einer Achse abtragen werden, aber das erschwert es, die Zeitpunkte mit den Werten der anderen Variablen zu vergleichen. Man kann auch für jeden Zeitpunkt eine Achse verwenden. Aufgrund der Kreisanordnung wäre das für zyklische Zeitverläufe gut geeignet, für lineare Zeitverläufe weniger. Dann können jedoch keine multivariaten Daten mehr dargestellt werden. Für die Darstellung großer Datenmengen ist dieses Diagramm nicht gut geeignet. Bei vielen Variablen werden die Abstände zwischen den Achsen sehr klein, so dass man sie irgendwann kaum voneinander unterscheiden kann, und bei vielen zu visualisierenden Objekten entstehen viele Linien, die sich überschneiden und das Diagramm sehr unübersichtlich machen können. Nach Chan [Cha06] sind Star Glyphs nützlich zur Visualisierung multivariater Datensätze angemessener Größe, bei großer Anzahl zu visualisierender Datenobjekte wird die Ansicht jedoch überladen.

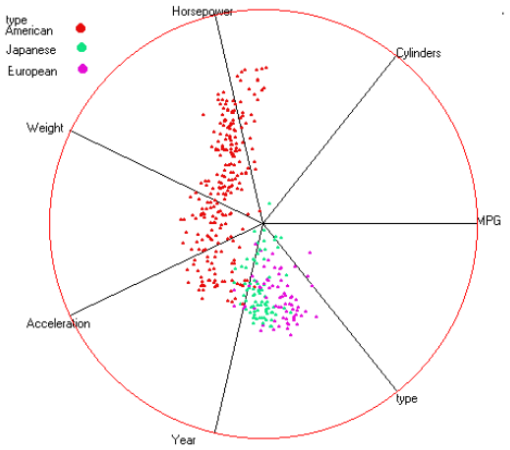

Abbildung 3.9.: Radial Coordinates [Hof77]

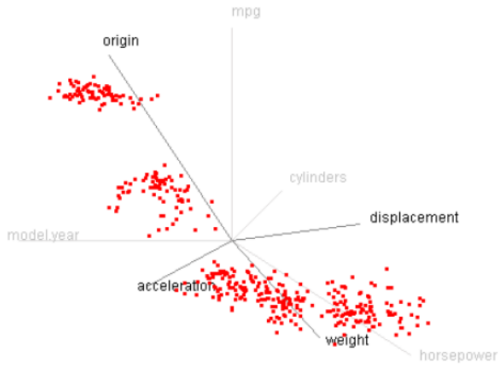

Abbildung 3.10.: Star Coordinates [Kan00]

3.2.9. Star Coordinates/Radial Coordinates

Star Coordinates (siehe Abbildung 3.10) setzen sich wie Star Plots aus im Kreis angeordneten Achsen zusammen, die von innen nach außen verlaufen. Im Gegensatz zu Star Plots werden die Werte in den Star Coordinates nicht auf den Achsen, sondern im Raum zwischen jeweils zwei benachbarten Achsen eingetragen. Dadurch erhält man im Kreis angeordnete Scatter Plot-ähnliche Diagramme, die jeweils den Zusammenhang zwischen den zwei Achsen beschreiben. Star Coordinates eigenen sich gut für die Darstellung multivariater Daten, da es pro Variable eine Achse gibt. Für zeitabhängige Daten ist diese Darstellung nicht gut geeignet. Man könnte eine der Achsen als Zeitachse verwenden, aber dann könnte man nur eine weitere Variable in deren Abhängigkeit darstellen und die anderen nicht. Große Datenmengen können einerseits in den Punktdarstellungen helfen, Muster herauszustellen. Andererseits führen viele Variablenachsen zu sehr schmalen Darstellungen, auf denen man nicht mehr viel erkennen kann. Dafür sind Star Coordinates also nur bedingt geeignet. Star Coordinates sind nützlich, um einen Einblick in hierarchisch gruppierte Datensets zu bekommen oder für die multidimensionale Analyse für Entscheidungsfällungen [Cha06].

3.2.10. Line Graph Explorer

Der Line Graph Explorer (siehe Abbildung 3.11) ist ein Liniendiagramm-Verwaltungssystem, das es ermöglicht, verschiedene Liniendiagramme zu vergleichen. Dabei werden die Liniendiagramme nicht in ihrer ursprünglichen Form dargestellt werden, sondern als waagerechter Balken, indem die Werte nicht über ihre Position im Diagramm, sondern über eine Farbkodierung dargestellt werden. So kann man viele Liniendiagramme untereinander darstellen, ohne, dass sie sich überdecken. Für die Visualisierung großer Datenmengen ist diese Darstellung wesentlich besser geeignet, als Liniendiagramme in ihrer ursprünglichen Form, da man auch

bei sehr vielen Linien keine Probleme mit sich überdeckenden Linien bekommt. Ähnlich wie in einer Pixelmap könnten die Linien entsprechend dünn dargestellt werden, wodurch sehr viele Liniendiagramme auf den Bildschirm passen. Für die Darstellung multivariater Daten müsste man für jede Variable ein Liniendiagramm erstellen. Das funktioniert nur, wenn alle Liniendiagramme die gleiche Achsenmetrik besitzen. Des Weiteren müsste man die Linien beschriften, um unterscheiden zu können, welche Variable in einer Linie dargestellt wird. Für die Darstellung von zeitlichen Daten sind die Line-Graph-Explorer-Darstellungen gut geeignet, da man in horizontaler Richtung die Zeit abtragen kann. Nach Kincaid et al. [KL06] eignet sich diese Darstellung bei der Visualisierung von zeitabhängigen Klimadaten gut, um auch in einer Übersicht auf hoher Ebene periodische Vorkommen in zeitlichen Daten sehr einfach zu erkennen.

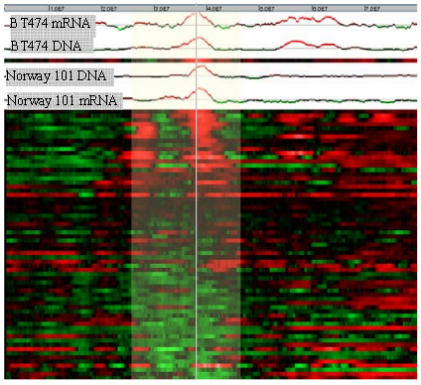

Abbildung 3.11.: Line Graph Explorer [KL06]

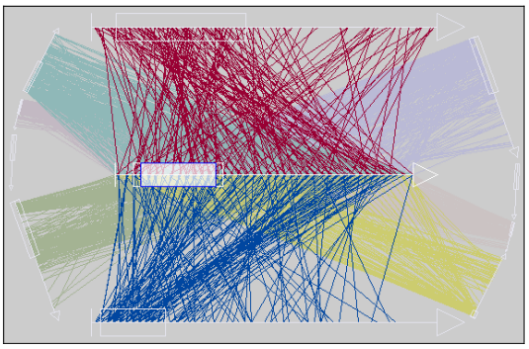

Abbildung 3.12.: TimeWheel [TAS04]

3.2.11. TimeWheel

Das TimeWheel (siehe Abbildung 3.12) setzt sich aus mehreren im Kreis angeordneten Variablenachsen und einer Zeitachse in deren Mitte zusammen. Jeder Variablenwert wird in Abhängigkeit der Zeit dargestellt, indem der entsprechende Zeitpunkt auf der Zeitachse mit dem zugehörigen Wert auf der entsprechenden Variablenachse verbunden wird. Um die Linien besser unterscheiden zu können, werden für die Linien der verschiedenen Variablen verschiedene Farben verwendet. Das TimeWheel eignet sich gut für die Visualisierung zeitabhängiger Daten, da die Zeitachse im Mittelpunkt steht und alle Variablen in Abhängigkeit dieser Achse eingetragen werden. Das TimeWheel eignet sich durch die Variablenachsen auch für die Darstellung multivariater Daten gut, allerdings ist es schwierig, verschiedene Objekte mit mehreren Variablen darzustellen, da sich diese irgendwie unterscheiden lassen müssten. Es kann jedes Objekt mit einer anderen Farbe gezeichnet werden, dann könnten die Variablen aber nicht mehr durch Farben unterschieden werden. Bei größeren Datenmengen ergeben sich sehr viele Linien, die sehr schwer zu unterscheiden sind und sich überlagern können. Durch viele Variablen erhält man außerdem entweder einen sehr großen Kreis, indem die Variablenachsen einen großen Abstand zur Zeitachse haben und es schwerer wird, den

Linien zu folgen, oder man erhält sehr kurze Achsen, auf denen es schwerer wird, die Werte zu unterscheiden und auf denen sich erst recht Kanten überlagern können. Daher ist das TimeWheel für große Datenmengen nicht gut geeignet. Laut Tominski et al. [TAS04] ist die TimeWheel-Visualisierung leicht zu verstehen. Für mehr als fünfzehn Achsen, also fünfzehn verschiedene Attribute eines Datensatzes, ist die Visualisierung jedoch nicht geeignet. Bei mehr als zehn Attributen wird die Extraktion von auffälligen Eigenschaften schon schwierig. TimeWheels scheinen aber insgesamt gut geeignet um verschiedene Variablen und deren zeitliche Abhängigkeiten zu vergleichen.

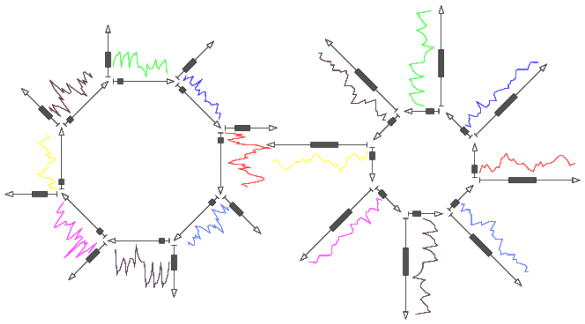

Abbildung 3.13.: MultiCombs [TAS04]

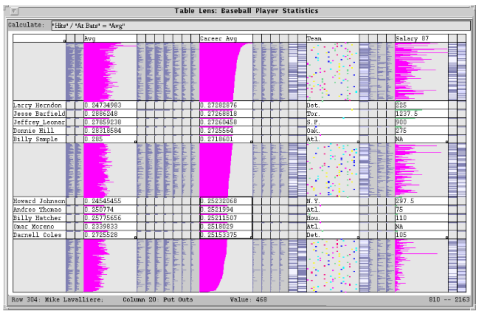

Abbildung 3.14.: Table Lens [RC94]

3.2.12. MultiCombs

In den MultiCombs (siehe Abbildung 3.13) werden, wie beim TimeWheel, mehrere Variablen in Abhängigkeit der Zeit dargestellt. Hier verlaufen die Variablenachsen nicht im Kreis, sondern von innen nach außen, und besitzen je eine eigene Zeitachse in Kreisrichtung. Dadurch erhält man mehrere, im Kreis angeordnete, Liniendiagramme mit zwei Achsen. Alternativ können die Zeitachsen von innen nach außen verlaufen und die Variablenachsen im Kreis. In der Mitte des Kreises wird in beiden Fällen eine Zusammenfassung oder Grobübersicht über die Daten angezeigt. Auch MultiCombs sind sehr gut für die Darstellung zeitlicher Daten geeignet, da alle Variablen in Abhängigkeit der Zeit visualisiert werden. Multivariate Daten lassen sich auch sehr gut darstellen, da es für jede Variable eine Achse gibt. Im Gegensatz zum TimeWheel überschneiden sich die Linien der verschiedenen Variablen nicht und man könnte mehrere Objekte darstellen, indem man nicht pro Variable, sondern pro Objekt eine Farbe wählt, da sich die Variablen durch ihre Anordnung im Raum dennoch gut unterscheiden lassen. Bei großen Datenmengen stößt man auf ähnliche Probleme, wie beim TimeWheel: Die Kreisform nutzt den Bildschirmplatz nicht optimal und bei vielen Variablen werden die einzelnen Diagramme sehr klein. Wie auch das TimeWheel sind MultiCombs nach Tominski et al. [TAS04] schnell zu verstehen, für die Analyse von Datensätzen mit mehr als zehn Attributen jedoch nicht mehr gut geeignet. Bei weniger Achsen sind auch MultiCombs für die Analyse mehrerer Variablen und ihrer zeitlichen Abhängigkeiten gut geeignet.

3.2.13. Table Lens

Die Table Lens-Visualisierung (siehe Abbildung 3.14) wurde speziell zur Darstellung tabellarischer Daten entwickelt. Die Daten werden in einer Tabelle mit einer Spalte pro Variable und Zeilen pro Objekt dargestellt. Statt in Textform werden die Werte verkleinert in Balkendiagrammen oder Punkten für die Werte der Variablen dargestellt. Beispiel: ein Punkt weiter rechts in der Spalte symbolisiert einen höheren Wert, ein Punkt weiter links einen niedrigeren. Einzelne Zeilen und Spalten können, ähnlich wie mit einer Lupe, vergrößert dargestellt werden. Dadurch ergibt sich eine Fokus + Kontext-Ansicht (siehe Kapitel 2.1). Diese Darstellung eignet sich gut für multivariate Daten, da es für jede Variable eine Spalte gibt. Zeitliche Daten können von oben nach unten dargestellt werden, dann wird es jedoch schwierig mehrere Objekte darzustellen. Für große Datenmengen ist die Visualisierung bedingt geeignet. Es lassen sich wesentlich mehr Daten darstellen, als in einer klassischen Tabelle in Textform, da die Spalten und Zeilen kleiner dargestellt werden können. Doch irgendwann stößt auch diese Darstellung an ihre Grenzen und Spalten könnten beispielsweise so schmal werden, dass sich die Werte in der Spalte nicht mehr erkennen und voneinander unterscheiden lassen. Laut Chan [Cha06] besteht der Vorteil der Table Lens-Visualisierung darin, dass es auf dem Konzept einer Tabelle beruht, welches allgemein bekannt ist. Sie ermöglicht dem Benutzer Relationen in den Daten zu erkennen, Tendenzen zu analysieren, mögliche Zusammenhänge zwischen den Daten herzustellen und auf einfache Art und Weise ganze Datensätze zu analysieren und bearbeiten.

3.3. Vergleich

In der folgenden Tabelle werden die oben aufgeführten Visualisierungen nach ihrer Eignung für diese Arbeit verglichen. Dabei bedeutet „+“, dass die Visualisierung gut für die entsprechenden Daten geeignet ist, „0“, dass die Visualisierung bedingt für diese Daten geeignet ist, das bedeutet, man kann diese Daten in dieser Visualisierung darstellen, es gehen dadurch aber Informationen verloren, und „-“, dass sich diese Daten nicht in dieser Visualisierung darstellen lassen.

Visualisierung	**multivariate Daten**	**zeitabhängige Daten**	**große Datenmengen**
Scatter Plot	-	0	+
Scatterplot Matrix	+	+	0
Parallele Koordinaten	+	0	-
GANTT/LifeLines	-	-	-
Theme River	-	+	-
Pixelmap	-	+	+
Spiral Graphs	-	+	0
Line Graph Explorer	-	+	+
Star Plots	+	0	0
Star Coordinates	+	0	0
TimeWheel	+	+	-
MultiCombs	+	+	0
Table Lens	+	0	+

Tabelle 3.1.: Vergleich der Visualisierungsmethoden

Die Tabelle zeigt, dass keine der Visualisierungen allen Ansprüchen gerecht wird. Einige sind beispielsweise für die Visualisierung multivariater Daten gut geeignet, aber nicht für zeitabhängige Daten, andere wiederum für große Datenmengen, aber nicht für multivariate Daten. Eine einzige Visualisierung reicht nicht aus, um solche komplexen Datensätze zu visualisieren. Das Ziel dieser Arbeit ist daher, dem Benutzer mehrere Visualisierungen zur Verfügung zu stellen, die unterschiedliche Aspekte des vorliegenden Datensatzes visualisieren.

4. Verwandte Arbeiten

In den letzten Jahren wurde viel im Bereich der Analyse von Leistungsdaten geforscht und entwickelt. Die Ziele dieser Arbeit umfassen die Visualisierung vieler, multivariater und zeitabhängiger Daten mithilfe visuell skalierbarer Visualisierungen, welche anwendungsbezogene Leistungsdaten mit infrastrukturellen Daten kombinieren. In diesem Kapitel werden bestehende Ansätze und Arbeiten zur Visualisierung von Supercomputer-Leistungsdaten beschrieben und mit den Zielen dieser Arbeit verglichen.

4.1. VAMPIR

VAMPIR wird seit 1996 am Forschungszentrum Jülich und an der technischen Universität Dresden entwickelt [BWNH01]. Während der Ausführung eines Programms können mithilfe von VampirTrace anwendungsbezogene Leistungsdaten gesammelt werden, beispielsweise Informationen über Benutzerevents, MPI-Events, zeitliche und örtliche Informationen, wie zum Beispiel Informationen über den Cluster-Knoten oder Thread [MKJ+07]. Die so gesammelten Informationen können von Vampir gelesen und visualisiert werden [GWT]. Abbildung 4.1 zeigt das Hauptfenster von VAMPIR mit verschiedenen geöffneten Visualisierungen. Das Master Timeline Chart (Abbildung 4.1 C) und das Process Timeline Chart (Abbildung 4.2) visualisieren Informationen über Funktionen, Kommunikation und Synchronisierungen von einzelnen oder mehreren Prozessen über die Zeit. Das Counter Timeline Chart und das Performance Radar Chart (Abbildungen 4.3 und 4.4) visualisieren Daten von Zählern, die über einen Zeitraum bestimmte Events gezählt haben, zum Beispiel Cache Misses, die beschreiben, das etwas nicht im Cache gefunden wurde. Während das Counter Timeline Chart einen Zähler für einen bestimmten Prozess als Liniendiagramm visualisiert, wird im Performance Radar Chart ein einzelner Zähler in Form von Balken für alle Prozesse untereinander dargestellt. Hier werden die Werte des Zählers durch ihre Farbgebung visualisiert. Master Timeline Chart und Performance Radar Chart können mit benutzerdefinierter Transparenz übereinandergelegt und verglichen werden. Für das Performance Radar Chart kann der Benutzer auch eigene Metriken definieren. Die sogenannten Function Summary Charts (siehe Abbildung 4.5) und Process Summary Charts bestehen aus Kuchen- und Balkendiagrammen, die anzeigen, wie viel der Zeit ein Prozess für eine bestimmte Funktion oder Funktionsgruppe aufgewendet hat. Die Message Summary Charts stellen mithilfe von Balkendiagrammen verschiedene Metriken zu gesendeten Nachrichten dar, beispielsweise die akkumulierte Nachrichtengröße oder die Anzahl der Nachrichten. Ähnlich visualisiert die IO Summary Informationen, wie die Anzahl der IO-Operationen oder die aggregierte IO-Transaktionszeit. Die Communication Matrix

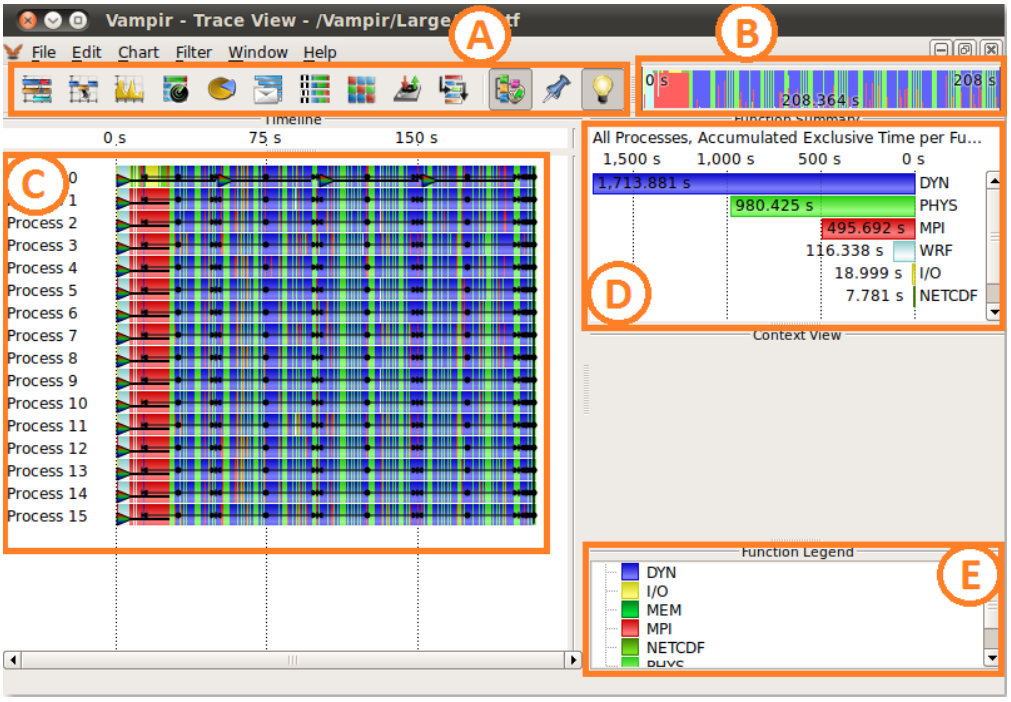

Abbildung 4.1.: Hauptfenster von VAMPIR [GWT]
A: Charts Toolbar zur Auswahl der Visualisierung
B: Zoom Toolbar für die Auswahl eines Vergrößerungsbereichs
C: Master Timeline Chart
D: Function Summary Chart
E: Legende für die Farben der Funktionen

View (siehe Abbildung 4.6) stellt mit einer Adjazenzmatrix die Kommunikation zwischen Prozessen dar.

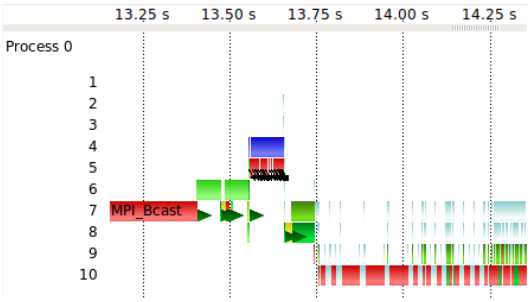

Abbildung 4.2.: Ausschnitt aus einem Process Timeline Chart [GWT]

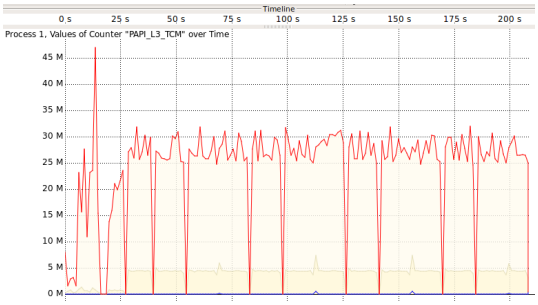

Abbildung 4.3.: Counter Timeline Chart aus VAMPIR [GWT]

Der Call Tree bildet die Vererbungshierarchie aller aufgezeichneten Funktionen als aufklappbaren Baum ab, ähnlich dem Verzeichnisbaum des Windows-Explorers. Das Tool Vampir

bietet Interaktionsmöglichkeiten, die über die Maus gesteuert werden. In der Charts Toolbar (Abbildung 4.1 A) kann die anzuzeigende Visualisierung angeklickt werden. In der Zoom Toolbar (Abbildung 4.1 B), die einen groben Überblick über den zeitlichen Verlauf der Daten gibt, kann mit der Maus oder dem Mausrad ein Zeitintervall markiert werden, um in dieses hineinzuzoomen. Auch direkt in der Visualisierung kann ein Zeitbereich und/oder, je nach Visualisierung, eine Teilmenge der visualisierten Prozesse oder anderer Variablen ausgewählt werden. VAMPIR verwendet Multiple Coordinated Views (siehe Kapitel 2.1). Durch Zoomen in einer Visualisierung werden beispielsweise andere Visualisierungen automatisch angepasst. Ein Kontextmenü bietet die Möglichkeit, die Zoom-Einstellungen wieder zurückzusetzen, die Sortierung zu ändern, oder die Metrik für ein Diagramm zu ändern. Dabei können auch eigene Metriken definiert werden, indem beispielsweise zwei bestehende Metriken kombiniert werden. Mehrere Visualisierungen können neben- und untereinander in einem Fenster frei angeordnet und so besser verglichen werden. Durch An- und Abdocken der Visualisierungen können diese auch in mehrere Fenster ausgelagert werden. Über diverse Filter können die anzuzeigenden Daten gefiltert werden, um die Menge der angezeigten Daten zu reduzieren. Über die Compare View können außerdem Trace-Dateien miteinander verglichen werden.

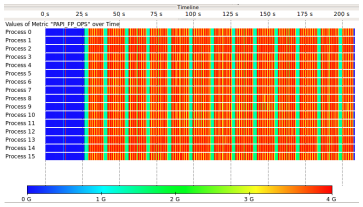

Abbildung 4.4.: Performance Radar Chart aus VAMPIR [GWT]

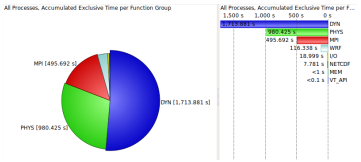

Abbildung 4.5.: Function Summary Chart aus VAMPIR [GWT]

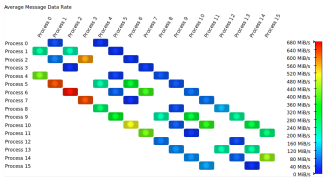

Abbildung 4.6.: Communication Matrix aus VAMPIR [GWT]

Vampir bietet dem Benutzer durch verschiedene Visualisierungen eine gute Übersicht über anwendungsorientierte Leistungsdaten eines Systems. Durch die Verwendung von Multiple Coordinated Views wird das Vergleichen von Daten in verschiedenen Visualisierungen erleichtert. Allerdings passen die vorhandenen Visualisierungen für große Datenmengen nicht mehr auf den Bildschirm. In Timeline Charts muss beispielsweise viel gescrollt werden, um alle Prozesse eines Systems zu betrachten, wodurch das Vergleichen verschiedener Prozesse erschwert wird, wenn diese nicht zufällig nah beieinander stehen. Im Zuge dieser Arbeit sollen deshalb Visualisierungen implementiert werden, die auch für große Datenmengen auf dem Bildschirm skalieren und des Weiteren anwendungsorientierte Daten mit infrastrukturellen Daten kombinieren, so dass der Nutzer diese vergleichen und in Relation zueinander stellen kann.

4.2. Nagios

Nagios [Nag] ist ein Open Source IT-Monitoring-System von Nagios Enterprises mit dem die gesamte Infrastruktur eines Systems visualisiert werden kann, um Probleme in Anwendungen, Prozessen und Diensten zu erkennen und den Benutzer darüber zu benachrichtigen. Mit Hilfe von Script-APIs sammelt die Software Daten zu Anwendungen, Diensten, Betriebssystemen, Netzwerkprotokollen, Systemmetriken und Infrastruktur-Komponenten.

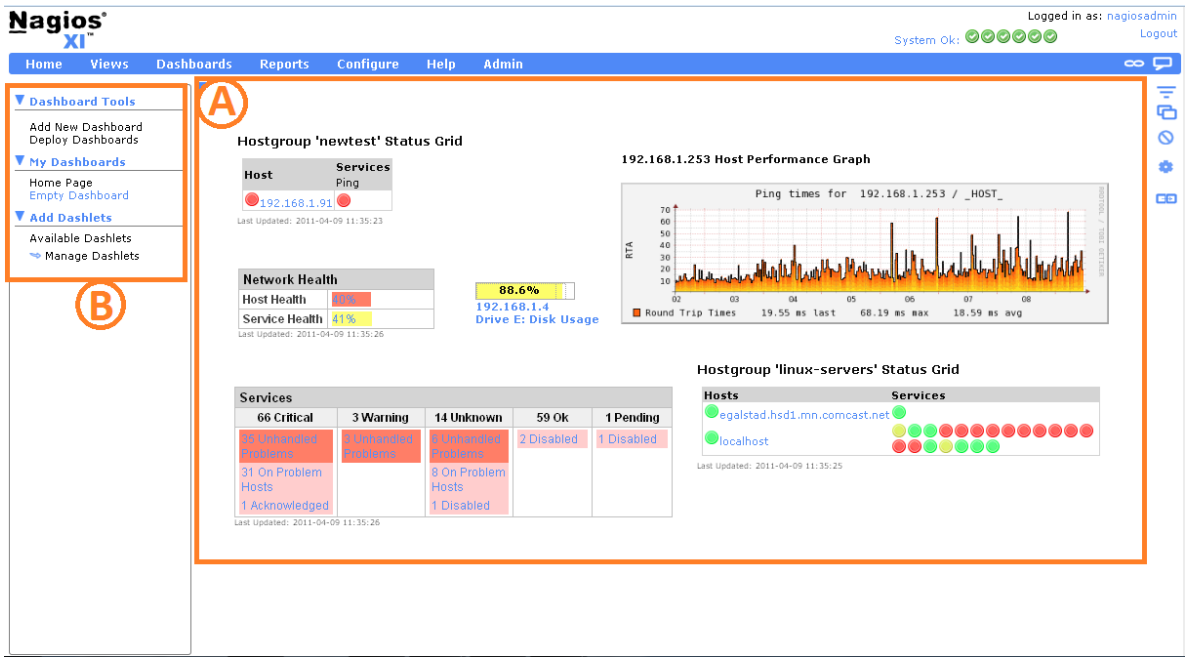

Abbildung 4.7.: Dashboard aus Nagios XI [Nag]
A: Dashboard mit Übersicht über verschiedene Visualisierungen in Nagios
B: Menü zur Verwaltung der Dashboards und Dashboard-Elemente

Nagios Core steht kostenlos zur Verfügung und enthält Kernfunktionalitäten zum Monitoring der Daten sowie ein einfaches Webinterface. Die Erweiterung Nagios XI ist käuflich erwerbbar und bietet viele Erweiterungen zu Nagios Core, zum Beispiel benutzerdefinierte Dashboards, Ansichten und Präferenzen. Nagios XI kann auch um benutzerdefinierte Komponenten und Plugins erweitert werden. Die Darstellung von Daten mit Nagios basiert zu einem großen Teil auf Text und Tabellen. An Visualisierungen stehen Histogramme, Knoten-Kanten-Diagramme, Linien- und Flächendiagramme zur Verfügung. Die Erweiterung Nagios XI bietet mehr Darstellungen, zum Beispiel eine HeatMap und einen ThemeRiver (siehe Abbildung 4.8), die die Menge der gemeldeten Alarme visualisieren. Abbildung 4.7 A zeigt das Dashboard von Nagios XI mit verschiedenen Visualisierungen. Die Darstellungen in Nagios sind statische Bilder, die einmal erzeugt werden. Es gibt daher keinerlei Möglichkeiten, mit ihnen zu interagieren. Um zum Beispiel die zu visualisierende Zeitspanne einzuschränken, müssen jedes mal neue Visualisierungen erstellt werden. Über das Menü in Abbildung 4.7 B können das

Dashboard und die Dashboard-Elemente verwaltet werden und es kann ausgewählt werden, was dargestellt werden soll. Es kann beispielsweise der Status verschiedener Komponenten angezeigt werden oder Berichte über Alarme, Meldungen, etc. die im System aufgetreten sind. Nagios stellt die Daten automatisch in entsprechenden Visualisierungen oder Tabellen dar. Es kann nicht ausgewählt werden, welche Darstellung für die Daten verwendet werden soll. Über Drop-Down-Menüs über der Visualisierung können die Daten teilweise gefiltert werden, indem zum Beispiel die visualisierte Zeitperiode ausgewählt wird.

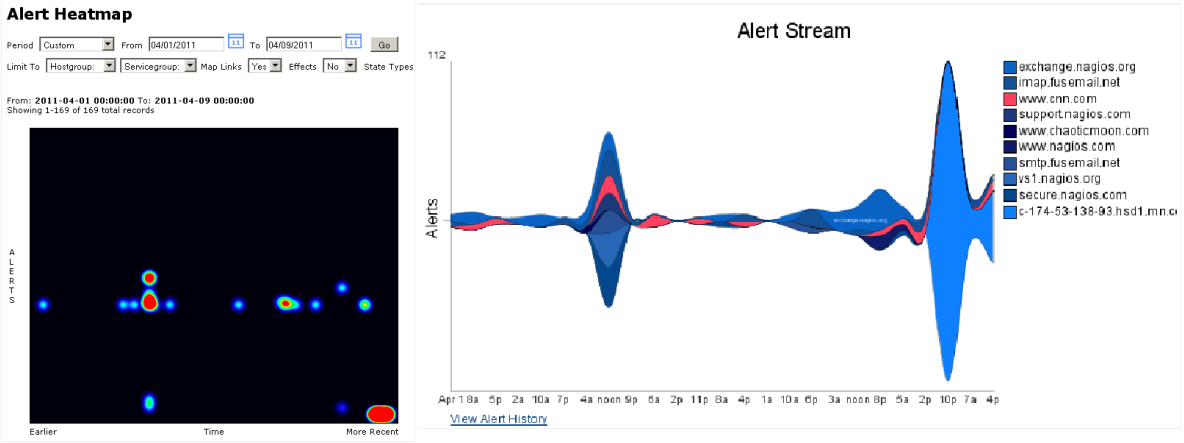

Abbildung 4.8.: HeatMap (links) und Alert Stream (rechts) aus Nagios XI [Nag]

Die Tabellen und Textdarstellungen der Daten in Nagios Core beanspruchen sehr viel Platz auf dem Bildschirm und es ist schwer, in sehr großen Tabellen Daten zu vergleichen oder Auffälligkeiten zu entdecken, ohne die Tabelle Zeile für Zeile durchzugehen. Die vorhandenen Diagramme skalieren auch nicht für große Datenmengen. In Liniendiagrammen beispielsweise sind kleine Details schwer erkennbar, wenn eine große Zeitspanne dargestellt wird. Nagios XI bietet mehr Visualisierungen an, die größere Datenmengen überschaubarer darstellen können. Was zur effektiven Exploration vieler multivariater Daten fehlt, sind geeignete Interaktionstechniken. Die erstellten Diagramme sind statische Bilder mit denen nicht direkt interagiert werden kann, um beispielsweise weitere Detailinformationen zu einem Prozess abzurufen. Nagios eignet sich, um schnell und ohne Aufwand wenige Leistungsdaten zu visualisieren. In dieser Arbeit sollen dagegen Visualisierungen für multivariate Daten und große Datenmengen miteinander kombiniert werden, damit der Benutzer sowohl eine Übersicht über die gesammelte Datenmenge erhält, als auch verschiedene Metriken in einer Visualisierung miteinander vergleichen kann. Des Weiteren soll der Benutzer durch geeignete Interaktionstechniken die Datenmenge explorieren und Details abrufen können.

4.3. Munin

Munin [SSDDS+a] ist ein kostenloses Tool zur Überwachung der Ressourcen in Rechnernetzwerken. Es soll helfen, Tendenzen zu erkennen und mögliche Ursachen für Performance-Probleme aufzudecken. Der zentrale Server Munin master wird auf einem Rechner installiert, welcher die anderen Rechner eines Netzwerks überwachen und Daten sammeln soll. Auf den überwachten Rechnern muss dafür der Netzwerkdienst Munin node installiert sein, der vom Munin master kontaktiert wird, um Daten zu erhalten. Die gesammelten Daten werden durch verschiedene Grafiken in einem Webinterface dargestellt. Um dem Benutzer so wenig Aufwand wie möglich zu machen, gibt es eine Standardinstallation, die bereits viele Grafiken zur Verfügung stellt. Munin verwendet das RRDtool, ein Open Source-Tool zum Aufzeichnen und Visualisieren von Zeitseriendaten [Oet13].

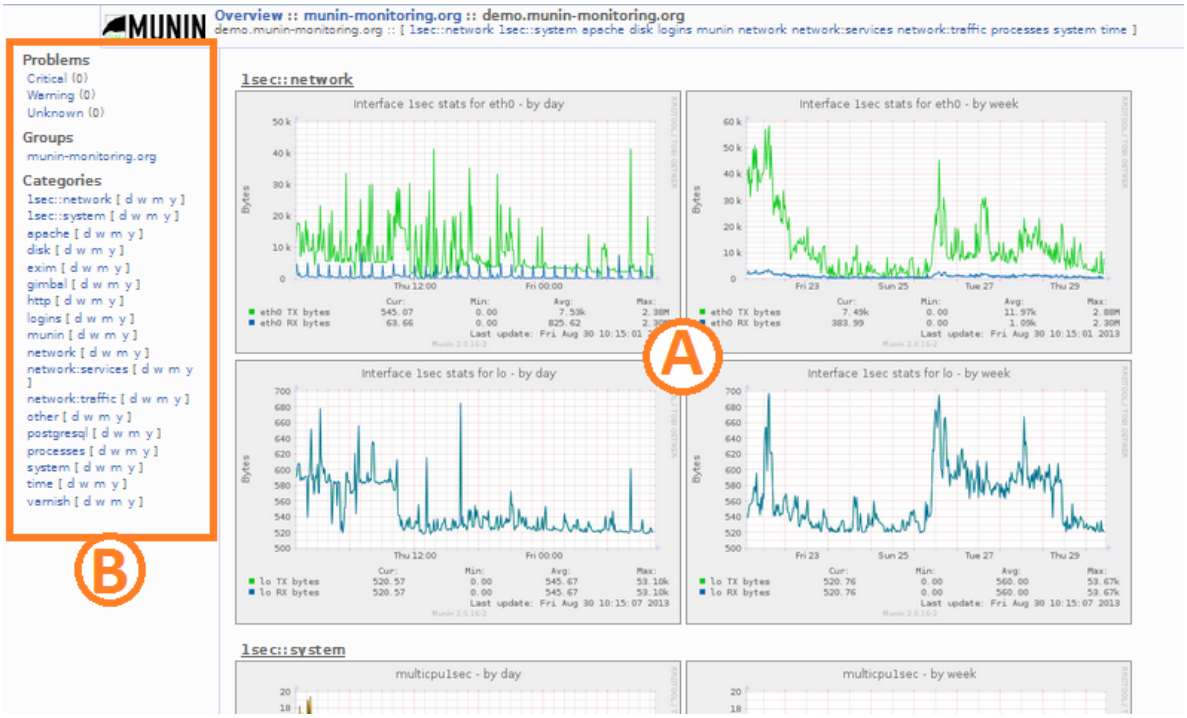

Abbildung 4.9.: Munin [SSDDS+b]
A: Übersicht über Visualisierungen in Munin
B: Menü zur Auswahl der visualisierten Attribute

Munins Visualisierungen werden über ein Webinterface dargestellt (siehe Abbildung 4.9). Munin bietet einfache Linien- und Flächendiagramme für die Darstellung der Daten an. Diese können für verschiedene Zeitintervalle angezeigt werden, zum Beispiel für einen Tag oder eine Stunde. Auch bei Munin sind die Visualisierungen nur statische Bilder, mit denen nicht direkt interagiert werden kann. Es gibt jedoch die Möglichkeit, die Darstellung durch die Wahl der angezeigten Zeitintervalle und Attribute (siehe Abbildung 4.9 B) zu beeinflussen.

Ein großer Vorteil von Munin ist, dass es durch das Installationsprogramm ohne großen Einrichtungsaufwand genutzt werden kann. So kann man einfach eine grobe Übersicht über die Ressourcen erhalten. Durch die stark beschränkte Wahl an Visualisierungen ist Munin jedoch nicht gut für die Darstellung multivariater Daten oder sehr großer Datenmengen geeignet. Die Diagramme werden für sehr viele Daten durch sich überdeckende Linien schnell sehr unübersichtlich und für multivariate Daten müssten mehrere Visualisierungen erstellt werden, was einerseits viel Platz beansprucht und andererseits das Vergleichen der Daten erheblich erschwert. Die Diagramme werden neben- und untereinander dargestellt, wodurch der Benutzer viel scrollen muss, um zwei Darstellungen zu vergleichen. In dieser Arbeit sollen, neben Visualisierungen für große Datenmengen und für mehrdimensionale Daten, geeignete Interaktionstechniken implementiert werden, die es erlauben den Datensatz zu explorieren und Detailinformationen abzurufen.

4.4. DataMeadow

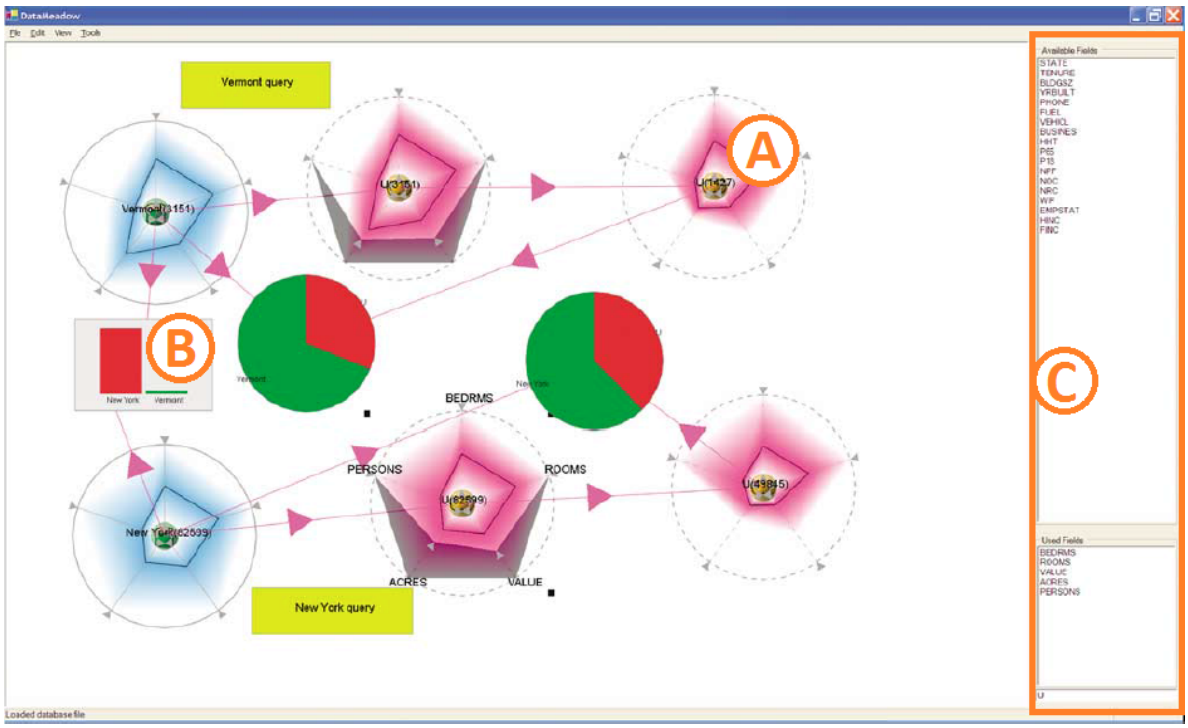

Abbildung 4.10.: DataMeadow [EST08]
A: StarGlyph
B: Balkendiagramm zum Vergleich von zwei StarGlyphs
C: Liste der verfügbaren Metriken

DataMeadow [EST08] ist ein Tool zur visuellen Analyse von mehrdimensionalen Daten. Der Hauptfokus des Tools liegt auf der Interaktion, Korrelation und dem Vergleich von multivariaten Daten aus unterschiedlichen Quellen. DataMeadow verwendet sogenannte StarGlyph-Visualisierungen, die, ähnlich wie Star Plots, ausgewählte Variablen der geladenen Daten visualisieren (siehe Abbildung 4.10 A). In einer Liste können aus verfügbaren Metriken mehrere ausgewählt werden, die in den StarGlyphs visualisiert werden sollen (siehe Abbildung 4.10 C). Durch dynamische Schieberegler an jeder Variablenachse können die Daten gefiltert werden, indem ein Bereich auf einer Variablenachse des StarGlyphs ausgewählt und ein neuer StarGlyph hinzugefügt wird, der nur die Daten in dem ausgewählten Bereich darstellt. Durch Ändern dieses Bereichs werden direkt die Auswirkungen auf die anderen Variablen sichtbar. Verschiedene StarGlyphs können verbunden und in Kuchen- oder Balkendiagrammen miteinander verglichen werden (siehe Abbildung 4.10 B). Des Weiteren können komplexe visuelle Anfragen erstellt werden, indem iterativ Teilbereiche der Daten ausgewählt und gefiltert werden und somit die Anfrage immer weiter verfeinert wird. DataMeadow ist gut auf die Visualisierung multivariater Daten ausgelegt. Durch die iterative Verfeinerung der Anfrage kann sehr tief in den zugrundeliegenden Datensatz hinein navigiert werden und sich so von einer groben Übersicht über die Daten zu einer detaillierten Darstellung ausgewählter Datenbereiche vorgearbeitet werden. Die StarGlyphs in DataMeadow werden neben- und untereinander auf dem Bildschirm und dadurch sehr klein dargestellt. Für zeitabhängige Daten muss die Zeit auf einer eigenen Achse des StarGlyphs dargestellt werden. Diese sind aber nicht so groß, dass man größere Zeitspannen übersichtlich auf einer einzelnen Achse darstellen kann. Es ist daher schwierig, einen zeitlichen Verlauf der Daten zu beobachten. In dieser Arbeit soll dem zeitliche Aspekt der Daten größere Bedeutung beigemessen werden. Dazu sollen Visualisierungsmethoden implementiert werden, die zeitabhängige Daten auch über einen großen Zeitraum übersichtlich darstellen und es ermöglichen den Verlauf verschiedener Attribute über die Zeit zu beobachten und dabei Abhängigkeiten zwischen den Attributen zu erkennen.

4.5. Tableau

Tableau basiert auf Polaris [STH08] und ist ein Tool zur Analyse von Daten, das von Tableau-Software in Seattle entwickelt wurde. Es verwendet VisQL-Statements um Daten aus einer Datenbank in Visualisierungen darzustellen. Neben Tableau Desktop, der Standardausführung von Tableau, existieren Tableau Server, die Business Intelligence-Version von Tableau, die Browser-basierte Analysemethoden verwendet, Tableau Online, die gehostete Version von Tableau Server und Tableau Public für die Veröffentlichung interaktiver Visualisierungen im Web [Tab]. Das Tool bietet neben einfachen Balken-, Linien- und Kuchendiagrammen unter anderem HeatMaps, Scatter Plots und geographische Projektionen. Die einzelnen Visualisierungen können Fenster-füllend dargestellt werden, oder es können mehrere Darstellungen nebeneinander und untereinander auf einem sogenannten Dashboard angezeigt werden, um verschiedene Aspekte der Daten miteinander zu vergleichen (Abbildung 4.11). Das Dashboard von Tableau realisiert das Prinzip „Brushing und Linking“ (siehe Kapitel 2.1). Wenn ein Bereich in einer Visualisierung markiert wird, wird dieser in den anderen Visualisierun-

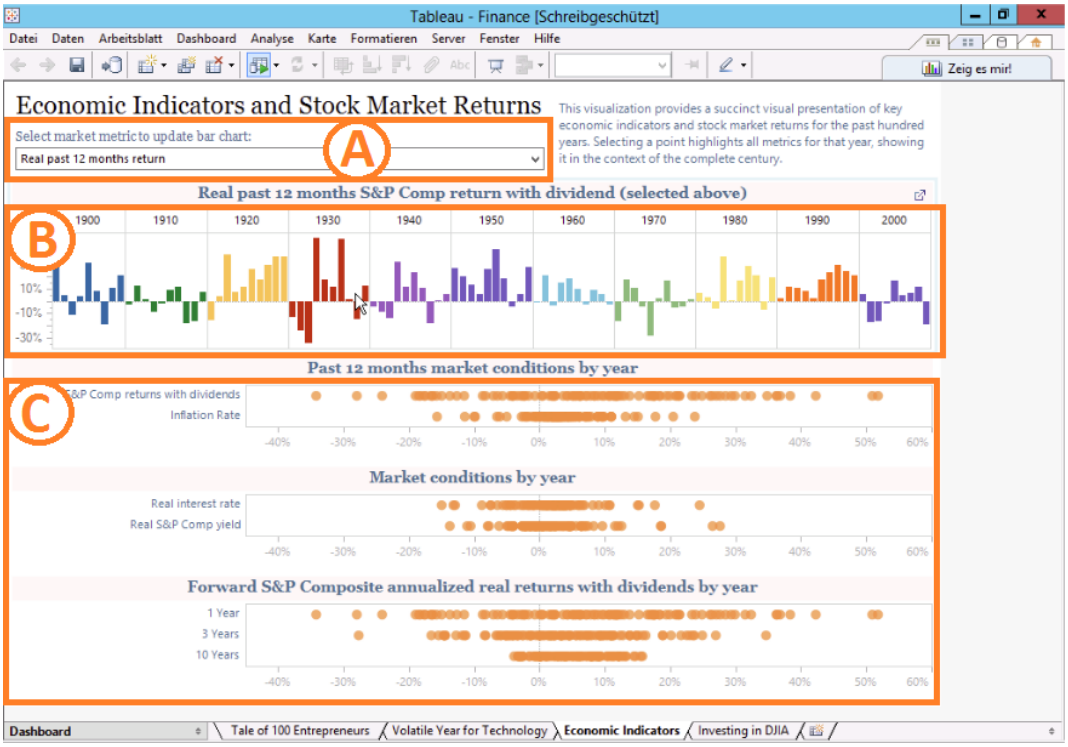

Abbildung 4.11.: Tableau Dashboard zur Visualisierung von Zusammenhängen [MSZ13]
A: Drop-Down zur Auswahl der visualisierten Metrik
B: Balkendiagramm für die ausgewählte Metrik
C: Punktdiagramme zur Visualisierung von prozentualen Verteilungen

gen auch hervorgehoben und kann so in verschiedenen Darstellungen einfacher verglichen werden [MSZ13]. Die Diagramme werden über „Drag & Drop“-Aktionen erstellt, indem verfügbare Datenbankfelder in das Fenster gezogen werden. Die Daten können gefiltert und sortiert und die Farben der Visualisierungen individuell angepasst werden. Eine Historie enthält eine Reihe von Kleinansichten mit zuletzt erstellten und betrachteten Visualisierungen und bietet damit die Möglichkeit, Schritte rückgängig zu machen. Tableau bietet viele verschiedene Diagramme, die unterschiedliche Aspekte der Daten visualisieren und über das Dashboard miteinander verglichen werden können. Die Diagramme eignen sich für die Darstellung von Zeitseriendaten und verschiedene Interaktionen ermöglichen das Explorieren der Daten. Das Tool visualisiert beliebige Datenbankeinträge und ist nicht auf die spezifischen Eigenschaften von Supercomputer-Leistungsdaten ausgelegt. Die Diagramme skalieren für die Datenmengen, die in Supercomputersystemen gesammelt werden können, nicht mehr auf dem Bildschirm, in den meisten Diagrammen kommt es dann zu Überlappungen, welche die Diagramme unübersichtlich erscheinen lassen (siehe „Visual Clutter“, Kapitel 2.1). In dieser Arbeit soll zwischen anwendungsorientierten und infrastrukturellen Daten aus Supercomputersystemen unterschieden werden und die großen Datenmengen und deren Multidimensionalität skalierbar und übersichtlich visualisiert werden.

4.6. BANKSAFE

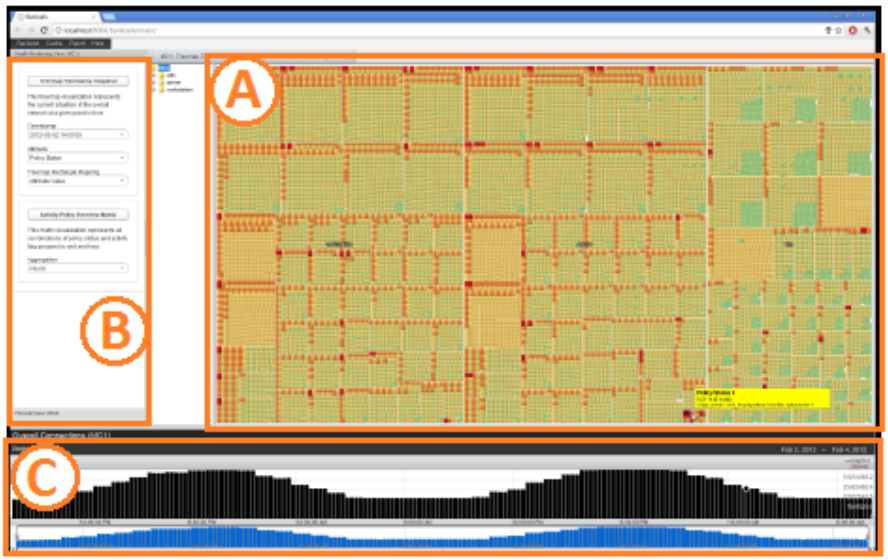

Abbildung 4.12.: Treemap Timestamp Snapshot aus BANKSAFE [FFMK12] zur Visualisierung des Status der Computer im Netzwerk zu einem bestimmten Zeitpunkt.
A: Treemap Timestamp Snapshot
B: Konfiguration der Visualisierung
C: Balkendiagramm

Abbildung 4.13.: Die Pixel-basierte Matrix aus BANKSAFE [FFMK12] repräsentiert die Anzahl der Rechner, die zu einem bestimmten Zeitpunkt eine bestimmte Kombination von Sicherheitsstufen und Aktivitätsraten haben.

BANKSAFE [FFMK12] ist ein skalierbares, verteiltes, webbasiertes Visualisierungssystem zur Analyse des Zustands von großen Computernetzwerken in Unternehmen. Um entsprechend große Datenmengen zu speichern und zu analysieren, liegt eine Cloud-basierte Datenbank zugrunde. Um Skalierbarkeit zu erzielen, verwendet BANKSAFE Google BigQuery, einen Cloud-basierten Datenbankdienst, der bereits viele Daten, wie Systemstatusüberprüfungen und IDS-Alarme, direkt importiert. BANKSAFE selbst ist eine Java Webanwendung, die durch Apache Tomcat gehostet wird und eine webbasierte grafische Benutzeroberfläche bereitstellt. Das Tool bietet verschiedene Visualisierungen für spezifische Arten von Daten an. Eine Treemap Timestamp Snapshot (siehe Abbildung 4.12 A) stellt eine Übersicht über den Status der Computer des Netzwerks zu einem bestimmten Zeitpunkt dar.

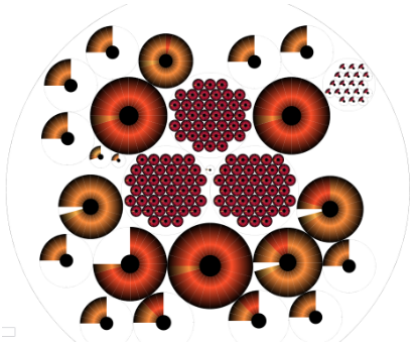

Abbildung 4.14.: Die ClockMap aus BANKSAFE [FFMK12] repräsentiert den Firewall-Status eines Netzwerkknotens über die letzten 24 Stunden.

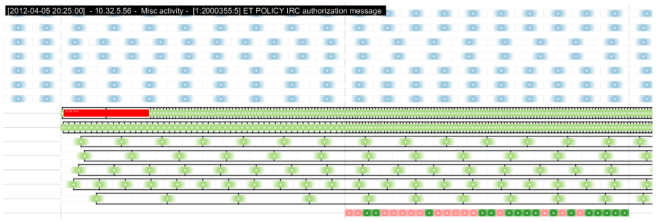

Abbildung 4.15.: Die Relaxed IDS Timeline aus BANKSAFE [FFMK12] repräsentiert IDS-Events pro Rechner über die Zeit.

Die Hierarchie der Treemap bezieht sich auf die organisatorische Struktur des Unternehmens. Eine Pixel-basierte Matrix stellt eine kompakte zeitliche Übersicht dar. In dieser Matrix können die möglichen Kombinationen aller Sicherheitsstufen und Aktivitätskategorien abgetragen werden. Über die Färbung kann die Anzahl der Rechner, auf die diese Kombination zu einem bestimmten Zeitpunkt zutrifft, visualisiert werden. Indem viele dieser Pixelmatrizen nebeneinander platziert werden, ergeben sich Muster, die den zeitlichen Verlauf sichtbar machen (siehe Abbildung 4.13). Die ClockMap (Abbildung 4.14) visualisiert den Zustand der Firewall eines Netzwerkknotens über einen 24-Stunden-Zyklus, der im Uhrzeigersinn abgetragen wird. Die Größe eines Glyphen gibt dabei die Anzahl der darunterliegenden Rechner an. In die ClockMap kann hineingezoomt werden und Glyphen können ausgeklappt werden, um darunterliegende Rechner im Einzelnen zu betrachten. Über Tooltips können weitere Informationen abgerufen werden [Fis]. Die IDS-Timeline (Abbildung 4.15) dient der Analyse von IDS-Events. Hier repräsentiert eine Zeile einen Rechner mit einer bestimmten IP-Adresse und eine Spalte eine Stunde. Die Farbe gibt die Klassifizierung des aufgetretenen Events an. In dieser Ansicht kann der Benutzer ein Event auswählen, um weitere Informationen zu erhalten und alle anderen Events dieses Typs durch Verbindungslinien hervorheben lassen. Die Pixel-basierte Matrix und die IDS-Timeline bieten auch für viele Daten eine Übersicht über den zeitlichen Verlauf. Die Darstellungen in BANKSAFE sind für bestimmte Daten implementiert. In dieser Arbeit soll ein Tool entwickelt werden, mit dem beliebige Supercomputer-Leistungsdaten importiert werden und in verschiedenen Visualisierungen betrachtet werden können.

4.7. VisTrails

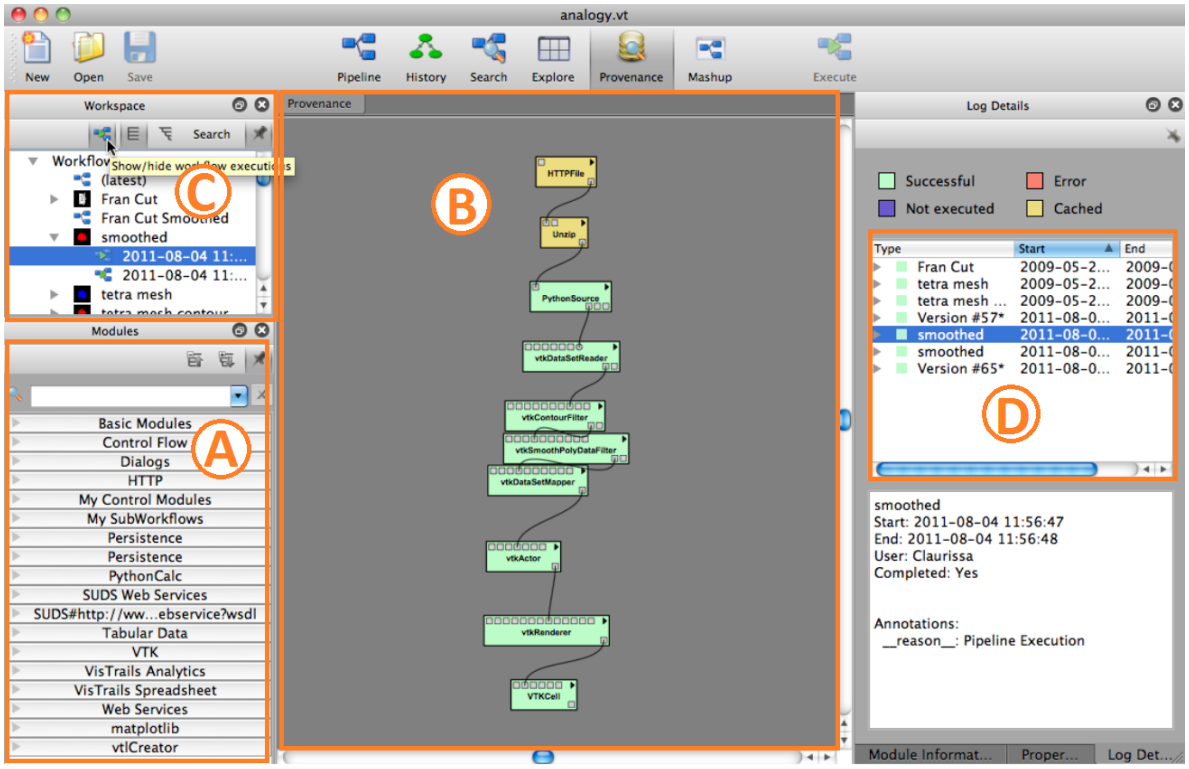

Abbildung 4.16.: Provenance-Ansicht von VisTrails [Vis]
A: Module zur Erstellung einer Pipeline
B: Pipeline des Arbeitsablaufs
C: Arbeitsumgebung
D: Provenance Browser mit den letzten Ausführungen der Pipeline

VisTrails [BCC+05] ist ein Open Source-System für die Verwaltung von wissenschaftlichen Arbeitsabläufen, welches Unterstützung bietet für Simulationen, Datenexploration und Visualisierung. Das Besondere an VisTrails ist eine detaillierte Historie über die ausgeführten Schritte und daraus gewonnenen Erkenntnisse, welche im XML-Format oder in relationalen Datenbanken gespeichert wird. Sie ermöglicht dem Benutzer, Schritte in Arbeitsabläufen rückgängig zu machen und abzuändern, um die unterschiedlichen daraus gewonnenen Ergebnisse zu vergleichen, was besonders in explorativen Aufgaben hilfreich ist. Damit zählt VisTrails zum Bereich der „Provenance Visualization“, welche sich damit beschäftigt, welche Schritte eines wissenschaftlichen Arbeitsablaufs zu einem bestimmten Ergebnis geführt haben. So können Ergebnisse reproduziert und verglichen werden und das daraus erlangte Wissen im wissenschaftlichen Umfeld wiederverwendet werden kann [DF08]. VisTrails bietet verschiedene Visualisierungen, die das Arbeiten mit und Entwickeln von Arbeitsabläufen erleichtern sollen. Eine Pipeline (Abbildung 4.16 B) stellt den momentanen Ablauf dar, bestehend aus verschiedenen Modulen und Verbindungen zwischen diesen.

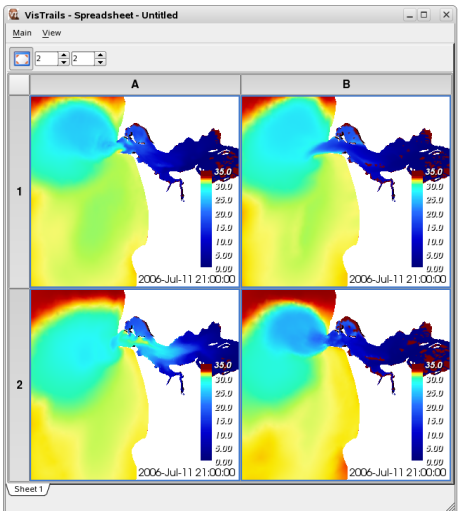

Abbildung 4.17.: Spreadsheet aus VisTrails [Vis]

Der Provenance Browser (Abbildung 4.16 D) zeigt dem Benutzer eine Historie der verschiedenen Ausführungen des Ablaufs. Das Spreadsheet (Abbildung 4.17) ist ein Fenster, in welchem verschiedene Ansichten neben- und untereinander angezeigt werden können, um zum Beispiel die Ergebnisse verschiedener Parameter miteinander zu vergleichen. VisTrails bietet viele Interaktionsmöglichkeiten, die je nach Ansicht variieren. In der Pipeline können beispielsweise Module miteinander verbunden werden, indem mit der Maus eine Verbindung vom Startmodul zum Zielmodul gezogen wird. Der Version Tree kann zwischen zwei Versionen ausgeklappt werden, um Details zu den Änderungen zwischen diesen anzusehen. Tags und Anmerkungen können zum Version Tree hinzugefügt werden und zwei Version Trees zusammengefügt werden. Im Interactive Mode des Spreadsheet kann mit den einzelnen Ansichten interagiert werden. Dabei steht für jede Ansicht eine eigene Toolbar zur Verfügung. Über den Editing Mode kann das Layout des Spreadsheets angepasst werden. Mit der Funktion Explore können verschiedene Parameter gewählt werden, für welche dann Werte generiert werden. Für jeden Wert wird der Arbeitsablauf einmal ausgeführt und ein Vergleich der Ergebnisse im Spreadsheet visualisiert. In der Provenance-Ansicht können einzelne Ausführungen ausgewählt werden, für welche die zugehörigen Module in der Pipeline markiert werden und mit der Option Mashup kann eine kleine Applikation erstellt werden, um für ein gegebenes Parameterset verschiedene Werte zu explorieren. Über den Execute-Button werden, je nach Ansicht, der momentane Ablauf, die Suche oder die Exploration für die ausgewählten Parameter ausgeführt. VisTrails bietet sehr viele verschiedene Interaktionen, die die Arbeit mit wissenschaftlichen Arbeitsabläufen erheblich erleichtern und viele Möglichkeiten bei der Exploration unterschiedlicher Parameter geben. Über das Spreadsheet können einfach unterschiedliche Ergebnisse miteinander verglichen werden und kollaboratives Arbeiten mit Arbeitsabläufen wird durch den Version Tree, bzw. dessen Merge-Funktion, erheblich erleichtert. VisTrails ist speziell auf die Arbeit mit wissenschaftlichen Arbeitsabläufen ausgerichtet, nicht auf die generelle Visualisierung von Daten. Einige der umgesetzten Konzepte, beispielsweise das Spreadsheet und die Historie, eignen sich jedoch auch für die Visualisierung und Interaktion mit Supercomputer-Leistungsdaten.

5. Entwicklung eines Prototyps

Dieses Kapitel beschreibt das Konzept und die Umsetzung eines Prototyps. Im Konzept wird zunächst die grundlegende Idee beschrieben. In der Umsetzung wird beschrieben, welche Ideen des Konzepts umgesetzt wurden und wie.

5.1. Konzept

Die Leistungsdaten eines Supercomputersystems stammen oft aus unterschiedlichen Quellen und liegen in unterschiedlichen Formaten vor. Für die Aufzeichnung und Ausgabe der Daten in eine Datei wurde Sysstat verwendet (siehe Kapitel 2.2). Der Prototyp liest die Daten durch SQL-Anfragen aus der Datei ein und generiert Visualisierungen (siehe Abbildung 5.1). Über SQL-Anfragen können ganz gezielt die gesuchten Daten abgefragt werden, indem die Anfrage schon so gestellt wird, dass nur Daten mit vorgegebenen Kriterien ausgelesen werden und diese nicht mehr in der Implementierung des Prototyps gefiltert werden müssen. Der Prototyp stellt diese Daten anschließend in verschiedenen Visualisierungen auf dem Bildschirm dar. Der Benutzer kann mit den Visualisierungen interagieren, verschiedene Visualisierungen miteinander vergleichen, seine Anfrage verfeinern und Detailinformationen abrufen.

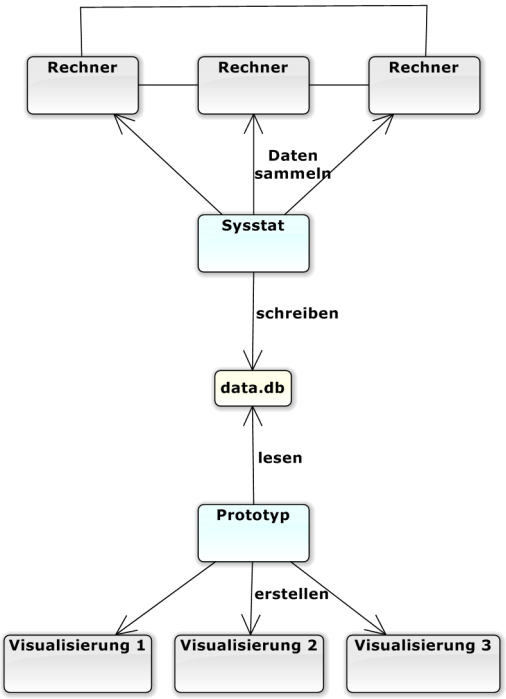

Abbildung 5.1.: Ablauf vom Datensammeln bis zur Visualisierung

5.1.1. Architekturüberlegungen

Der Prototyp soll einige ausgewählte Visualisierungen implementieren. Dabei soll es jedoch möglich sein, den Prototyp später um weitere Darstellungen zu erweitern. Die Visualisierungen sollen daher als einzelne Module implementiert werden, die vom Hauptmodul eingebunden werden.

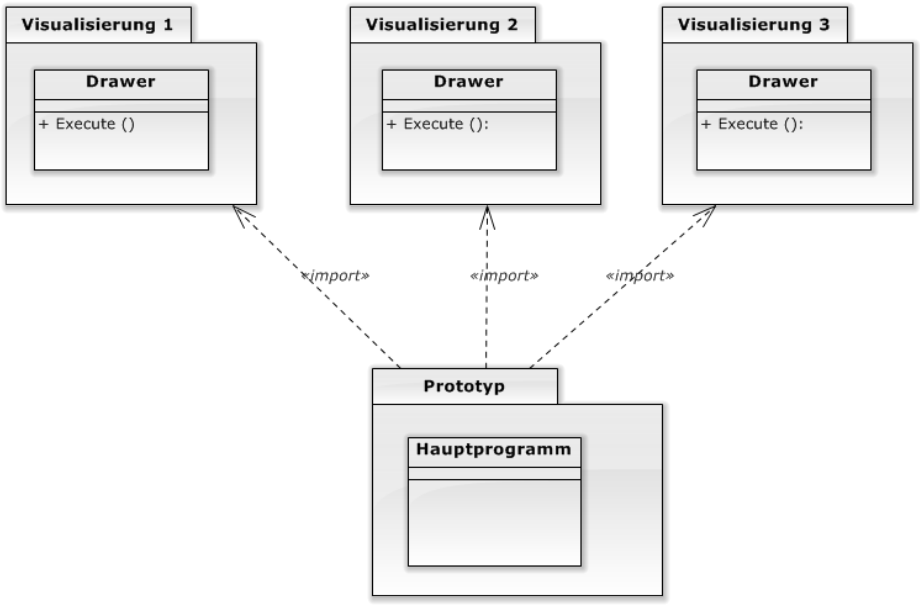

Abbildung 5.2.: Paketdiagramm für die Beziehungen zwischen dem Hauptmodul und den Visualisierungen. Das Hauptprogramm importiert die Visualisierungsprojekte.

Dadurch können sie später auch in andere Programme eingebunden und dort verwendet werden. Die Implementierung der Visualisierungen selbst soll einem einheitlichen Schema folgen, anhand dessen weitere implementiert werden können. Das garantiert, dass neue Visualisierungen ohne große Anpassungen im Hauptprogramm eingebunden werden können. Sie sollen deshalb eine Art „Drawer“-Klasse erhalten, die wiederum eine „Draw“- oder „Execute“-Methode enthält, über welche das Zeichnen der jeweiligen Darstellung ausgeführt wird. Dieser Methode sollen in allen Visualisierungen die gleichen Parameter übergeben werden, ausgenommen einzelne Parameter, die nur in bestimmten Visualisierungen berücksichtigt werden. Einer Visualisierung, die beispielsweise nur Knoten eines Supercomputersystems darstellen kann, müssen keine Parameter für Kanten übergeben werden. Das Hauptmodul stellt die Benutzeroberfläche zur Verfügung und bindet unter anderem die Visualisierungen ein (siehe Abbildung 5.2). Es behandelt sämtliche Interaktionen zwischen dem Benutzer und den Daten.

5.1.2. Interaktionstechniken

Die Implementierung verschiedener Interaktionstechniken ermöglicht erst die Analyse des gesamten Datensatzes. Da die zu analysierenden Datensätze mitunter sehr groß sind, muss die Datenmenge auf interessante Teilbereiche einschränkt werden können. Dies bezieht sich einerseits auf die visualisierte Zeitspanne, andererseits auf die Knoten und Kanten des Supercomputersystems. Dafür bieten sich Datumsfelder an, in welchen Start- und Enddatum ausgewählt werden können, oder eine Zeitleiste auf der ein Zeitbereich ausgewählt werden

kann. Für das Einschränken der Knoten und Kanten bietet sich das Markieren und Zuschneiden mit der Maus direkt in der Visualisierung an, sowie die Anwendung eines Filters, welcher nach verschiedenen Attributen filtern kann. Zu einzelnen Elementen sollen Detailinformationen abgerufen werden können, beispielsweise Name und ID eines Knotens. Dies kann zum Teil, sofern es die Visualisierung und die dargestellte Datenmenge erlauben, durch Labels in der Visualisierung abgebildet werden, oder über Tooltips eingeblendet werden. Letzteres hat den Vorteil, dass wesentlich mehr Informationen angezeigt werden können, da diese keinen zusätzlichen Platz in der Visualisierung beanspruchen, gleichzeitig werden aber Elemente der Visualisierung verdeckt. In Darstellungen mit sehr vielen Elementen ist die Implementierung einer Hervorhebung oder Markierung sinnvoll. Durch Auswählen von Elementen in einer Visualisierung sollen diese hervorgehoben werden. Hier bietet sich auch die Umsetzung des Interaktionskonzepts „Brushing und Linking“ (siehe Kapitel 2.1) an, wodurch die hervorgehobenen Elemente einer Darstellung automatisch auch in allen anderen Darstellungen hervorgehoben werden. Dadurch können diese Elemente den anderen Visualisierungen wesentlich schneller gefunden und leichter verglichen werden.

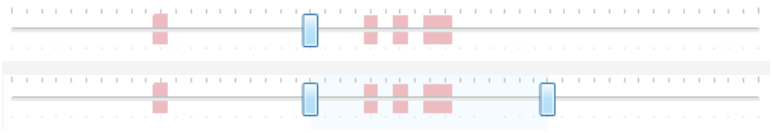

Abbildung 5.3.: Konzeptuelles Bild des Zeitsliders.
oben: für einzelne Zeitschritte
unten: für einen Zeitbereich
Die farbige Hinterlegung markiert die Vorkommen von Filterergebnissen.

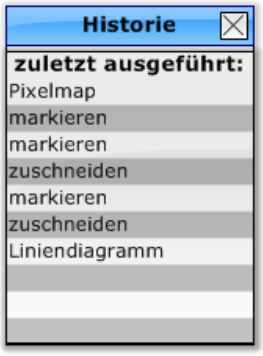

Abbildung 5.4.: Konzept einer Historie mit den zuletzt ausgeführten Interaktionen

Für die Interaktion mit der zeitlichen Dimension der Daten wurde das Konzept eines Zeitsliders entwickelt. Dabei handelt es sich um einen Schieberegler, mit dessen Hilfe je nach Visualisierung einzelne Zeitpunkte oder ein Zeitbereich ausgewählt werden können. Durch Verschieben des Schiebereglers können entweder die einzelnen Schritte durchlaufen oder die Zeitspanne verschoben werden, während die Darstellung dynamisch angepasst wird. Alternativ kann der Zeitslider automatisch abgespielt werden. Er könnte des Weiteren mit den Filtern kombiniert werden, indem die Vorkommen der Filterergebnisse auf dem Zeitslider farbig hinterlegt werden. So kann nach interessanten Attributen gefiltert werden und auf dem Zeitslider angezeigt werden, in welchen Bereichen diese auftraten. Abbildung 5.3 zeigt das Konzept des Zeitsliders mit Bereichsauswahl und farbiger Unterlegung. Bei der Arbeit mit vielen Daten ist eine Möglichkeit, Zwischenschritte zu speichern und rückgängig zu machen, oftmals sehr hilfreich. Durch das Speichern des analysierten Standes kann der Benutzer zu einem anderen Zeitpunkt weiterarbeiten, ohne nochmal von vorne beginnen zu müssen.

Des Weiteren kann der gespeicherte Stand an verschiedene Nutzer weitergegeben werden, die parallel daran weiterarbeiten können. Durch eine Art Historie der zuletzt ausgeführten Schritte, wie sie Programme wie Photoshop verwenden [Obe], muss sich der Benutzer nicht genau erinnern, welche Schritte er zuvor ausgeführt hat. Abbildung 5.4 zeigt, wie so eine Historie aussehen kann. Er kann verschiedene Interaktionen mit den Daten ausprobieren und falls diese nicht zum gewünschten Ergebnis führen, zum vorherigen Schritt zurückspringen und etwas anderes versuchen.

5.1.3. Benutzerschnittstelle

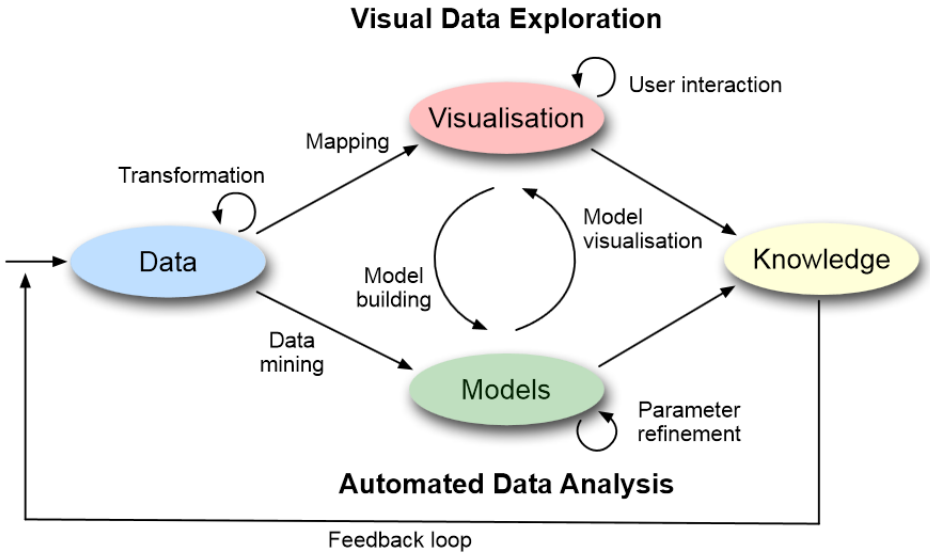

Abbildung 5.5.: Visual Analytics Process [KKEM10]

Der Aufbau der Benutzerschnittstelle orientiert sich an den Prinzipien und Richtlinien, die im Nachfolgenden beschrieben werden. Die Benutzeroberfläche soll außerdem möglichst übersichtlich und selbsterklärend sein. Für die Visualisierung soll möglichst viel Platz eingeplant werden, damit möglichst viele Daten in der Anwendung dargestellt werden können. Die Steuerung der Anwendung soll daher, sofern es der Umfang erlaubt, ausschließlich über eine Werkzeugleiste am oberen Bildschirmrand und Kontextmenüs stattfinden. Es sollen mehrere Fenster geöffnet werden können, damit mehrere Visualisierungen gleichzeitig angezeigt und miteinander verglichen werden können. Die Konzeption der Benutzerschnittstelle orientiert sich in erster Linie am „Information Seeking Mantra“ von Ben Shneiderman [Shn96] und dem „Visual Analytics Process“ von Keim et al. [KKEM10]. Sie soll dem Benutzer ermöglichen, zunächst einen Überblick über die große Datenmenge zu bekommen und sich dann Schritt für Schritt zu detaillierteren Darstellungen von Teilmengen der Daten vor zuarbeiten. Die Gestaltung der Benutzeroberfläche orientiert sich des Weiteren an den „Eight Golden Rules of Interface Design“ von Ben Shneiderman [SP10].

Information Seeking Mantra [Shn96]

1. Overview first
   Der Benutzer benötigt zunächst eine Übersicht über alle Daten. Diese Übersicht soll ihn Auffälligkeiten und Anomalien in den Daten erkennen lassen. Durch die Masse an visualisierten Daten werden in der Übersicht noch keine weiteren Details angezeigt, Auffälligkeiten in den Daten lassen sich aber erkennen.
2. Filter + Zoom
   Mit Hilfe von Filtern und Zoomen soll der Benutzer irrelevante Elemente ausblenden und die Ansicht auf interessante Elemente in den Daten einschränken können. Dies kann zum Beispiel das Hineinzoomen mittels Mausrad, ein Auswahlrechteck in der Übersicht oder das Filtern nach bestimmten Merkmalen, zum Beispiel Elementnamen oder Variablen mit bestimmten Wertüberschreitungen, sein. Durch die Verringerung der visualisierten Datenmenge können die Daten hier detailreicher dargestellt werden, um den Benutzer beispielsweise Zusammenhänge zwischen den Elementen erkennen zu lassen.
3. Details on Demand
   Nach der Auswahl und Fokussierung der interessanten Daten soll der Benutzer Detailinformationen zu einzelnen Elementen anzeigen lassen können, um zu erkennen, was für eine Anomalie genau an dieser Stelle aufgetreten ist. Dies kann zum Beispiel ein Tooltip oder ein Klick auf ein Element sein.

Visual Analytics Process Der Visual Analytics Process von Keim et al. [KKEM10] beschreibt den Prozess der visuellen Datenanalyse (siehe Abbildung 5.5). Die vorhandenen Daten werden zunächst für die weitere Analyse transformiert, zum Beispiel durch Filtern. Anschließend werden die Daten analysiert. Dies kann durch automatische Analysemethoden, beispielsweise Data-Mining-Techniken, passieren, oder indem die Daten auf eine visuelle Repräsentation abgebildet werden und der Benutzer selbst mit dieser interagiert [SWLL13]. Mit dem dadurch gewonnenen Wissen über die Daten können diese wiederum transformiert werden, zum Beispiel indem die Datenmenge auf Teilbereiche eingeschränkt wird, und der Kreislauf beginnt von vorne.

5.2. Umsetzung

5.2.1. Architektur

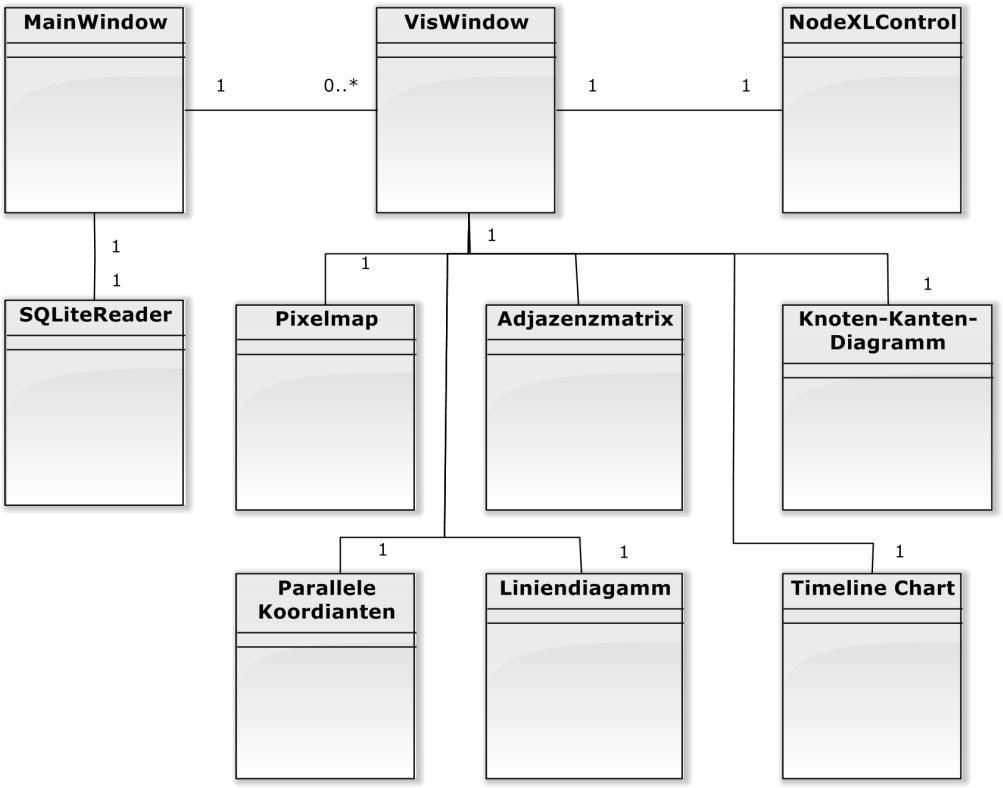

Abbildung 5.6.: Klassendiagramm mit den Beziehungen zwischen den wichtigsten Klassen des Prototyps.

Der Prototyp wurde in C# implementiert. Die Projektmappe besteht aus insgesamt neun Einzelprojekten. Diese umfassen das Hauptprogramm, den SQLiteReader, die Datenstruktur und sechs implementierte Visualisierungen.

SQLiteReader Die mit Sysstat gesammelten Leistungsdaten liegen in einer SQLite-Datenbank vor (siehe Kapitel 2.2). Der SQLiteReader verwendet die SQLite-Bibliothek „System.Data.SQLite.dll“ [HKM], die die Daten und Bearbeitung von SQL-Anfragen in Dateien organisiert und dadurch keine weitere Infrastruktur benötigt. Mit SQLite werden die Daten am Anfang einmal komplett aus der Datei eingelesen und daraus die in diesem Kapitel beschriebenen Objekte generiert. Die Visualisierungen arbeiten nur noch auf den generierten Objekten.

Datenstruktur Für die Verwendung der gesammelten Daten aus einem Supercomputersystem wurde eine eigene Datenstruktur entwickelt, die Objekte für die Zustände der Knoten und Kanten eines Systems zu einem bestimmten Zeitpunkt erstellt. Durch die Verwendung einer einheitlichen Datenstruktur soll eine einfache Erweiterbarkeit des Prototyps um weitere Visualisierungen erzielt werden.

NodeDescription-Objekte beschreiben Knoten eines Supercomputersystems. Ein NodeDescription-Objekt erhält einen Namen und eine ID, einen Zeitstempel und Werte für verschiedene Knotenmetriken zu diesem Zeitstempel. In der konkreten Implementierung des Prototyps beschreibt eine NodeDescription einen Rechner des Systems, sie kann jedoch auch zur Beschreibung eines CPU-Kerns des Systems verwendet werden.

EdgeDescription-Objekte beschreiben Kanten eines Supercomputersystems. Ein EdgeDescription-Objekt erhält die IDs des zugehörigen Start- und Endknotens, einen Zeitstempel und Werte für verschiedene Kantenmetriken zu diesem Zeitstempel. Dabei kann eine EdgeDescription sowohl die Kommunikation zwischen zwei Rechnern, als auch zwischen anderen Knotentypen, beispielsweise zwei CPU-Kernen, beschreiben.

Visualisierungen Die Visualisierungen werden mit NodeXL [HSS10] gezeichnet. Dabei handelt es sich um ein Framework, das als C#-Klassenbibliothek vorliegt und bereits viele grundlegende Funktionen für die Erstellung von Visualisierungen mitbringt. So konnte die Zeit für die Entwicklung eines eigenen Grafik-Frameworks eingespart werden. Jede der implementierten Visualisierungen liegt in einem eigenen Projekt vor und arbeitet auf der oben beschriebenen Datenstruktur. Zusätzlich zu den darzustellenden Knoten- und Kantenobjekten erhalten alle Visualisierungen eine Liste von darzustellenden Zeitpunkten, die zu visualisierenden Metriken und ein NodeXLControl als Eingabe, auf welchem die Visualisierung gezeichnet wird. Dadurch, dass alle Darstellungen mit derselben Datenstruktur als Eingabe arbeiten, können beliebige weitere Visualisierungen auf dieser Datenstruktur implementiert und in den Prototyp eingebunden werden. Jede Visualisierung enthält eine Methode namens „Draw“, die die Visualisierung auf ein NodeXLControl zeichnet.

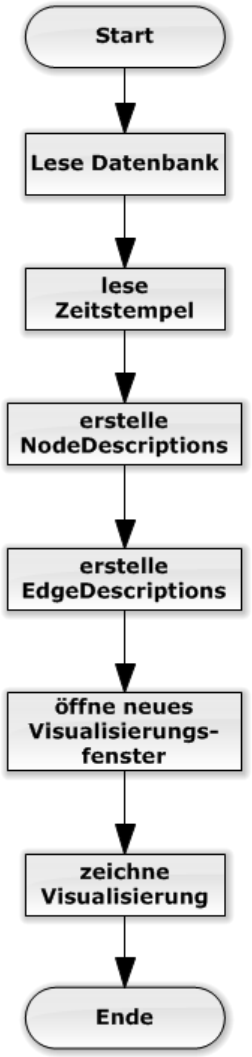

Abbildung 5.7.: Flussdiagramm des Prototyps, das das Laden der Datenbank und Erstellen einer Visualisierung zeigt.

Hauptprogramm Das Hauptprogramm stellt die Benutzeroberfläche zur Verfügung. Vom Startfenster des Prototyps aus können eine Datei geladen und Visualisierungen geöffnet werden. Für ersteres verwendet das Startfenster den SQLiteReader, für letzteres die Projekte der implementierten Visualisierungen. Eine Visualisierung wird in einem eigenen Fenster geöffnet, in dem mit ihr interagiert werden kann. Das Visualisierungsfenster enthält eine Toolbar und ein NodeXLControl, auf welches die Visualisierung gezeichnet wird. Des Weiteren enthält diese Klasse die NodeDescriptions, EdgeDescriptions und Zeitstempel, die visualisiert werden sollen. Das Startfenster enthält Verweise zu allen geöffneten Visualisierungsfenstern und kann diese in den Vordergrund holen. Das Flussdiagramm in Abbildung 5.7 beschreibt, wie der Benutzer die gesammelten Daten lädt und Visualisierungen erstellt. In Anhang A.1 befindet sich ein ausführlicheres Sequenzdiagramm, dass den Ablauf zwischen den entsprechenden Klassen beschreibt. Der Benutzer wählt über das Startfenster eine Datei aus, aus der die Daten geladen werden sollen. Das Startfenster ruft den SQLiteReader auf, welcher die Datei einliest, aus den Informationen in der Datei NodeDescriptions und EdgeDescriptions erstellt, sowie die Zeitpunkte ausliest, zu denen Daten vorliegen, und gibt diese Informationen an das Startfenster zurück. Anschließend kann der Benutzer über das Startfenster eine Visualisierung, die zu visualisierenden Metriken und einen Zeitraum auswählen, um eine Visualisierung zu erstellen. Das Startfenster erstellt ein neues Visualisierungsfenster und gibt die ausgewählten Informationen an dieses weiter. Das Visualisierungsfenster erstellt daraufhin mit diesen Informationen eine Visualisierung und lässt sie auf sein NodeXLControl zeichnen. Abbildung 5.8 zeigt beispielhaft für das Interagieren mit Visualisierungen das Markieren und Einschränken der Daten in einer Visualisierung. In Anhang A.1 befindet sich ein Sequenzdiagramm, dass den Ablauf zwischen den Klassen beschreibt. Der Benutzer markiert zunächst Elemente im NodeXLControl des Visualisierungsfensters, zum Beispiel mit der Maus. Dann lässt er das Visualisierungsfenster die Visualisierung auf diese Elemente einschränken. Das Visualisierungsfenster liest die markierten Elemente aus dem NodeXLControl aus und sucht die entsprechenden NodeDescriptions, EdgeDescriptions und Zeitpunkte heraus, die durch die markierten Elemente repräsentiert werden. Dann lässt es die Visualisierung mit diesen Elementen auf seinem NodeXLControl neu zeichnen.

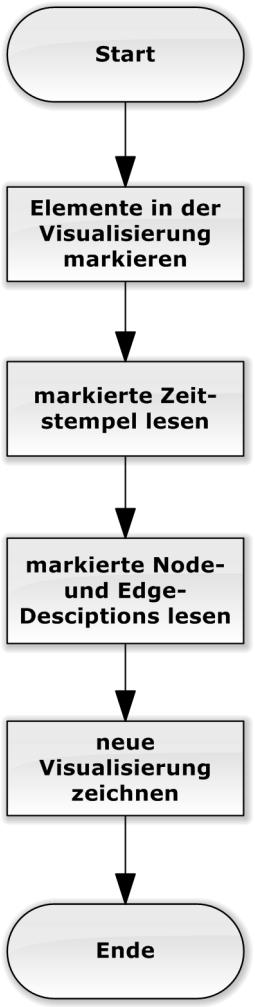

Abbildung 5.8.: Flussdiagramm für das Markieren und Zuschneiden in einer Visualisierung.

5.2.2. Benutzerschnittstelle

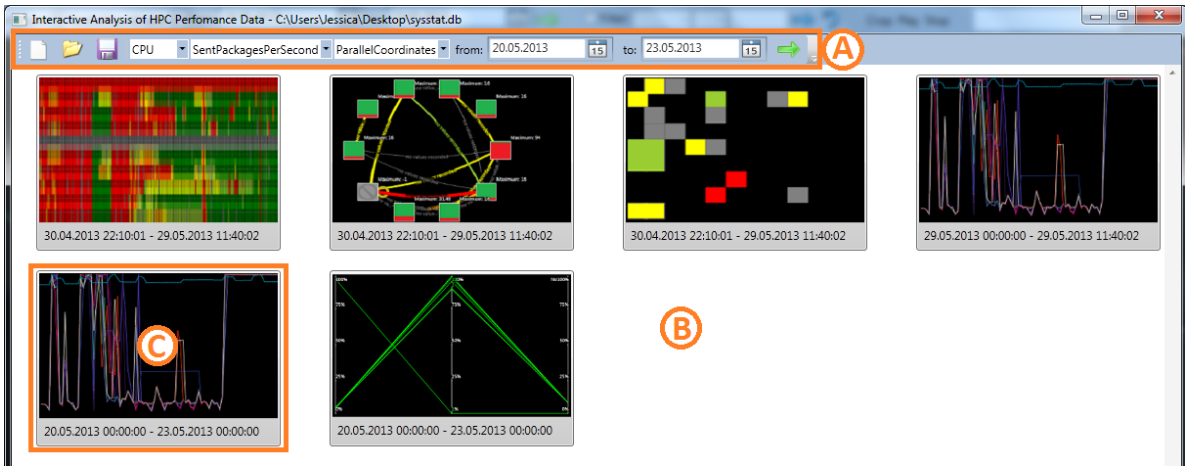

Abbildung 5.9.: Startfenster des Prototyps. Hier können eine Datenbank geladen und Visualisierungen geöffnet und verwaltet werden.
A: Toolbar zum Laden von Datenbanken und Öffnen von Visualisierungen
B: Dashboard mit einer Übersicht über geöffnete Visualisierungen
C: Kleinansicht einer geöffneten Visualisierung

Die Benutzerschnittstelle soll dem Benutzer das Arbeiten mit den Daten und Visualisierungen so angenehm wie möglich machen. Dabei soll sie so einfach und intuitiv wie möglich gehalten werden. Abbildung 5.9 zeigt das Startfenster des Prototyps. Über die Toolbar (siehe Abbildung 5.9 A) kann der Benutzer zunächst eine Datenbank laden, deren Daten er analysieren möchte. Da dies einen Augenblick dauern kann, wird der Benutzer in einer Statuszeile am unteren Fensterrand darüber informiert, dass die Datenbank gerade geladen wird. Wurde die Datenbank erfolgreich geladen, kann der Benutzer über die Drop-Down-Menüs und Datumsfelder der Toolbar eine Visualisierung und die zu visualisierenden Knoten- und Kantenmetriken, sowie einen Zeitraum auswählen, um eine neue Visualisierung zu erstellen. An Visualisierungen stehen eine Pixelmap, eine Adjazenzmatrix, ein Knoten-Kanten-Diagramm, ein Liniendiagramm, Parallele Koordinaten und ein Timeline Chart zur Verfügung. Eine genauere Beschreibung der einzelnen Visualisierungen findet sich in den nachfolgenden Abschnitten (siehe 5.11 - 5.22). Die Knotenmetriken beziehen sich auf die Knoten des Supercomputernetzwerks, das den geladenen Daten zugrunde liegt, die Kantenmetriken auf die Kommunikation zwischen diesen Knoten (siehe Kapitel 2.4). In der Implementierung des Prototyps stehen als Knotenmetriken die durchschnittliche CPU-Auslastung der Kerne eines Rechners, die Speicherauslastung eines Rechners und die Netzwerkauslastung zur Verfügung, sowie das Maximum als kombinierte Metrik aus allen drei. Als Kantenmetrik stehen zwei Test-Metriken zur Verfügung, welche durch einen Zufallsgenerator generiert wurden, da für diese Arbeit keine echten Daten für die Kommunikation innerhalb des Supercomputersystems zur Verfügung standen. Auch hier kann das Maximum beider Werte als kombinierte Metrik verwendet werden. In der Implementierung dieses Prototyps wurden diese Metriken

verwendet, sowohl Knoten- als auch Kantenmetriken lassen sich aber einfach erweitern. Die Datumsfelder beinhalten den gesamten Zeitraum, für den Daten in der geladenen Datenbank vorliegen.

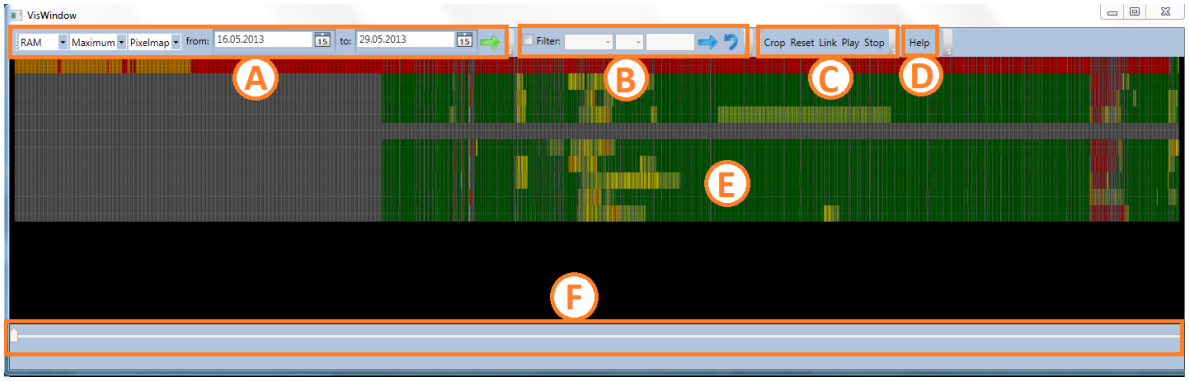

Abbildung 5.10.: Visualisierungsfenster des Prototyps. Hier kann der Benutzer mit einer geöffneten Visualisierung interagieren.
A: Toolbar zum Konfigurieren der angezeigten Visualisierung
B: Auswahl zum Filtern der visualisierten Elemente
C: Buttons zur Interaktion mit der Visualisierung
D: Hilfe zur angezeigten Visualisierung
E: Visualisierung der Daten
F: Schieberegler zum Durchlaufen einzelner Zeitschritte in Visualisierungen für einzelne Zeitschritte

Unter der Toolbar des Startfensters befindet sich ein Dashboard (siehe Abbildung 5.9 B). Beim Erstellen einer neuen Visualisierung öffnet sich ein neues Fenster für diese Visualisierung. Dabei wird gleichzeitig eine Kleinansicht auf dem Dashboard erstellt, die mit dem Visualisierungsfenster verknüpft ist (Abbildung 5.9 C). Die Kleinansicht besteht aus einem Bild, das den Visualisierungstyp angibt, und einem Label, das den Zeitraum angibt, für den diese Visualisierung erstellt wurde. Beim Schließen eines Visualisierungsfensters wird dieses versteckt und kann durch einen Klick auf die zugehörige Kleinansicht wieder in den Vordergrund geholt werden. Das soll dem Benutzer die Arbeit mit vielen geöffneten Visualisierungen erleichtern, da er nicht alle Fenster gleichzeitig sichtbar auf dem Bildschirm haben muss, sondern nur diejenigen anzeigen lassen kann, mit denen er gerade arbeitet. Gleichzeitig hat er über die Kleinansichten im Dashboard eine Übersicht über alle erstellten Visualisierungen und kann dadurch leicht und schnell auf diese zugreifen und zwischen diesen wechseln. Beim Erstellen einer Visualisierung öffnet sich für diese ein neues Fenster (siehe Abbildung 5.10). Dieses besteht aus einer dreigeteilten Toolbar, der Visualisierung selbst und einem Zeitslider. Über Drop-Down-Menüs und Datumsfelder, die denen im Hauptfenster gleichen, kann der Benutzer die Visualisierung konfigurieren (Abbildung 5.10 A). Die Datumsfelder umfassen den Zeitbereich, der im Hauptfenster bei der Erstellung der Visualisierung ausgewählt wurde. Abbildung 5.10 B zeigt einen Filter, über den der Benutzer die visualisierten Elemente des zugrundeliegenden Supercomputersystems filtern kann.

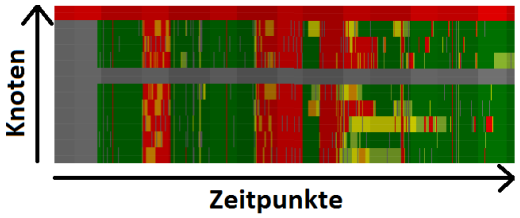

Abbildung 5.11.: Pixelmap des Prototyps für die Speicherauslastung von zehn Rechnern über einen Zeitraum von einer Woche. Eine Zeile repräsentiert einen Rechner, eine Spalte einen Zeitpunkt. Eine Farbskala repräsentiert den Metrikwert.

Beim Setzen des Häkchens im Filterkontrollkästchen werden die nebenstehenden Drop-Down-Menüs und ein Textfeld aktiviert. Im ersten Drop-Down-Menü kann der Benutzer auswählen, wonach die Visualisierung gefiltert werden soll. Dabei stehen ihm in der Implementierung des Prototyps die ID eines Rechners, der Name eines Rechners und die Knoten- und Kantenmetriken zur Verfügung. Im zweiten Drop-Down-Menü stehen dem Benutzer die Operatoren „größer“, „kleiner“, „gleich“ und „enthält“ zur Verfügung. Im Textfeld kann anschließend der Wert eingetragen werden, nach dem gefiltert werden soll. Über die beiden Pfeile daneben kann der Filter angewendet oder zurückgesetzt werden. Abbildung 5.10 C zeigt verschiedene Buttons zur Interaktion mit der Visualisierung. Über den „Hilfe“-Button in Abbildung 5.10 D kann eine Beschreibung der angezeigten Visualisierung geöffnet werden, die dem Benutzer erklärt, was in der Visualisierung dargestellt wird. Unterhalb der Toolbar befindet sich die Visualisierung selbst (siehe Abbildung 5.10 E). Wie mit den unterschiedlichen Visualisierungen interagiert werden kann, wird in den nachfolgenden Abschnitten im Detail erklärt (siehe 5.11 - 5.22). Unterhalb der Visualisierung befindet sich der Zeitslider (Abbildung 5.10 F). Dabei handelt es sich um einen Schieberegler, auf dem die einzelnen Zeitpunkte, zu denen Daten aufgezeichnet wurden, abgetragen sind. Der Zeitslider umfasst dabei immer den Zeitbereich, der in den Datumsfeldern aus Abbildung 5.10 A ausgewählt ist. Mit der Maus können die Zeitschritte von Hand durchgeschaltet werden. Da das Zeichnen einzelner Visualisierungen unter Umständen einige Sekunden dauert, wird die Visualisierung erst dann gezeichnet, wenn der Benutzer den Zeitslider wieder loslässt. Alternativ kann der Benutzer die Zeitschritte über den „Play“-Button in Abbildung 5.10 C automatisch abspielen lassen. Beginnend vom gerade ausgewählten Zeitpunkt springt der Zeitslider automatisch zum nächsten Zeitpunkt und zeichnet die zugehörige Visualisierung. Über den „Stop“-Button daneben kann das automatische Abspielen beendet werden. Diese Funktion soll es dem Benutzer erleichtern, den zeitlichen Verlauf der Daten zu betrachten, indem er sich nicht auf den Schieberegler konzentrieren muss, sondern sich allein auf die Änderungen in der Visualisierung konzentrieren kann. Der „Link“-Button in Abbildung 5.10 C realisiert das Konzept „Brushing und Linking“ (siehe Kapitel 2.1). Wenn der Benutzer Elemente in einer Visualisierung markiert und auf diesen Button klickt, werden die entsprechenden Elemente in den anderen geöffneten Visualisierungen auch markiert, so dass der Benutzer sie dort sofort sieht und vergleichen kann.

Pixelmap Die Pixelmap stellt den Verlauf einer Metrik für die Knoten des Supercomputersystems über die Zeit dar. Die Knoten werden untereinander dargestellt, so dass eine waagerechte Zeile genau einem Knoten entspricht. Von links nach rechts sind die Zeitpunkte, zu denen Werte gemessen wurden, abgetragen. Ein „Pixel“, in diesem Fall ein senkrechter Streifen, repräsentiert daher den Zustand eines Knotens zu einem bestimmten Zeitpunkt. Abbildung 5.11 zeigt eine Pixelmap, die den Verlauf der Speicherauslastung für zehn Rechner und einen Zeitraum von einer Woche darstellt. In der Implementierung des Prototyps stellt eine Zeile in der Pixelmap einen bestimmten Rechner des Supercomputersystems dar. Die Pixelmap kann aber einfach erweitert werden, so dass eine Zeile zum Beispiel auch einen CPU-Kern eines Rechners repräsentiert. Der Wert der Metrik wird über die Einfärbung angegeben, die auf einer Farbskala beruht. In der Implementierung des Prototyps wurde eine Farbskala von grün, für niedrige Werte, über gelb zu rot, für hohe Werte implementiert. Die Farbskala kann jedoch auf beliebige Farbverläufe angepasst werden. Diese Skala wird auf den Wertebereich zwischen null und dem größten gemessenen Wert der visualisierten Metrik in der geladenen Datenbank abgebildet. Abbildung 5.12 zeigt, wie sich auffällige Daten (siehe Kapitel 2.4) in der Pixelmap äußern können. In den schwarz umrandeten Bereichen ist zu sehen, dass der Speicher eines Rechners fast gar nicht ausgelastet ist (grün dargestellt), während alle anderen Rechner fast voll ausgelastet sind (rot dargestellt). Das kann darauf schließen lassen, dass die Kapazität eines Rechners gar nicht genutzt wird und die Leistung des Supercomputersystems gesteigert werden könnte, wenn die Operationen, die auf den anderen Rechnern ausgeführt werden, auf alle Rechner verteilt werden. In der Pixelmap kann der Benutzer mit der Maus ein Auswahlrechteck ziehen, um Teilbereiche auszuwählen. Mit gleichzeitig gedrückter Steuerungstaste können mehrere Bereiche selektiert werden. Durch einen Klick auf den „crop“-Button in der Toolbar (siehe Abbildung 5.10 C) wird die Pixelmap auf die ausgewählten Zeitpunkte und Knoten beschränkt. Abbildung 5.13 zeigt links eine Pixelmap, in der mit Hilfe des Auswahlrechtecks ein Ausschnitt markiert wurde und rechts das Ergebnis des Zuschneidens auf den ausgewählten Bereich. Mit dieser Funktion kann der Benutzer eine unter Umständen sehr große Datenmenge auf einen kleineren Bereich zuschneiden, um nur diesen zu betrachten. Die Pixelmap wurde in erster Linie deshalb als Visualisierung ausgewählt, weil sie eine sehr große

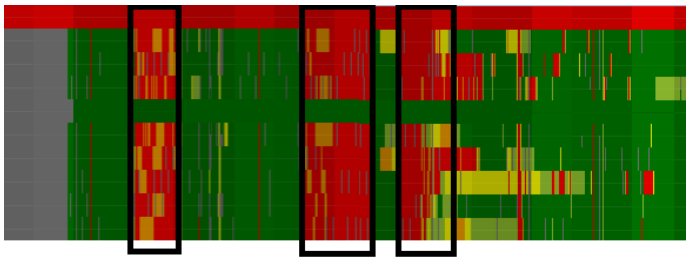

Abbildung 5.12.: Pixelmap mit markierten „auffälligen“ Daten. In den umrandeten Bereichen ist sichtbar, dass ein Rechner kaum ausgelastet ist (grün), während die anderen Rechner fast voll ausgelastet sind (rot).

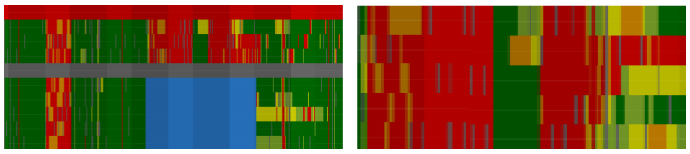

Abbildung 5.13.: Vergleich der Pixelmap vor und nach dem Zuschneiden auf den markierten Bereich. links: Pixelmap mit markierten (blauen) Elementen, rechts: Pixelmap auf die markierten Elemente zugeschnitten

Menge an Daten darstellen kann. Durch den Verzicht auf Beschriftungen können die „Pixel“ sehr schmal bzw. flach dargestellt werden, so dass sehr viele Knoten untereinander und sehr viele Zeitpunkte nebeneinander dargestellt werden können. Durch diese Visualisierung kann der Benutzer alle Knoten und alle Zeitpunkte, die in der Datenbank aufgezeichnet wurden, auf einmal betrachten und von dieser Übersicht beginnend den Datensatz auf kleinere Teilbereiche einschränken.

Adjazenzmatrix Die Adjazenzmatrix stellt die Kommunikation zwischen Knoten des Supercomputersystems zu einem bestimmten Zeitpunkt dar. Dabei werden sowohl senkrecht, als auch waagrecht, die Knoten aufgetragen. Ein Rechteck beschreibt die Kommunikation vom Knoten in dieser Zeile zum Knoten in dieser Spalte. Der Wert der Kantenmetrik wird durch die Färbung des Rechtecks dargestellt. Wie in der Pixelmap wurde in der Implementierung des Prototyps ein Farbverlauf von grün über gelb zu rot implementiert, der jedoch für andere Farbskalen angepasst werden kann. Dabei wird die Farbskala auf den Wertebereich zwischen null und dem größten aufgezeichneten Wert der visualisierten Metrik in der Datenbank abgebildet. Auch die Adjazenzmatrix wurde für den Prototyp so implementiert, dass eine Zeile bzw. eine Spalte einen Rechner des Supercomputersystems repräsentiert. Auch hier könnte eine Zeile bzw. Spalte stattdessen einen CPU-Kern eines Rechners und deren Kommunikation untereinander darstellen.Abbildung 5.14 zeigt eine Adjazenzmatrix, die die gesendeten Pakete pro Sekunde zwischen zehn Rechnern zu einem bestimmten Zeitpunkt visualisiert. Die Adjazenzmatrix verzichtet auf Beschriftungen, so dass die Zeilen und Spalten sehr schmal dargestellt werden können. So können sehr viele Knoten unter- und nebeneinander dargestellt werden. Dadurch kann der Benutzer auch in großen Supercomputersystemen mit sehr vielen Knoten die Kommunikation zwischen allen diesen Knoten auf einmal betrachten. Die Adjazenzmatrix stellt immer einen bestimmten Zeitpunkt dar. Mit Hilfe des Zeitsliders (siehe Abbildung 5.10 F) und des „Play“-Buttons (siehe Abbildung 5.10 C) kann der Benutzer die einzelnen Zeitpunkte der ausgewählten Zeitspanne manuell durchlaufen oder automatisch abspielen lassen. In der Adjazenzmatrix kann der Benutzer mit der Maus ein Auswahlrechteck um Teilbereiche der Matrix ziehen, welche dann markiert werden. Durch einen Klick auf den „crop“-Button (Abbildung 5.10 C) wird die Matrix

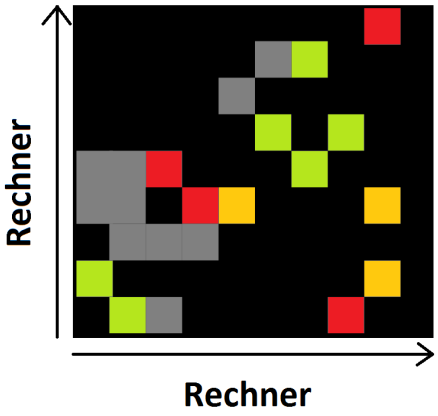

Abbildung 5.14.: Adjazenzmatrix des Prototyps für die gesendeten Pakete pro Sekunde zwischen zehn Rechnern zu einem bestimmten Zeitpunkt. Jede Zeile und jede Spalte repräsentiert einen Rechner. Eine Farbskala repräsentiert den Metrikwert.

Abbildung 5.15.: Tooltip in der Adjazenzmatrix mit Informationen zur dargestellten Kante

auf die ausgewählten Zeilen und Spalten zugeschnitten. Es werden dann unter Umständen in den Zeilen andere Rechner dargestellt, als in den Spalten. Eine Symmetrie der Matrixachsen ist also nicht zwingend gegeben. Wenn der Benutzer mit dem Mauszeiger über einen Matrixeintrag fährt, erscheint ein Tooltip, welcher Informationen zur dargestellten Kante enthält. In der Implementierung des Prototyps werden dem Benutzer ID und Name des Start- und des Endknotens angezeigt, sowie die Werte der beiden implementierten Kantenmetriken zum visualisierten Zeitpunkt. Denkbar sind aber auch andere Informationen.

Knoten-Kanten-Diagramm Das Knoten-Kanten-Diagramm stellt sowohl Knoten eines Supercomputersystems, als auch deren Kommunikation untereinander zu einem bestimmten Zeitpunkt dar. Das Knoten-Kanten-Diagramm ist weder auf die Visualisierung multivariater, noch auf die Visualisierung zeitabhängiger Daten ausgelegt. Es ist jedoch eine verbreitete Visualisierung für die Darstellung von Netzwerken und wird dem Benutzer daher als zusätzliche Option zur Verfügung gestellt. Ein Knoten des Knoten-Kanten-Diagramms repräsentiert einen Knoten des Supercomputersystems, eine Kante die Kommunikation zwischen diesen. In dieser Darstellung wird sowohl eine Knoten- als auch eine Kantenmetrik visualisiert. In der Implementierung des Prototyps werden beide Metriken durch einen Farbverlauf von grün über gelb nach rot dargestellt, welcher auf den Wertebereich zwischen null und dem jeweils größten gemessenen Wert der entsprechenden Metrik in der geladenen Datenbank abgebildet wird. Der Farbverlauf kann, wie in der Pixelmap und in der Adjazenzmatrix, auch auf andere Farben angepasst werden. Der Wert der Kantenmetrik wird gleichzeitig durch die Dicke der Kante verdeutlicht. Je höher der Wert, desto dicker die Kante. Knoten, die innerhalb des ausgewählten Zeitbereichs miteinander kommunizieren, jedoch nicht zum dargestellten Zeitpunkt, werden durch eine dünne farblose Kante miteinander verbunden. Dadurch kann der Benutzer sehen, welche Knoten überhaupt irgendwann miteinander kommunizieren und welche zu genau diesem Zeitpunkt. Knoten, für die zum dargestellten Zeitpunkt kein Wert für die visualisierte Knotenmetrik gemessen wurde, werden farblos dargestellt. Liegt die Anzahl der visualisierten Knoten unterhalb eine definierten Grenze, wird die Knotenmetrik nicht durch eine Farbe des Farbverlaufs dargestellt, sondern durch eine detailliertere, zweifarbige Darstellung (siehe Abbildung 5.16). Im Prototyp wurde eine Rot-Grün-Darstellung implementiert. Je größer der Rotanteil des Knotens, desto größer der Wert der visualisierten Metrik, wobei eine vollständig grüne Färbung den Wert null repräsentiert und eine vollständig rote Färbung den größten gemessenen Wert der Metrik in der geladenen Datenbank. Die Knoten

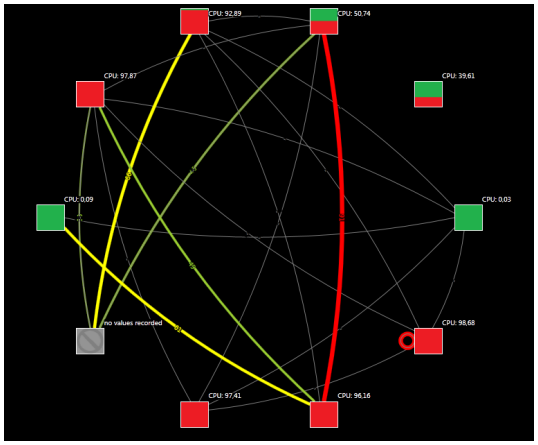

Abbildung 5.16.: Knoten-Kanten-Diagramm des Prototyps zur Visualisierung der Rechner und deren Kommunikation untereinander zu einem bestimmten Zeitpunkt. Die Metrikwerte werden über die Färbung und die Kantendicke repräsentiert.

und Kanten enthalten Beschriftungen. In der Implementierung des Prototyps geben diese den genauen Wert der visualisierten Metrik an, denkbar sind jedoch auch andere Beschriftungen, zum Beispiel Name und ID eines Rechners, den ein Knoten repräsentiert. Fährt der Benutzer mit dem Mauszeiger über einen Knoten, wird ein Tooltip angezeigt, welcher Name und ID des Rechners, sowie die genauen Werte der implementierten Knotenmetriken zum angezeigten Zeitpunkt darstellt (siehe Abbildung 5.17). Denkbar sind auch hier andere Informationen. Durch einen Klick auf einen Knoten werden dieser und alle von ihm ausgehenden Kanten markiert. Über den „crop“-Button wird das Diagramm auf die markierten Knoten und die Kanten zwischen diesen Knoten zugeschnitten. Bei der Erstellung der Visualisierung werden die Knoten automatisch im Kreis angeordnet. Durch Klicken auf einen Knoten und gleichzeitiges Ziehen der Maus über den Bildschirm, können Knoten in der Visualisierung verschoben werden. So kann der Benutzer die Knoten beliebig auf dem Bildschirm anordnen, um die Darstellung übersichtlicher zu gestalten. Über den Zeitslider und den „Play“-Button (Abbildungen 5.10 F und 5.10 C) kann der Benutzer die einzelnen Zeitschritte manuell durchschalten oder automatisch abspielen lassen.

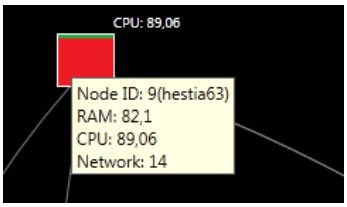

Abbildung 5.17.: Knoten-Tooltip im Knoten-Kanten-Diagramm mit Informationen zu einem Rechner des Supercomputersystems

Parallele Koordinaten Parallele Koordinaten stellen die Werte verschiedener Knoten- und Kantenmetriken für die Knoten des Supercomputersystems zu einem bestimmten Zeitpunkt dar. Auf der ersten senkrechten Achse werden die visualisierten Knoten in gleichmäßigen Abständen abgebildet, so dass jede eingezeichnete Linie in den Parallelen Koordinaten einem bestimmten Knoten zugeordnet werden kann. Auf den restlichen parallelen Koordinatenachsen werden die Knoten- und die Kantenmetriken abgebildet. Dabei werden die Wertebereiche zwischen null und dem größten gemessenen Wert der jeweiligen Metrik in der geladenen Datenbank auf der entsprechenden Achse abgetragen. Für jeden dargestellten Knoten werden die entsprechenden Knotenmetriken auf den zugehörigen Achsen eingetragen und die Kantenmetriken von den Kanten, die von diesem Knoten ausgehen. In dieser Visualisierung können verschiedene Metriken gleichzeitig dargestellt werden. Das soll dem Benutzer ermöglichen, Zusammenhänge zwischen den Metriken leichter erkennen zu können. Abbildung 5.18 zeigt Parallele Koordinaten für zehn Rechner, drei Knotenmetriken und zwei Kantenmetriken zu einem bestimmten Zeitpunkt. Standardmäßig werden die eingetragenen Linien durch einen Farbverlauf auf der ersten Achse eingefärbt, so dass jeder eingezeichnete Knoten eine andere Farbe erhält, damit der Benutzer die Knoten gut voneinander unterscheiden kann. Dieser Farbverlauf kann durch einen Mausklick auf das Quadrat am oberen Ende der Achse auf die entsprechende Achse übertragen werden. So kann der Benutzer erkennen, welche Rechner für diese Metrik einen hohen und welche einen niedrigeren Wert haben. In dieser Abbildung fällt beispielsweise auf, dass der Speicher eines Rechners fast vollständig ausgelastet ist, während der Speicher der anderen Rechner kaum ausgelastet ist.

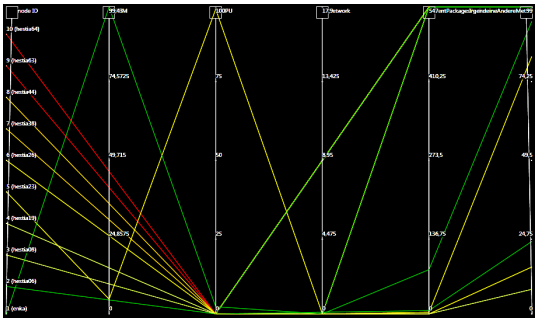

Abbildung 5.18.: Parallele Koordinaten des Prototyps für Knoten und Kantenmetriken von zehn Rechnern zu einem bestimmten Zeitpunkt. Auf der ersten Achse werden die Rechner dargestellt, auf den restlichen Achsen drei Knoten- und zwei Kantenmetriken

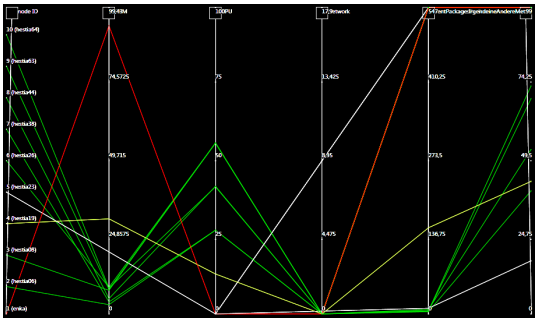

Abbildung 5.19.: Parallele Koordinaten in denen eine Farbskala von grün nach rot auf der Speicher-Achse abgetragen wurde. Hier fällt auf, dass die Speicherkapazität eines Rechners fast voll ausgelastet ist, während die Speicherkapazität der anderen Rechner kaum ausgelastet ist.

Auch hier wurde ein Farbverlauf von grün über gelb nach rot implementiert, wobei Kanten, für die zu diesem Zeitpunkt kein Wert für die entsprechende Metrik vorliegt, weiß dargestellt werden. Über den Zeitslider und den „Play“-Button kann der Benutzer auch hier einzelne Zeitschritte durchklicken oder automatisch abspielen lassen.

Liniendiagramm Das Liniendiagramm stellt den Verlauf einer Knotenmetrik für die Knoten des Supercomputersystems über die Zeit des ausgewählten Zeitraums dar. Dabei entspricht jede Linie einem Knoten. Die Werte der Knotenmetrik werden auf der Vertikalen des Diagramms abgebildet, wobei der Wert null auf den unteren Rand der Visualisierung und der größte gemessene Metrikwert in der geladenen Datenbank auf den oberen Rand der Visualisierung abgebildet werden. Abbildung 5.20 zeigt ein Liniendiagramm für zehn Rechner eines Supercomputersystems und den Verlauf der Speicherauslastung dieser Rechner über einen Zeitraum von zwei Tagen. Auffällig ist hier, dass ein Rechner durchgehend eine sehr hohe Speicherauslastung hat, während der Speicher der anderen Rechner nur hin und wieder genutzt wird. Die Färbung der Linien dient hier allein der besseren Unterscheidbarkeit voneinander und sagt nichts über den Metrikwert aus. Dieser wird allein über die Positionierung im Diagramm dargestellt. In der Implementierung des Prototyps stellt eine Linie einen Rechner des Supercomputersystems dar. Labels am Beginn der Linien geben den Namen und die ID des entsprechenden Rechners an. Mit Hilfe des Mausrads kann der Benutzer Teilbereiche der Visualisierung vergrößern, um kleine Schwankungen in den Werten detaillierter zu betrachten (siehe Abbildung 5.21).

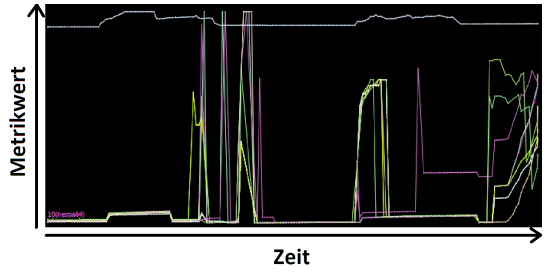

Abbildung 5.20.: Liniendiagramm des Prototyps für die Speicherauslastung von zehn Rechnern über einen Zeitraum von zwei Tagen. Eine Linie repräsentiert einen Rechner.

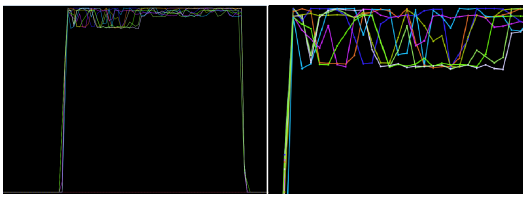

Abbildung 5.21.: Vergleich unvergrößertes und vergrößertes Liniendiagramm. links: Liniendiagramm in Standardgröße, rechts: vergrößerter Ausschnitt des Liniendiagramms

Timeline Chart Das Timeline Chart ist der Master Timeline aus VAMPIR (siehe Abbildung 4.1 C in Kapitel 4) nachempfunden. Untereinander werden die Knoten des Supercomputersystems dargestellt. Jeder Knoten erhält eine waagerechte gestrichelte Linie, die den ausgewählten Zeitraum darstellt. Auf diesen Zeitlinien werden die Zeitpunkte, zu denen ein Knoten eine Anfrage an einen anderen Knoten gestellt hat und zu denen ein Knoten auf eine Anfrage eines anderen Knotens geantwortet hat, eingetragen. Anfragezeitpunkt und Antwortzeitpunkt für die Kommunikation zwischen zwei Rechnern werden durch eine Linie miteinander verbunden. Dadurch kann der Benutzer sehen, welche Rechner sehr lange auf eine Antwort gewartet haben. Die Zeitlinien sind zur Unterscheidung der Knoten unterschiedlich eingefärbt. Eine Kommunikationslinie zwischen zwei Knoten erhält automatisch die Farbe des Knotens, der eine Anfrage an einen anderen Knoten stellt. Abbildung 5.22 zeigt die Kommunikation zwischen fünf Rechnern innerhalb eines Tages.

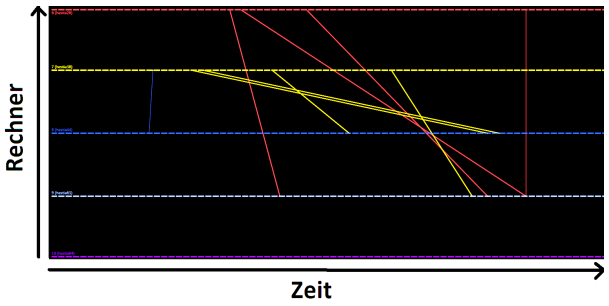

Abbildung 5.22.: Timeline Chart des Prototyps zur Visualisierung der Kommunikationszeiten zwischen fünf Rechnern eines Supercomputersystems innerhalb eines Tages.

5.2.3. Herausforderungen

Bei der Aufzeichnung der Daten aus einem verteilten System, wie einem Supercomputersystem, können Probleme auftreten, die beim Zusammenfügen der Daten in einer Visualisierung gelöst werden müssen. Drei häufig auftretende Probleme und mögliche Lösungsansätze werden im Folgenden beschrieben.

NodeXL Die Verwendung einer vorhandenen Grafikbibliothek brachte auch Einschränkungen mit sich, die in dieser Arbeit behandelt werden mussten. NodeXL bietet keine Kantenselektion oder Kanten-Tooltips, die beispielsweise in einem Knoten-Kanten-Diagramm sinnvoll sein können. Dies konnte gut umgangen werden, indem auf die Kante des Diagramms bezogene Informationen in den Tooltips der Knoten, die durch diese Kante verbunden sind, beschrieben werden. Die Kanten können selektiert werden, indem die zugehörigen Knoten selektiert werden. Dadurch kann eine Kante nur selektiert sein, wenn gleichzeitig die zugehörigen Knoten selektiert sind. Dies stellt jedoch kein Problem dar, da eine Kante überhaupt nur mit einem Start- und Zielknoten existieren kann.
Um eine Visualisierung in NodeXL zu zeichnen, müssen zunächst alle Knoten und Kanten, die das NodeXLControl zeichnen soll, erstellt und diesem hinzugefügt werden. Da das NodeXLControl ein UI-Steuerelement ist, muss das Erzeugen einer Visualisierung im UI-Thread passieren und kann nicht in einen externen Thread ausgelagert werden. Da das Erzeugen ein paar Sekunden dauern kann, wird dem Benutzer über einen Sanduhr-Mauszeiger symbolisiert, dass der Prototyp noch beschäftigt und nicht abgestürzt ist. Das eigentliche Zeichnen, nachdem alle Elemente hinzugefügt wurden, wird von der Bibliothek in einem asynchronen Thread ausgeführt, auf den von außen nicht ohne Weiteres zugegriffen werden kann. In der Implementierung der Autoplay-Funktion des Zeitsliders führte das dazu, dass bereits die Visualisierung für den nachfolgenden Zeitschritt erstellt wurde, bevor die Visualisierung für den aktuellen Zeitschritt gezeichnet werden konnte. Der Zeitslider durchlief die Zeitschritte, ohne dass die Darstellung aktualisiert wurde. Dieses Problem konnte durch die Verwendung eines DispatcherTimers gelöst werden. Über die Eigenschaft „IsLayingOutGraph“ kann abgefragt werden, ob das Steuerelement noch noch zeichnet. Das wird jedes mal vor dem Erzeugen einer Visualisierung, während des Autoplays, überprüft. Zeichnet das NodeXLControl noch, wird der Timer auf eine Sekunde aufgezogen und bei Ablauf des Timers erneut geprüft, ob das Steuerelement noch zeichnet. Wenn nicht, wird der Timer gestoppt, eine Sekunde gewartet, damit der Benutzer die Visualisierung mindestens eine Sekunde lang sieht, und dann die nächste Visualisierung gezeichnet.

Asynchrone Systemuhren In einem Supercomputersystem ist die Wahrscheinlichkeit sehr hoch, dass die Systemuhren der einzelnen Systeme nicht synchron laufen. So erhält man bei der Aufzeichnung der Daten unter Umständen enorm viele Zeitschritte, die jedoch nur sehr dünn mit Informationen besetzt sind, da nur für einen kleinen Bruchteil der Systeme zu exakt diesem Zeitpunkt Daten erfasst wurden. In Visualisierungen, die ganze Zeitspannen visualisieren, wie zum Beispiel in der Pixelmap, müssen dadurch enorm viele Elemente dargestellt werden, für die gleichzeitig entsprechend weniger Platz zur Verfügung steht.

Auf dem Zeitslider müssten extrem viele Zeitschritte eingetragen werden, die alle einzeln durchlaufen werden müssten. Es bietet sich daher an, Zeitschritte zusammenzufassen. Würden beispielsweise alle zehn Minuten Daten aufgezeichnet, bietet es sich an, Zeitspannen von zehn Minuten zu einem Zeitschritt zusammenzufassen. Ein Element auf dem Zeitslider repräsentiert dann nicht mehr einen bestimmten Zeitpunkt, sondern ein Zeitintervall. Die Abbildungen 5.23 und 5.24 zeigen beispielhaft, wie sich das Zusammenfassen der Zeitschritte auf eine Pixelmap und auf den Zeitslider auswirken.

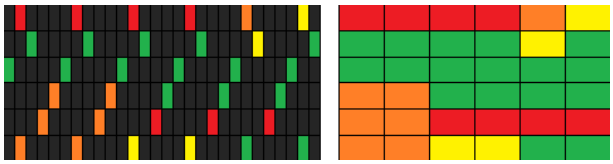

Abbildung 5.23.: Auswirkung aggregierter Zeitschritte am Beispiel der Pixelmap
links: Pixelmap ohne Aggregation, rechts: Pixelmap mit Aggregation von jeweils fünf Zeitschritten

Abbildung 5.24.: Auswirkung aggregierter Zeitschritte auf den Zeitslider
links: Zeitslider ohne Aggregation, rechts: Zeitslider mit Aggregation von jeweils fünf Zeitschritten

Fehlende Daten Es kann passieren, dass aus unbekannten Gründen für einen Knoten des Supercomputersystems in einer bestimmten Zeitspanne keine Daten aufgezeichnet wurden. Wird der Knoten während dieser Zeitspanne überhaupt nicht in der Visualisierung repräsentiert, kann es passieren, dass der Benutzer genau diese Zeitspanne betrachtet und nicht bemerkt, dass dieser Knoten existiert und nur keine Daten aufgezeichnet wurden. Daher sollten auch Elemente, für die zu einem Zeitpunkt keine Daten vorliegen, in den Visualisierungen repräsentiert werden. Gleichzeitig sollte sichtbar gemacht werden, dass keine Daten vorliegen. Dies lässt sich je nach Visualisierung unterschiedlich umsetzen. In der Pixelmap wird durch das Einfügen einer farblosen oder ausgegrauten Zeile verdeutlicht, dass der Knoten existiert. Dasselbe lässt sich auf die Zeilen und Spalten der Adjazenzmatrix anwenden. Im Liniendiagramm und den Parallelen Koordinaten könnten die Labels der entsprechenden Knoten angezeigt werden, ohne dass diese eine Linie besitzen. Im Knoten-Kanten-Diagramm könnte ein farbloser Knoten abgebildet werden und im Timeline Chart eine leere Zeitachse. Statt gar keine Werte zu liefern, kann aber auch eine Interpolation zwischen dem vorherigen und nachfolgenden Zeitpunkt, zu dem Daten vorliegen, durchgeführt werden. Dadurch kann der Benutzer zumindest grob einschätzen, wie die Werte verlaufen. Es ist jedoch wichtig zu verdeutlichen, dass es sich um berechnete Werte und nicht um echte Messungen handelt, da diese Annahme zu Fehlschlüssen führen kann. Die interpolierten Werte sollten daher von den anderen Werten abgehoben werden, beispielsweise durch eine geringere Sättigung bzw. Helligkeit der Farbskala, oder durch gestrichelte Linien anstelle von durchgezogenen Linien.

Unterschiedliche Datenformate Wenn Daten aus unterschiedlichen Quellen gesammelt werden, ist die Wahrscheinlichkeit sehr hoch, dass diese nicht alle im selben Format gespeichert werden. Das heißt, für manche Quellen stehen eventuell mehr oder detailliertere Informationen zur Verfügung, als für andere Quellen. Um sie dennoch in derselben Visualisierung darstellen zu können, müssen die Daten auf einen gemeinsamen Nenner gebracht werden. Visualisiert man nur Informationen, die für alle Quellen vorliegen, gehen unter Umständen wichtige Informationen verloren. Visualisiert man hingegen alle Informationen, die für irgendeine der Quellen gesammelt wurden, erhält man unter Umständen sehr viele Attribute, für die ein Großteil der Quellen jedoch keine Informationen enthält. In Parallelen Koordinaten kann das dazu führen, dass sehr viele Achsen in der Visualisierung dargestellt werden, aber für viele der Achsen nur sehr wenig Werte vorliegen. Hier muss man abwägen, ob man riskieren will, dass Informationen verloren gehen, oder in Kauf nimmt, dass Visualisierungen durch die Menge an Attributen überladen und möglicherweise unübersichtlich werden. Da dies stark von den vorliegenden Daten und dem Ziel, das der Benutzer verfolgt, abhängt, sollte man den Benutzer selbst entscheiden lassen, beispielsweise durch einen Umschalter in der Benutzeroberfläche. In der Implementierung des Prototyps wurden gezielt einige wenige Metriken ausgewählt, die für alle Knoten des Systems vorlagen, sollte der Prototyp in Zukunft jedoch um beliebige Metriken erweitert werden, sollte diesem Problem große Bedeutung beigemessen werden.

6. Benutzerstudie

Dieses Kapitel beschreibt die Durchführung einer Benutzerstudie zur Evaluation einiger Konzepte, die im Prototyp umgesetzt wurden. Aufgrund des zeitlichen Rahmens und des Umfangs der Arbeit war es nicht möglich, den gesamten Prototyp zu evaluieren. Daher wurde der Fokus in der Studie auf einzelne umgesetzte Konzepte und den Gesamteindruck des Prototyps gelegt.

6.1. Grundlagen der Studie

Die Benutzerstudie wurde an drei Tagen vom 02.10.2013 bis zum 04.10.2013 mit insgesamt 9 Teilnehmern durchgeführt. Alle Teilnehmer hatten zum Zeitpunkt der Teilnahme ähnliche Vorkenntnisse im Bereich der Informationsvisualisierung und Visual Analytics. Die Teilnehmer wurden mündlich auf der Grundlage eines Fragebogens befragt, welcher während der Durchführung vom Leiter der Studie ausgefüllt wurde. Die Datengrundlage der Benutzerstudie entspricht der Datengrundlage dieser Arbeit (siehe Kapitel 2.2). In der Vorbereitung der Studie wurde bereits die Datenbank und für die Aufgaben notwendige Visualisierungen geladen. Als Ausgangslage für die einzelnen Aufgaben standen jedem Teilnehmer exakt die gleichen Visualisierungen und visualisierten Daten zur Verfügung.

6.2. Ablauf

Die Teilnehmer nahmen nacheinander jeweils 30 Minuten an der Studie teil. Jeder Teilnehmer erhielt zunächst eine mündliche Erklärung des Tools. Dabei wurden der Sinn und Zweck des Prototyps beschrieben und der Aufbau der Benutzeroberfläche erklärt. Vor der Durchführung der einzelnen Aufgaben erhielt jeder Teilnehmer jeweils eine kurze Erklärung der entsprechenden Visualisierung und die Möglichkeit maximal eine Minute lang selbst mit der Visualisierung zu interagieren. Anschließend folgte die Aufgabenstellung inklusive Durchführung der Aufgabe. Während der Durchführung der einzelnen Aufgaben wurde jeweils die Zeit gestoppt. Nach Ausführung der Aufgabe wurde die Zeit notiert, der Teilnehmer zur Aufgabe befragt und die Antworten stichwortartig auf dem Fragebogen notiert. Zu einzelnen Aufgaben wurden auch Bewertungen auf einer, auf dem Fragebogen vorgegebenen, Skala eingetragen. Am Ende jeder Aufgabe erhielt der Teilnehmer die Möglichkeit weitere freie Anmerkungen anzubringen. Am Ende der Durchführung aller Aufgaben wurden dem Teilnehmer weitere nicht-aufgabenspezifische Fragen gestellt und nochmals die Möglichkeit gegeben, freie Anmerkungen zur gesamten Durchführung der Studie zu machen.

6.3. Aufgabenbeschreibung

Die Studie besteht aus insgesamt vier Aufgaben, die von allen Teilnehmern durchgeführt wurden. In der ersten Aufgabe soll der Teilnehmer in den Parallelen Koordinaten zunächst mit der Standardkantenfärbung nach Knotennamen einen Wert auslesen und anschließend mit Färbung nach der entsprechenden Metrik einen anderen Wert auslesen. Diese Aufgabe soll testen, ob die Kantenfärbung auf den Achsen der Parallelen Koordinaten hilfreich für das Auslesen der Werte ist. In der zweiten Aufgabe soll der Teilnehmer in den Parallelen Koordinaten zunächst manuell einige Schritte mit dem Zeitslider durchlaufen und die Zeitschritte anschließend automatisch abspielen lassen. In beiden Fällen soll er versuchen den groben Verlauf der Werte zu beschreiben. Anschließend soll der Teilnehmer bewerten, ob sich ein zeitlicher Verlauf in den Daten einfacher von Hand oder durch automatisches Abspielen erkennen lässt. Damit soll getestet werden, ob die Implementierung des Autoplays sinnvoll ist und Vorteile bringt. Die dritte Aufgabe beschäftigt sich mit den implementierten Interaktionstechniken. Hier soll der Benutzer ausgehend von einer Pixelmap selbstständig herausfinden, wie der Rechner zu einem bestimmten vorgegebenen Segment der Pixelmap heißt. Diese Aufgabe soll testen, ob die implementierten Interaktionen intuitiv sind bzw. von den Teilnehmern ohne weitere Erklärungen gefunden werden, oder Hilfestellung notwendig ist. In der vierten Aufgabe soll der Teilnehmer zuerst den Verlauf der Speicherauslastung in einem Liniendiagramm und anschließend den Verlauf der Speicherauslastung in einer Pixelmap beschreiben. Diese Aufgabe soll einerseits testen, ob die Teilnehmer unterschiedliche Eigenschaften des Werteverlaufs beschreiben und andererseits, welche Visualisierung die Teilnehmer nützlicher finden, um den Verlauf der Werte zu erkennen. Anschließend werden dem Teilnehmer freie Fragen zum Prototyp gestellt. Diese sollen prüfen, ob Teilnehmer etwas störend fanden oder etwas anderes erwartet haben und insgesamt einen positiven oder negativen Eindruck vom Prototyp haben. Die ausformulierte Aufgabenstellung ist in Anhang A.2 nachzulesen.

6.4. Auswertung

Ausnahmslos alle Teilnehmer fanden das automatische Abspielen des Zeitsliders hilfreich (siehe Tabelle 6.1), davon vier sogar sehr hilfreich, um den zeitlichen Verlauf der Daten zu betrachten. Es wurde angemerkt, dass man sich besser auf die Veränderungen in der Visualisierung konzentrieren kann, wenn man sich nicht auf den Zeitslider konzentrieren muss und dass das manuelle Durchschalten anstrengender ist, da man mit der Maus genau treffen muss. Einfacher wäre es, wenn sich der Zeitslider über die Pfeiltasten der Tastatur bedienen ließe.

sehr hilfreich	hilfreich	neutral	weniger hilfreich	störend
4	5	-	-	-

Tabelle 6.1.: Auswertung des Zeitsliders

Acht von neun Teilnehmern empfanden die Kantenfärbung in den Parallelen Koordinaten als hilfreich, davon sechs Teilnehmer als sehr hilfreich (siehe Tabelle 6.2). Es wurde geäußert, dass diese Färbung bei einem größeren Datensatz noch hilfreicher sein könnte, wenn das Diagramm sehr viele Linien enthält, da sich diese beim getesteten Datensatz auch ohne Färbung unterscheiden lassen, wenn auch etwas schwieriger als mit Färbung. Des Weiteren wäre es hilfreich, wenn die Farbskala im Prototyp eingeblendet würde.

sehr hilfreich	hilfreich	neutral	weniger hilfreich	störend
6	2	1	-	-

Tabelle 6.2.: Auswertung der Kantenfärbung in Parallelen Koordinaten

Fünf von neun Teilnehmern bevorzugten die Pixelmap gegenüber dem Liniendiagramm, um den Verlauf der Speicherauslastung zu betrachten. Drei von neun Teilnehmern bevorzugten weder das Liniendiagramm, noch die Pixelmap. Ein Teilnehmer bevorzugte das Liniendiagramm gegenüber der Pixelmap (siehe Tabelle 6.3). Begründet wurden die Antworten damit, dass in der Pixelmap die einzelnen Rechner besser voneinander unterschieden werden könnten, da sie sich nicht überdecken, wie die Linien im Liniendiagramm, das Liniendiagramm eigne sich jedoch besser, um sehr kleine Wertunterschiede zu erkennen, bei welchen die Farben auf der Farbskala in der Pixelmap sehr ähnlich aussehen und dadurch schwerer zu unterscheiden sind. Mehrfach vorgeschlagen wurde daher die Kombination aus beiden Visualisierungen, um mit der Pixelmap einen Überblick zu erhalten und mit dem Liniendiagramm Details zu betrachten.

Pixelmap	keins von beidem	Liniendiagramm
5	3	1

Tabelle 6.3.: Auswertung des Vergleichs zwischen Pixelmap und Liniendiagramm

Sieben von neun Teilnehmern wünschten sich insgesamt mehr Beschriftungen in den Visualisierungen. In der Pixelmap wurde mehrfach angemerkt, dass Beschriftungen vor den Zeilen sehr hilfreich wären. Tooltips in den Visualisierungen wurden von vier Teilnehmern gewünscht, Kontextmenüs von drei Teilnehmern. Fünf von neun Teilnehmern wünschten sich die Umsetzung von „Brushing und Linking“ (siehe Kapitel 2.1) zwischen den Visualisierungen. Der Button in der Toolbar (siehe Kapitel 5, Abbildung 5.10 C), welcher diese Funktion bereits realisiert, wurde von keinem der Teilnehmer bemerkt. Stattdessen wurde das automatische Einfärben in anderen Visualisierungen erwartet. Insgesamt hatten die Teilnehmer einen positiven Eindruck vom Prototyp, mit einzelnen Verbesserungsvorschlägen.

6.5. Schlussfolgerung

Die Kantenfärbung in den Parallelen Koordinaten und das automatische Abspielen des Zeitsliders haben sich als sinnvolle Interaktionstechniken herausgestellt, die das Analysieren des Datensatzes vereinfachen. Die Kombination unterschiedlicher Visualisierungen hilft, unterschiedliche Informationen aus den Daten zu lesen, indem beispielsweise eine Pixelmap als Übersicht und ein Liniendiagramm als Detailansicht angeboten werden. Der Prototyp stellte sich damit sinnvoller als Ansatz für das Analysieren und Explorieren komplexer Datensätze mit Supercomputer-Leistungsdaten heraus. Mehr Interaktionstechniken, wie automatisches „Brushing und Linking“, sowie mehr Beschriftungen und direkte Interaktionen in der Visualisierung durch Tooltips und Kontextmenüs können dies weiter vereinfachen. Weitere Anknüpfungspunkte und Erweiterungsmöglichkeiten werden nachfolgend in Kapitel 7.2 beschrieben.

7. Zusammenfassung und Ausblick

7.1. Zusammenfassung

In heutigen Supercomputersystemen liegen eine Vielzahl unterschiedlicher Leistungsdaten in unterschiedlichen Formaten vor. Bestehende Ansätze betrachten meist nur applikationsspezifische Daten oder bieten Visualisierungen, die die Komplexität dieser Datenmenge nicht mehr erfassen können. In dieser Arbeit wurden die Probleme bestehender Lösungen vorgestellt und Visualisierungstechniken für die Analyse vieler, multivariater und zeitabhängiger Daten analysiert. Darauf basierend wurde ein Konzept für einen Prototyp entwickelt, welches sowohl applikationsspezifische Daten als auch infrastrukturelle Daten einbezieht und visualisiert. Für die Visualisierung wurden verschiedene Visualisierungen ausgewählt und implementiert, die unterschiedliche Aspekte der Daten darstellen, so dass der Benutzer sowohl die Mehrdimensionalität der Daten als auch den zeitlichen Verlauf der Daten betrachten kann und die Visualisierungen auch für solch große Datenmengen noch skalieren. Durch die Implementierung ausgewählter Interaktionstechniken kann der Benutzer diese Daten betrachten, explorieren, in verschiedenen Visualisierungen miteinander vergleichen und Detailinformationen abrufen. In einer anschließenden Nutzerstudie wurden ausgewählte Interaktionstechniken evaluiert und als sinnvoll herausgestellt und Anregungen für zukünftige Erweiterungen und Verbesserungen gesammelt.

7.2. Ausblick

Einige Ideen des Konzepts konnten in der vorgegebenen Zeit nicht umgesetzt werden, könnten in einer zukünftigen Erweiterung des Tools aber umgesetzt werden. Des Weiteren ergab die Nutzerstudie ein insgesamt positives Feedback und sinnvolle Hinweise auf Verbesserungsmöglichkeiten des Prototyps.

In der Implementierung des Prototyps wurde auf die Erweiterbarkeit der vorhandenen Visualisierungen geachtet. Diese kann weiter verbessert werden, indem das Entwurfsmuster „Factory Method“ von Johnson et al. [JHVG95] implementiert wird, das ein Interface für die Implementierung weiterer Visualisierungen bereitstellt und es ermöglicht diese sehr leicht in das bestehende Projekt einzubinden. Abbildung 7.1 zeigt ein Klassendiagramm für das Factory Method Pattern am Beispiel der Pixelmap. Die bestehenden Visualisierungen können erweitert werden, so dass ein Liniendiagramm oder eine Pixelmap auch Verläufe von Kantenwerten und nicht nur von Knotenwerten darstellen können.

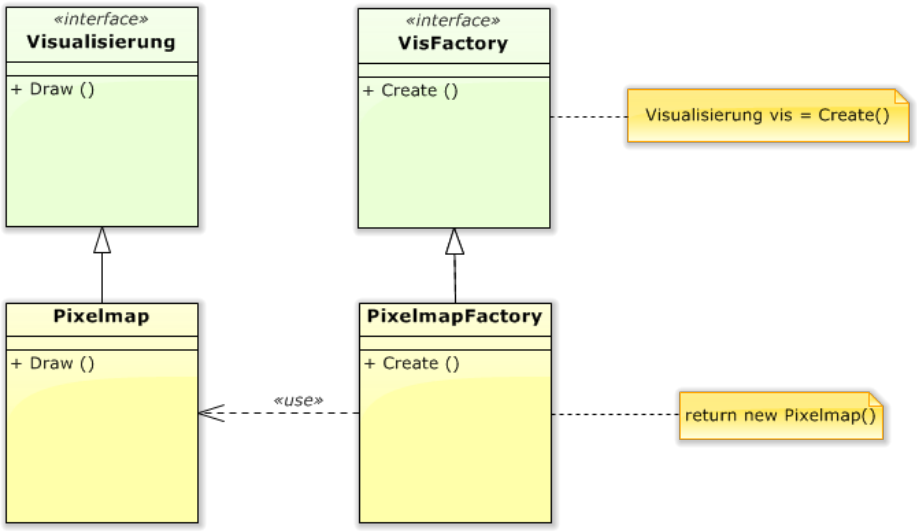

Abbildung 7.1.: Beispiel des Factory Method Pattern von Johnson et al. [JHVG95] für die Erweiterbarkeit der Visualisierungen, hier am Beispiel der Pixelmap.

Im Prototyp wurden ausgewählte Knoten- und Kantenmetriken implementiert. Der SQLiteReader kann erweitert werden und dynamisch alle in der Datenbank vorhandenen Metriken einlesen und in den Visualisierungen bereitstellen, um alle Datenaspekte betrachten zu können. Wenn die Datenbank sehr viele Daten enthält, kann es hilfreich sein, dass der Benutzer über eine Vorauswahl entscheiden kann, welche Metriken geladen werden sollen, damit unnötig lange Wartezeiten vermieden werden. Das birgt jedoch das Risiko, dass dem Benutzer wichtige Informationen entgehen, daher ist es sinnvoll, wenn in einer möglichen Vorauswahl standardmäßig alle Metriken ausgewählt sind. Das Explorieren der Daten kann durch weitere Interaktionsmöglichkeiten vereinfacht werden. Mit der Implementierung einer Undo-/Redo-Funktion, mit deren Hilfe größere Schritte rückgängig gemacht werden können, kann der Benutzer jederzeit zum vorherigen Explorationsschritt zurückkehren. Das ist beispielsweise hilfreich, wenn er versehentlich einen kleinen Zeitbereich der Daten ausgewählt hat und zur vorherigen Auswahl zurückkehren möchte, aber nicht mehr weiß, welcher Bereich das war. Dafür können ausgeführte Aktionen in einer Liste gespeichert werden, die in einer Historie angezeigt wird, ähnlich wie es im „history panel“ von Photoshop [Obe] realisiert ist. Abbildung 7.2 zeigt, wie die Historie im Prototyp aussehen könnte. Drop-Down-Menüs können dynamisch angepasst werden, indem sich die Inhalte des Filters dynamisch an die Vorauswahl anpassen, damit der Benutzer nur sinnvolle Kombinationen filtern kann und die Menüs nicht überladen wirken.

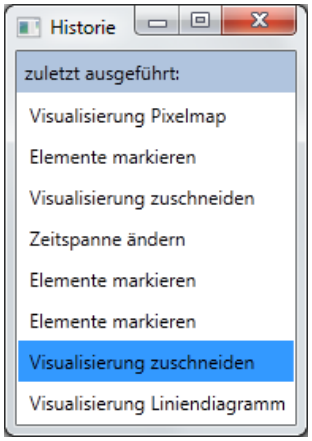

Abbildung 7.2.: Beispiel für eine Historie im Prototyp, die die zuletzt ausgeführten Aktionen anzeigt und es ermöglicht zum angeklickten Schritt zurückzuspringen.

In Visualisierungen, die Zeitspannen darstellen, kann ein Schieberegler für Zeitspannen, wie er in Kapitel 5.1 beschrieben wird, implementiert werden, um ganze Zeitspannen zu verschieben. So kann zum Beispiel die Pixelmap horizontal gescrollt werden und nicht nur ein fester Zeitbereich angezeigt werden. Die in einer Visualisierung verwendete Farbskala kann neben oder unter der Visualisierung angezeigt werden und dem Benutzer die Option gegeben werden, diese Skala an andere Farbwerte anzupassen. Im Prototyp kann der Benutzer mehrere Visualisierungen vergleichen, indem er die Fenstergrößen ändert und nebeneinander anordnet. Dies kann erleichtert werden, indem mehrere Visualisierungen automatisch in einem Fenster angeordnet werden können. Auf diese Weise kann auch eine Fokus+Kontext-Ansicht generiert werden, indem beispielsweise eine Pixelmap und ein Liniendiagramm in einem Fenster angeordnet werden. Markierte Bereiche in der Pixelmap können dann in detaillierterer Form im Liniendiagramm angezeigt werden. Abbildung 7.3 zeigt, wie eine Fokus+Kontext-Ansicht mit einer Pixelmap und einem Liniendiagramm aussehen kann. Für die Unterscheidung der Elemente in den Visualisierungen, wie dem Liniendiagramm, ist es hilfreich, wenn Kanten selektiert werden können und diese, beispielsweise farbig oder durch eine unterschiedliche Kantendicke, von den anderen abgehoben werden.

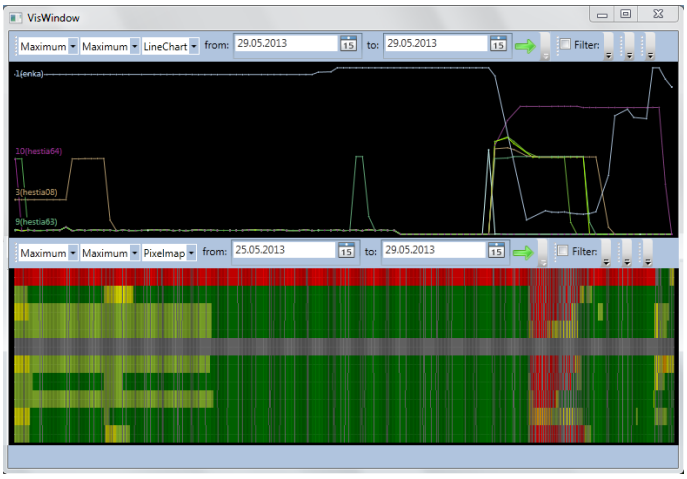

Abbildung 7.3.: Beispiel für eine Fokus+Kontext-Ansicht im Prototyp mit der Pixelmap als Übersicht und dem Liniendiagramm als Detailansicht.

A. Ein Anhang

A.1. Sequenzdiagramme

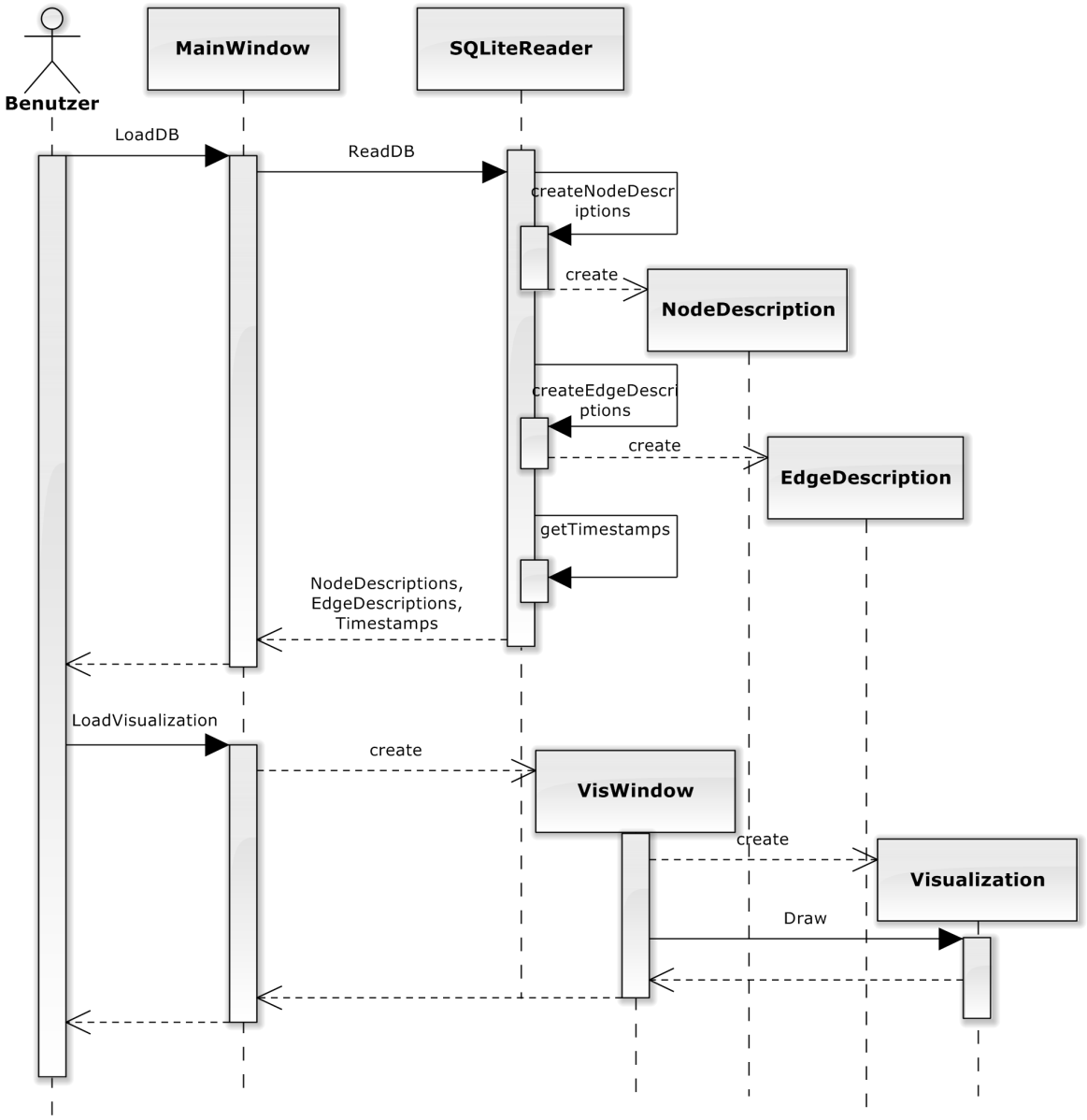

Abbildung A.1.: Sequenzdiagramm des Prototyps, das das Laden der Datenbank und Erstellen einer Visualisierung zeigt.

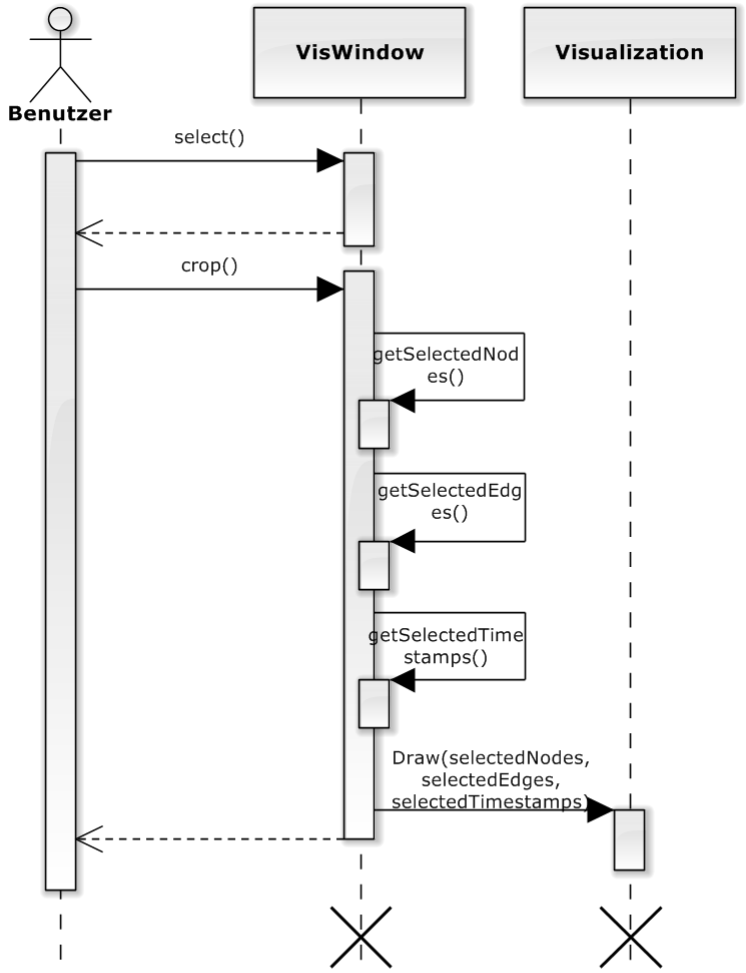

Abbildung A.2.: Sequenzdiagramm des Prototyps, das das Markieren von Elementen in einer Visualisierung und das Zuschneiden auf diese Elemente zeigt.

A.2. Evaluationsbogen

Nutzerstudie der Diplomarbeit Nr. 3483 „Interaktive Analyse von Supercomputerleistungsdaten“

Evaluationsbogen

Aufgabe 1 – Parallele Koordinaten mit Kantenfärbung

a) Parallele Koordinaten mit Standardfärbung zu einem bestimmten Zeitpunkt werden angezeigt. Finde den Rechner mit der höchsten Speicherauslastung zum dargestellten Zeitpunkt.

b) Parallele Koordinaten mit Färbung nach Speicherauslastung zu einem bestimmten Zeitpunkt werden angezeigt. Finde den Rechner mit der höchsten Speicherauslastung zum dargestellten Zeitpunkt.

Aufgabe gelöst: [ ] *ja* [ ] *nein*

gestoppte Zeit:

Frage: Lässt sich der Recher mit der höchsten Speicherauslastung durch die Kantenfärbung besser finden?

Ja, deutlich	*Ja, ein wenig*	*Kein Unterschied*	*Nein, eher schlechter*	*Nein, wesentlich schlechter*
[ ]	[ ]	[ ]	[ ]	[ ]

Anmerkungen (freiwillig):

Aufgabe 2 – Zeitslider/Autoplay

a) Parallele Koordinaten mit Standardfärbung werden angezeigt. Schalte einige Zeitschritte mit dem Zeitslider durch und beschreibe den Verlauf der CPU-Auslastung von Rechner 1.

b) Spiele nun die Autoplayfunktion ab und beschreibe den Verlauf der CPU-Auslastung von Rechner 1.

Aufgabe gelöst: [ ] *ja* [ ] *nein*

gestoppte Zeit:

Frage: Hat dir die Autoplayfunktion das Beschreiben des Verlaufs der CPU-Auslastung erleichtert?

Ja, deutlich	*Ja, ein wenig*	*Kein Unterschied*	*Nein, eher schlechter*	*Nein, wesentlich schlechter*
[ ]	[ ]	[ ]	[ ]	[ ]

Anmerkungen (freiwillig):

Aufgabe 3 – Interaktionstechniken (crop, filtern, Tooltips)

Eine Pixelmap für die CPU-Auslastung wird angezeigt. Finde heraus, welcher Rechner im dargestellten Zeitbereich die höchste CPU-Auslastung hat (Name).

Nutzerstudie der Diplomarbeit Nr. 3483 „Interaktive Analyse von Supercomputerleistungsdaten“

Hinweis: Teilbereiche mit der Maus markieren, zuschneiden („crop“), Tooltips, Visualisierung wechseln.

Aufgabe gelöst: [ ] *ja* [ ] *nein*

gestoppte Zeit:

Anmerkungen (freiwillig):

Aufgabe 4 – Pixelmap/Liniendiagramm

Eine Pixelmap und ein Liniendiagramm für die Speicherauslastung werden angezeigt. Zeige den Bereich mit der höchsten Speicherauslastung.

Aufgabe gelöst: [ ] *ja* [ ] *nein*

gestoppte Zeit:

Frage: In welcher der beiden Visualisierungen fiel es dir leichter, diesen Bereich zu erkennen?

[ ] *Pixelmap* [ ] *in beiden etwa gleich* [ ] *Liniendiagramm*

Anmerkungen (freiwillig):

Allgemeine Fragen:

Hat dir etwas gefehlt/ Hast du etwas vermisst? Wenn ja, was?

Wie fandest du die Bedienung des Tools? (angenehm zu bedienen, nervig, etc.)

Was hättest du anders gemacht?

Was fandest du unnötig?

Weitere Anmerkungen:

Literaturverzeichnis

[BCC+05] L. Bavoil, S. P. Callahan, P. J. Crossno, J. Freire, C. E. Scheidegger, C. T. Silva, H. T. Vo. Vistrails: Enabling interactive multiple-view visualizations. In *Visualization, 2005. VIS 05. IEEE*, S. 135–142. IEEE, 2005. (Zitiert auf Seite 38)

[BDW05] M. Burch, S. Diehl, P. Weißgerber. Visual data mining in software archives. In *Proceedings of the 2005 ACM symposium on Software visualization*, S. 37–46. ACM, 2005. (Zitiert auf den Seiten 15, 18 und 20)

[BTG+05] S. Biffl, B. Thurnher, G. Goluch, D. Winkler, W. Aigner, S. Miksch. An empirical investigation on the visualization of temporal uncertainties in software engineering project planning. In *Empirical Software Engineering, 2005. 2005 International Symposium on*, S. 10–pp. IEEE, 2005. (Zitiert auf Seite 19)

[BWNH01] H. Brunst, M. Winkler, W. E. Nagel, H.-C. Hoppe. Performance optimization for large scale computing: The scalable VAMPIR approach. In *Computational Science-ICCS 2001*, S. 751–760. Springer, 2001. (Zitiert auf Seite 27)

[Cha06] W. W.-Y. Chan. A survey on multivariate data visualization. *Department of Computer Science and Engineering. Hong Kong University of Science and Technology*, 8(6):1–29, 2006. (Zitiert auf den Seiten 10, 15, 17, 21, 22 und 25)

[CK98] J. V. Carlis, J. A. Konstan. Interactive visualization of serial periodic data. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, S. 29–38. ACM, 1998. (Zitiert auf den Seiten 15 und 21)

[CMS99] S. K. Card, J. D. Mackinlay, B. Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999. (Zitiert auf Seite 10)

[COM12] COMPUTERWORLD - the voice of IT management. Supercomputers face growing resilience problems. http://www.computerworld.com.au/article/442703/supercomputers_face_growing_resilience_problems/, 2012. Accessed: 2013-11-03. (Zitiert auf Seite 7)

[DEC11] DECOI 2007 - PC & Multimedia: Ratgeber und Wissen. Das Phänomen der Supercomputer. http://www.decoi2007.nl/2011/02/das-phaenomen-der-supercomputer/, 2011. Accessed: 2013-11-03. (Zitiert auf Seite 9)

[Deu12] Deutschlandradio. Rechenleistung für die Wissenschaft. http://www.dradio.de/dlf/sendungen/forschak/1762855/, 2012. Accessed: 2013-10-14. (Zitiert auf Seite 9)

[DF08] S. B. Davidson, J. Freire. Provenance and scientific workflows: challenges and opportunities. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, S. 1345–1350. ACM, 2008. (Zitiert auf Seite 38)

[DFN02] C. Daassi, M.-C. Fauvet, L. Nigay. Multiple visual representation of temporal data. In *Database and Expert Systems Applications*, S. 701–709. Springer, 2002. (Zitiert auf Seite 9)

[DLR09] G. M. Draper, Y. Livnat, R. F. Riesenfeld. A survey of radial methods for information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 15(5):759–776, 2009. (Zitiert auf Seite 20)

[DR11] A.-S. Dadzie, M. Rowe. Approaches to visualising linked data: A survey. *Semantic Web*, 2(2):89–124, 2011. (Zitiert auf den Seiten 15 und 16)

[ED07] G. Ellis, A. Dix. A taxonomy of clutter reduction for information visualisation. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1216–1223, 2007. (Zitiert auf Seite 11)

[EST08] N. Elmqvist, J. Stasko, P. Tsigas. DataMeadow: a visual canvas for analysis of large-scale multivariate data. *Information visualization*, 7(1):18–33, 2008. (Zitiert auf den Seiten 33 und 34)

[FFMK12] F. Fischer, J. Fuchs, F. Mansmann, D. A. Keim. BANKSAFE: A Visual Situational Awareness Tool for Large-Scale Computer Networks. *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, October 14–19 2012. (Zitiert auf den Seiten 36 und 37)

[Fis] F. Fischer. Fabian Fischer. http://ff.cx/banksafe/. Accessed: 2013-11-03. (Zitiert auf Seite 37)

[God10] S. Godard. SYSSTAT utilities home page. http://sebastien.godard.pagesperso-orange.fr/, 2010. Accessed: 2013-11-03. (Zitiert auf Seite 11)

[GWT] GWT-TUD GmbH. VAMPIR 8.1. http://vampir.eu/. Accessed: 2013-11-03. (Zitiert auf den Seiten 27, 28 und 29)

[HHN00] S. Havre, B. Hetzler, L. Nowell. ThemeRiver: Visualizing theme changes over time. In *Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*, S. 115–123. IEEE, 2000. (Zitiert auf den Seiten 15, 18 und 19)

[HKM] D. R. Hipp, D. Kennedy, J. Mistachkin. SQLite. http://www.sqlite.org/. Accessed: 2013-11-03. (Zitiert auf Seite 46)

[Hof77] P. E. Hoffman. *Table visualizations: a formal model and its applications*. Dissertation, UNIVERSITY OF MASSACHUSETTS, 1977. (Zitiert auf Seite 22)

[How10] M. Howison. Getting Started What is HPC? http://www.brown.edu/Departments/CCV/sites/brown.edu.Departments.CCV/files/CCV-WhatIsHPC.pdf, 2010. Accessed: 2013-11-03. (Zitiert auf Seite 9)

[HSS10] D. Hansen, B. Shneiderman, M. A. Smith. *Analyzing social media networks with NodeXL: Insights from a connected world*. Morgan Kaufmann, 2010. (Zitiert auf Seite 47)

[HW12] J. Heinrich, D. Weiskopf. State of the Art of Parallel Coordinates. In *Eurographics 2013-State of the Art Reports*, S. 95–116. The Eurographics Association, 2012. (Zitiert auf Seite 17)

[JHVG95] R. Johnson, R. Helm, J. Vlissides, E. Gamma. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1995. (Zitiert auf den Seiten 65 und 66)

[Kan00] E. Kandogan. Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions. In *Proceedings of the IEEE Information Visualization Symposium*, Band 650. Citeseer, 2000. (Zitiert auf Seite 22)

[Kei02] D. A. Keim. Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8, 2002. (Zitiert auf Seite 10)

[KKEM10] D. A. Keim, J. Kohlhammer, G. Ellis, F. Mansmann. *Mastering The Information Age-Solving Problems with Visual Analytics*. Florian Mansmann, 2010. (Zitiert auf den Seiten 44 und 45)

[KL06] R. Kincaid, H. Lam. Line graph explorer: scalable display of line graphs using Focus+Context. In *Proceedings of the working conference on Advanced visual interfaces*, S. 404–411. ACM, 2006. (Zitiert auf Seite 23)

[Kos] R. Kosara. eagereyes - Visualization and Visual Communication. http://eagereyes.org/techniques/parallel-coordinates. Accessed: 2013-11-03. (Zitiert auf Seite 18)

[LL07] J. Ludewig, H. Lichter. *Software Engineering: Grundlagen, Menschen, Prozesse, Techniken*. dpunkt.verlag GmbH, 2007. (Zitiert auf Seite 11)

[Mat] MathWorks. Visualizing higher dimensional data. http://www.mathworks.de/products/demos/statistics/mvplotdemo.html. Accessed: 2013-11-03. (Zitiert auf Seite 16)

[MKJ+07] M. S. Müller, A. Knüpfer, M. Jurenz, M. Lieber, H. Brunst, H. Mix, W. E. Nagel. Developing Scalable Applications with Vampir, VampirServer and VampirTrace. In *PARCO*, S. 637–644. Citeseer, 2007. (Zitiert auf Seite 27)

[MSZ13] F. Merkle, H. Schäfer, S. Zillessen. Evaluation verfügbarer Visual Analytics Toolkits anhand von Benchmark-Datensätzen, 2013. URL http://elib.uni-stuttgart.de/opus/volltexte/2013/8389. Accessed: 2013-11-03. (Zitiert auf Seite 35)

[Nag] Nagios Enterprises. Nagios - The Industry Standard in IT Infrastructure Monitoring. http://www.nagios.com/. Accessed: 2013-11-03. (Zitiert auf den Seiten 30 und 31)

[Obe] B. Obermeier. How to Use the History Panel in Photoshop CS6. http://www.dummies.com/how-to/content/how-to-use-the-history-panel-in-photoshop-cs6.html. Accessed: 2013-11-03. (Zitiert auf den Seiten 44 und 66)

[Oet13] T. Oetiker. RRDtool - logging and graphing. http://oss.oetiker.ch/rrdtool/, 2013. Accessed: 2013-08-25. (Zitiert auf Seite 32)

[Ött08] S. Öttl. *Visualisierungs-und Interaktionsdesign für multivariate, zeitabhängige Daten in sozialen Netzwerken*. Dissertation, Universität Konstanz, 2008. (Zitiert auf Seite 9)

[PMS+98] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, B. Shneiderman. LifeLines: using visualization to enhance navigation and analysis of patient records. In *Proceedings of the AMIA Symposium*, S. 76. American Medical Informatics Association, 1998. (Zitiert auf Seite 18)

[RC94] R. Rao, S. K. Card. The table lens: merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, S. 318–322. ACM, 1994. (Zitiert auf den Seiten 15 und 24)

[Rob07] J. C. Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV'07. Fifth International Conference on*, S. 61–71. IEEE, 2007. (Zitiert auf Seite 10)

[Shn96] B. Shneiderman. *A task by data type taxonomy for information visualizations*, S. 336–3432. IEEE Symposium on Visual Languages, 1996. (Zitiert auf den Seiten 44 und 45)

[SP10] B. Shneiderman, C. Plaisant. *Designing the User Interface: Strategies for Effective Human-Computer Interaction: Fifth Edition*. Addison-Wesley Publ. Co., 2010. (Zitiert auf Seite 44)

[SSDDS+a] P.-A. Steve Schnepp (Developer, S. S. M. (Developer, Sysadmin), H. L. (Developer), G. P. (Developer). Munin. http://munin-monitoring.org/. Accessed: 2013-11-03. (Zitiert auf Seite 32)

[SSDDS+b] P.-A. Steve Schnepp (Developer, S. S. M. (Developer, Sysadmin), H. L. (Developer), G. P. (Developer). Munin. http://demo.munin-monitoring.org/munin-monitoring.org/demo.munin-monitoring.org/index.html. Accessed: 2013-11-03. (Zitiert auf Seite 32)

[STH08] C. Stolte, D. Tang, P. Hanrahan. Polaris: a system for query, analysis, and visualization of multidimensional databases. *Communications of the ACM*, 51(11):75–84, 2008. (Zitiert auf Seite 34)

[SWLL13] G.-D. Sun, Y.-C. Wu, R.-H. Liang, S.-X. Liu. A Survey of Visual Analytics Techniques and Applications: State-of-the-Art Research and Future Challenges. *Journal of Computer Science and Technology*, 28(5):852–867, 2013. (Zitiert auf Seite 45)

[Tab] Tableau Software. tableau SOFTWARE. http://www.tableausoftware.com/de-de. Accessed: 2013-11-03. (Zitiert auf Seite 34)

[TAS04] C. Tominski, J. Abello, H. Schumann. Axes-based visualizations with radial layouts. In *Proceedings of the 2004 ACM symposium on Applied computing*, S. 1242–1247. ACM, 2004. (Zitiert auf den Seiten 23 und 24)

[Vis] VisTrails. VisTrails. http://www.vistrails.org/index.php/Main_Page. Accessed: 2013-08-14. (Zitiert auf den Seiten 38 und 39)

[WAM01] M. Weber, M. Alexa, W. Müller. Visualizing Time-Series on Spirals. In *Infovis*, Band 1, S. 7–14. 2001. (Zitiert auf Seite 20)

[WMR+11] D. Weißkopf, C. Müller, G. Reina, M. Burch, C. Buchgraber, L. Bruder, D. Exner, S. Gerzmann, R. Goldberg, M. Greis, J. Hackländer, S. Leonhardt, N. Rodrigues, H. Schmauder, C. Schmid, B. Schmidt. Studienprojekt „Holistische Anzeige von Softwaresystemen“. http://www.vis.uni-stuttgart.de/lehre/details/typ/praktikum/710/46.html, 2011. Accessed: 2013-11-03. (Zitiert auf Seite 8)

Alle URLs wurden zuletzt am 03. 11. 2013 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift