Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Visualisierung approximativer und unscharfer Sortierungen auf Basis von Randbedingungen

Ba-Anh Vu

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Max Franke, M.Sc., Dr. Guido Reina, Dr. Steffen Koch
Beginn am:	23. Juni 2020
Beendet am:	13. Januar 2021

Kurzfassung

Diese Arbeit ist aus dem Bereich der Informationsvisualisierung und Digital Humanities, und beschäftigt sich mit einem visuellen und interaktiven Lösungsansatz, der entwickelt wurde, um die Sortierung von historischen Eponymdaten zu erleichtern. Diese basieren insbesondere auf verschiedenen Randbedingungen und enthalten unterschiedliche Grade an Unsicherheiten. In der Arbeit wird zunächst auf verwandte Arbeiten eingegangen, welche entweder eine ähnliche Problemstellung lösen, in gewisser Weise als Vorbild dienen, oder in sonstiger Weise relevant für diese Arbeit sind. Dann wird im Detail der entwickelte Lösungsansatz und im Zusammenhang dazu der implementierte Prototyp beleuchtet. Es wird erläutert, wie verschiedene Teilprobleme der Sortierung mit verschiedenen Funktionen angegangen werden. Hierbei wird für die Lösung vor allem auf graphische Darstellungsformen und visuelle Metaphern zurückgegriffen. Aber auch die Interaktion mit der Visualisierung durch den Nutzer stellt einen der Hauptaspekte des Lösungsansatzes dar. Interessant ist hierbei auch, wie die historischen Daten zunächst im Datenmodell modelliert werden und dann in der Visualisierung dargestellt werden. Weiterhin werden die Ergebnisse einer kleinen Expert-Feedback-Studie erläutert, worin ein Historiker als Domänenexperte sein Feedback zu den entwickelten Lösungskonzepten gibt.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Digitalisierung verändert unser Leben zunehmend. In wichtigen Bereichen wie beispielsweise in der Wirtschaft oder in der Bildung wird sie so oft beschworen. Aber auch bei ganz alltäglichen Dingen spüren wir die Veränderungen, wie z. B. beim Bezahlen im Supermarkt per Smartphone, beim Bestellen der Weihnachtsgeschenke im Online-Shop oder auch die Kommunikation über das Internet sind nicht mehr aus unserem Leben wegzudenken. Auch bei der Automobilität sehen wir durch den Fortschritt der künstlichen Intelligenz eine allmähliche Revolution, indem sie das autonome Fahren vorantreibt.

Ein weiteres Beispiel, wo die Digitalisierung nicht halt gemacht hat, sind die Geistes- und Kulturwissenschaften. Hier werden systematisch computergestützte und informationstechnische Verfahren genutzt, um die Forschungsarbeit zu unterstützen. Diese neuartige, interdisziplinäre Verbindung, von sowohl traditionellen Geistes- und Kulturwissenschaften als auch Verfahren aus der Informatik, wird auch als „Digital Humanities“ (digitale Geisteswissenschaften) bezeichnet.

Ein konkretes Problem aus diesem Gebiet, womit sich diese Arbeit beschäftigt, ist aus der Geschichtswissenschaft: Im antiken Griechenland wurden in manchen Regionen Kalenderjahre nicht wie bei uns heutzutage numerisch, sondern durch die Namen von hohen Persönlichkeiten kategorisiert, nämlich den sogenannten „Eponymen“. Die klare Sortierung von diesen Eponymen birgt jedoch viele Probleme, da keine klare Umrechnung in unsere heutigen, numerischen Kalenderjahre existiert. Die korrekte Sortierung kann nur aus einer Vielzahl von Informationsbruchstücken aus vielen verschiedenen Quellen rekonstruiert werden. Diese können beispielsweise Ereignisse darstellen, deren Sortierung bekannt ist, oder auch Quer-Referenzen oder zeitliche Horizonte. Solche Listen können sich je nach geographischer Herkunft auch voneinander unterscheiden: So gibt es in der Literatur z. B. für Athen eine andere Sortierung als für Rhodos. Ein weiteres Problem ist die teilweise stark variierende Qualität der vorhandenen Quellen. Durch all diese Randbedingungen ergeben sich also viele Unschärfen, Unsicherheiten und Freiheitsgrade, was eine korrekte Sortierung dieser Eponyme erschwert.

Bei der Lösung dieses Problems könnte auch hier die Interdisziplinarität der Digital Humanities behilflich sein: Konkret soll hierzu im Rahmen dieser Arbeit ein softwaregestützter Lösungsansatz vorgestellt werden, welcher sich verschiedener Konzepte aus der Informatik bedient — allem voran aus dem Bereich der Informationsvisualisierung. In dem eigens für diesen Zweck entwickelten Prototyp wird veranschaulicht, wie der Einsatz von verschiedenen Methoden aus der Informationsvisualisierung den Sortierungsprozess der Eponyme unterstützen kann. Dies beinhaltet nicht nur eine Vielzahl von verschiedenen visuellen Elementen und Darstellungen, sondern auch eine Reihe von Interaktionsmöglichkeiten, um die Sortierung der historischen Daten möglichst einfach und effizient zu gestalten.

In der Arbeit wird also dieser entwickelte Lösungsansatz im Detail vorgestellt: Sowohl die verschiedenen visuellen und interaktiven Elemente, als auch andere, technische Aspekte werden ausführlich beleuchtet, wie beispielsweise das verwendete Format für die Daten. Das Ergebnis ist eine softwaregestützte und interaktive Visualisierung der historischen Daten, welche deren Sortierungsprozess unterstützen soll.

Auf diese Weise könnte also auch in diesem Fall — durch die Verwendung von Techniken aus der Informatik und Informationsvisualisierung — das zuvor beschriebene Problem der approximativen und unscharfen Sortierung auf Basis der genannten Randbedingungen gelöst werden.

Um den beschriebenen Lösungsansatz näher vorzustellen, wird diese Arbeit konkret wie folgt unterteilt:

Kapitel 2 — Verwandte Arbeiten beleuchtet andere, bereits existierende Arbeiten, welche relevant für diese Arbeit sind.

Kapitel 3 — Lösungskonzept und Prototyp: Hier werden sowohl der in dieser Arbeit entwickelte Prototyp erläutert als auch die zu Grunde liegenden Konzepte, auf die der Prototyp basiert.

Kapitel 4 — Expert Feedback: Dieses Kapitel enthält das Feedback eines Experten aus der Domäne, nämlich eines Historikers, welcher sich mit der beschriebenen Problemstellung der Sortierung der Eponyme beschäftigt.

Kapitel 5 — Zusammenfassung und Ausblick: Hier erfolgt eine Rekapitulation dieser Arbeit und es erfolgt ein Ausblick auf weitere, potentielle Weiterentwicklungsmöglichkeiten des vorgestellten Lösungsansatzes.

2 Verwandte Arbeiten

In diesem Kapitel werden verwandte Arbeiten behandelt, welche jeweils auf ihre bestimmte Art und Weise relevant für diese Arbeit sind. Die Relevanz kann zum Beispiel dadurch gegeben sein, dass diese Arbeit auf deren beinhalteten Konzepten aufbaut. Aber auch einfach die bloße thematische Verwandtschaft kann es für den Leser sinnvoll machen, diese zu erwähnen, um die vorgestellten Konzepte besser in die existierende Forschung einordnen zu können und in einen passenden Kontext zu setzen.

Zu diesem Zweck wird also im Folgenden eine Auswahl an Arbeiten präsentiert — dies beinhaltet sowohl einen Umriss ihrer Inhalte als auch eine kurze Beschreibung, wie sich die Relevanz zur dieser Arbeit ergibt.

2.1 Information Visualization Reference Model

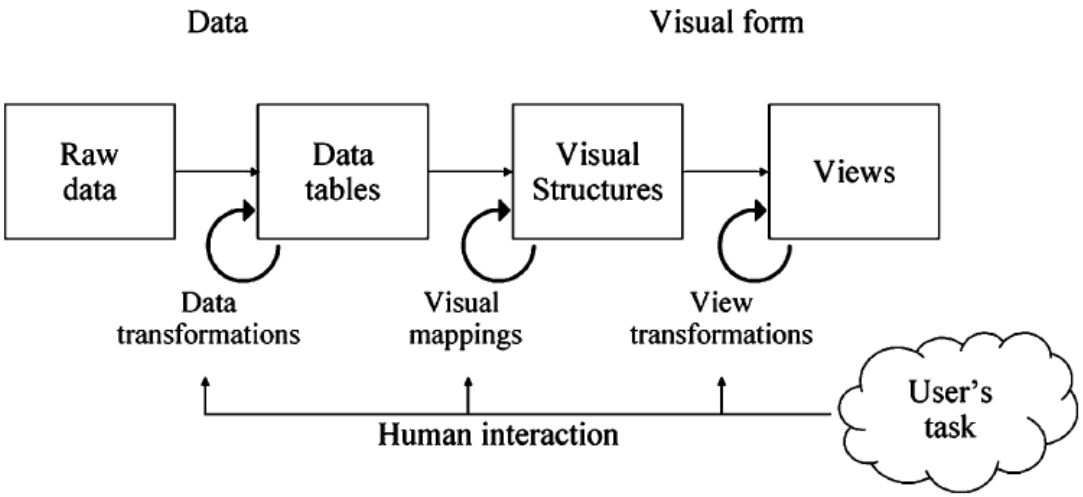


Abbildung 2.1: Das Informationsvisualisierungsreferenzmodell: Zu sehen ist die Aufteilung in die verschiedenen Zwischenschritte mit ihren jeweiligen Zwischenprodukten. Das Ziel ist es von den rohen Daten zu einer geeigneten visuellen Darstellung zu gelangen. [Car99]

Card definiert [Car99] das „Information Visualization Reference Model“ (im Folgenden mit der deutschen Bezeichnung „Informationsvisualisierungsreferenzmodell“ genannt). Dieses Modell beschreibt den Weg, wie man von naturbelassenen Rohdaten zu einer geeigneten Visualisierung

gelangen kann. Hierbei liegt eine besondere Betonung auf der Möglichkeit, dass der Nutzer in die einzelnen Zwischenschritte eingreifen kann, um so die Visualisierung wie gewünscht anzupassen (siehe Abbildung 2.1).

Zunächst stehen am Anfang die rohen Daten, welche in ihrer unveränderten Form noch nicht zur Erzeugung der Visualisierung geeignet sind. Diese müssen nun einige Zwischenschritte durchlaufen, welche durch das Informationsvisualisierungsreferenzmodell beschrieben werden: Als erstes findet die „Data Transformation“ statt, welche zum Ziel hat, die Daten auf eine einheitliche Form, die sogenannten „Data Tables“, zu bringen. Dies kann auch unter anderem eine geeignete Aggregation oder auch eine entsprechende Filterung der Daten beinhalten.

Als nächstes findet ein Mapping auf passende „Visual Structures“ statt. Hierbei werden bestimmte Werte der zuvor berechneten Datentabelle auf diese visuellen Strukturen abgebildet. Das können verschiedene visuelle Elemente sein, wie beispielsweise die räumliche Position, spezielle Farben, besondere Markierungen oder auch andere grafische Eigenschaften.

Die so erhaltenen visuellen Strukturen können dann letztendlich dafür verwendet werden, um den letzten Schritt des Informationsvisualisierungsreferenzmodells durchzuführen: Die „View Transformation“, welche die eigentlichen „Views“ (Ansichten) erzeugt. Im Wesentlichen beinhaltet dieser Schritt das Rendern der grafischen Darstellungen, kann aber auch aus anderen Operationen wie das Wechseln des Betrachtungswinkels der bestehenden Ansicht u. ä. bestehen.

Wie zuvor erwähnt kann im Visualisierungsreferenzmodell der Nutzer auch in die einzelnen Zwischenschritte eingreifen und so die resultierende Visualisierung verändern und anpassen. Je nach Zwischenschritt unterscheiden sich die möglichen Interaktionen: Bei der Datentransformation könnte das z. B. die Anpassung von Filterkriterien sein, bei den visuellen Mappings die Anpassung von Farbpaletten, und bei View-Transformationen könnte das das Verschieben oder das Drehen der aktuellen Darstellung sein.

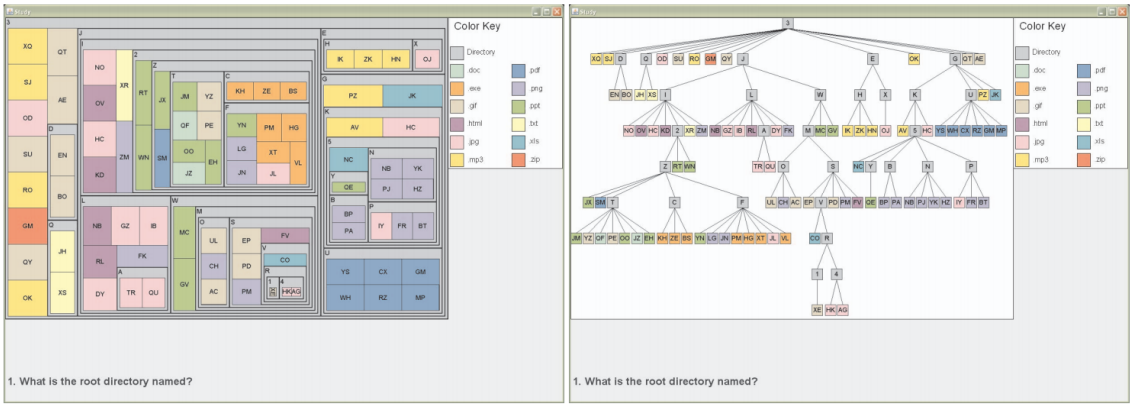
Das Informationsvisualisierungsreferenzmodell ist insofern für uns relevant, weil es im Grunde genommen genau wie zuvor beschrieben in dem entwickelten Ansatz dieser Arbeit Anwendung findet. Einzig der erste Schritt, die Datentransformation der rohen Daten, fällt in dieser Arbeit in den Hintergrund. Das rührt daher, dass die zahlreichen historischen Quellen, welche die rohen Daten darstellen, korrekt gesammelt, gefiltert und interpretiert werden müssen. Dies erfordert die Expertise von fachlich geeigneten Historikern, was im Rahmen dieser Arbeit nur nachgeahmt werden kann, indem stattdessen ein möglichst realistischer Dummy-Datensatz verwendet wird, der bereits in einer einheitlichen Form ist und so für das visuelle Mapping verwendet werden kann. Wie dies im Einzelnen funktioniert und auch wie die anderen Schritte des Informationsvisualisierungsreferenzmodells in dieser Arbeit umgesetzt werden, wird in Kapitel 3 näher beschrieben.

2.2 The Shaping of Information by Visual Metaphors

Ziemkiewicz und Kosara beschreiben die Wichtigkeit von visuellen Metaphern in der Informationsvisualisierung [ZK08]. Diese sind für die Verständlichkeit und den Nutzen einer gegebenen Visualisierung von entscheidender Bedeutung. Je nachdem, welche visuellen Metaphern für die darzustellenden Daten gewählt werden, kann es sich als leichter oder schwieriger für den Nutzer

gestalten, die resultierende Visualisierung zu lesen und korrekt zu interpretieren, was sich wiederum positiv oder negativ auf die Aufgaben auswirken kann, welche auf der gegebenen Visualisierung aufbauen. Beispiele wären hier die Exploration von den Daten oder auch deren Analyse.

Im Rahmen ihrer Arbeit haben Ziemkiewicz und Kosara ein Experiment durchgeführt, um diese Hypothese zu überprüfen. Sie entwickelten einige Fragen und Aufgaben, welche sich auf einen gegebenen Datensatz bezogen, und ließen diese von den Versuchsteilnehmern lösen. Als Hilfe standen den Teilnehmern auf Basis des Datensatzes zwei verschiedene Visualisierungen zur Verfügung: Eine Gruppe bekam sogenannte „Treemaps“, während die andere Gruppe mit „Node-Link-Diagrammen“ arbeiten durfte. Konkret ging es um die Daten bzw. die Darstellung eines Ordner- und Dateisystems (siehe Abbildung 2.2).



(a) Treemap (b) Node-link

Abbildung 2.2: Hier sind die zwei verschiedenen Visualisierungen im Experiment von Ziemkiewicz und Kosara zu sehen: a) stellt die verwendete Treemap dar; b) das verwendete Node-Link-Diagramm. [ZK08]

Beispiele für die gestellten Aufgaben waren:

- Wie heißt das oberste Wurzelverzeichnis?
- Wie viele Dateien hat das Verzeichnis x?
- Welches Verzeichnis hat eine größere Ordnertiefe, x oder y?

Die Ergebnisse des Experiments haben ergeben, dass tatsächlich ein Zusammenhang zwischen der gewählten visuellen Metapher, und der Geschwindigkeit und Korrektheit bei der Beantwortung der Fragen besteht. Je nachdem, welcher Gruppe die Teilnehmer angehörten, waren sie unterschiedlich gut: Diejenigen, die das Node-Link-Diagramm benutzt haben, haben die Fragen, die sich auf die Tiefe der Ordnerstruktur bezogen, signifikant besser und schneller beantwortet. Im Gegensatz dazu haben die Teilnehmer mit der Treemap bei den Fragen bezogen auf die enthaltenen Elemente signifikant besser abgeschnitten. Die Ergebnisse des Experiments weisen somit auf den Einfluss und die Relevanz von visuellen Metaphern bezüglich der Qualität und der Verständlichkeit einer Visualisierung hin.

Diese Erkenntnisse sind insofern für uns interessant und relevant, weil der entwickelte Ansatz dieser Arbeit sich im relativ großen Umfang an visuellen Metaphern bedient, um verschiedene Elemente und Aspekte der historischen Eponymdaten darzustellen. Dies betrifft sowohl den Basisdatensatz der Eponyme selbst, als auch implizierte, fortgeschrittenere Informationen, wie z. B. die Qualität der aktuellen Sortierung. Insofern hat dies eine große Auswirkung auf das Ziel, was mit dieser Arbeit verfolgt wird, nämlich mit Hilfe einer Visualisierung die Sortierung von großen Eponymdatensätzen zu vereinfachen und effizient zu gestalten. Welche visuellen Metaphern hierzu gewählt wurden und wie sie genau in dieser Arbeit umgesetzt sind, ist detailliert im Kapitel 3 nachzulesen.

2.3 SchemaLine: Timeline Visualization for Sensemaking

Im Abschnitt 2.2 sind wir bereits auf die Bedeutung von visuellen Metaphern in der Informationsvisualisierung eingegangen. SchemaLine [NXWW14] von Nguyen et al. stellt in gewisser Weise ein konkretes Beispiel für eine solche geschickte Verwendung von visuellen Metaphern dar. Dies kann uns insofern als Orientierung in dieser Arbeit dienen, da es sich in unserem Fall bei der Problemstellung ebenfalls um eine Visualisierung von historischen, chronologischen Daten handelt.

Bei SchemaLine geht es nämlich um eine interaktive Darstellung von chronologischen Elementen, in diesem Fall handelt es sich um Ereignisse, welche aus textuellen Daten stammen. Diese Daten können sowohl fiktiv sein, sie können aber auch realistische Ereignisse abbilden, wie z. B. ein Datensatz von realen Nachrichtenmeldungen. Mit Hilfe von automatisierten, computergestützten Verfahren, wie beispielsweise Text Mining u. ä., werden Kategorien und Gruppen gebildet und es entsteht eine entsprechende Aufteilung der Ereignisse.

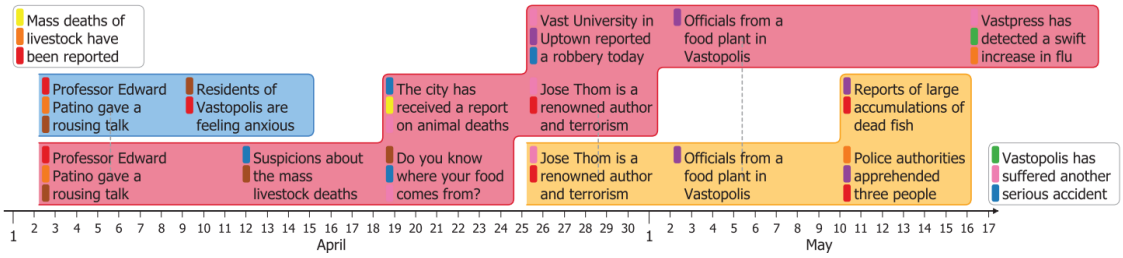


Abbildung 2.3: In der Abbildung ist ein Beispiel aus SchemaLine [NXWW14]: Hier sind verschiedene Gruppen von Ereignissen dargestellt, welche ihre eigene visuelle Metapher bekommen: Durch die klare, räumliche Abgrenzung durch die rechteckigen Formen und die zusätzliche Zuweisung von verschiedenen Hintergrundfarben lassen sich schnell die einzelnen Gruppen erkennen und zuordnen.

Diese Aufteilung erhält ihre eigenen visuellen Metaphern (siehe Abbildung 2.3): Durch die Verwendung von rechteckigen Formen, welche eine räumliche Begrenzung voneinander darstellen, und die Zuweisung von unterschiedlichen Farben, lassen sich die so aufgeteilten Gruppen von Ereignissen klar voneinander trennen. Die Ereignisgruppen wiederum enthalten dann jeweils die Ereignisse, die entsprechend zu ihrem zeitlichen Auftreten entlang der Zeitachse angeordnet ist.

Weiterhin kann es auch vorkommen, dass bestimmte Ereignisse in mehreren Gruppen vorkommen — diese werden dann zusätzlich durch eine entsprechende, vertikale Verbindung zwischen ihnen gekennzeichnet.

Tatsächlich unterscheiden sich die Domänen und die Problemstellungen von unserer Arbeit und von Nguyen et al. zu sehr, als dass man die beschriebenen visuellen Metaphern ohne weiteres übernehmen könnte. Dennoch kann ihre Arbeit uns als Vorbild dienen: Sie zeigen, wie man chronologische Daten nicht einfach nur auf einem Zeitstrahl anordnen kann, sondern dass man unter geschickter Verwendung von visuellen Metaphern und Elementen auch zusätzliche, nützliche Informationen in die Visualisierung mit hineincodieren kann, um so bestimmte Elemente besser in den Kontext setzen zu können.

Die tatsächlichen Aspekte, die letztendlich bei uns dargestellt werden sollen, unterscheiden sich durch die unterschiedlichen Domänen wesentlich von denjenigen in SchemaLine. Dennoch kann diese Arbeit uns in dieser Hinsicht als Orientierung dienen, um die zahlreichen Aspekte des Eponymdatensatzes auf geeignete Weise zu visualisieren. Ein genauerer Einblick in die verwendeten visuellen Metaphern innerhalb dieser Arbeit ist in Kapitel 3 zu finden.

2.4 A Review of Overview+Detail, Zooming, and Focus+Context Interfaces

Cockburn et al. [CKB09] diskutieren verschiedene Ansätze, wie man Kontextinformationen von unterschiedlichen Detailebenen einer Visualisierung geeignet darstellen kann. Hierzu werden vier verschiedene Ansätze vorgestellt:

- Overview+Detail
- Zooming
- Focus+Context
- Cue-basiert

Aufgrund der praktischen Anwendung innerhalb dieser Arbeit steht vor allem der Ansatz „Overview+Detail“ und „Zooming“ im Vordergrund, weshalb hierauf auch der Fokus liegen soll. Diese Ansätze werden im Folgenden näher erläutert.

Overview+Detail. Bei diesem Ansatz teilen Cockburn et al. den darzustellenden Informationsraum in eine Übersicht (engl. overview) und eine detaillierte Ansicht (engl. detail) auf, wobei hier die Besonderheit ist, dass die beiden Ansichten räumlich voneinander getrennt sind.

Wie der Name vermuten lässt, wird in der Übersicht typischerweise eine umfassende Ansicht des gesamten anzuzeigenden Informationsraums angezeigt, oder es wird zumindest eine Skalierung gewählt, die es erlaubt, einen großen Teil des Informationsraums auf einen Blick einzusehen. Hierbei werden jedoch zu jedem einzelnen Element des Informationsraums vergleichsweise wenig Details angezeigt.

Dagegen stellt die Detailansicht nur einen relativ kleinen Bruchteil des Informationsraums dar, dafür jedoch in einer deutlich höheren, informationellen „Auflösung“, d. h. für den aktuellen Ausschnitt sind deutlich mehr Informationen und Details zu sehen. Im Gegenzug rücken jedoch die Kontextinformationen in den Hintergrund, weil hier der Teil des umliegenden Informationsraums typischerweise gar nicht dargestellt wird.

Bei Overview+Detail werden diese beiden Ansichten, die zunächst etwas gegensätzlich erscheinen, kombiniert, sodass die Vorteile beider Ansichten sich gegenseitig ergänzen und so eine effektive Gesamtvisualisierung ergeben. Ein typisches Beispiel hierfür ist (eine ältere Version von) Google Maps (siehe Abbildung 2.4).

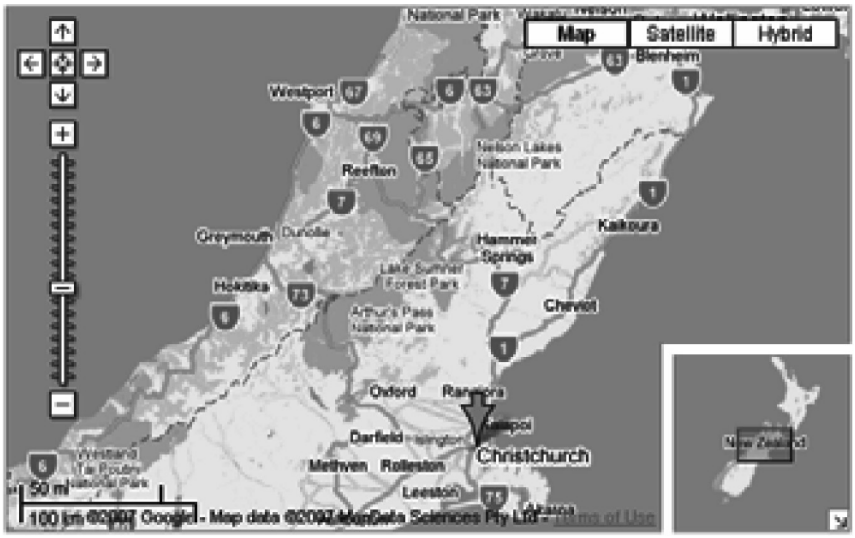


Abbildung 2.4: Zu sehen ist eine typisches Beispiel für die Anwendung von Overview+Detail: In (dieser älteren Version von) Google Maps gibt es neben der detaillierten Hauptansicht der Karte auch eine Übersicht in der rechten unteren Ecke, welche zwar weniger Details enthält, dafür aber einen größeren Kartenausschnitt darstellt. [CKB09]

Hier wird eine Kombination der beschriebenen Ansichten verwendet: In der großen Hauptansicht werden alle Elemente dargestellt, welche typisch für eine Karte sind, wie z. B. Städtenamen, Straßen und deren Namen bzw. Nummern usw. Es fehlt jedoch die Kontextinformation der umliegenden Gebiete des gezeigten Ausschnitts.

An dieser Stelle kommt die Ansicht ins Spiel, welche die Übersicht enthält (Abbildung 2.4, rechte untere Ecke): Hier wird ein größerer Ausschnitt der Karte gezeigt, der dem Nutzer dabei hilft, den Ausschnitt aus der Hauptansicht geographisch einzuordnen zu können. In der Übersicht fehlen zwar Informationen, wie z. B. Städtenamen und Straßen, jedoch kann der Nutzer hierzu auf die Hauptansicht zurückgreifen. Auf diese Weise ergänzen sich die beiden Ansichten und ergeben so eine effektive Visualisierung von Karteninformationen.

Diese beschriebene Kombination einer detaillierten Ansicht mit einem separaten Übersichtsbereich wird auch in dieser Arbeit angewendet. Die große Anzahl an zu sortierenden Eponymen erfordert eine geschickte Darstellung von Detailinformationen, ohne dass dabei der Nutzer wichtige

Kontextinformationen aus dem Blick verlieren soll. Dazu soll in der Manier von Overview+Detail eine ähnliche Darstellung wie in dem vorangegangenen Beispiel verwendet werden (mehr dazu in Kapitel 3).

Zooming. Ein weiterer Ansatz, der in der Arbeit von Cockburn et al. vorgestellt wird, wird als „Zooming“ bezeichnet. Hier gibt es zwar auch verschiedene Detaillevel an angezeigten Informationen wie bei Overview+Detail, jedoch liegt hier der Fokus auf einer einzigen Ansicht, statt auf mehreren getrennten Bereichen.

Zu den verschiedenen Detailstufen an Informationen gelangt der Nutzer mittels Interaktion mit der Visualisierung, d. h. er kann interaktiv zu einer Ansicht gelangen, die entweder vergrößert ist und mehr Informationen enthält (hereinzoomen), oder verkleinert ist, dafür aber einen größeren Ausschnitt darstellt (herauszoomen). Typischerweise gibt es hier nicht nur zwei verschiedene Stufen an Details wie bei Overview+Detail, sondern mehrere oder gar ein kontinuierlicher Übergang an Detailstufen (stufenloser Zoom). Auf diese Weise kann der Nutzer zwischen den verschiedenen Stufen hin und her wechseln, je nachdem ob er gerade mehr Detailinformationen benötigt oder eher einen größeren Ausschnitt mit mehr Kontextinformationen.

Google Maps bietet auch hier wieder eine Veranschaulichung dieses Ansatzes (siehe Abbildung 2.5): Je nach eingestellter Zoomstufe, enthält die Kartenansicht entweder einen größeren Ausschnitt mit mehr Kontextinformationen (links), oder einen kleineren Ausschnitt, dafür aber mit mehr Details (rechts).

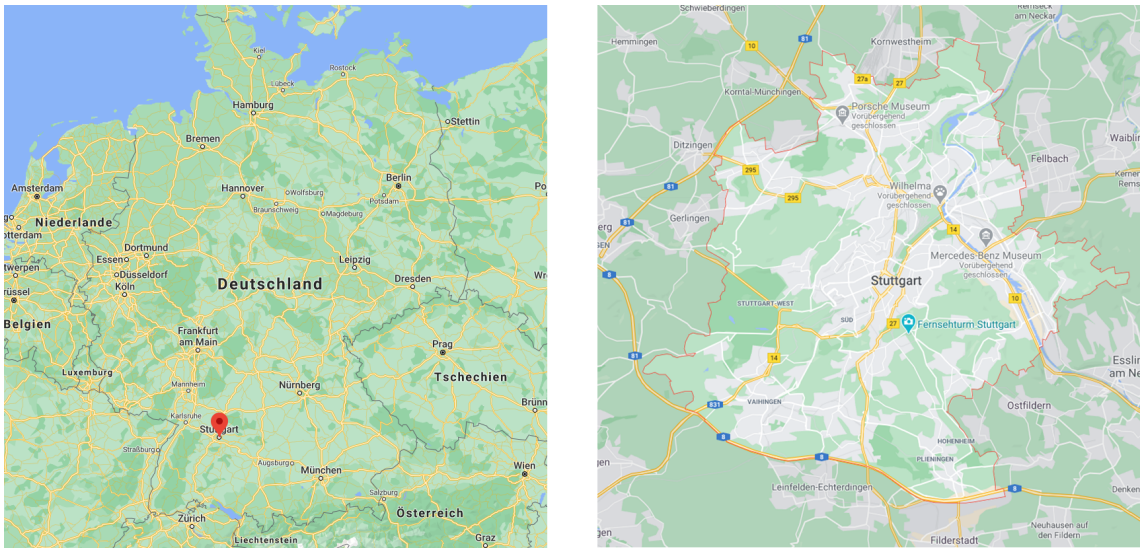


Abbildung 2.5: Wie von Cockburn et al. im Zooming-Ansatz beschrieben, sind hier verschiedene Zoomstufen zu sehen: Links herausgezoomt mit Blick auf ganz Deutschland, rechts hereingezoomt auf die Stadt Stuttgart mit mehr Details zu Stadtteilen, Autobahnen usw. [Goo20]

Diese Art von Interaktion und Darstellung von Detail- und Kontextinformationen soll auch in dieser Arbeit Anwendung finden. Je nach Situation des Sortierungsprozesses der Eponyme kann entweder der Bedarf an Detailinformationen oder an Kontextinformationen größer sein. Mit Hilfe einer Zoomfunktion, wie von Cockburn et al. beschrieben, soll der Nutzer interaktiv die für ihn adäquate Menge an Detail- bzw. Kontextinformationen anpassen können (näheres dazu in Kapitel 3).

2.5 Toward a Deeper Understanding of the Role of Interaction in Information Visualization

In dieser Arbeit von Yi et al. [YKSJ07] wird die Bedeutung von Interaktionen mit einer gegebenen Informationsvisualisierung hervorgehoben. In der Vergangenheit wurde zwar schon in anderen Arbeiten über die Interaktion des Nutzers mit Visualisierungen geschrieben [BMMS91] [DE98] [KHG03] [Twe97], jedoch hat man sich hier auf einem relativ niedrigem Abstraktionslevel bewegt, d. h. die möglichen Interaktionen wurden eher auf technischer Ebene betrachtet, wobei die eigentliche Motivation durch den Nutzer etwas in den Hintergrund rückte.

Yi et al. wollen in ihrer Arbeit hingegen auf eine höhere Abstraktionsebene gehen, und bei den Interaktionen den Nutzer mehr in den Vordergrund rücken. Hierzu definieren sie 7 verschiedene Oberkategorien, die die Interaktionen mit der Visualisierung nach der Motivation des Nutzers und dessen eigentlichem Ziel einteilen. Diese Kategorien lauten wie folgt:

- Selektieren
- Explorieren
- Rekonfigurieren
- Encodieren
- Abstrahieren/Elaborieren
- Filtern
- Verbinden

Selektieren. Bei dieser Interaktionskategorie hat der Nutzer das Ziel, ein gewünschtes Element der dargestellten Daten auf bestimmte Art und Weise hervorzuheben, sodass es einfacher ist dieses in der Menge der anderen Datenelemente im Auge zu behalten und wiederzufinden.

In Abbildung 2.6 ist ein Beispiel hierfür zu sehen: Die Datenelemente werden in der Visualisierung als schwarze Punkte dargestellt. Möchte man hier, vor allem bei einer Veränderung der Punktewolke, bestimmte Datenpunkte wiederfinden, kann sich das als außerordentlich schwierig gestalten. Durch das Selektieren kann an dieser Stelle eine rotes Label als visuelle Hervorhebung angebracht werden, sodass das Verfolgen bestimmter Punkte deutlich einfacher wird.

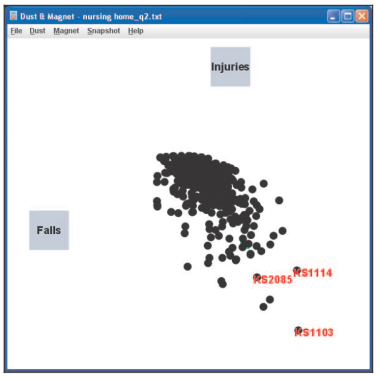


Abbildung 2.6: In der Menge der schwarzen Datenpunkte sind bestimmte, *selektierte* Elemente mit einem roten Label hervorgehoben, sodass diese leichter wiederzufinden sind. [YKSJ07]

Explorieren. Das Ziel der Interaktionen dieser Kategorie besteht im Wesentlichen darin, eine andere Untermenge an Datenelementen darzustellen als gerade in der Visualisierung angezeigt wird. Diese Interaktion wird typischerweise bei Visualisierungen benötigt, da durch verschiedene Einschränkung — entweder durch die Visualisierung selbst oder auch durch die kognitiven Grenzen des Nutzers — nur eine begrenzte Teilmenge der Daten angezeigt werden kann. Durch eine Explorieren-Interaktion wird es dem Nutzer möglich, die Teilmenge der gezeigten Daten dynamisch zu ändern und nach den aktuellen Bedürfnissen anzupassen.

Eines der häufigsten Interaktionen aus dieser Kategorie ist das sogenannte „Pannen“ (deutsch: schwenken). Hierbei wird einfach der aktuelle Ausschnitt der gezeigten Visualisierung räumlich verschoben, sodass sich als Resultat auch der Ausschnitt der gezeigten Datenelemente verändert. So kann der Nutzer auf einfache Weise die gesamte Datenmenge explorieren, obwohl die Visualisierung zunächst nur einen bestimmten Ausschnitt davon darstellen kann.

Rekonfigurieren. Diese Kategorie bezieht sich auf Interaktionen, die die räumliche Anordnung und Platzierung der Datenelemente anpassen bzw. *rekonfigurieren*. Auf diese Weise sollen neue Perspektiven, Kontexte und Beziehungen zwischen den Daten sichtbar werden.

Ein Beispiel könnte hier die Sortierung von Daten nach einem bestimmten Attribut sein, wie in Abbildung 2.7 zu sehen. In dieser Visualisierung werden Daten von verschiedenen Fahrzeugen dargestellt. Durch die Sortierung der Daten nach der Leistung in PS, wird eine scheinbare Korrelation zwischen der Fahrzeugleistung und anderen Größen sichtbar, wie die Zylinderanzahl, der Hubraum oder das Gewicht. Ohne diese Art von Rekonfiguration der Daten wäre bei dieser Visualisierung das Finden solcher Korrelationen deutlich schwieriger.

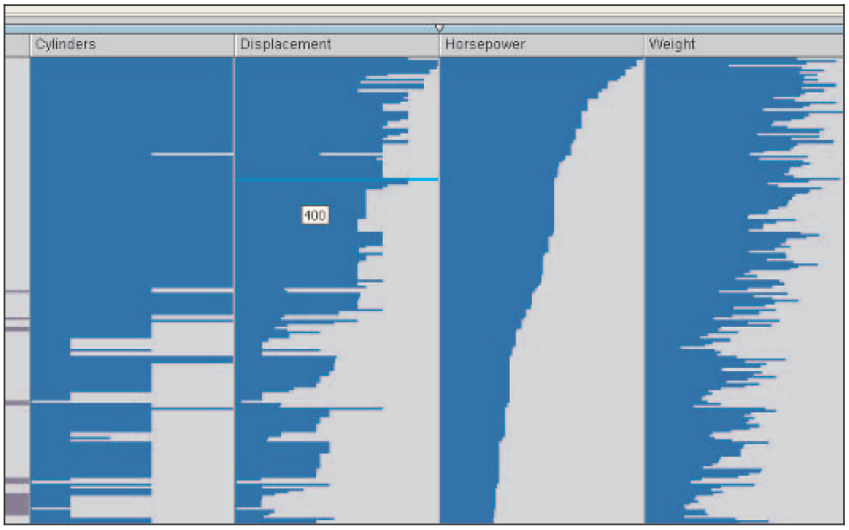


Abbildung 2.7: Durch das *Rekonfigurieren*, in diesem Fall eine entsprechende Sortierung der Datenelemente, werden Korrelationen zwischen den verschiedenen Größen der Daten sichtbar, hier: Leistung, Zylinderanzahl, Hubraum und Gewicht von verschiedenen Fahrzeugen. [YKSJ07]

Encodieren. Das Encodieren bezeichnet die Interaktionen, die sich auf die verwendeten visuellen Elemente auswirken, welche für die Darstellung der Datenmenge verwendet werden. Die visuellen Elemente können z. B. die Farbe, Form oder Größe beinhalten. Dieses Encodieren kann verschieden motiviert sein: Beispielsweise kann durch die Änderung der verwendeten visuellen Elemente eine neue Sicht auf die Daten ermöglicht werden, welche potentiell zu neuen Erkenntnissen führen könnte.

Ein anderes Beispiel könnte auch die Encodierung von farbenblindenfreundlichen Farben sein, sodass eine Visualisierung, die normalerweise für Farbenblinde nicht gemacht worden ist, im Nachhinein dann für Farbenblinde nutzbar ist (siehe auch Kapitel 3).

Abstrahieren/Elaborieren. Hier stehen die verschiedenen Ebenen an Detail- und Kontextinformationen im Vordergrund und deren dynamischer Wechsel zwischen ihnen (ähnlich wie in Abschnitt 2.4). Je nach Situation kann sich der benötigte Detailgrad der dargestellten Daten für den Nutzer ändern. In diesem Fall möchte er mit den Interaktionen aus dieser Kategorie den Detailgrad anpassen können, sodass er entweder einen größeren Datenausschnitt und Kontext sehen kann, oder aber eine detailliertere Ansicht von weniger individuellen Elementen.

Ein häufiges Beispiel für diese Kategorie ist das klassische Zoomen, wie auch schon in Abschnitt 2.4 beschrieben.

Filtern. Beim Filtern möchte der Nutzer bestimmte Datenelemente ein- und ausblenden können. Hierbei wird insbesondere nicht der betrachtete Ausschnitt der Visualisierung verändert, wie z. B. beim Explorieren, sondern die Betonung liegt auf der Verwendung von *Filterkriterien*. Diese

Kriterien können beispielsweise bestimmte Werteintervalle oder andere logische Bedingungen darstellen. Datenelemente, welche die Filterkriterien nicht erfüllen, können dann z. B. ausgeblendet werden, oder zumindest schwächer dargestellt als die Zielelemente.

In Abbildung 2.8 ist hierfür ein Beispiel aus dem Attribute Explorer [ST98] zu sehen: In a) befindet sich der Zustand vor der Anwendung des Filters. In b) ist die Visualisierung nach dem Filtern zu sehen, wobei die eigentlich herausgefilterten Elemente sichtbar bleiben, aber dafür schwächer dargestellt werden, um so die Kontextinformationen für den Nutzer zu erhalten.

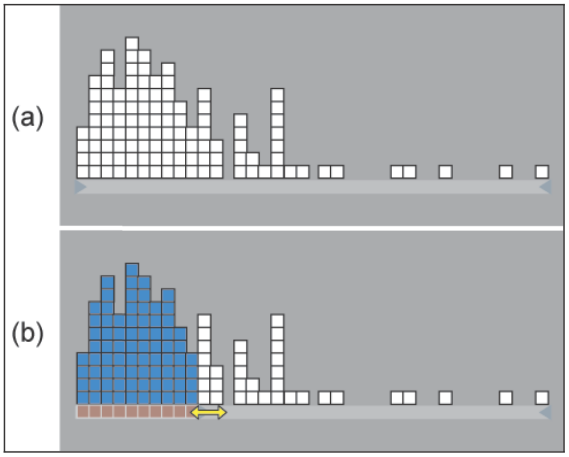


Abbildung 2.8: Hier ist eine mögliche Darstellung nach dem Filtern zu sehen: Nachdem in a) der Filter angewendet wurde, wird in b) das Resultat angezeigt. Hierbei ist die Besonderheit, dass die eigentlich herausgefilterten Daten trotzdem schwächer dargestellt werden, damit die Kontextinformationen erhalten bleiben. [YKSJ07]

Verbinden. Bei dieser Interaktionskategorie geht es um die Darstellung von Relationen von Datenelementen. Ein Beispiel hierfür kann sein, dass eine Visualisierung mehrere Ansichten eines gleichen Datensatzes verwendet, wie das z. B. in Spotfire [Ahl96] der Fall ist (siehe Abbildung 2.9).

Hier kann es schwierig sein, von einem bestimmten Datenelement einer Ansicht, das Pendant in der anderen Ansicht zu finden. Mit Hilfe von sogenannten „Brushing“-Interaktionen und geeigneten visuellen Hervorhebungen kann dieses Problem gelöst werden, indem zu einem bestimmten Element der einen Ansicht auch immer das entsprechende Element der anderen Ansicht visuell unterstrichen wird, in unserem Beispiel mit einem Kreis.

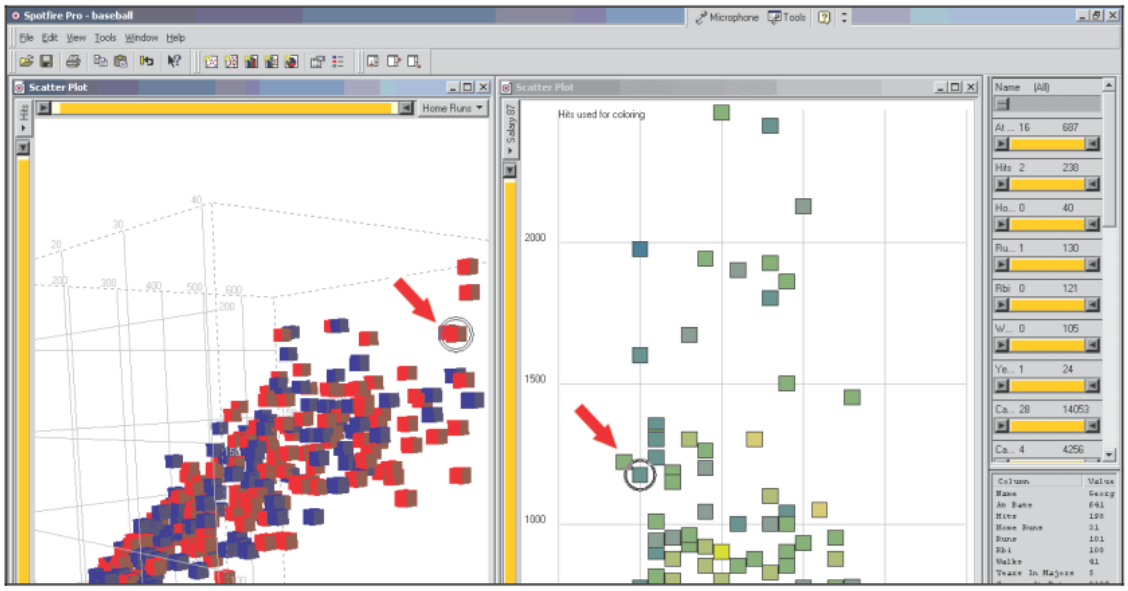


Abbildung 2.9: Ein Beispiel für die Verbinden-Interaktionen: In zwei verschiedenen Ansichten desselben Datensatzes werden die gleichen Datenelemente mit einem Kreis hervorgehoben, um diese einfacher wiederfinden zu können. [YKSJ07]

Aber auch wenn es nur eine Ansicht gibt, kann die Verbinden-Kategorie sinnvoll sein: Beispielsweise könnte in einem Personennetzwerk die Verbindung von Datenelementen durch eine „Freundschaft“-Beziehung gegeben sein. Bei der Auswahl einer bestimmten Person könnten dann z. B. die Freunde hervorgehoben werden, die eine direkte Freundschaftsbeziehung mit ihr besitzen. So könnten solche Netzwerke einfacher auf Freundschaften oder auch anderen Relationen untersucht werden.

Diese vorgestellten Arten von Interaktionen sind insofern für uns relevant, da in dieser Arbeit zu beinahe jeder genannten Kategorie eine ähnliche Interaktion entwickelt wurde. Diese erfüllen jeweils ihren speziellen Zweck und sollen auf ihre Art und Weise die Sortierung der Eponymdaten unterstützen und erleichtern (mehr dazu siehe Kapitel 3).

3 Lösungskonzept und Prototyp

In diesem Kapitel beleuchten wir detailliert den Prototypen, der im Rahmen dieser Arbeit entwickelt wurde, einschließlich der Konzepte, auf die er basiert, um die Problemstellung dieser Arbeit zu lösen. Zur Erinnerung: Das Ziel ist es, mit Hilfe eines visuellen und interaktiven Software-Ansatzes, die Sortierung von Eponymen aus einem historischen Datensatz aus dem antiken Griechenland zu unterstützen (siehe auch Kapitel 1).

3.1 Verwendete Technologien

Bevor wir zu dem eigentlichen Lösungsansatz gelangen, sollen hier die technischen Fragen geklärt werden, die den entwickelten Prototyp betreffen, vor allem im Bezug auf die verwendeten Technologien. Da der Lösungsansatz hauptsächlich Experten aus der Geschichtswissenschaft dienen soll und diese im Allgemeinen relativ wenig Kenntnisse in der Informatik haben, soll der Prototyp mit möglichst wenig technischen Vorkenntnissen benutzbar sein. Deshalb stellt der Prototyp im Wesentlichen eine Web-Anwendung dar, die in herkömmlichen Web-Browsern aufgerufen und benutzt werden kann. So soll sichergestellt werden, dass der Nutzer sich nicht selbst um die technischen Voraussetzungen kümmern muss, um die Anwendung zu benutzen, wie z. B. das Installieren von diversen Dependencies usw. Weiterhin soll die Ausführung auch möglichst plattformunabhängig und diesbezüglich konsistent sein. Für die Benutzung der Anwendung genügt lediglich das Aufrufen der entsprechenden URL[1] in einem gängigem und aktuellem Web-Browser, wie z. B. Firefox oder Google Chrome.

Die Web-Anwendung selbst basiert hauptsächlich auf HTML, CSS und JavaScript und ist als „Single-Page Application“ (SPA) gehalten. Eine Besonderheit ist die Verwendung der JavaScript-Bibliothek D3[2], mit dessen Hilfe der Großteil der entwickelten Visualisierungen umgesetzt wurde. Als weiteres Framework wurde Bootstrap[3] verwendet, das für die Gestaltung von allgemeinen Bedienelementen der Anwendung genutzt wurde, wie z. B. Menüs, Buttons usw.

Besonderes Augenmerk richtet sich auf den Datensatz, welche die historischen Informationen enthalten. Die Entwicklung des Lösungsansatzes fand in Kooperation mit Historikern statt und langfristiges Ziel war und ist es, dass die Anwendung mit einer Datenbank und mit realen Daten funktioniert. Im Rahmen dieser Arbeit stand eine solche Datenbank jedoch leider noch nicht zur Verfügung, weshalb stattdessen mit einem Dummydatensatz gearbeitet wurde, dessen Format und Inhalt den echten Daten nachempfunden ist, sodass ein späterer Umstieg auf eine reale Datenbank

[1] https://www2.visus.uni-stuttgart.de/eponyms/ (Zugang nur mit Login)
[2] https://d3js.org/
[3] https://getbootstrap.com/

keine größeren Probleme darstellen sollte. Hierbei handelt es sich konkret um einen Datensatz, welcher im JSON-Format vorliegt. Im Verlauf des Kapitels wird noch näher auf den verwendeten Datensatz eingegangen.

Weiterhin wurden noch Apache Maven[4] als Build-Management-Tool für das Software-Projekt verwendet und Git[5] als Versionsverwaltungssystem.

3.2 Domainenproblem und Datenstruktur

Um im Folgenden das Verständnis des Prototypen zu erleichtern, wird in diesem Abschnitt das Domainenproblem etwas näher erläutert. Insbesondere gehen wir auch darauf ein, wie der im vorigen Abschnitt bereits genannte Dummydatensatz darauf basiert, da die grundlegende Gestaltung des Lösungsansatzes davon abhängt.

Wie in Kapitel 1 bereits erwähnt, soll eine Menge von Eponymen in eine historisch korrekte Reihenfolge gebracht werden. Ein Eponym steht hierbei für einer Jahresperiode, die nach einer bestimmten Persönlichkeit aus der damaligen Zeit benannt ist. Diese Eponyme werden zunächst als einfache Eponym-Objekte in den Daten modelliert.

3.2.1 Constraints

Interessant wird es nun bei den bereits zuvor erwähnten Randbedingungen, die die korrekte Sortierung der Eponyme eingrenzen. Durch eine Vielzahl von verschiedenen Informationen aus unterschiedlichen historischen Quellen, lassen sich diese Randbedingungen ableiten. Diese können dabei unterschiedlich aussehen — Beispiele hierfür wären: „Eponym *A* muss zeitlich vor Eponym *B* liegen“, oder „Eponym *C* muss zeitlich nach Jahr *x* kommen“. Diese Randbedingungen müssen zum einen im Datenmodell simpel genug modelliert werden, sodass ein automatisierte Verarbeitung damit ermöglicht wird. Zum anderen muss das Datenmodell aber auch dazu in der Lage sein, alle nötigen Informationen der historischen Fakten korrekt und umfassend zu speichern.

Im Zuge dessen und in Absprache mit Historikern, mit denen im Zusammenhang dieser Arbeit kooperiert wurde, wurde ein Datenmodell definiert, welches beide Anforderungen gleichermaßen erfüllt. Hierbei fiel die Entscheidung auf die Definition von sechs verschiedenen Typen von Randbedingungen, oder im Folgenden auch *Constraints* genannt:

Before-Constraint. Dieses Constraint sagt aus, dass ein Eponym *A* **vor** einem Eponym *B* liegen muss. Der genaue zeitliche Abstand spielt hierbei keine Rolle, ausschlaggebend ist nur die korrekte Reihenfolge der beiden Eponyme, auf die sich das Before-Constraint bezieht.

[4] https://maven.apache.org/
[5] https://git-scm.com/

Separation-Constraint. Dieses Constraint bezieht sich wie das Before-Constraint auf zwei Eponyme *A* und *B*. Es sagt hierbei aus, wie groß der zeitliche Abstand zwischen den beiden Eponymen sein muss, also der Abstand, der die beiden „separiert“ (daher der Name). Dieses Constraint beinhaltet dabei sowohl einen Minimum- als auch einen Maximum-Wert. Die Reihenfolge spielt hierbei keine Rolle, ausschlaggebend ist nur der zeitliche Abstand zwischen den Eponymen.

Ein einfaches Beispiel wäre: „Zwischen Eponym *A* und Eponym *B* liegen mindestens 3, aber höchstens 6 Jahresperioden.“ Im Sonderfall, dass der Minimum- und Maximum-Wert identisch ist, muss der zeitliche Abstand genau diesen Wert haben. Der kleinstmögliche Wert für das Minimum und Maximum ist hierbei 1, was bedeutet, dass die Eponyme direkt aufeinander folgen.

Year-Before-Constraint. Dieses Constraint bezieht sich auf ein einzelnes Eponym und sagt aus, dass dieses **vor** einem bestimmten Jahr kommen muss. Zum Beispiel: „Eponym *A* muss vor dem Jahr *x* liegen.“

Year-After-Constraint. Dieses Constraint bezieht sich ebenfalls auf ein einzelnes Eponym und sagt aus, dass dieses **nach** einem bestimmten Jahr kommen muss. Zum Beispiel: „Eponym *A* muss nach dem Jahr *x* kommen.“

Leap-Year-Constraint. Dieses Constraint bezieht sich auf ein einzelnes Eponym und sagt aus, dass dieses in einem Schaltjahr liegen muss. In Verbindung mit diesem Constraint wird in den Daten auch modelliert und gespeichert, welche Jahre genau Schaltjahre darstellen und welche nicht. Wenn ein Eponym ein solches Leap-Year-Constraint besitzt, dann muss es in einem Schaltjahr liegen. Falls nicht, dann kann es zwar in einem Schaltjahr liegen, muss aber nicht zwingend. Hierbei ist nicht das konkrete Jahr ausschlaggebend, sondern nur die Tatsache, *dass* es in einem liegt.

Phyle-Constraint. Dieses Constraint sagt über ein einzelnes Eponym aus, in welcher *Phyle* es liegen muss. Eine Phyle stellt hierbei sozusagen eine Gruppierung von einzelnen Jahresperioden dar, die sich gegenseitig abwechseln. Im Zusammenhang dazu wird in den Daten auch gespeichert, welches Jahr hierbei zu welcher Phyle gehört. Drei Phylen werden dabei in den Dummydaten definiert: Kamiros, Lindos und Ialysos. Ein Phyle-Constraint könnte z. B. aussagen: „Eponym *A* muss in der Ialysos-Phyle liegen.“ In diesem Fall, muss also das Eponym *A* in einem Jahr liegen, dass zur Ialysos-Phyle gehört.

Neben den einzelnen Constraint-Typen selbst ist auch wichtig zu erwähnen, dass die Eponyme mit jeweils unterschiedlich vielen Constraints behaftet sein können, je nach Datenlage der historischen Quellen. Hierdurch können zum einen die möglichen Jahre für ein Eponym stärker eingegrenzt werden, sofern ein Eponym entsprechend viele Constraints besitzt. Zum anderen kann der mögliche Zeitraum für ein Eponym auch sehr groß sein, wenn es mangels historischer Quellen nur wenige Constraints besitzt.

3.2.2 Confidence-Levels

Die bis hierher beschriebenen Daten enthalten eine weitere Komponente in Form von sogenannten *Confidence-Levels*. Diese beschreiben, wie sicher oder unsicher ein bestimmtes Constraint ist, je nachdem wie sicher es durch historische Beweise belegt werden kann oder wie sicher die Quellen selbst sind. Hierbei werden fünf verschiedene Stufen in den Daten für die Constraints definiert:

- *Certain* — Das Constraint gilt als 100% gesichert.
- *Confident* — Das Constraint gilt zwar nicht als 100% sicher, jedoch ist es mit sehr hoher Wahrscheinlichkeit korrekt.
- *Contested* — Für das Constraint gibt es gleichermaßen Gründe anzuführen, warum es korrekt oder auch falsch sein könnte („50/50“).
- *Uncertain* — Das Constraint ist mit relativ hoher Wahrscheinlichkeit falsch.
- *False* — Das Constraint ist auf jeden Fall falsch.

Es wurde bewusst eine solche ordinale Einordnung der Confidence-Levels verwendet, da nach Aussage der Historiker, mit denen wir zusammengearbeitet haben, es schwierig ist, genaue Zahlenwerte für die Constraints festzulegen. Deshalb ist die Entscheidung auf diese offenere Abstufung gefallen.

Jedes Constraint, unabhängig vom jeweiligen Constraint-Typ, besitzt ein solches Confidence-Level. Hierdurch ergeben sich für jedes Eponym nicht nur eine Vielzahl von möglichen Zeiträumen, in denen es liegen könnte, sondern diese Zeiträume erhalten durch die Confidence-Levels auch eine Gewichtung; denn ein Zeitraum, der durch ein sicheres Constraint eingegrenzt ist, ist sicherlich höher zu bewerten als ein Zeitraum, der nur durch ein unsicheres Constraint bestimmt ist.

Auf diese Weise werden in den verwendeten Daten die Eponyme mit den zugehörigen Randbedingungen und Unsicherheiten modelliert, was als Grundlage für die weiteren Erläuterungen des Lösungsansatzes dient. Wie dieser im Detail funktioniert, wird nun in den folgenden Abschnitten beschrieben.

3.3 Übersicht der Benutzeroberfläche

Zunächst soll die allgemeine Arbeitsfläche des Lösungsansatzes in einem Überblick vorgestellt werden (detaillierte Erläuterungen von einzelnen Funktionen erfolgen noch in weiteren Abschnitten). In Abbildung 3.1 ist eine Übersicht darüber dargestellt. Der Fokus der Visualisierung stellt eine große Hauptansicht in der Mitte dar, die die Eponyme enthält. Hier kann der Nutzer sowohl den aktuellen Zustand der Sortierung der Eponyme einsehen als auch mit der Sortierung interagieren. Die Eponyme werden hierbei als Rechtecke dargestellt, welche mit ihrem jeweiligen Namen beschriftet sind (Abb. 3.1(1)).

Die Hauptansicht ist mit Hilfe von gestrichelten, horizontalen Linien in mehrere Worklanes unterteilt, welche auf der linken Seite beschriftet sind (Abb. 3.1(2)). Dabei dienen als Vorbild klassische Videoschnittprogramme, welche typischerweise ebenfalls eine Einteilung der Arbeitsfläche in mehrere Spuren verwenden, um je nach Bedarf die unterschiedlichen Szenen, Audiosequenzen usw.

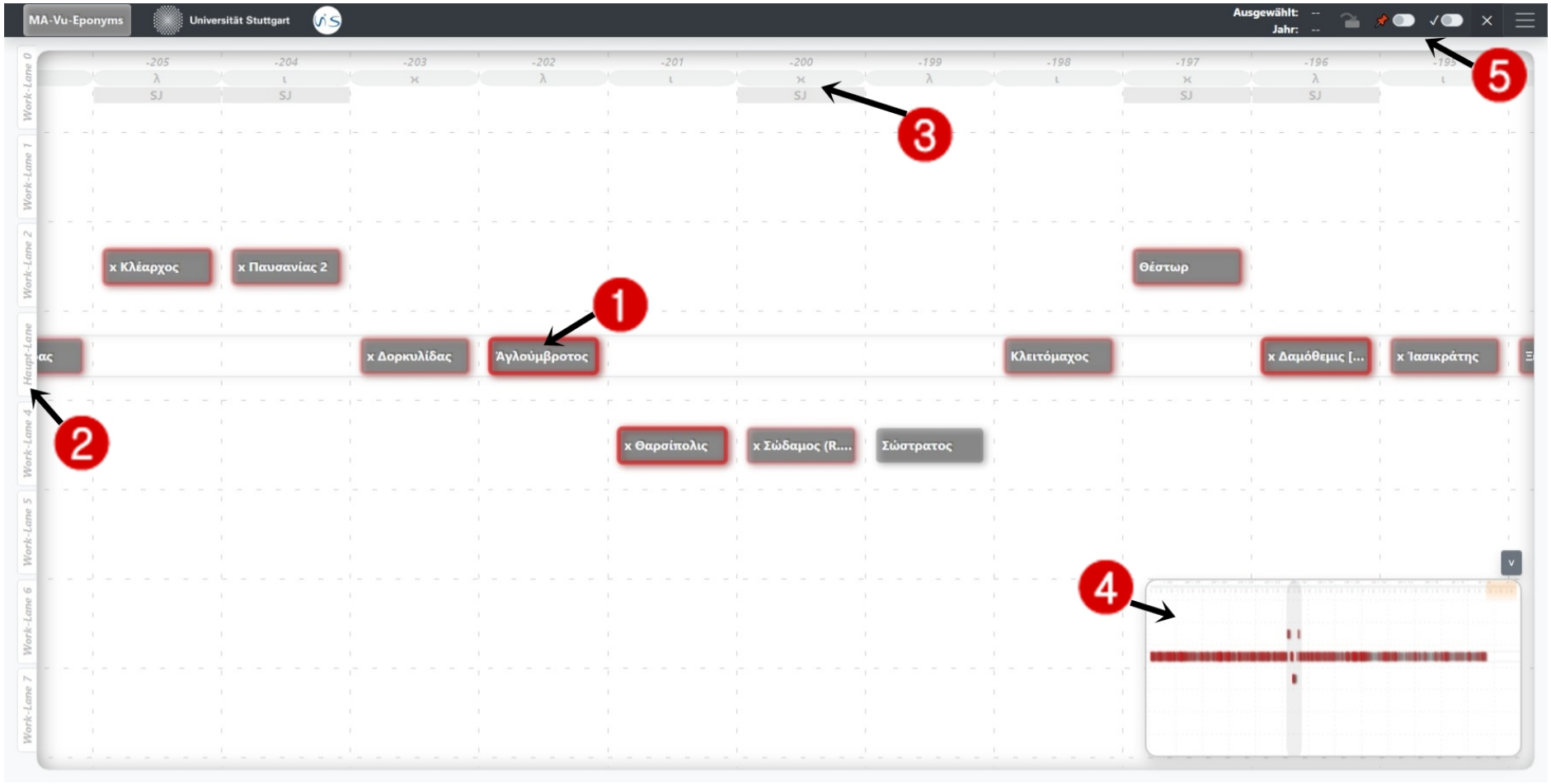


Abbildung 3.1: Zu sehen ist eine Übersicht der Arbeitsfläche: (1) Eponyme werden als beschriftete Rechtecke dargestellt. (2) Diese liegen in verschiedenen Worklanes, welche links beschriftet sind. (3) Die oberste Worklane ist für den Zeitstrahl und Informationen zu Schaltjahren und Phylen reserviert. (4) Die Minimap zeigt eine Übersicht über die gesamten Eponyme. (5) In der Toolbar sind diverse Funktionen zu der Visualisierung verfügbar.

anordnen zu können, um so dem Nutzer mehr Flexibilität beim Arbeiten zu geben (mehr dazu noch in den folgenden Abschnitten). Die Hauptworklane, als diejenige Lane in der Mitte des Sichtfeldes, wird zusätzliche durch separate, durchgezogene Linien leicht hervorgehoben. Die Eponyme können hier je nach Bedarf in verschiedene Worklanes verschoben werden.

Die oberste Worklane hat noch weitere Funktionen: Hier werden die Informationen zu dem Zeitstrahl angezeigt (Abb. 3.1(3)). Diese beinhalten zum einen die Beschriftung mit den entsprechenden Jahren (wobei Jahre v. Chr. mit negativen Jahreszahlen gekennzeichnet werden). Zum anderen werden hier auch die Schaltjahre und Phylen durch entsprechende Markierungen gekennzeichnet: Schaltjahre werden mit *SJ* markiert; Phylen mit dem entsprechenden griechischen Anfangsbuchstaben (ι = Ialysos, κ = Kamiros, λ = Lindos). Die Jahre werden hierbei mit gestrichelten, vertikalen Linien in verschiedene Spalten voneinander getrennt, worin sich die Eponyme einordnen.

In der rechten unteren Ecke befindet sich eine Minimap, welche nach dem Overview+Detail-Prinzip (Kapitel 2.4) zu jedem Zeitpunkt eine Übersicht über die gesamte Visualisierung zeigt (Abb. 3.1(4)). Weiterhin befindet sich in der rechten oberen Ecke die Toolbar, wo diverse Funktionen zu der Visualisierung verfügbar sind (Abb. 3.1(5)). Auf beide Punkte wird in den späteren Abschnitten noch genauer eingegangen.

Erläuterung der Lösungskonzepte und des Prototypen

Im Folgenden soll nun detailliert auf die einzelnen Funktionen des entwickelten Prototypen eingegangen werden. Hierbei werden nicht nur die Funktionsweisen an sich beschrieben, sondern an den entsprechenden Stellen auch die zu Grunde liegenden Prinzipien auf konzeptioneller Ebene betrachtet. Dies betrifft beispielsweise verwendete visuelle Metaphern oder auch verschiedene Interaktionen mit der Visualisierung. Wir gehen hierbei wie folgt vor: Zunächst wird ein vorliegendes Teilproblem beschrieben, das der Nutzer lösen will. Dazu wird dann ein entsprechender Teil des Prototypen beleuchtet, der als Lösung des Teilproblems dienen soll. So werden wir die verschiedenen Aspekte und Funktionen des entwickelten Lösungsansatzes nach und nach kennenlernen und auch besser in Kontext setzen können.

3.4 Übersicht und Einschätzung über aktuellen Zustand

Zu Beginn steht der Nutzer vor einer gegebenen Sortierung, mit der er potentiell gar nicht oder nur teilweise vertraut ist. Eine solche Sortierung kann beispielsweise aus der Literatur kommen, wie z. B. von Badoud [Bad15]. Der Nutzer kann aber auch einfach eine eigene Sortierung an einer bestimmten Stelle abgebrochen haben und nun fortsetzen wollen. An dieser Stelle muss er einen Überblick über den aktuellen Zustand der Sortierung gewinnen und diese einschätzen können.

Hierzu kann der Nutzer in der Hauptansicht (Abb. 3.1) zunächst die Maus verwenden, um die Ansicht herauszuzoomen, indem er entsprechend mit dem Mausrad scrollt. Wie auch im Abschnitt 2.4 beschrieben, kann er so die gewünschte Zoomstufe einstellen, um mehr oder weniger große Ausschnitte der Visualisierung einzustellen (siehe Abbildung 3.2 und 3.3).

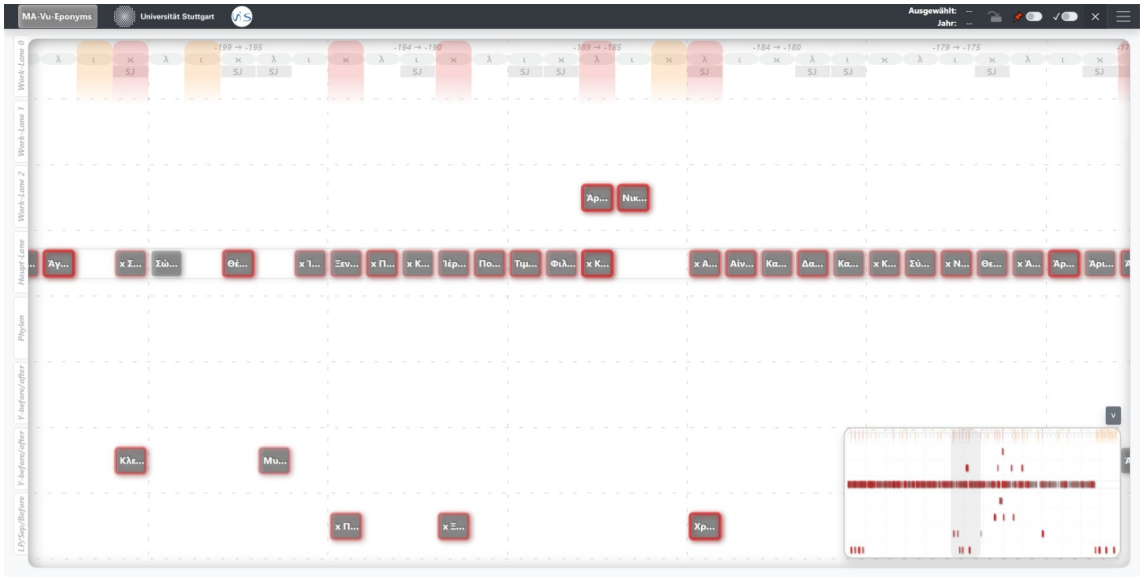


Abbildung 3.2: Per Scrollen mit dem Mausrad kann der Nutzer in der Hauptansicht das Zoomlevel einstellen. Entweder mehr hereingezoomt. . .

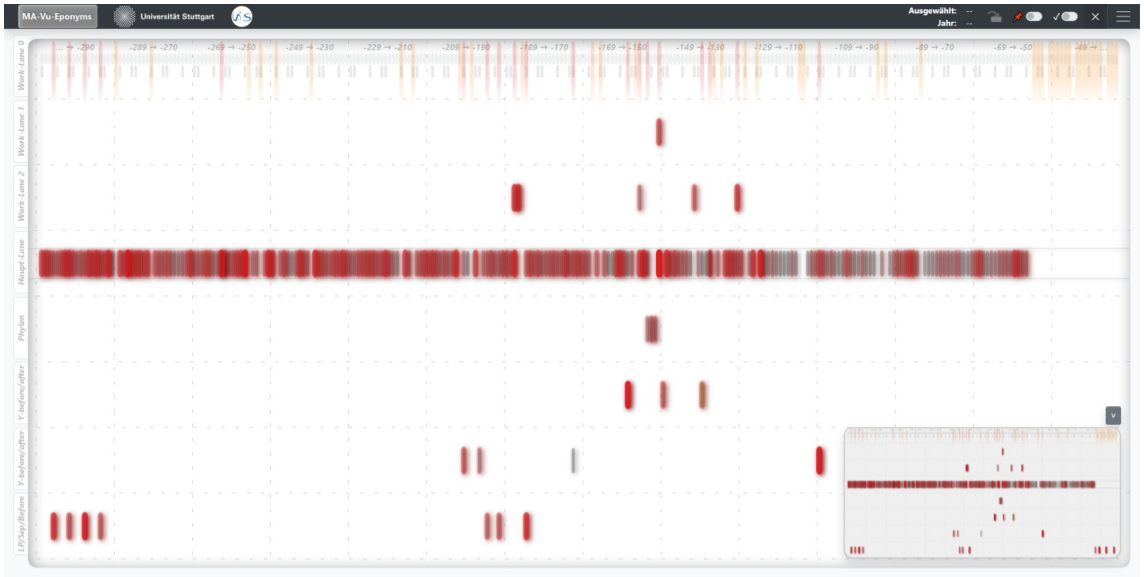


Abbildung 3.3: . . . oder weiter herausgezoomt.

Die Veränderung der Zoomstufe wirkt sich hierbei vor allem auf den horizontal gezeigten Ausschnitt aus (Zeitachse), da die Anzahl der Worklanes in der Vertikalen immer konstant ist und immer sichtbar sein sollen. Dies bewirkt, dass die Größe der Eponyme vor allem in der Horizontalen variiert: Näher herangezoomt sind diese länger, weiter herausgezoomt werden diese kürzer. Da im herausgezoomten Zustand weniger Platz für die Eponymbeschriftung vorhanden ist, wird diese entsprechend abgekürzt — im sehr weit herausgezoomten Zustand werden diese auch ganz

ausgeblendet. Um zusätzlich bei Bedarf ein einzelnes Eponym identifizieren zu können, auch wenn die Beschriftungen teilweise oder ganz ausgeblendet sind, besteht auch die Möglichkeit ein Eponym mit der Maus zu hovern; in diesem Fall wird dann die Beschriftung per Tooltip eingeblendet (siehe Abbildung 3.4).

Abbildung 3.4: Beim Hovern der Eponyme mit der Maus wird ein Tooltip mit dessen vollständigen Namen angezeigt. Alternativ kann auch einfach weiter hereingezoomt werden.

Ähnlich wie zu den Eponymen, verhält sich auch der Zeitstrahl: Dieser wird beim Zoomen entsprechend gestaucht bzw. gestreckt, die Jahresintervalle entsprechend mehr oder weniger zusammengefasst, und in Proportion dazu auch die Markierungen für die Phylen und Schaltjahre vergrößert oder verkleinert (siehe Abbildungen 3.5 und 3.6). Wie auch schon im Abschnitt 2.4 beschrieben, kann der Nutzer hier mit Hilfe von Zoomen also den gewünschten Grad an Detailinformationen oder auch Kontextinformationen einstellen.

Abbildung 3.5: Beim Zoomen ändert sich auch der Detailgrad in dem Zeitstrahl. Einmal mehr hereingezoomt. . .

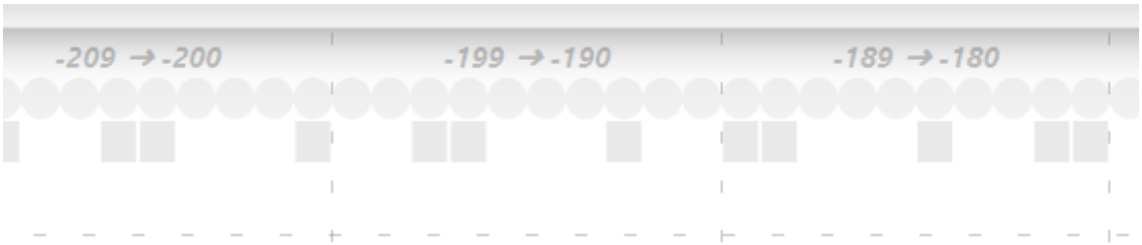


Abbildung 3.6: . . . und weiter herausgezoomt.

Um im hereingezoomten Zustand den aktuellen Ausschnitt auch verändern zu können, kann der Nutzer mit Linksklick die Hauptansicht hin und her ziehen und so den aktuellen Ausschnitt der Visualisierung verschieben. Analog zu oben funktioniert dieses Pannen nur in der Horizontalen, also in der zeitlichen Dimension, da die Worklanes fest bleiben.

Um nun den aktuellen Zustand der Sortierung nicht nur einzusehen, sondern auch einschätzen und bewerten zu können, kann sich der Nutzer an den roten Rahmen orientieren, die sich um diese herum befinden. Diese geben dem Nutzer Aufschluss darüber, wie sehr die zugehörigen Constraints eines Eponyms verletzt sind: Je intensiver und dicker dieser Rahmen ist, desto mehr von den zugehörigen Constraints werden verletzt. Hierbei werden auch die einzelnen Confidence-Levels der jeweiligen Constraints berücksichtigt: Je sicherer ein Constraint ist, desto mehr wirkt sich das auch auf den Grad der Constraint-Verletzung aus.

Anhand der Anzahl und der Intensität der roten Eponymrahmen innerhalb der Visualisierung, gewinnt der Nutzer also einen Eindruck über die aktuelle Sortierung und dessen Qualität bzw. Korrektheit im Bezug auf die Randbedingungen. So kann er nicht nur den aktuellen Gesamtzustand einschätzen, sondern hat auch direkte Anhaltspunkte, bei welchen Eponymen er mit einer Korrektur beginnen könnte.

3.5 Einsehen von einzelnen Eponymen und Constraints

Nachdem der Nutzer nun ein Eponym identifiziert hat, bei dem eine Korrektur nötig wäre, möchte er nun dessen Constraints im Einzelnen betrachten, um den Grad der Constraint-Verletzung und dessen Zustandekommen genauer einzusehen. Hierzu kann er den Auswahlmodus verwenden (ähnlich wie im Abschnitt 2.5 beschrieben): Als erstes wählt der Nutzer das gewünschte Eponym aus, indem er es mit der linken Maustaste anklickt. Zunächst wird das ausgewählte Eponym dunkler eingefärbt, um es so als aktuelle Auswahl hervorzuheben (siehe Abbildung 3.7). Weiterhin werden alle Eponyme in der Visualisierung ausgegraut, die in keiner direkten Beziehung zu ihm stehen, d. h. es gibt kein verbindendes Constraint zu ihm, wodurch diese Eponyme für dessen Constraint-Verletzung eher irrelevant sind. Trotzdem sollen diese zumindest noch leicht sichtbar sein, um den Gesamtkontext weiterhin noch einsehen zu können. Schließlich werden nun die einzelnen Constraints eingeblendet, die zu dem ausgewählten Eponym gehören, wodurch der Nutzer nun im Detail einsehen kann, welche Constraints verletzt sind und welche nicht.

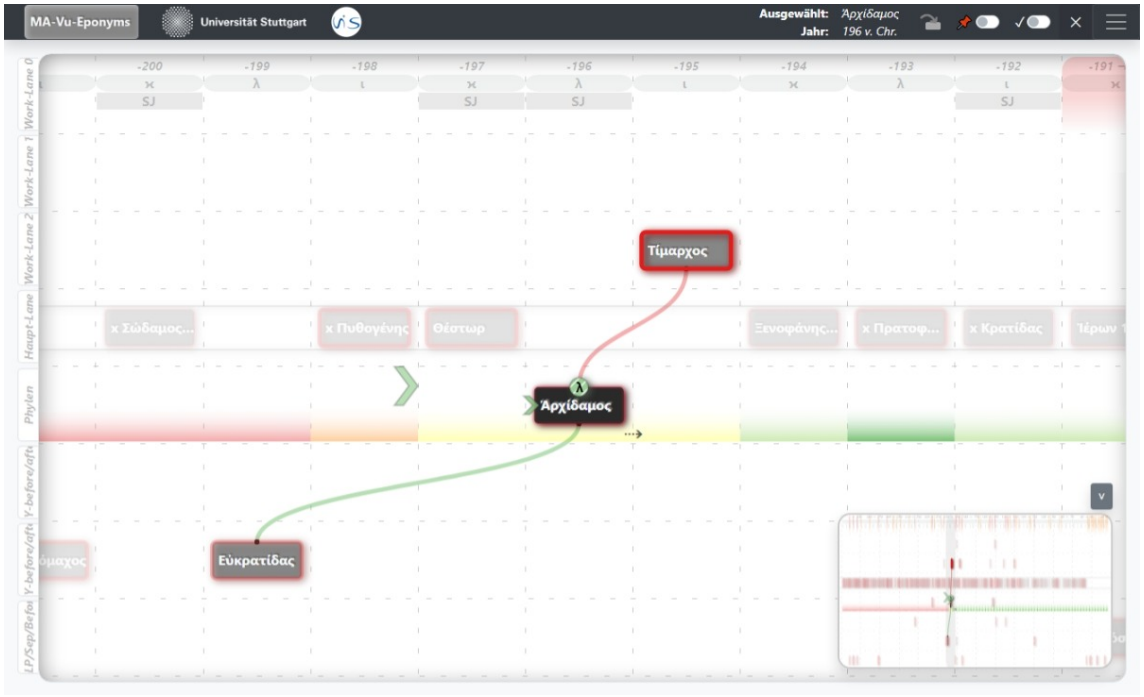


Abbildung 3.7: Der Auswahlmodus: Das ausgewählte Eponym wird mit einer dunkleren Farbe hervorgehoben und irrelevante Eponyme fallen durch das Ausgrauen in den Hintergrund. Weiterhin werden nun die zugehörigen Constraints im Einzelnen angezeigt.

Die Visualisierung der einzelnen Constraints schließt hierbei mehrere Komponenten aus den Daten ein: Den zu Grunde liegenden Constraint-Typ, den aktuellen Zustand des Erfüllt-Seins, sowie das jeweilige Confidence-Level. Diese Aspekte der Constraints sollen im Folgenden etwas genauer erläutert werden:

3.5.1 Darstellung der Constraint-Typen

Nach den im Abschnitt 2.2 beschriebenen Prinzipien, wurden für die Visualisierung der unterschiedlichen Constraint-Typen visuelle Metaphern definiert, welche möglichst einfach und intuitiv zu verstehen sind. Diese sind wie folgt:

- **Before-Constraint**: Diese Constraints werden durch eine einfache Verbindung zwischen den beiden beteiligten Eponymen dargestellt (siehe Abbildung 3.8), welche je nach Erfüllt-Status entsprechend eingefärbt sind (mehr zum verwendeten Farbschema noch in den folgenden Abschnitten).

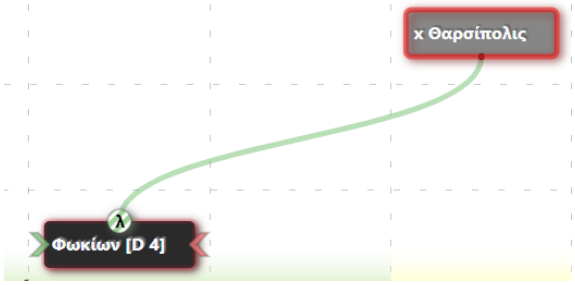


Abbildung 3.8: Die visuelle Metapher für Before-Constraints: Eine Verbindung zwischen den beteiligten Eponymen, eingefärbt je nachdem, ob das Constraint erfüllt ist oder nicht.

- **Year-Before-Constraint**: Dieser Constraint-Typ wird durch Pfeile repräsentiert, die in der Visualisierung an der entsprechenden Jahreszahl positioniert sind, um anzuzeigen, dass das entsprechende Eponym vor der betroffenen Jahreszahl liegen muss (siehe Abbildung 3.9), d. h. die Pfeil zeigen nach links.

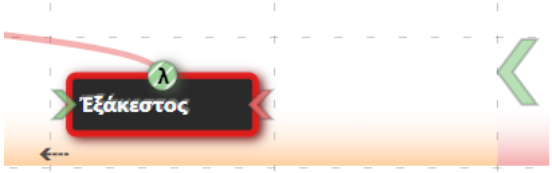


Abbildung 3.9: Die visuelle Metapher für Year-Before-Constraints: Auf der rechten Seite ist ein großer Pfeil zu sehen, der andeutet, in welche Richtung ein Eponym bezogen auf eine bestimmte Jahreszahl liegen muss (nach links).

- **Year-After-Constraint**: Ähnlich wie beim Year-Before-Constraint wird dieser Constraint-Typ durch Pfeile repräsentiert, welche an der entsprechenden Jahreszahl positioniert sind. Allerdings zeigen die Pfeile in die entgegengesetzte Richtung, also nach rechts, da das beteiligte Eponym *nach* der entsprechenden Jahreszahl liegen muss (siehe Abbildung 3.10).

Abbildung 3.10: Die visuelle Metapher für Year-After-Constraints: Links befindet sich ein großer Pfeil, der andeutet, in welche Richtung ein Eponym bezogen auf eine bestimmte Jahreszahl liegen muss (in die entgegengesetzte Richtung von Year-Before-Constraints, also nach rechts).

- **Separation-Constraint**: Diese grenzen ähnlich wie Year-Before- und Year-After-Constraints bestimmte Zeiträume ein, allerdings handelt es sich hierbei um geschlossene Zeiträume (im Gegensatz zu den „halboffenen“ Zeiträumen bei Year-Before- und Year-After-Constraints, welche in eine Richtung jeweils theoretisch unendlich groß sind). Deshalb werden hier stattdessen Rechtecke verwendet, die den gültigen Bereich eingrenzen. Aufgrund der Richtungsunabhängigkeit dieses Constraints, werden in der Visualisierung spiegelsymmetrisch auf beiden Seiten die entsprechenden Rechtecke angezeigt. Zusätzlich werden diese Rechtecke durch leichte, graue Linien mit dem relevanten Eponym verbunden, sodass ersichtlich ist, welche Eponyme beteiligt sind (nur zum nicht-ausgewählten Eponym, da das ausgewählte Eponym bereits als aktuelle Auswahl entsprechend hervorgehoben ist). Ein Beispiel ist in Abbildung 3.11 dargestellt.

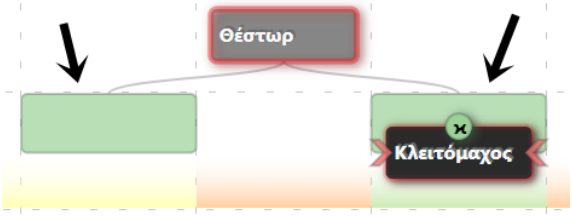


Abbildung 3.11: Die visuelle Metapher für Separation-Constraints: Der gültige Bereich für ein Eponym wird durch entsprechende Rechtecke markiert.

- **Phyle-Constraint**: Da Phyle-Constraints unabhängig von anderen Eponymen oder konkreten Jahreszahlen sind, kann dieses Constraint direkt am Eponym selbst visualisiert werden: Es wird durch einen Kreis dargestellt, welcher an dem Eponym angehängt ist (siehe Abbildung 3.12). Hierbei ist die Position am Eponym immer gleichbleibend, nämlich oben in der Mitte vom Eponym. Dies ist konsistent mit dem Zeitstrahl, wo die Position der Phyle-Markierungen ebenfalls immer oben ist (direkt unter den Jahreszahlen). Die konkrete Phylenzugehörigkeit wird, ebenfalls wie im Zeitstrahl, durch den griechischen Anfangsbuchstaben symbolisiert, welcher sich innerhalb des Kreises befindet.

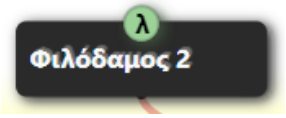


Abbildung 3.12: Die visuelle Metapher für Phyle-Constraints: Ein kleiner Kreis oberhalb des Eponyms mit dem Anfangsbuchstaben der entsprechenden Phyle.

- **Leap-Year-Constraint**: Dieser Constraint-Typ ist ähnlich zu den Phylen-Constraints unabhängig von anderen Eponymen und konkreten Jahreszahlen und kann deshalb ebenfalls direkt am Eponym visualisiert werden. Es wird durch ein Quadrat dargestellt, welches auch an dem Eponym angehängt ist (siehe Abbildung 3.13). Die Position ist hier ebenso gleichbleibend und konsistent mit dem Zeitstrahl, mit dem Unterschied, dass diese immer *unterhalb* des Eponyms ist. Genau wie im Zeitstrahl wird das Constraint mit *SJ* (für Schaltjahr) beschriftet.

Abbildung 3.13: Die visuelle Metapher für Leap-Year-Constraints: Ein kleines Quadrat unterhalb des Eponyms, beschriftet mit *SJ*.

3.5.2 Darstellung des aktuellen Erfüllt-Status.

Alle zuvor genannten Constraints beinhalten jeweils auch immer einen aktuellen Status im Bezug auf das Erfüllt-Sein des Constraints. Hierbei wird als visuelle Metapher ein einfaches Ampelsystem mit den Farben Rot und Grün verwendet. Je nachdem, ob ein Eponym in den vom Constraint eingegrenzten Jahren liegt oder nicht, erhält es eine entsprechende Einfärbung. Rot steht hier für: „Warnung, hier Bedarf es Aufmerksamkeit“ — das Constraint ist also nicht erfüllt. Grün hat entsprechend die umgekehrte Bedeutung: „Hier ist alles ok, du kannst weitermachen/weitergehen“ — das Constraint ist erfüllt. Die Einfärbung passt sich außerdem dynamisch entsprechend dem aktuellen Zustand an: Je nachdem, wie das ausgewählte Eponym verschoben wird, ändert sich entsprechend die Einfärbung der jeweiligen Constraints, abhängig vom neuen Erfüllt-Status.

3.5.3 Darstellung des Confidence-Levels.

Neben dem aktuellen Erfüllt-Status enthält jedes Constraint außerdem noch das zugehörige Confidence-Level. Um diese zu visualisieren, verwenden wir die „Kontinuität“ der Constraints als visuelle Metapher. Das Prinzip lautet hier wie folgt: Je kontinuierlicher ein Constraint dargestellt ist, desto sicherer ist es. Umgekehrt ist ein Constraint umso unsicherer, je mehr Unterbrechungen es in der Darstellungen hat. Diese Unterbrechungen werden im Prototyp konkret als Schraffurmuster dargestellt, welche abhängig vom Confidence-Level unterschiedliche Ausprägungen haben. Ist ein Constraint sehr unsicher, so ist der weiße Anteil (die Unterbrechungen) sehr groß, und der farbige Anteil entsprechend kleiner. Je sicherer ein Constraint ist, desto mehr dreht sich dieses Verhältnis um: Der weiße Anteil im Constraint schrumpft, sprich die Unterbrechungen werden kleiner, und der farbige Anteil steigt. Constraints mit dem höchsten Confidence-Level „certain“ haben hierbei entsprechend gar keine Unterbrechungen mehr und sind zur Gänze eingefärbt.

Hierbei folgen alle Constraints diesem Muster. Einzig die Before-Constraints weichen hiervon leicht ab: Da die Before-Constraints als Verbindungslinien zwischen den Eponymen dargestellt werden, ist es schwierig diese mit einem Schraffurmuster zu versehen. Deshalb werden stattdessen die Linien entsprechend gestrichelt, was nach dem gleichen Prinzip wie zuvor funktioniert: Je unsicherer ein Before-Constraint ist, desto größer werden die Lücken zwischen den Strichen. Das höchste Confidence-Level ist analog dazu durch eine vollständig durchgezogene Linie dargestellt.

3.5.4 Darstellung von Kontextinformationen.

Besondere Aufmerksamkeit bedarf es bei der Visualisierung von Kontextinformationen im Bezug auf spezielle Constraints. Insbesondere wenn der Nutzer stark in die Hauptansicht hineingezoomt ist, ist es schwierig Constraints auszumachen, die an bestimmte Positionen gebunden sind, die sich nicht im aktuellen Ausschnitt befinden. Bei Phylen-, Leap-Year- und Before-Constraints ist dieses Problem nicht so ausgeprägt, da hier entweder das Constraint direkt am Eponym dargestellt wird, oder zumindest eine Teil des Constraints beim ausgewählten Eponym startet. So ist die Wahrscheinlichkeit niedrig, dass man solche Constraints als Nutzer übersieht.

Bei Separation-, Year-Before- und Year-After-Constraints jedoch muss darauf geachtet werden, dass der Nutzer diese nicht übersieht, da diese entweder an feste Jahreszahlen gebunden sind, oder im Fall von Separation-Constraints an Eponymen, die außerhalb des aktuellen Ausschnitts liegen können. Deshalb werden direkt an den Eponymen links und rechts zusätzliche visuelle Elemente angebracht, welche den aggregierten Zustand der zuvor genannten Constraints anzeigen. Diese haben dabei dieselbe, aber verkleinerte Darstellung der Year-Before- bzw. Year-After-Constraints, um die enge Verwandtschaft der Bedeutungen zu betonen. Besitzt nämlich ein Eponym mindestens ein Year-Before-Constraint, dann wird auch der entsprechende Pfeil direkt am Eponym angezeigt. Analog dazu wird direkt am Eponym ein Year-After-Constraint-Pfeil angezeigt, sobald das Eponym mindestens ein Year-After-Constraint besitzt.

Sobald mindestens eines der Year-Before- bzw. Year-After-Constraints verletzt ist, wird auch der zuvor beschriebene Aggregationspfeil direkt am Eponym entsprechend rot eingefärbt. Nur im Fall, dass alle Year-Before- bzw. Year-After-Constraints erfüllt sind, wird der Aggregationspfeil direkt am Eponym auch grün eingefärbt. So sieht der Nutzer auch direkt am Eponym, dass bestimmte Constraints momentan verletzt sind, auch wenn diese nicht im aktuellen Ausschnitt der Hauptansicht liegen. Der Nutzer hat hierbei auch den Hinweis, in welche Richtung das verletzte Constraint liegt, da der Aggregationspfeil immer in die Richtung des verletzten Constraints zeigt. Diese Aggregationspfeile funktionieren auch für Separation-Constraints: Liegt ein verletztes Separation-Constraint vor, dann zeigt der entsprechend rote Aggregationspfeil in die Richtung des verletzten Separation-Constraint-Bereiches.

Um die Navigation zu solchen verletzten Constraints zu erleichtern, kann der Nutzer zusätzlich die Minimap in der unteren rechten Ecke nutzen. Diese zeigt, wie zuvor bereits erwähnt, einen Gesamtüberblick über die Visualisierung, sowie als graues Rechteck angedeutet den aktuell gezeigten Ausschnitt der Hauptansicht. Der Nutzer kann per Drag-and-Drop mit der Maus diesen aktuellen Ausschnitt schnell und effizient verschieben, ohne langwierig herauszuzoomen oder große Strecken in der Hauptansicht per Pannen zurücklegen zu müssen. Weiterhin erleichtert die Minimap auch das Einsehen von Kontextinformationen, da potentiell verletzte Constraints und auch alle weiteren Elemente der Visualisierung permanent dargestellt werden (wenn auch nur in verkleinerter Form). Im Fall, dass die Minimap in bestimmten Situationen stören sollte, ist diese so konzipiert, dass diese per Klick auf den entsprechenden Button über der Minimap zu (und auch wieder auf-)geklappt werden kann.

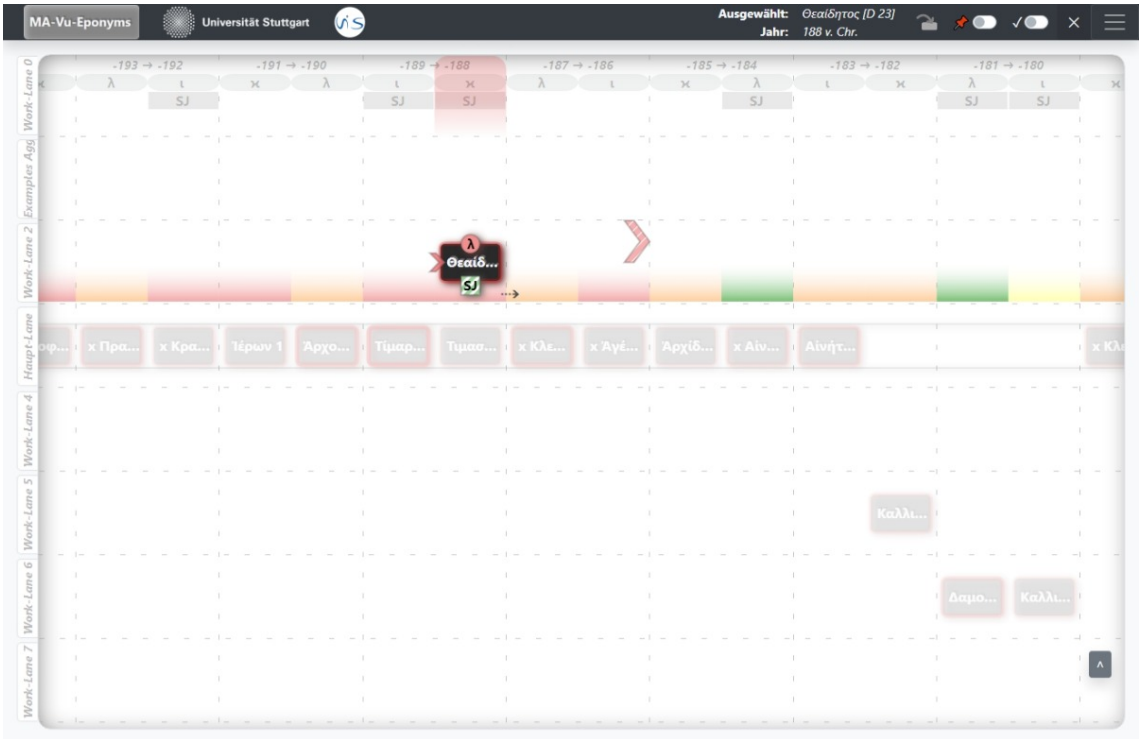


Abbildung 3.14: Beispiel für den Auswahlmodus: Das ausgewählte Eponym besitzt drei verschiedene Constraints: Ein Phyle-, ein Leap-Year- und ein Year-After-Constraint. Nicht-erfüllte Constraints sind rot eingefärbt, erfüllte grün. Sichtbar sind auch die unterschiedlichen Confidence-Levels: Je weniger ausgeprägt die Unterbrechungen bzw. das Schraffurmuster der Constraints ist, desto sicherer sind sie. Das Phyle-Constraint ist beispielsweise vollständig ausgefüllt ohne jegliche Unterbrechung, und hat somit das höchste Confidence-Level, das Eponym muss also sicher in der entsprechenden Lindos-Phyle liegen. Weiterhin ist links direkt am Eponym ein Aggregationspfeil angebracht, welcher andeutet, dass in die Zeigerichtung mindestens ein verletztes Constraint liegt. Die Minimap ist via dem Button in der unteren rechten Ecke eingeklappt, sodass Elemente dahinter bei Bedarf sichtbar werden.

Abbildung 3.15: Ein weiteres Beispiel für den Auswahlmodus: Das ausgewählte Eponym besitzt eine Vielzahl von Constraints mit unterschiedlichen Confidence-Levels, u.a. ein Year-Before-Constraint, das eine sehr niedrige Sicherheit hat (uncertain), dargestellt durch die relativ großen Unterbrechungen und dem kleinen Farbanteil. Auf der ausgeklappten Minimap ist außerdem erkennbar, wohin das linke Before-Constraint hinführt, was in der Hauptansicht nicht mehr zu sehen ist. Per Drag-and-Drop kann der Nutzer das graue Rechteck der Minimap entsprechend verschieben, um den aktuellen Ausschnit der Hauptansicht nach Bedarf zu anzupassen.

3.6 Anpassen der Sortierung

Nachdem der Nutzer ein gewünschtes Eponym und dessen zugehörige Constraints näher betrachtet hat, möchte er nun eine Anpassung der Sortierung vornehmen, mit dem Ziel, diese im Bezug auf die nicht-erfüllten Randbedingungen zu verbessern. Hierzu stehen zunächst einige grundlegende Interaktionen zur Verfügung. Der Nutzer kann mit Hilfe der Maus per Drag-and-Drop ein Eponym an eine neue Position verschieben. Hierbei werden, wie zuvor schon erwähnt, dynamisch alle Erfüllt-Zustände der zugehörigen Constraints aktualisiert, sodass der Nutzer immer unmittelbares Feedback über die Auswirkung seiner Änderung hat. Als zusätzliche Information wird in der Status-Leiste neben der Toolbar das aktuelle Jahr angezeigt, sodass dieses leicht ablesbar ist, auch wenn man weiter herausgezoomt sein sollte. Gleiches gilt für den Namen des ausgewählten Eponyms.

Sollte der Nutzer sich in dem Fall befinden, dass er den aktuellen Ausschnitt soweit verschoben hat, dass das ausgewählte Eponym nicht mehr sichtbar ist, so kann er die Eponym-Springen-Funktionen nutzen, welche in der Toolbar erreichbar ist (siehe Abbildung 3.16). Hiermit kann der Nutzer das ausgewählte Eponym an die gewünschte Position per Mausklick springen lassen, ohne nochmal zurück zur aktuellen Auswahl zurückpannen zu müssen, um es mit der Maus zu ziehen. Als Alternative kann der Nutzer auch jederzeit die gewünschte Position in der Hauptansicht direkt Doppelklicken um das Eponym dorthin springen zu lassen, was die Geschwindigkeit und die Benutzerfreundlichkeit nochmal etwas erhöht. Auf diese Weise soll dem Nutzer unnötiges Zoomen und Pannen erspart werden.

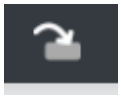


Abbildung 3.16: Button in der Toolbar zum Eponym-Springen

Eine weitere Frage, die sich dem Nutzer stellt, ist an welche Position er am besten ein bestimmtes Eponym verschieben sollte. Hierzu wurde eine visuelle Unterstützung entwickelt, die wie folgt funktioniert: Beim Selektieren eines Eponyms wird für jedes einzelne Jahr berechnet, wie sehr die zugehörigen Constraints verletzt wären, wenn es in dem jeweiligen Jahr liegen würde. Hierbei werden die Constraints mit Hilfe ihres Confidence-Levels gewichtet, woraus sich ein interner Wert für den Grad der Constraint-Verletzung für jedes Jahr ergibt. Je höher der Grad der Verletzung ist, desto mehr sollte dem Nutzer davon abgeraten werden, das Eponym dorthin zu verschieben. Um dies dem Nutzer anzuzeigen, wird als visuelle Metapher ein erweitertes Ampelsystem verwendet, welches folgende fünf Farben enthält: Rot, Orange, Gelb, Hellgrün, Dunkelgrün (in der Reihenfolge von schlecht bis gut). Dies stellt einen Kompromiss aus ausreichend vielen Abstufungen und noch unterscheidbaren Farben dar. Dunkelgrün bedeutet hier insbesondere, dass alle zugehörigen Constraints vollständig erfüllt sind.

Mit Hilfe dieses erweiterten Ampelsystems wird nun in der Worklane des ausgewählten Eponyms ein horizontales Empfehlungsband erzeugt, worin alle Jahre mit der entsprechenden Farbe markiert werden. So ist für den Nutzer direkt ersichtlich, welche Positionen anhand den Constraints in die engere Auswahl kommen würden. Für den Fall, dass der Nutzer näher hereingezoomt ist und nur einen Ausschnitt dieses Empfehlungsbands sehen kann, werden zusätzlich kleine Hinweispfeile neben dem ausgewählten Eponym angezeigt, sodass er einen richtigen Anstoß bekommt, wohin er navigieren sollte. Hierbei sei auch betont, dass der Nutzer sich nicht zwingend an diese Empfehlungen halten muss: Er kann weiterhin zu jeder Zeit die zugehörigen Constraints im Einzelnen einsehen, und mit Hilfe seiner eigenen Expertise selbstständig entscheiden, welche Position er am besten für ein Eponym hält. Das ist vor allem wichtig, wenn mehrere Jahre für ein Eponym in Frage kommen würden.

Abbildung 3.17: Hier ist ein Beispiel für die Empfehlungsbänder dargestellt: Die zugehörigen Constraints des aktuell ausgewählten Eponyms grenzen implizit unterschiedliche Bereiche ein, wo der Grad der Constraint-Verletztung variiert. Erkennbar ist dies an den Teilbereichen, die in dem Empfehlungsband unterschiedlich eingefärbt sind (das Empfehlungsband wird in der unteren Hälfte der Worklane angezeigt, die obere Hälfte ist für die einzelnen Constraints reserviert). Die Jahre, in denen alle Constraints vollständig erfüllt wären, sind mit dunkelgrün eingefärbt. Zusätzlich zeigt ein kleiner Hinweispfeil unterhalb des ausgewählten Eponyms, wohin der Nutzer am besten navigieren sollte.

Da das Verschieben eines Eponyms sich im Allgemeinen auch auf die Constraints von anderen Eponymen auswirkt, kann es für den Nutzer relevant sein, die Auswirkungen einer Änderung auf solche Eponyme einsehen zu können. Um dies dem Nutzer zu ermöglichen, wurde eine Anpinnen-Funktion umgesetzt, womit dies bewerkstelligt werden kann.

Zunächst besteht das Problem, dass die Empfehlungsbänder nur für das ausgewählte Eponym angezeigt werden können. Um auch die Empfehlungsbänder von mehreren Eponymen gleichzeitig sehen zu können, kann die Anpinnen-Funktion genutzt werden: Der Nutzer klickt das gewünschte Eponym mit Rechtsklick an, woraufhin das Eponym als angepinnt gilt. Nun wird unabhängig davon, welches andere Eponym aktuell ausgewählt ist, sein Empfehlungsband angezeigt, einschließlich der zugehörigen Constraints. Im Grunde genommen ist ein angepinntes Eponym fast wie ein selektiertes Eponym, mit dem Unterschied, dass sich alle Positionsveränderungen nur auf die Selektion beziehen. Angepinnte Eponyme sind hiervon ausgenommen, einzig deren Constraints und Empfehlungsband wird angezeigt. Da ein angepinntes Eponym wegen des Empfehlungsbandes eine

ganze Worklane beansprucht, können maximal sieben Eponyme gleichzeitig angepinnt werden (acht existierende Worklanes abzüglich eines ausgewählten Eponyms). Ein Eponym kann per nochmaligen Rechtsklicken auch wieder entpinnt werden.

Mit Hilfe der Anpinnen-Funktion kann der Nutzer also die Constraints von mehreren Eponymen gleichzeitig betrachten. Insbesondere kann er diejenigen Eponyme anpinnen, die durch ein Constraint mit der aktuellen Auswahl verbunden sind, und so deren Zustand der Constraint-Verletzung einsehen. Das gibt ihm nun die Möglichkeit, die Auswirkungen seiner Änderungen auf andere Eponyme on-the-fly sehen zu können, was ihm zusätzlichen Input dabei gibt, sich für eine bestimmte Position zu entscheiden, vor allem, wenn mehrere Möglichkeiten zur Verfügung stehen.

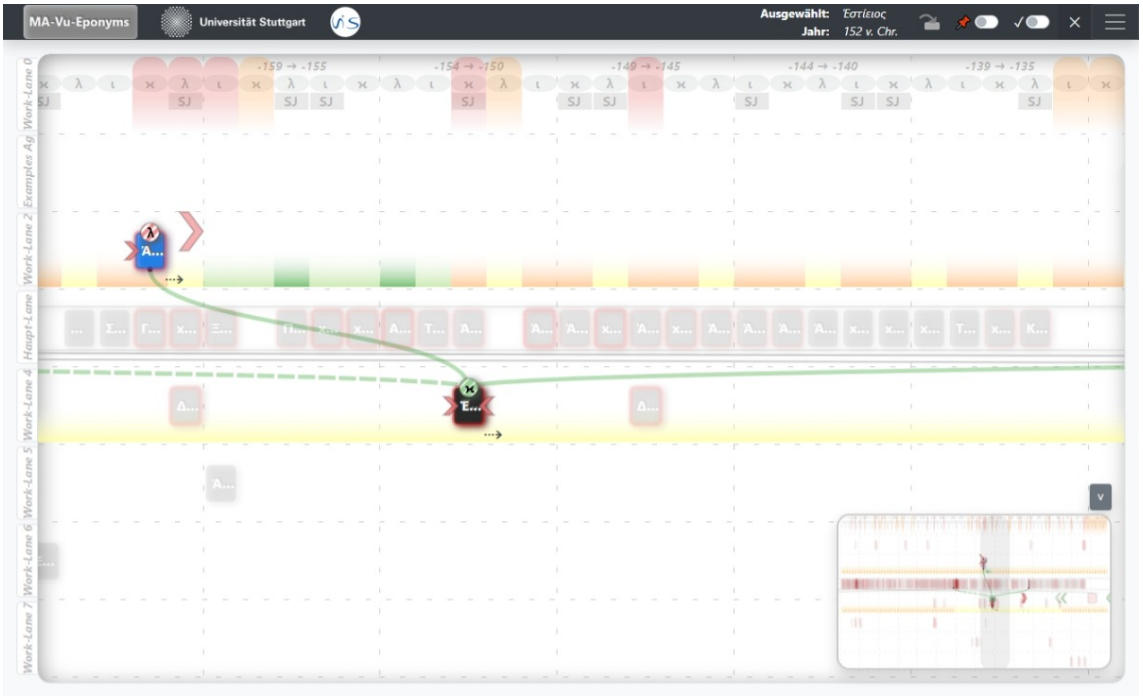


Abbildung 3.18: Hier wird ein Beispiel für das Anpinnen von Eponymen dargestellt: Durch das Rechtsklicken wird ein Eponym zunächst blau als angepinnt markiert. Gleichzeitig werden dessen Constraints eingeblendet, sowie sein Empfehlungsband. Dort kann man sehen, wie das Before-Constraint zwischen ihm und dem ausgewählten Eponym (schwarz markiert) implizit einen relativ guten Bereich eingrenzt (hellgrün/dunkelgrün).

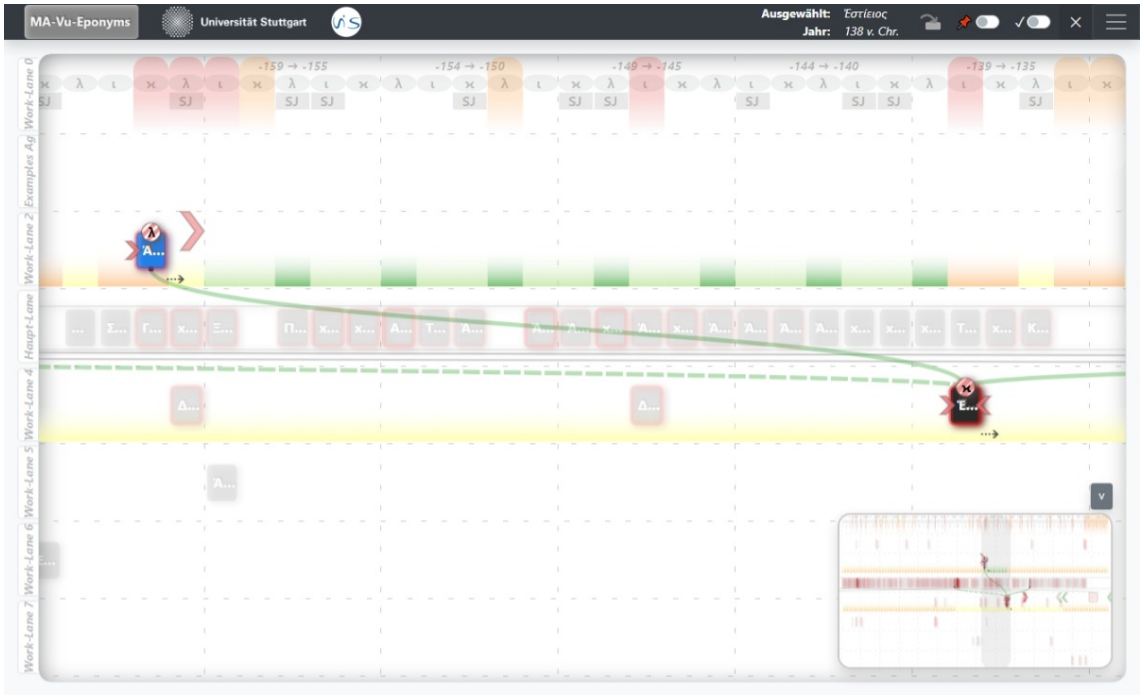


Abbildung 3.19: Nachdem das ausgewählte Eponym in die Empfehlungsrichtung verschoben wurde (angedeutet durch den kleinen Pfeil darunter), lässt sich erkennen, dass der zuvor erwähnte grüne Bereich nun größer geworden ist, was positiv für die Auswahlmöglichkeiten des angepinnten Eponyms ist.

3.7 Abschließen der Positionsanpassung eines Eponyms

Nachdem der Nutzer sich für eine neue Position für ein Eponym entschieden hat und nun erst einmal die Anpassung hierfür abschließen will, möchte er wieder den Auswahlmodus verlassen und zurück zu der Übersicht, von der wir eingangs ìn diesem Kapitel gestartet sind. Hierzu kann er die aktuelle Auswahl von Eponymen aufheben, indem er entweder den *X*-Button in der oberen rechten Ecke innerhalb der Toolbar benutzen, oder auch die *Escape*-Taste auf der Tastatur betätigt.

Wünschenswert wäre es, wenn sich der Grad der Constraint-Verletzung für das angepasste Eponym verbessert hat, sodass die rote Umrandung um das Eponym herum sich abschwächt. Im besten Fall sind nun alle zugehörigen Constraints erfüllt und der Rahmen ist nun vollständig verschwunden. Es kann aber auch durchaus vorkommen, dass der Rahmen durch eine Verschlechterung des Eponyms stärker wird — alles sind erlaubte Fälle und wird bewusst innerhalb des Prototypen zugelassen. Insbesondere kann es durchaus vorkommen, dass aufgrund der Konstellation der anderen Eponyme noch nicht alle Constraints erfüllt werden können, sodass vorerst der rote Rahmen bestehen bleiben muss, welcher — im unbelassen Zustand — in Zukunft den Eindruck erwecken könnte, dass er noch nicht behandelt worden ist.

Um dem entgegenzuwirken, wird eine dedizierte Funktion dem Nutzer zur Verfügung gestellt, mit der er Eponyme als erledigt markieren kann. Hierzu kann er mit der Maus das gewünschte Eponym mit der mittleren Maustaste anklicken. Das Eponym wird zunächst mit grüner Farbe als erledigt markiert (konsistent mit dem vorher verwendeten Ampelsystem). Neben der visuellen Markierung, bewirkt dies auch, dass das Eponym nicht mehr von dem zugewiesenen Jahr entfernt werden kann. Lediglich die Worklane kann noch gewechselt werden. Dies hat den Effekt, dass der Nutzer sich zum einen hierdurch merkt, dass er das Eponym schon einmal bearbeitet hat und mit guten Grund an die aktuelle Position platziert hat. Zum anderen verhindert dies auch, dass beim weiteren Sortierungsprozess dieses Eponym versehentlich in ein anderes Jahr verschoben wird.

Bei der Verwendung dieser Funktion bleibt jedoch weiterhin die rote Umrandung des Eponyms bestehen, sofern sie schon vorher bestand. Dies dient dem Zweck, dass der Nutzer weiterhin einsehen kann, dass noch zugehörige Constraints verletzt sind, auch wenn es sich wahrscheinlich nur um wenige und unsichere handelt. Hierdurch kann der Nutzer zu einem späteren solche Eponyme wiederfinden und versuchen, die verbleibenden Constraints zu erfüllen, nachdem sich z. B. die restlichen Eponyme verschoben haben und so nun eine Chance besteht, dass diese nun erfüllbar sind. Denkbar wäre auch, dass der Nutzer nochmal eine andere mögliche Sortierung in Verbindung mit anderen Eponymen in Erwägung ziehen und testen möchte.

Für diesen Zweck ist es auch jederzeit möglich, die Erledigt-Markierung wieder per nochmaligen Mittelklick auf das Eponym wieder zu entfernen. Hierdurch wird sowohl die grüne Einfärbung entfernt als auch das Verschieben des Eponyms wieder verfügbar. Das langfristige Ziel ist es aber, dass für möglichst viele Eponyme die rote Umrandung entfernt wird, indem deren Constraints vollständig erfüllt werden, und dass zusätzlich alle Positionen von entsprechenden Experten mit Hilfe der Erledigt-Markierung bestätigt werden.

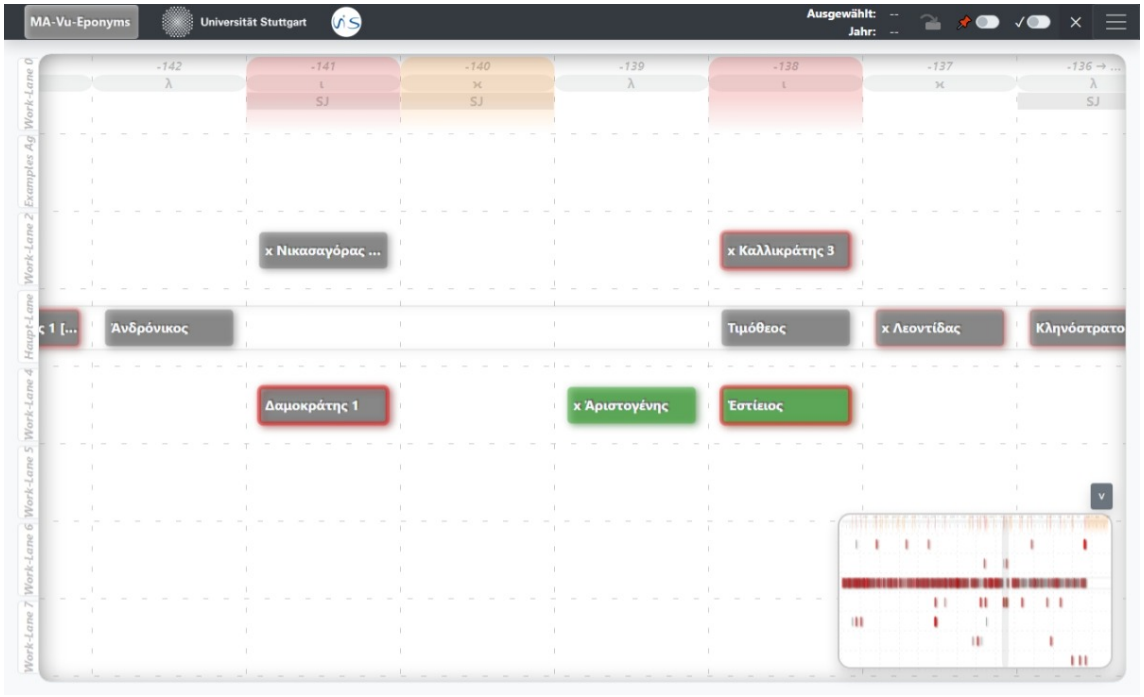


Abbildung 3.20: Hier sind verschiedene Kombinationsmöglichkeiten mit roten Umrandungen und Erledigt-Markierungen zu sehen: Die roten Umrandungen, welche unerfüllte Constraints anzeigen, können sowohl bei Eponymen auftreten, die noch nicht als erledigt markiert sind, als auch bei solchen, die schon eine solche Markierung besitzen. Doch auch bei noch nicht erledigten Eponyme kann die rote Umrandung schon vollständig verschwunden sein, was bedeutet, dass für das Eponym zwar schon alle Constraints erfüllt worden sind, es aber sein kann, dass der Nutzer beabsichtigt, dieses Eponym in naher Zukunft nochmal zu bearbeiten.

3.8 Sonstige Funktionen

In diesem Abschnitt werden einige Funktionen vorgestellt, die nicht nur spezifisch zu einem Teilschritt der Sortierung gehören, sondern allgemein als Unterstützung dienen können. Weiterhin dienen sie auch der Verbesserung der Benutzerfreundlichkeit.

Als erstes wird dem Nutzer eine zusätzliche visuelle Metapher für ein implizites Constraint zur Verfügung gestellt, welche nicht explizit in den Daten vorhanden sind. Hierbei handelt es sich, um die Tatsache, innerhalb eines Jahres nicht zwei verschiedene Eponyme zugewiesen sein können. Wenn dieses implizite Constraint für ein Jahr verletzt ist, so wird im Zeitstrahl eine entsprechende rote Markierung hinzugefügt (siehe Abbildung 3.21). Ein weiteres ähnlich implizites Constraint ist, dass ein Jahr nicht ohne ein Eponym bestehen darf. Solche leeren Jahre werden mit einer orangenen Markierung versehen — wieder in Anlehnung an das zuvor verwendete Ampelsystem. Die orangene Farbe soll im Vergleich zum Rot andeuten, dass die Verletzung dieses Constraints im Verhältnis nicht ganz so gravierend ist, wie das erstere, da es durchaus vorkommen kann, dass mehr Jahre als

Eponyme in den Daten definiert werden und sie deshalb übrig bleiben. Beide Constraints bieten dem Nutzer jedoch Hinweise darauf, dass an entsprechender Stelle noch Änderungsbedarf besteht, sowohl im Auswahlmodus als auch in der Übersicht.

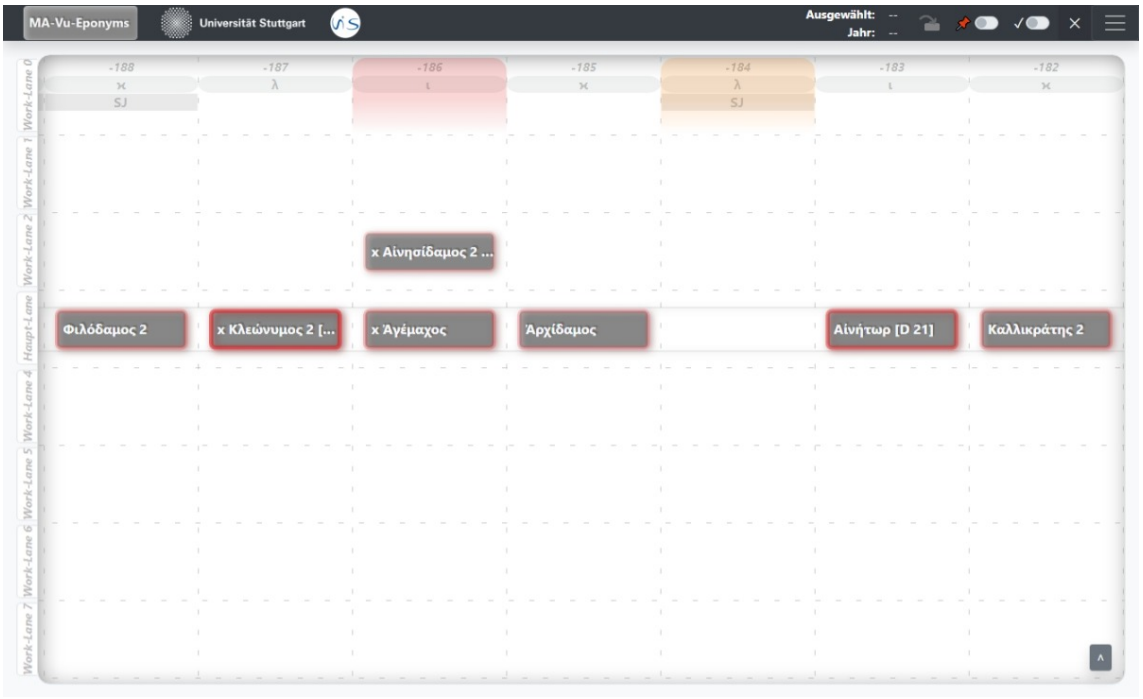


Abbildung 3.21: Im Zeitstrahl werden Jahre rot markiert, wenn mehr als ein Eponym zugewiesen ist. Analog dazu werden Jahre orange markiert, wenn sie noch leer sind.

Zuvor wurden schon die Anpinnen-Funktion und die Erledigt-Markierungen erläutert, welche auf die mittlere bzw. rechte Maustaste gelegt sind. Um sie auch Nutzern nahe zu bringen, die noch nicht viel Erfahrung mit dem Prototypen haben, wurden diese Funktionen auch in die Toolbar oben rechts abgelegt, sodass sie jederzeit sichtbar für den Nutzer sind (siehe Abbildung 3.22). Die Toggle-Buttons für die Anpinnen- und Erledigt-Funktion haben noch den Nebeneffekt, dass sie den aktuellen Status eines ausgewählten Eponyms anzeigen. Dies ist vor allem aus dem Grund nützlich, weil bei einer Selektion dessen schwarze Farbe die grüne bzw. blaue Markierung überlagert. Hier können die jeweiligen Status weiterhin eingesehen werden, auch wenn das Eponym bereits durch die Selektion markiert wurde.

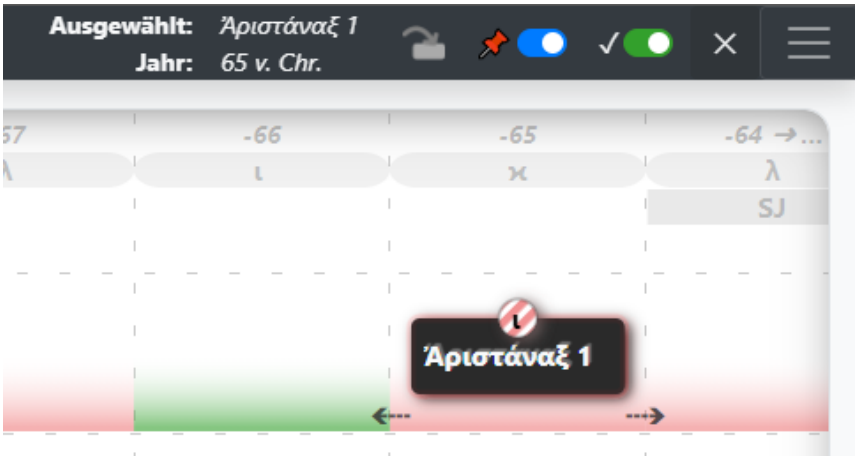


Abbildung 3.22: Die Toolbar-Funktionen von links nach rechts: Informationen zum ausgewählten Eponym und dem aktuellen Jahr, die Eponym-Springen-Funktion, der Anpinnen-Toggle, der Toggle für Erledigt-Markierungen, der Auswahl-Aufheben-Button und der Zugriff auf das Burgermenü.

Das Bürgermenü, das durch den Button der oberen rechten Ecke in der Toolbar erreichbar ist, enthält zum einen verschiedene Einstellungsmöglichkeiten, welche die Visualisierung betreffen als auch eine Speichern- bzw. Laden-Funktion (siehe Abbildung 3.23). Mit Letzteren kann der aktuelle Zustand als Datei heruntergeladen werden und zu einem späteren Zeitpunkt geöffnet werden. Der gespeicherte Zustand beinhaltet hierbei nicht nur die aktuelle Sortierung, sondern auch die jeweiligen Worklanes, die ausgewählten, angepinnten und erledigten Eponyme, sowie den aktuellen Ausschnitt und Zoomstufe in der Hauptansicht. In dieser Hinsicht stellt dies ein exaktes Abbild des aktuellen Zustandes dar. Hiermit können beispielsweise verschiedene Sortierungen miteinander verglichen werden oder auch „lebende Screenshots“ unter Experten oder Nutzern der Anwendung miteinander geteilt werden, welche direkt modifizierbar sind.

Weiterhin enthält das Burgermenü verschiedene Einstellungsmöglichkeiten, die die Visualisierung betreffen. Beispielsweise können die verschiedenen Hilfslinien der Visualisierung ausgeschaltet werden, um so bei Bedarf den „Visual Clutter“ zu reduzieren, wenn dies in bestimmten Situationen gewünscht ist. Es kann auch das gesamte Farbschema der Anwendung in ein farbenblindenfreundliches Farbschema geändert werden, um so Nutzer mit Farbenblindheit nicht zu benachteiligen. Denn wie zuvor erläutert basiert der Großteil der verwendeten Farben im Lösungsansatz auf einem Ampelsystem, womit die meisten Menschen vertraut sind. Allerdings ist ein solches Ampelsystem nicht brauchbar für Farbenblinde, weshalb mit dieser Option die verwendeten Grüntöne durch entsprechende Blautöne ersetzt werden, was farbenblindenfreundlicher ist. Das resultierende Ampelsystem ist dann wie folgt: Rot, Orange, Gelb, Hellblau, Dunkelblau (von schlecht zu gut). Außerdem wird die Farbe für angepinnte Eponyme von blau zu gelb geändert, da dies sonst im Widerspruch zu den erledigten Eponymen steht.

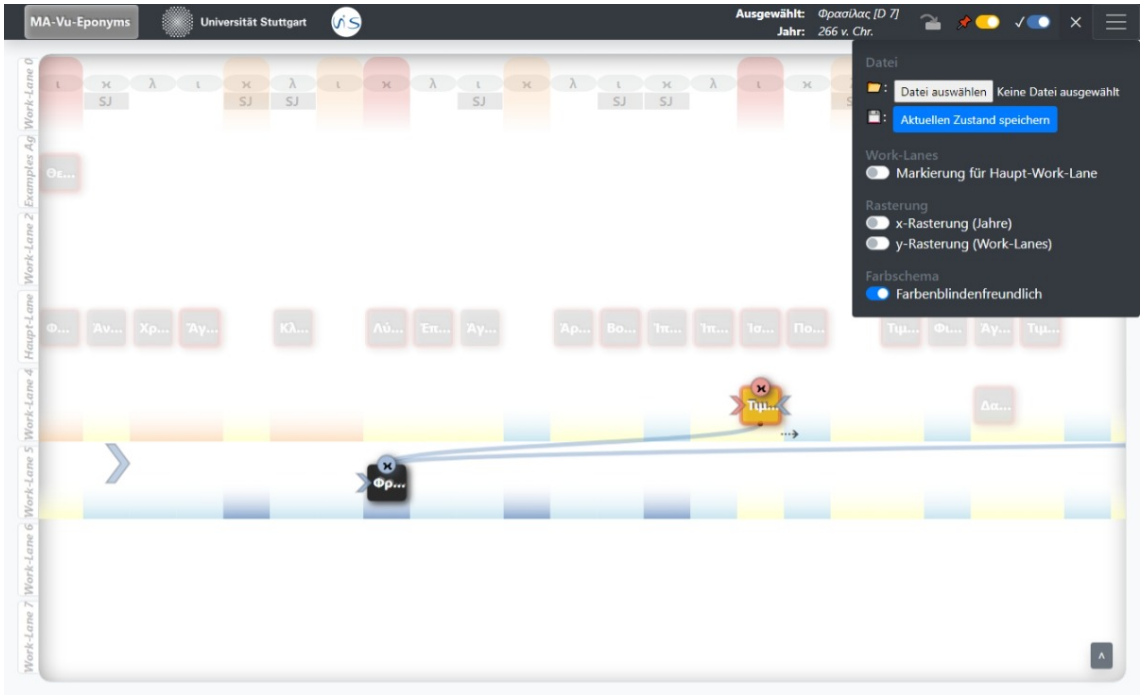


Abbildung 3.23: Hier ist das Burgermenü zu sehen: Es beinhaltet zum eine Speichern- und Laden-Funktion. Zum anderen kann man hier auch diverse Hilfslinien ausschalten und auch das Farbschema ändern. Dessen Änderungen sind hier in der Hauptansicht zu sehen.

Eine weitere Funktion, die den visuellen Clutter bei Bedarf reduzieren soll, ist die Hover-Funktion für Constraints, die eine Verbindungslinie zu anderen Eponymen besitzen, nämlich Separation- und Before-Constraints. Wenn sehr viele davon in der Hauptansicht visualisiert werden, kann es sich als schwierig erweisen, die einzelnen Start- und Endpunkte und die dazugehörigen Eponyme der Constraints zu identifizieren. Hierbei kann die Hover-Funktion Abhilfe schaffen, welche *alles* ausgegraut, ausgenommen die gehoverte Verbindung und dessen Start- und End-Eponyme. Bei Separation-Constraints bleiben zusätzlich noch die Begrenzungsrechtecke sichtbar. Hierdurch sollen die einzelnen Verbindungen besser zugeordnet werden können. Wird das entsprechende Constraint nicht mehr gehovert, wird wieder alles wie zuvor eingeblendet.

Außerdem sind die Beschriftungen für die Worklanes (links von der Hauptansicht) durch den Nutzer editierbar. Hierdurch kann der Nutzer selbst bestimmten und zuweisen, welche Worklane für welchen Zweck eingesetzt werden soll. Denkbar wäre z. B. eine dedizierte Worklane für bereits erledigte Eponyme, eine Worklane für sichere/unsichere Eponyme, eine Worklane für jeden Constraint-Typ usw. Die Beschriftungen werden auch in der Speicherfunktion inkludiert, wodurch diese auch dafür geeignet sind, die Empfänger von solchen Speicherzuständen auf eine bestimmte Gruppe von Eponymen hinzuweisen o.ä.

4 Expert Feedback

Im Rahmen dieser Arbeit fand eine Feedback-Sitzung zusammen mit einem Experten statt, nämlich einem der Historiker, mit denen im Zusammenhang dieser Arbeit kooperiert wurde. Hierbei wurde zunächst der entwickelte Lösungsansatz vorgestellt und die genaue Verwendung anhand des Prototypen erläutert. Anschließend durfte der Experte selbst Hand anlegen: Mit geringfügigen Anweisungen unsererseits testete er, wie gut er mit dem Lösungsansatz zurecht kommt und hat uns dabei seine Eindrücke und sein Feedback hierzu mitgeteilt.

Vorab muss an dieser Stelle erwähnt werden, dass die Sitzung mit unserem Partner leider nicht ganz wie erhofft verlieft, was zum Teil auch durch die Auswirkungen der Corona-Pandemie geschuldet ist, welche zum Zeitpunkt der Arbeit vorherrschte. Durch die Einschränkungen der offiziellen Lockdown-Regeln, war es nicht möglich ein persönliches Treffen vor Ort mit direktem Menschenkontakt zu organisieren. Stattdessen musste auf ein Online-Meeting per Video-Konferenz gewechselt werden, was einige Aspekte der Sitzung leider nicht ganz so verlaufen ließ wie geplant.

In einem Online-Meeting hatten wir leider wenig Kontrolle über die Umgebung, in der die Sitzung stattfand. Der ausschlaggebendste Punkt war hierbei, dass wir kein eigenes Testsystem zur Verfügung stellen konnten, wo wir hätten sicherstellen können, dass alle Bedingungen für eine reibungslose Feedback-Sitzung erfüllt sind. Stattdessen musste unser Experte den Prototypen auf seinem privaten Laptop ausführen, welcher nach seiner Aussage relativ in die Jahre gekommen war. Deshalb war die Feedback-Sitzung nach der Präsentation und Einführung von unserer Seite relativ schnell beendet, da durch die resultierenden Performance-Probleme nur ein Bruchteil der geplanten Tests durchgeführt werden konnte. Nichtsdestotrotz konnte eine gewisse Menge von Eindrücken seitens unseres Partners festgestellt werden, welche im Folgenden näher erläutert werden soll.

Zunächst der offensichtlichste Punkt: Der entwickelte Prototyp ist in Hinsicht auf die Performance-Optimierung nicht für schwächere, ältere Systeme ausgelegt. Die Entwicklung des Prototypen fand auf relativ aktueller Hardware statt und verlief in dieser Hinsicht relativ reibungslos, und sonst fand auch eher selten ein Testen auf anderen System statt. Hierdurch ist der Aspekt der Performance in den Hintergrund gerückt. Eines der Ziele der Gestaltung des Prototypen als Web-Anwendung beinhaltete, eine möglichst einfache und plattformunabhängige Ausführung zu ermöglichen, ohne ausgefallene technische Rahmenbedingungen oder technische Vorkenntnisse voraussetzen zu müssen. Das ist in diesem Fall bisher wohl nicht zu 100% erreicht worden.

Ein weiterer Punkt stellt die Erlernbarkeit des Lösungsansatzes dar: Bereits die erste Aufgabe, welche wir dem Experten gestellt haben, hatte ihm schon sichtlich Schwierigkeiten bereitet. Diese bestand darin ein Eponym auszusuchen, das zunächst nur wenig verletzte Constraints hatte und diese sollten erfüllt werden. Sicherlich haben sich auch die Performance-Probleme negativ darauf ausgewirkt. Trotzdem musste er als erstes fragen, wie man die Hauptansicht zoomt und verschiebt. Dies ist ein Hinweis auf eine eher hohe Hürde bei der Erlernbarkeit unseres Lösungsansatzes. Im Vergleich zu der relativ großen Menge an anderen möglichen Interaktionen, stellen das Pannen und Zoomen der Hauptansicht noch relativ simple Aufgaben dar. Trotzdem fiel es unserem Experten

schwer, sich selbst diese relativ grundlegenden Funktionen zu merken. Wie bereits erwähnt, mussten aus Performance-Gründen die Tests bereits an dieser Stelle unterbrochen werden. Jedoch lässt sich hier schon vermuten, dass die Schwierigkeiten bei den restlichen Tests im Bezug auf die anderen Funktionen auch relativ hoch gewesen wären.

Womit lässt sich dies erklären? Meine naheliegendste Vermutung ist schlicht, dass die Fülle an Informationen und Funktionen einfach zu hoch ist, um sie nach nur einmaligem Hören umfassend und tiefgreifend genug zu verstehen. Es gibt zugegebenermaßen alleine schon eine hohe Zahl an visuellen Metaphern, Elementen und zugehörigen Interaktionen zu den verschiedensten Aspekten, welche sich bereits schwer nach nur einmaligen Hören merken lassen. Weiterhin kommt noch dazu, dass diese in Kontext zueinander gesetzt werden müssen — man muss zum einen wissen, in welchem Punkt des Sortierungsprozesses man sich gerade befindet, und zum anderen auch, welche Teilfunktionen aktuell als Lösung des Teilproblems zur Verfügung stehen. Ich vermute, dass dies neue, unerfahrene Nutzer zu Anfang schlicht überfordert. Um den entwickelten Lösungsansatz effektiv nutzen zu können, muss man sich eine gewisse Zeit nehmen, um nach und nach die unterschiedlichen Teilfunktionen und visuellen Elemente kennenzulernen. Außerdem muss man zumindest einmal wie in einer Art Tutorial Schritt für Schritt durch die einzelnen Teilschritte in der Sortierung geführt werden, um so das Ganze in Kontext setzen zu können.

Nichtsdestotrotz gab es auch positives nach der Feedback-Sitzung zu vermelden: Die restlichen entwickelten Funktionen und Visualisierungen, die unser Experte nur in unserer Einführung erleben durfte (und leider nicht im Praxistest), hat er als gelungen empfunden. Die Entwicklung des Lösungsansatzes fand unter anderem auch in Absprache mit ihm statt, wovon unter anderem auch die Definition der Struktur der verwendeten Dummydaten stammt. Nach seiner Aussage hätte er sich die Darstellungen des Lösungsansatzes so vorstellen können, was darauf deutet, dass die verwendeten visuellen Metaphern relativ einfach und intuitiv zu verstehen sind.

Insgesamt würde ich aus der Feedback-Sitzung schließen, dass der entwickelte Lösungsansatz von den Funktionen her relativ komplex ist und dessen Erlernbarkeit mit gewissem Aufwand verbunden ist. Grund hierfür ist schlicht die Komplexität der Daten und die daraus folgenden benötigten Funktionen und Interaktionen. Wenn man aber einmal eingearbeitet und mit allen Aspekten vertraut ist, dann kann sich der Ansatz durchaus als effektiv erweisen, um die Sortierungen der Eponyme mit all seinen Aspekten zu Randbedingungen und Unsicherheiten darzustellen.

5 Zusammenfassung und Ausblick

In dieser Arbeit wurde ein Lösungsansatz entwickelt, der sich mit der Visualisierung von historischen Eponymdaten beschäftigt. Besonderes Augenmerk lag hierbei auf zugehörigen Randbedingungen, den Constraints, und Unsicherheiten, den Confidence-Levels.

Bevor wir den eigentlichen Lösungsansatz vorgestellt haben, haben wir zunächst einige verwandte Arbeiten betrachtet. Wir sind verschiedene Aspekte von ihnen durchgegangen und haben beleuchtet, warum diese Arbeiten für unseren Fall relevant sind und welche Dinge uns als Vorbild dienen konnten.

Daraufhin sind wir im Detail auf das entwickelte Konzept der Lösung eingegangen und stellten Schritt für Schritt verschiedene Teile des Prototypen vor, welche auf verschiedene Teilprobleme gerichtet sein sollten. Unter anderem sind wir auf die zu Grunde liegenden Daten eingegangen, wir haben die unterschiedlichsten visuellen Formen und Metaphern vorgestellt und haben eine Bandbreite an verschiedenen Interaktionsmöglichkeiten mit der Visualisierung kennengelernt.-

Schließlich wurde auch eine kleine Evaluation des Lösungsansatzes in Form einer Expert-Feedback-Sitzung durchgeführt. Hierbei ist zu Tage getreten, dass zwar die verwendeten visuellen Metaphern und entwickelten Funktionen an sich gut gelungen sind, jedoch die Erlernbarkeit durch eine relativ hohe Komplexität des Lösungsansatzes einen gewissen Aufwand erfordert.

Ausblick

Diese Arbeit kann im Anbetracht verschiedenster Punkte auf vielfältige Weise fortgeführt werden. Um zunächst die angesprochenen Probleme aus dem vorigen Kapitel anzusprechen: Die erste und wohl einfachste Möglichkeit den Lösungsansatz zu verbessern, stellt eine Performance-Optimierung des Prototypen dar. Weiterhin könnte auch eine Entwicklung eines eigenständigen Tutorials dazu beitragen, die relativ schwierige Erlernbarkeit zu verbessern. Als weitere relativ grundlegende Funktion könnte eine Zustandskontrolle entwickelt werden, um so dem Nutzer zu ermöglichen Teilschritte vor- und rückgängig zu machen.

Um dem User noch mehr Hilfestellungen und Empfehlungen beim Sortierungsprozess der Eponyme geben zu können, könnte die geschickte Verwendung eines Force-Directed-Layout behilflich sein. Im Zusammenhang dazu könnte man noch einen Schritt weiter gehen, und einen Algorithmus implementieren, der eine oder sogar mehrere mögliche Vorsortierungen berechnet, sodass nur noch „Feinheiten“ angepasst werden müssen. Wiederum einen weiteren Schritt könnte man gehen und die Visualisierung so erweitern, dass der direkte Vergleich zwischen verschiedenen Sortierungen gezogen werden könnte, und so bewerten könnte, welche die bessere und historisch korrektere ist.

Denkbar wäre auch eine Funktion womit man innerhalb der Visualisierung auf Wunsch bestimmte Daten editieren könnte, wenn man z. B während des Sortierungsprozesses feststellt, dass bestimmte Daten plötzlich fragwürdig erscheinen und im Konflikt stehen, oder wenn auch schlichtweg Fehler in den Daten bestehen. Wenn man die Plattformunabhängigkeit und die Benutzerfreundlichkeit weiter treiben möchte, dann könnte man auch in Erwägung ziehen, die Anwendung für Stylus- oder Touch-Eingaben zu optimieren, sodass die Benutzung auf Tablets oder auch auf Touchbildschirmen im Großformat funktionieren würde. So könnte man überprüfen, ob die Bedienung per Touch eventuell intuitiver und effektiver als eine Mausbedienung wäre.

Literaturverzeichnis

[Ahl96] C. Ahlberg. „Spotfire: an information exploration environment“. In: *ACM SIGMOD Record* 25.4 (1996), S. 25–29 (zitiert auf S. 21).

[Bad15] N. Badoud. *Le temps de Rhodes: une chronologie des inscriptions de la cité fondée sur l'étude de ses institutions*. CH Beck, 2015 (zitiert auf S. 28).

[BMMS91] A. Buja, J. A. McDonald, J. Michalak, W. Stuetzle. „Interactive data visualization using focusing and linking.“ In: *IEEE Visualization*. Bd. 91. 1991, S. 156–163 (zitiert auf S. 18).

[Car99] M. Card. *Readings in information visualization: using vision to think*. Morgan Kaufmann, 1999 (zitiert auf S. 11).

[CKB09] A. Cockburn, A. Karlson, B. B. Bederson. „A review of overview+detail, zooming, and focus+context interfaces“. In: *ACM Computing Surveys (CSUR)* 41.1 (2009), S. 1–31 (zitiert auf S. 15–18).

[DE98] A. Dix, G. Ellis. „Starting simple: adding value to static visualisation through simple interaction“. In: *Proceedings of the working conference on Advanced visual interfaces*. 1998, S. 124–134 (zitiert auf S. 18).

[Goo20] Google. „Google Maps“. In: (2020). Online: https://www.google.de/maps (zitiert auf S. 17).

[KHG03] R. Kosara, H. Hauser, D. L. Gresh. „An interaction view on information visualization“. In: *State-of-the-Art Report. Proceedings of EUROGRAPHICS* (2003), S. 123–137 (zitiert auf S. 18).

[NXWW14] P. H. Nguyen, K. Xu, R. Walker, B. L. W. Wong. „SchemaLine: Timeline Visualization for Sensemaking“. In: *2014 18th International Conference on Information Visualisation*. 2014, S. 225–233. DOI: 10.1109/IV.2014.14 (zitiert auf S. 14, 15).

[ST98] R. Spence, L. Tweedie. „The Attribute Explorer: information synthesis via exploration“. In: *Interacting with Computers* 11.2 (1998), S. 137–146 (zitiert auf S. 21).

[Twe97] L. Tweedie. „Characterizing interactive externalizations“. In: *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*. 1997, S. 375–382 (zitiert auf S. 18).

[YKSJ07] J. S. Yi, Y. a. Kang, J. Stasko, J. A. Jacko. „Toward a Deeper Understanding of the Role of Interaction in Information Visualization“. In: *IEEE Transactions on Visualization and Computer Graphics* 13.6 (2007), S. 1224–1231. DOI: 10.1109/TVCG.2007.70515 (zitiert auf S. 18–22).

[ZK08] C. Ziemkiewicz, R. Kosara. „The shaping of information by visual metaphors“. In: *IEEE transactions on visualization and computer graphics* 14.6 (2008), S. 1269–1276 (zitiert auf S. 12, 13).

Alle URLs wurden zuletzt am 13. 01. 2021 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