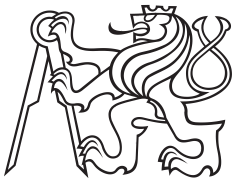
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra Počítačové grafiky a interakce

Vizualizace historických událostí světa Asterion

Bc. Jan Tislický

Vedoucí: Ing. Radek Richtr, Ph.D.
Obor: MP44 - Otevřená informatika
Studijní program: MPOI318 - Počítačová grafika
Leden 2023

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Tislický** Jméno: **Jan** Osobní číslo: **457874**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Specializace: **Počítačová grafika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Vizualizace historických událostí světa Asterion

Název diplomové práce anglicky:

Visualization of historical events in the world of Asterion

Pokyny pro vypracování:

Asterion je rozsáhlý svět s bohatou a komplikovanou historií. Cílem práce je vizualizovat historické události (reprezentované na vstupu pomocí otagovaných odstavců beletristického textu) tohoto fiktivního světa pomocí vícera (alespoň pěti) časových os vázaných na konkrétní tag jako je město, postava, událost, válka, apod. Práce navazuje (nahrazením frontendu) na existující prototyp.
1) Prostudujte stávající prototyp práce a bakalářské práce Tomáše Růžičky a Michaely Zimmermannové. Analyzujte je a navrhněte vhodnější fronend aplikace. Vyberte jeden výsledný způsob vizualizace.
2) Proveďte rešerši způsobů vizualizace dat o historických událostech. Zaměřte se převážně na přehlednou vizualizaci pořadí, v jakém se historické události odehrály v čase a to, jak spolu souvisí.
3) Dále proveďte rešerši a analýzu technologického řešení pro vizualizaci událostí v čase ve webovém prohlížeči.
4) Navrhněte a implementujte frontend webové aplikace umožňující vizualizaci událostí odehrávajících se v čase a jejich souvislostí.
5) Funkčnost výsledné aplikace ověřte na datech o událostech světa Asterion a demonstrujte na databázi minimálně událostí základního modulu (400-500 událostí a obdobný počet souvislostí). Na výsledné aplikaci proveďte kvalitativní uživatelské testování s alespoň šesti uživateli.

Seznam doporučené literatury:

1) Tamara Munzner. Visualization Design and Analysis, CRC Press, 2015.
2) Aigner, Wolfgang, et al. Visualizing time-oriented data—a systematic view. Computers & Graphics 31(3), pages 401-409, 2007.
3) Huntington, Matthew. D3.js Quick Start Guide: Create Amazing, Interactive Visualizations in the Browser with JavaScript, Packt Publishing, 2018.
4) Iglesias, Marcos. Pro D3.js: Use D3.js to Create Maintainable, Modular, and Testable Charts, Apress L. P., 2019.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Radek Richtr, Ph.D. katedra softwarového inženýrství FIT

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **23.07.2022** Termín odevzdání diplomové práce: **10.01.2023**

Platnost zadání diplomové práce: **19.02.2024**

Ing. Radek Richtr, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____
Datum převzetí zadání

_____
Podpis studenta

Poděkování

Chtěl bych zde poděkovat svému vedoucímu Ing. Radku Richtrovi, Ph.D., za veškerou podporu během psaní této diplomové práce. Jeho vedení a konstruktivní komentáře k celé práci byly velkým přínosem a posouvaly práci vpřed velkými skoky. Dále bych také rád poděkoval Ing. Davidovi Bernhauerovi, Ph.D., za konzultace a technickou pomoc s novou technologií. Jeho rady a zkušenější pohled na problematiku webového vývoje byly velmi přínosné. Za jazykovou kontrolu práce patří můj dík Bc. Petře Svíčkové. Závěrem bych chtěl poděkovat všem mým spolužákům a rodině za podporu během celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací. Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. §46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užit éı této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen "Dílo"), a to všem osobám, které si přej éı Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli éúčelem (včetně užití k výdělečným éúčelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byt' jen z části) na základě Díla, éúpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový soubor takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový soubor Díla.

.....

Jan Tislický
V Praze, 9. ledna 2023

Abstrakt

Tato diplomová práce pokračuje v již běžící vizualizaci časových os ze světa Asterion. Rozebírá základy vizualizací a snaží se doplnit diskuze chybějící v předchozí práci. Na základě těchto teoretických znalostí jsou diskutovány a navrženy změny dosavadní vizualizace, hlavně frontendu webové aplikace. Kromě toho jsou také diskutovány nástroje na zpřehlednění celé vizualizace, jako například pořadí časových os nebo volba vhodnějších vizuálních prvků. Aplikace pak nabízí možnost různých pohledů na uživatelem vybrané časové osy, ty umožňují sestavit příběh ve hře na hrdiny právě ze světa Asterion.

Klíčová slova: vizualizace, časově orientovaná data, relační data, časové osy, webová aplikace, pořadí časových os, Asterion

Vedoucí: Ing. Radek Richtr, Ph.D.

Abstract

This thesis continues with a timeline visualization of temporal data of the Asterion RPG world, which is already accessible. It goes through the basics of visualizations and tries to apply this knowledge to the previous bachelor theses. Based on this theoretical information, a new form of visualization is created and discussed in detail, mainly changes in the front end of the current web application. Besides this, a new way of creating a better overview of the visualization is discussed, for example, a tool for making a better order of the timelines or more suitable visual elements. The application offers a way to create different views on user-defined timelines. These are essential for a story-driven RPG experience in the world of Asterion.

Keywords: visualization, time-oriented data, relation data, timelines, web application, order of timelines, Asterion

Title translation: Visualization of historical events in the world of Asterion

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Současná doba v mnoha případech nahrává k rozšiřování stolních her o technické prvky jako jsou aplikace, které např. řídí pohyb protivníka, udržují přehled o hře nebo vyprávějí příběh a větví ho do různých konců. Výjimkou nejsou ani větší hry na hrdiny[1], mezi které patří např. velmi populární americká hra Dungeons & Dragons, nebo česká Dračí Hlídka. V českých končinách je rozšířena ještě jedna varianta této hry a tou je Asterion. V rámci těchto her se hráči chopí role hrdiny, kterého si vytvoří, a následně prožijí dobrodružství se svými spoluhráči pod vedením tzv. pána jeskyně, který si pro toto dobrodružství připraví podklady. Těmito podklady je myšlen průchod určitou oblastí herního světa, souboje a diplomatická setkání, posezení v hospodách a další drobné detaily takového příběhu. Celý příběh je vždy zasazen do určitého časového období a na určité území. Tím pádem musí mít pán jeskyně vcelku dobrý přehled o historii herního světa, o všech významných událostech, významných postavách, diplomatických vztazích různých organizací a vládců. Ne všechny informace jsou vždy přístupné v jedné knížce. Typicky se tyto světy stále vyvíjí a přidávají se další části historie, případně se pokračuje aktuálním rokem do budoucna. To znamená, že může být velmi těžké udržet si přehled o všech částech herního světa. Zde by tedy bylo vhodné mít k dispozici nástroj, který by umožňoval vidět tyto události, vztahy mezi jednotlivými entitami, cesty postav a tak dále.

Tímto nástrojem se právě pro svět Asterionu stala webová aplikace Asterion-Timelines, která je výsledkem dvou bakalářských prací [1, 2]. Tato aplikace umožňuje zobrazit časové osy dle výběru a sledovat na nich různé události, jejich postup a souvislosti mezi nimi. Nicméně vždy záleží na tom, v jakém

[1] Označovány jako role-playing game, která se zkracuje na RPG.

pořadí uživatel vloží osy do vizualizace. To nemusí být vždy jednoduchý úkon, pokud je cílem dosáhnout co nejpřehlednější časové osy. Proto přichází na scénu rozšíření stávající aplikace, a to o možnost automatického řazení dvěma metodami. Ty se snaží minimalizovat počty křížení na základě různých principů. Kromě toho také došlo k zásadní změně celého uživatelského rozhraní.

Kapitoly 2, 3, 4 a 5 rozebírají různé aspekty vizualizací. Na základě teoretických znalostí z těchto kapitol se snaží identifikovat problémy, které se ve výstupní verzi bakalářské práce [2] vyskytují. Dále také doplňují diskuze tam, kde chybí v [2], která se právě vizualizací zabývala. Bakalářská práce [2] se obecně zabývala frontendem[2] a zejména na tuto část navazuje tato diplomová práce. V kapitole 6 jsou rozebrány různé vizualizační techniky pro časově orientovaná data. Kapitola 7 pojednává o samotných datech Asterionu, jakého jsou typu, jaké jsou použité prvky a které by bylo vhodnější využít, interakční schémata a další. Na jejím konci je také provedena diskuze nad tím, která metoda kapitoly 6 bude nejvhodnější pro data Asterion. V kapitole 8 jsou probrány technologie, které souvisí s webovým návrhem a je vhodné je použít, případně byly použity a budou využity znovu, protože splňují požadavky. V této kapitole je pak také provedena diskuze nad knihovnou, s pomocí které bude celá vizualizace vytvořena. Kapitoly 9 a 10 obsahují návrh nového rozhraní i jeho výslednou realizaci. Kapitola 11 pak obsahuje shrnutí z testování s novým návrhem na základě výsledků testování.

[2]Běžný pojem v prostředí webových aplikací, jedná se o rozhraní, které je prezentováno uživateli.

Kapitola 2

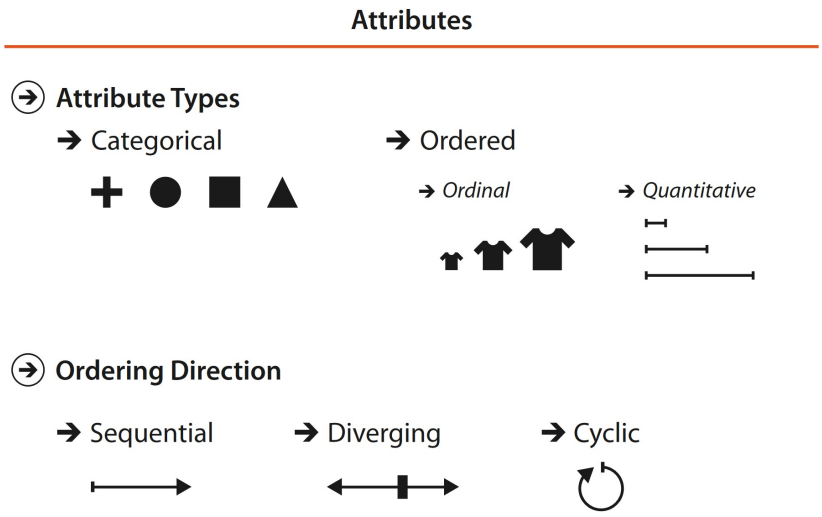
Analýza dat, datové typy

Tato kapitola obsahuje informace z literatury o různých datových typech a jejich přibližný popis. Mimo jiné jsou u datových typů uvedeny příklady, to platí zejména tam, kde je hůře představitelné, co obsahují.

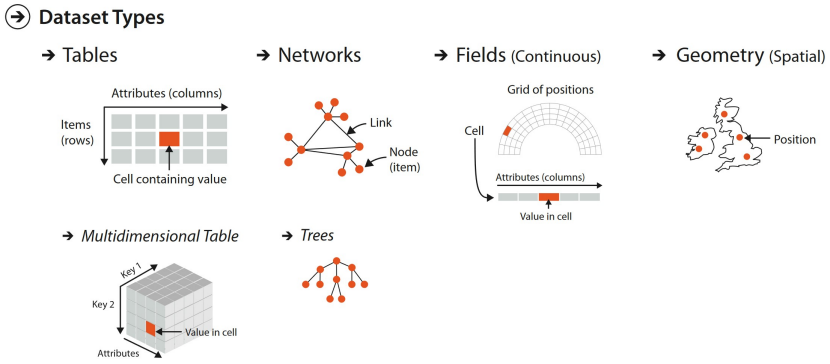
První kroky vizualizace dat by měly obsahovat zhodnocení dat a zvolení vhodných reprezentací vzhledem k datům i požadovanému výsledku. Tato sekce obsahuje informace o datových typech, které jsou ve vizualizacích nejčastější. Jako příklad je možné uvést plat, cenu, počet prodejů nebo teplotu. Celá tato část vychází zejména z [3] kapitola dva, [4] a jako doplňkový zdroj pak [5].

Dle [3] jsou rozlišovány různé entity pro vizualizace, které jsou označeny jako datové typy. Tento typ říká na jaký element jsou data navázána. Mezi tyto datové typy patří jednotlivé položky, atributy, hrany, pozice nebo mřížky. Jako **atribut** je v [3] označena jakýkoliv naměřená, sledovaná nebo zaznamenaná vlastnost dat. Jako příklad poslouží plat, cena, množství prodejů nebo teplota. Atributy samotné se dělí na kategorické nebo řaditelné. **Kategorické atributy** jsou například lidé, společnosti, typy nemocí a podobné. Dále platí, že nemají žádné základní řazení. **Řaditelné atributy** jsou dále rozděleny na dvě další podkategorie, a to ordinální a kvantitativní. **Ordinální atributy** jsou řaditelné, jako příklad postačí velikosti oblečení nebo pořadí dnů v týdnu. **Kvantitativní atributy** jsou také řaditelné a navíc podporují různé aritmetické operace. Například diskrétní číselná posloupnost nebo množinu reálných čísel. Zbývající datové typy jsou snadněji pochopitelné. **Jednotlivé položky**[1] reprezentují například řádky v tabulkách, uzly v sítích nebo lidi. **Hrany**

[1] V [3] značené jako „item“.



Obrázek 2.1: přehled různých druhů atributů [3]

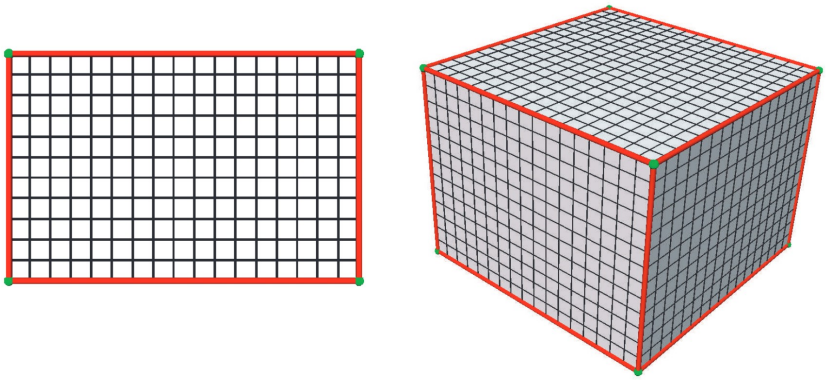


Obrázek 2.2: Ukázka základních typů dat pro vizualizace [3]

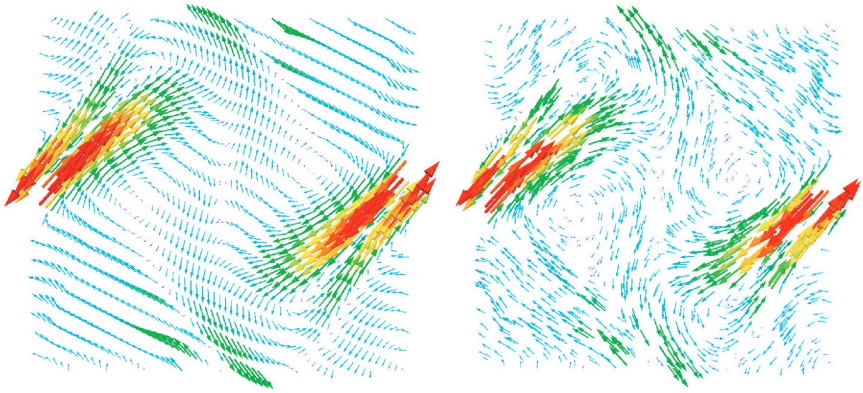
pak udávají vztah mezi položkami, typicky vztahy v síti. **Mřížka** udává specifickou strategii vzorkování spojitých dat v geometrickém i topologickém pohledu na vztahy mezi buňkami. **Pozici** je například zeměpisná šířku a délku nebo jiná pozici ve dvou, nebo třech rozměrech. Následující sekce obsahují rozbor další části vizualizací a tou jsou typy množin dat. Pro představu je k dispozici množina dat a ty mohou být různého typu, například prostorová, nebo tabulková.

2.1 Prostorová data

Data ve vizualizacích jsou v nejjednodušším pohledu rozdělena na dvě části, prostorová a abstraktní. Prostorová data jsou taková data, která jsou vztažena



Obrázek 2.3: Ukázka uniformní mřížky ve které mohou být naměřena data [4]

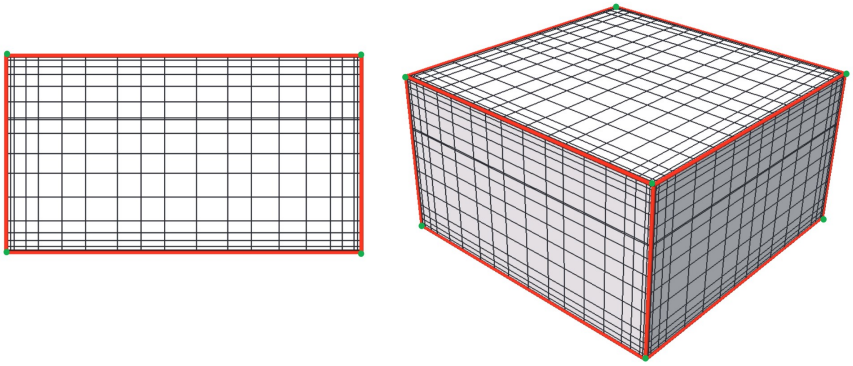


Obrázek 2.4: Ukázka vektorových dat [4]

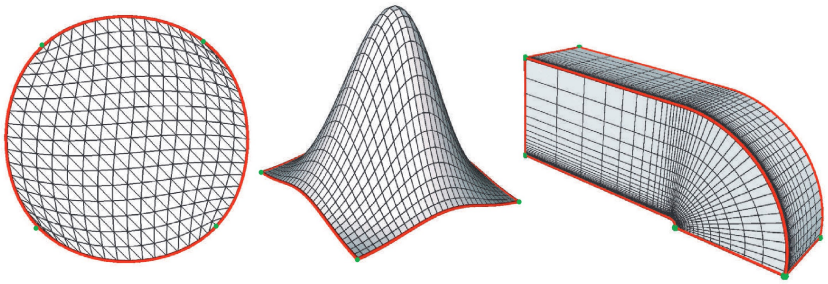
k nějaké geografické poloze na mapě nebo obecně k nějakému místu v prostoru (2D nebo 3D). Konceptuálně mohou být měřená místa až nekonečně malá, ale vzhledem k vysokým paměťovým nárokům se využívá menší hustoty jednotlivých buněk. Tyto buňky pak nesou informace ze spojité domény. V případě potřeby jemnějších dat se provede nové jemnější měření na příslušné doméně[2]. Spojitá data musí být zpracovávána velmi opatrně, jinak řečeno je nutné brát v potaz vzorkování dané domény. Jak se bude interpolovat, nebo jak například zobrazit hodnoty mezi vzorkovanými body tak, aby vzniklo co nejmenší zkreslení. Samotná data, která jsou změřena v těchto buňkách, bývají různého typu, ale všechny buňky tento typ sdílí. Může se jednat o tlak, hustotu, rychlost a mnoho dalších veličin.

Prostorová pole. Prostorová data se velmi často dají nalézt ve formě prostorových polí. Jednotlivé buňky jsou konstruovány z naměřených vzorků a jejich třídění s následnou interpolací. Vzorky jsou měřeny v uniformních intervalech

[2]To nemusí být možné u všech typů datových množin. Například se nelze vrátit v čase a měřit teplotu s větším množstvím měření.



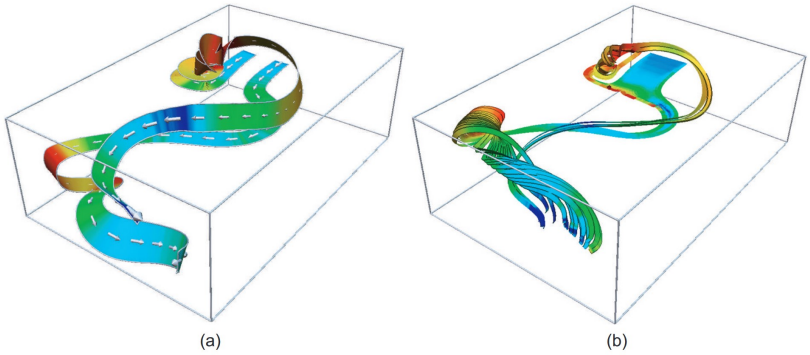
Obrázek 2.5: Neuniformní mřížka je dalším typem možného měření [4]



Obrázek 2.6: Ukázka dalšího typu mřížky [4]

a tím vytváří strukturu daného měření. Proto v případech prostorových dat vzniká mřížka. Tato mřížka je velmi podobná geometrii, ale nemá žádné informace o povrchu. Tyto mřížky jsou různého druhu a tím pádem je se vypouštějí některé informace o topologii. Jako příklad neuložené informace poslouží poloha v prostoru pro uniformní mřížky, jelikož data jsou zaznamenána na určitém místě. Kromě uniformních mřížek existují i další struktury, například strukturované mřížky, nestrukturované mřížky nebo neuniformní mřížky. Ve většině případů se data pojí k určité struktuře, které je nutné porozumět a dále s ní pracovat. Typická zobrazovaná data jsou výstupy rentgenu nebo měření rychlosti proudění vzduchu, vlhkosti a podobných veličin. Zde je tedy vidět, že měřená data nemusí být nutně jedna hodnota, ale i vektor. Tento způsob reprezentace dat umožňuje ukládat naměřené hodnoty jako atributy vizualizace.

Geometrie. Další možností, jak vizualizovat data v prostoru, je geometrie. Jedná se o vizualizaci, která specifikuje tvar zkoumaného předmětu velmi jednoduše. Celý předmět je rozdělen na jednotlivá grafická primitiva, jako jsou vrcholy, hrany, plošky a další. Tuto vizualizaci lze typicky nalézt v případech, kdy je nutné porozumět povrchu zkoumaného objektu. Často je na tuto reprezentaci napojena hierarchická informace s různým rozměrem. Hlavním rozdílem oproti prostorovým polím je absence měřených dat, tento typ dat



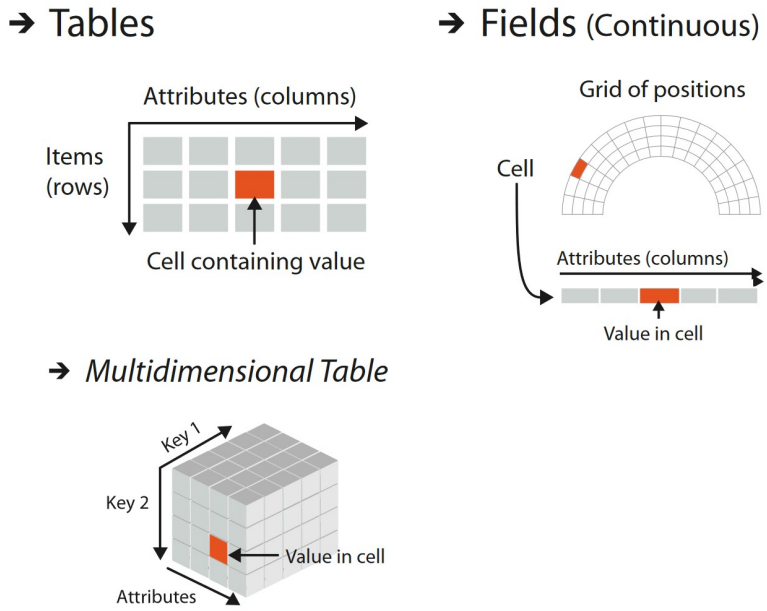
Obrázek 2.7: Vykreslení dat pomocí stream ribbon, obrázek (a) obsahuje dva a obrázek (b) dvacet jednotlivých „stužek“ [4]

nese pouze informaci o topologii daného objektu. Tato reprezentace se typicky využívá v počítačové grafice jako výsledek modelování nebo skenování. Ve vizualizaci je geometrie výsledkem určitých operací nebo algoritmů, např.: konturování nebo proudové pásky[3]. Vizualizace tohoto typu bývá využita samostatně, ale i jako podklad pro další vrstvy, které ukáží doplňkovou informaci.

2.2 Tabulková a relační data

Jedná se o data abstraktního typu. U takového typu dat není přítomna žádná informace o poloze, kde daná data byla získána. V případě známé polohy u takovýchto dat se jedná o mix prostorových a abstraktních dat. Pro abstraktní datové typy také platí, že tato data se neinterpolují. Důvod je vidět na příkladu, ve kterém je každá položka dat jeden člověk, tím pádem provádět interpolaci mezi dvěma lidmi pozbývá smyslu. Jako další příklad abstraktních datových typů poslouží porovnávání dokumentů nebo knih. Tabulková data mají v řádcích uvedené jednotlivé záznamy. Tyto záznamy mají vyplněné jednotlivé atributy[4]. Průnik atributu a položky je pak nazýván buňka. V této buňce se nachází hodnota daného atributu. Samotné tabulky mohou být i multidimenzionální[5] a mít mnohem větší a složitější strukturu, která se může týkat i jiné tabulky[6]. Hodnoty zaznamenané v tabulkách pak mohou být buď nominální, nebo ordinální.

[3] V angličtině označené jako stream ribbon.
[4] Atribut je jeden sloupec hodnot.
[5] Dimenze udává počet kategorií, které daná tabulka má.
[6] Tyto tabulky jsou pak spojeny klíčem, což je odkaz do jiné tabulky, tedy relační data. Podobné jako cizí klíče v SQL.



Obrázek 2.8: Popis tabulkových dat [3]

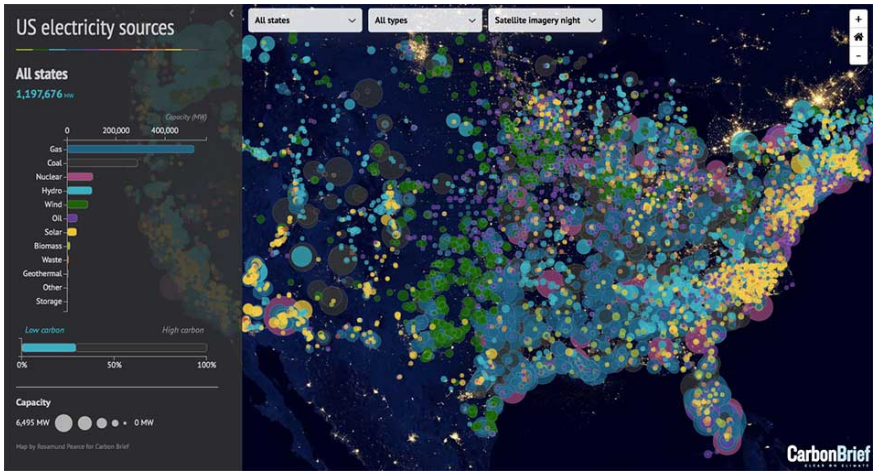
Relační data jsou velmi podobná tabulkovým s tím rozdílem, že se jedná o několik tabulek spojených vztahy. Opět se jedná o abstraktní data, ačkoliv samotné uzly[7] mohou jako své hodnoty obsahovat prostorové souřadnice, obohacení dat se zde také nevyužívá. Data zde jsou navázána na jednotlivé uzly, nebo na spojnice mezi nimi, typicky je počet atributů menší. Jednotlivé vazby jsou různého typu a velikosti[8].

2.3 Stromy a sítě

Tato část dat je dalším zástupcem abstraktního typu. Typicky jsou tato data reprezentována jako uzly a hrany. Jako jednoduchý příklad je možné uvést lidi a vztahy mezi nimi. Jednotliví lidé jsou reprezentováni jako uzly a známost mezi lidmi pak je pak reprezentována jako hrana. Uzly k sobě mají typicky vztažené různé další hodnoty, které jsou s nimi svázány. Vzhledem k tomu, že se jedná o abstraktní data, tak je možné využít jako vizuální kanál pozici jednotlivých uzlů.

[7] Jako uzly jsou zde označovány jednotlivé tabulky.

[8] Jako velikost je myšlena násobnost vazby. Například 1 záznam může mít v přidružené tabulce až n záznamů. V literatuře se toto běžně označuje jako 1:N, M:1 nebo M:N.



Obrázek 2.9: Geografická data, která ukazují energetické zdroje v USA [6]

V momentě, kdy začne mít síť nějakou hierarchickou strukturu, tak je takový typ nazýván stromem. Tento typ grafu je při srovnání s obecným grafem acyklický a každé dítě má jen jednoho rodiče. Jako příklad lze uvést hierarchii v obchodní společnosti nebo evoluční vztah mezi jednotlivými druhy.

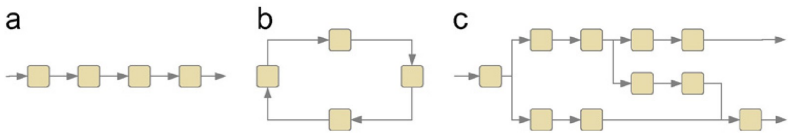
2.4 Směs abstraktních a prostorových dat

Tato část dat je, jak již název sekce napovídá, směs dvou typů dat. Zástupci tohoto typu dat jsou například zločinnost vztaženou k nějakému území, výsledky voleb nebo data o počasí. V těchto datech je přítomna určitá forma geometrie, jako například zajímavá místa, úsečky, křivky, oblasti. Každá z těchto částí mívá různý význam. Úsečky a křivky například reprezentují ulice či řeky a plochy území, parky nebo pozemky. Tato primitiva jsou spojována do větších celků, a tím vzniká geometrie těchto dat.

K vytvořené geometrii jsou pak přiřazena data jako je například text s názvem ulice či oblasti, populace, volební preference, zločinnost nebo data o počasí, tlaku apod. Je zde veliký rozdíl mezi tím, jakého typu jsou získaná data na dané geografické pozici. Přítomnost kvantitativních dat umožňuje využít techniky jako s rozptýlenými prostorovými daty. To znamená, že je použitelná rekonstrukce s pomocí obohacení dat[9]. V případě přítomnosti nominálních[10] dat není možné využít metody pro prostorová data. Například

[9] Obohacení dat je proces, kdy se například pro teplotu dopočítají mezilehlé hodnoty mezi dvěma vzorky a tím se získají zrekonstruované hodnoty.

[10] Kategorických



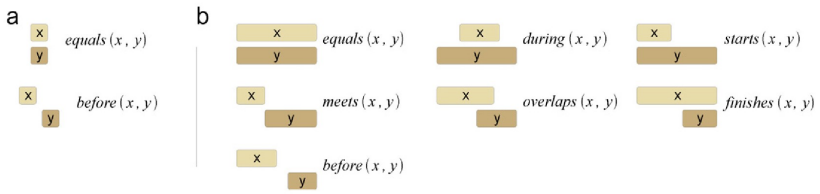
Obrázek 2.10: Struktura časových událostí, (a) je lineární časová posloupnost, (b) je cyklická časová posloupnost a (c) je pak větvící se časová posloupnost [7]

se nevyužije obohacení dat. Pro porovnání kvantitativních a nominálních dat je vhodné uvést několik případů. Nominální data jsou například typy zajímavých míst nebo seznam nezaměstnaných lidí s jejich adresami. Kvantitativní data pak údaje o počasí, tedy teplotě, tlaku, rychlosti proudění vzduchu.

2.5 Časově proměnná data

Tento datový typ nemá formální definici. Obecně je popisován jako množina dat, kde každá položka této množiny má navíc časovou značku. Samotná data pak jsou prostorového nebo abstraktního typu. Struktura těchto dat se mění v různých směrech. Pro prostorová data dochází ke změnám v topologii geometrie nebo mřížky případně v hodnotách. U abstraktních dat pak ke změnám v topologii sítě nebo struktuře dokumentů. Pro tuto část dat je velmi důležitá myšlenka provázání dat. Většinou se tato data budou brát jako hodnoty přiřazené k bodu v čase. Také platí, že data mají často větší význam, pokud jsou ve správném kontextu, tedy například prostor a čas. Je zde velký rozdíl mezi postupem v prostoru a v čase. V prostoru je pohyb možný ve všech směrech. V čase je však tento pohyb omezen na pohyb směrem dopředu, nelze jít zpět a pak opět dopředu. Jinak řečeno, čas je určité seřazení událostí, které dávají smysl jen v pořadí, ve kterém se odehrály. Jako dobré příklady těchto dat poslouží EKG, sledování počasí (jeho vývoj), monitorování různých hodnot, jako jsou například kryptoměny nebo investice.

Na časově orientovaná data je pohlíženo jako na data, která se mění v průběhu času, nebo události, které hrají hlavní roli, případně jsou středem zájmu. To, jaká data jsou vázána na čas, má velký dopad na návrh samotné vizualizace. Jak již bylo řečeno, k časovým jednotkám lze přidat různé datové typy, atributy nebo charakteristiky. Tedy je například možné pracovat s daty, která popisují vývoj na určitém území v daném čase (popis významné události). Nebo je možné získat data, která ukazují na vznik významných dokumentů k určitému datu.



Obrázek 2.11: Časové vztahy mezi jednotlivými časovými událostmi (a), které jsou jednorázové, (b) jsou pak vztahy mezi časovými úseky.[7]

Zatím zde nebyla definována žádná časová primitiva, se kterými by se dalo pracovat. Časová primitiva jsou ukotvená, nebo neukotvená. Příklad pro neukotvená data je, že hovor bude trvat hodinu. Zde není jasné, kdy začne, jen jaká bude jeho délka. Ukotvená časová data jsou pak jednotlivé události, například narození významné osobnosti. Nebo jako intervaly, například délka vlády nebo v menším měřítku délka výuky s ukotvením k nějakému datu, a to jak začátkem, tak koncem svého trvání. Ukotvená data jsou vždy pevně definována, u neukotvených dat tento aspekt chybí, a tím pádem je možné definovat vztah těchto dat pro lepší představu.

Struktura času se pak liší v závislosti na případu užití. Při vizualizaci je možné pracovat s lineární, cyklickou nebo větvící se strukturou. Lineární struktura odpovídá přirozenému vnímání času. Jde o posloupnost časových primitiv a časový průchod je pak od minulosti do budoucnosti. Cyklická struktura je sestavena z konečně mnoha opakujících se časových primitiv. Jako příklad poslouží roční období. Poslední strukturou je větvící se časová osa. Tato struktura je zobrazována jako graf, kde časová primitiva jsou vrcholy grafu. Může jít o různé události, které mohou nastat. Jedná se o orientovaný graf a vrcholy, které mají více než jednu odchozí hranu, označují místa, kde je přítomný alternativní scénář, který je částečně relevantní vůči plánování nebo predikcím. To, k jaké struktuře daná časová data patří, nemusí být vždy plně rozhodnuto a může být závislé na uživateli. V některých případech je lepší poskytnou možnost zobrazit daná data ve více strukturách a z nich může být možné vyčíst zajímavé úkazy.

Dalším důležitým aspektem časových dat je granularita času, se kterou jsou tato data měřená. Tato granularita bývá různá a tím pádem se s danými daty nemusí pracovat příliš dobře. Ne všechny události se musí nutně měřit stejně a také platí, že granularita se může měnit v čase. Proto je v některých případech nutné provádět konverzi, která nemusí být vždy jednoduchá. Vždy je jednoduché přecházet od časových bodů s nižší granularitou k bodům s vyšší. Pokud je nějaký časový bod změřen s přesností například na vteřiny, je jednoduché přejít na minuty, hodiny, dny a tak dále. Pokud však je časový bod změřen ve dnech, pak přesné určení hodiny a minuty není možné, jelikož tyto údaje chybí.

Kapitola 3

Klasifikace úloh

Tato kapitola obsahuje stále obecnější pohled na práci s daty. Budou zde probrány jednotlivé úlohy a související pojmy. Úlohy budou probrány pouze u typů dat, které jsou popsány výše. Jedná se tedy o úlohy nad prostorovými daty, tabulkovými daty, směsí prostorových a abstraktních dat a nakonec nad časově proměnnými daty. Zde je opět čerpáno z [3, 4] a jako doplňkový zdroj [5].

Při vizualizacích se musí velmi často řešit otázka, proč se vlastně daná data vizualizují. Většinou se vizualizacemi chce poukázat na různé fenomény, které se v datech vyskytují. Úlohy, které se řeší nad jednotlivými daty, je nutné oddělit od domény, ve které se pracuje. Samotné úlohy mohou mít různou podobu, obecně platí, že se v každé úloze vyskytují dvě části. Akce udává, co se bude dělat s daty, a typicky se jedná o sloveso. Cíl je pak předmět v dané větě. Jako jednoduchý příklad poslouží úloha „Potřebuji srovnat hodnoty mezi dvěma skupinami“.

Za akce se považuje analýza, dotaz či vyhledávání. Tyto podčásti mají své další možnosti, které zde budou postupně rozebrány. Jako cíle úloh pak například jsou všechna data, jednotlivé atributy, sítě, prostorová data a další. Jednotlivé cíle sledují například trendy, jednotlivé atributy, topologie nebo tvary.

3.1 Analytické akce

Mezi tyto akce se počítají dvě, a to konzumace a produkce. Konzumace znamená využití cílovým publikem, které může objevovat, prezentovat nebo si danou vizualizaci užívat. **Objevování** většinou využijí experti, kteří si chtějí ověřit své hypotézy nebo nějaké formulovat. **Prezentace** slouží především k předání znalostí a vizualizace je pomocný prvek určený k lepšímu pochopení dané látky. **Užívání** si některé vizualizace znamená, že uživatelé nemají potřebu nic ověřovat ani generovat (hypotézy), ale je uspokojována samotná zvědavost. Uživatel tím pádem získává bližší povědomí o dané disciplíně a proniknout do jejích zákoutí. Vizualizace samotné bývají využity k **produkci** nových dat. Lze v nich **anotovat** různé prvky nebo přiřazovat názvy uzlům v síti, typicky se jedná o manuální akci ze strany uživatele. Z vizualizací lze **vytvářet** různé snímky nebo nahrávky, které pak jsou nápomocné při výuce. Tyto záznamy jsou trvalejšího charakteru než anotace, u které se počítá s dočasným označením. Případně lze **odvozovat** nové atributy z těch existujících. V mnoha případech se pak tato akce provádí z důvodu lepší vizualizace dat. V některých případech je totiž vhodné vstupní data nejdříve transformovat a až poté zobrazit.

3.2 Dotazové akce

U tohoto typu akce je snaha identifikovat nějaké části, srovnat je nebo poskytnou přehled/shrnutí. Při **identifikaci** je pozornost upoutána na jeden cíl. Zde je veškerá jeho charakteristika odvozena z vizuální reprezentace. Při **srovnávání** je pak cílem několik bodů zájmu, které se srovnávají, což je mnohem náročnější než identifikace. Při zkoumání **přehledu** jsou cílem všechna data, se snahou poskytnout celkový pohled na data.

3.3 Vyhledávací akce

Tyto akce jsou rozsáhlejší. Je také nutné podotknout, že tento typ akce je velmi často součástí analytické akce. Vyhledávací akce jsou tyto: lookup[1], nalezení, prohlížení a průzkum. **Lookup** znamená, že cíl i poloha jsou známé a stačí přečíst výsledek. Jako příklad poslouží „Kdo vyhrál volby ve středočeském

[1] Jednoduše řečeno „kouknu a vidím“.

kraji“. **Nalezení** znamená, že je známý cíl, ale jeho poloha ne, a je nutné dané místo nalézt. Opět příklad „Kdo vyhrál volby v Břeclavi“. Při **prohlížení** je cíl neznámý, ale jsou známy jeho charakteristiky, a je třeba najít příslušné cíle. Příklad „Ve kterých městech prohrálo hnutí ANO s velkým rozdílem od prvního“. Poslední **vyhledávací** akce je průzkum. Během něj nejsou známy cíle ani charakteristiky a využívá se k vyhledávání zajímavých dat ve větší množině dat.

3.4 Všechna data jako cíl

V případě práce se všemi zkoumanými daty je cílem získat zajímavé informace. Například informace o trendech, kdy se charakterizují vzorce v datech, zisky nebo poklesy, maxima, průměry a další. V datech se také objevují tzv. outliers, což jsou body, které jsou mimo trend ve vzoru dat. Data samotná mají své charakteristiky a ty je možné z nich získat. Typicky doménově specifické. Tyto cíle jsou validní pro veškerá data.

3.5 Atributy jako cíl

V rámci atributů jsou cíle vztáhnuty k jednotlivým atributům nebo k nějaké jejich skupině. Při zkoumání jednoho atribut, je hledána distribuce pro daný atribut nebo extrémy, které prozrazují další informace. Cílem také bývá větší množství atributů, kdy z nich lze vyčíst závislost jednoho atributu na jiném. Korelace, tedy tendence pro hodnoty jednoho atributu blížit se k hodnotám jiného atributu. V některých případech jsou zkoumány i podobnosti, jedná se typicky o kvantitativní měření získané pro dva atributy.

3.6 Sítě jako cíl

Některé cíle se týkají specifických množin dat. Síťová data specifikují vztahy mezi jednotlivými uzly a jejich hranami. Zde se pak zkoumá například topologie, což znamená strukturu samotné sítě. Případně je možné zkoumat mnohem specifičtější topologický cíl, a to cesty mezi dvojicemi uzlů v síti.

3.7 Úlohy s daty

Tato sekce se pak zabývá přiřazováním úloh k dříve zmíněným typům datových množin. Postupně zde budou rozebrány úlohy pro prostorová data, tabulková a relační data, směs abstraktních a prostorových dat a časově orientovaných dat. Každá z těchto částí má své úlohy, některé specifické, jiné sdílené s ostatními typy.

Důležitou otázkou ve vizualizaci je, jak je uživateli umožněno docílit jeho úkolu. K tomu slouží celá řada pomocných procesů, kterých si uživatel nevšimne. Mezi ně patří zakódování atributů do různých grafických elementů. Manipulace s daty pomocí změn parametrů, výběru nebo navigace v datech. Samotná organizace v pohledech na data může být velmi nápomocná. O těchto částech bude řeč v dalších kapitolách.

Prostorová data. Tato data mají jako nejčastější úlohu zjištění tvaru zkoumaných dat. Mimo to jsou sestavovány úlohy, kterými budou uživatelé zkoumat zajímavá místa a hledat určité zvláštnosti v datech. Zde je vidět zastoupení úlohy identifikace hodnot, které jsou v dané oblasti významné. Dalším zástupcem je porovnání hodnot dvou míst, která mohou být sousední, nebo naopak v naprosto odlišných místech. Příkladem k této úloze je dotaz „V jakých místech je teplota vyšší než 50°C“. To hraje kritickou roli pro některé látky. Je také možné využít analytických úloh na objevování nebo prezentování. Kromě hodnot v prostorovém poli bývá zkoumána i geometrie. Kdy se uživatel snaží porozumět tvaru v datech, a srovnává ho s ostatními geometrickými tvary. Kromě tvarů lze také zkoumat vzdálenosti a blízkosti těchto geometrických objektů.

Tabulková a relační data. Tabulková data mají typický požadavek na identifikaci[2] různého počtu atributů nebo položek. Uživatel hledá jednotlivé atributy a k tomuto úkonu mu pomůže právě vizualizace. Ta pomáhá se zodpovězením otázek ohledně hodnoty nebo rozsahu hodnot, jaká je hodnota atributu nějaké položky nebo jaké je rozložení položek pro daný atribut (statistické metody). Dále jsou také na základě úloh zkoumány jednotlivé položky, například které položky mají určitou hodnotu některého svého atributu (pro všechny položky stejný atribut), stejné je to pro rozsah hodnot. Případně lze pomocí vizualizace splnit úlohy, které zahrnují mnoho atributů i cílů a chtějí zjistit vzájemné vztahy. Příkladem je zodpovězení otázky ohledně korelace nebo podobných statistických veličin. Tato část také zvládne

[2] Což je základní úlohou u tohoto typu dat.

zodpovědět úlohy, která se týkají vztahů mezi atributy. Co je také velmi důležité a nemělo by být opomenuto je, že s větším množstvím atributů roste obtížnost dané úlohy.

Pro relační data je pak možné mít úlohy pro jeden atribut, položky, vztahy mezi jednotlivými částmi nebo obecné charakteristiky vytvořeného grafu (sítě). Při úlohách s jedním atributem je opět doptávána hodnota jednotlivých atributů nebo hodnoty u některé z položek. S více položkami pak platí, že se úlohy týkají hodnot pro nějaký atribut u všech položek, rozsahy hodnot atributů nebo incidence se spojnicemi[3]. Pokud se uživatel bude zabývat úlohami ohledně spojnic pak typicky zkoumat které spojnice mají určitou hodnotu případně rozsah nebo jaké položky spojuje. Při popisu sítě bývá cílem popis určité cesty nebo komponenty. Tato data mají také vztah se stromy a sítěmi a proto je možné využít některé úlohy i v daných datech.

Směs abstraktních a prostorových dat. Tato data jsou zajímavější pro mnoho oblastí. Umožňují podávat informace vztažené k regionům nebo zajímavým místům. Od toho se také odvíjí typické úlohy. Opět je využíváno identifikace, kde se uživatel pokouší nalézt například charakteristiky nebo hledá nějaký cíl s danou charakteristikou. Případně je využito srovnávání hodnot více cílů a získání celkového pohledu na prezentovaná data. U srovnání jde zejména o porovnání hodnot u více cílů. Mimo tyto úlohy bývá řešena i vzdálenost. Uživatel zkoumá nejbližší instituty, restaurace a podobně. Dále úlohy zaměřené na dopravu z místa A do místa B, kdy uživatel hledá cestu s nejlepšími parametry. Případně úlohy zaměřené na pohyb nebo tok, kde jako příklad poslouží migrace druhů nebo přesun lidí mezi městy.

Časově proměnná data. U tohoto typu dat je dobré rozdělit úlohy do dvou podčástí. Jedna se zabývá úlohami, které jsou přímo vztaženy k časovým jednotkám, se kterými se pracuje. Druhá pak už k atributům, které jsou na tyto časové jednotky navázány.

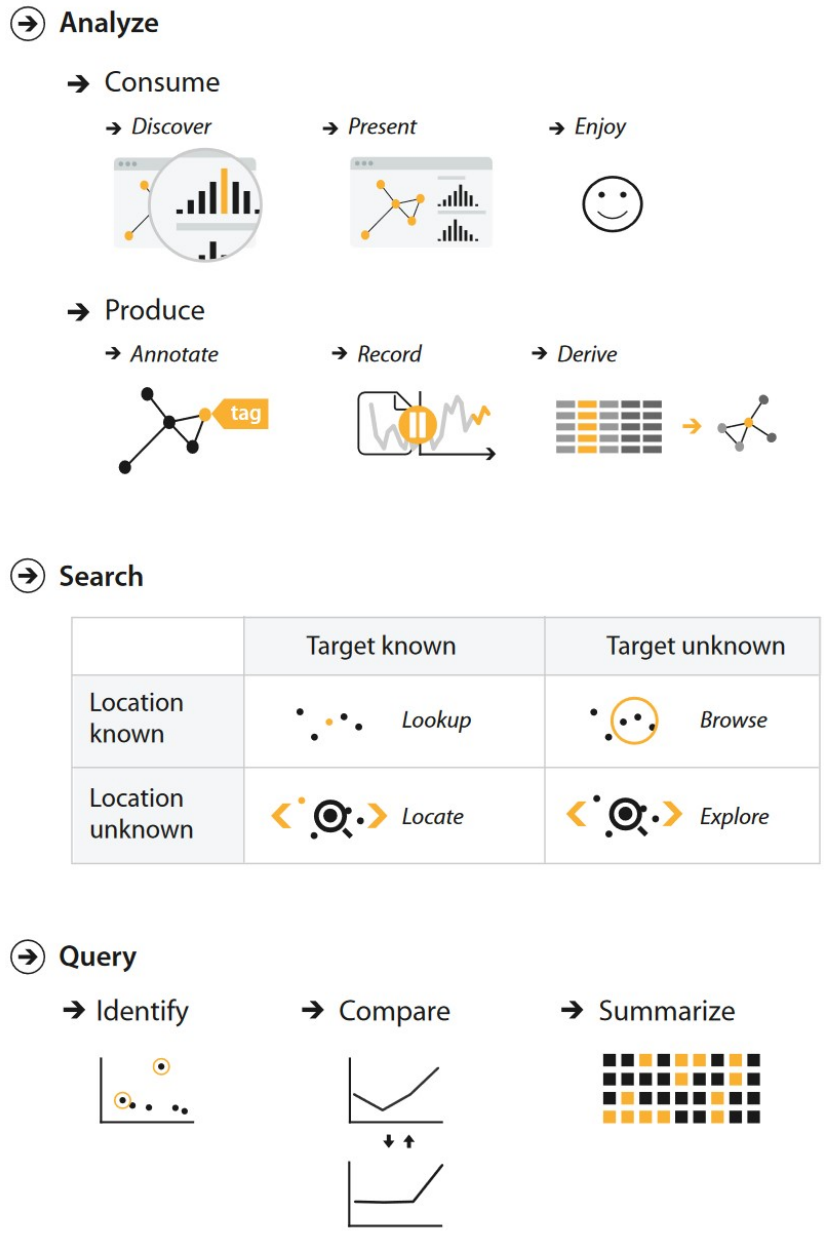
S časovými jednotkami jsou prováděny úlohy typu existence časových primitiv, kdy se zkoumá, zda bylo například provedeno nějaké měření ve specifický čas. Kdy má nastat nějaká událost nebo akce, čili otázka typu „Kdy probíhají státní závěrečné zkoušky.“ Úlohy se zabývají i samotnými intervaly a řeší například, jak dlouhý je nějaký interval. Příkladem může být „Jak dlouhé je zkouškové období.“ Jako další typ úlohy je studování pořadí jednotlivých událostí. Nebo určitý druh synchronizace, kde se uživatel doptává, zda je nějaká událost v daném časovém úseku nebo v daném časovém primitivu. V případě více časových os vedle sebe je možné provádět kontextové ověřování

[3] Co je kam napojeno.

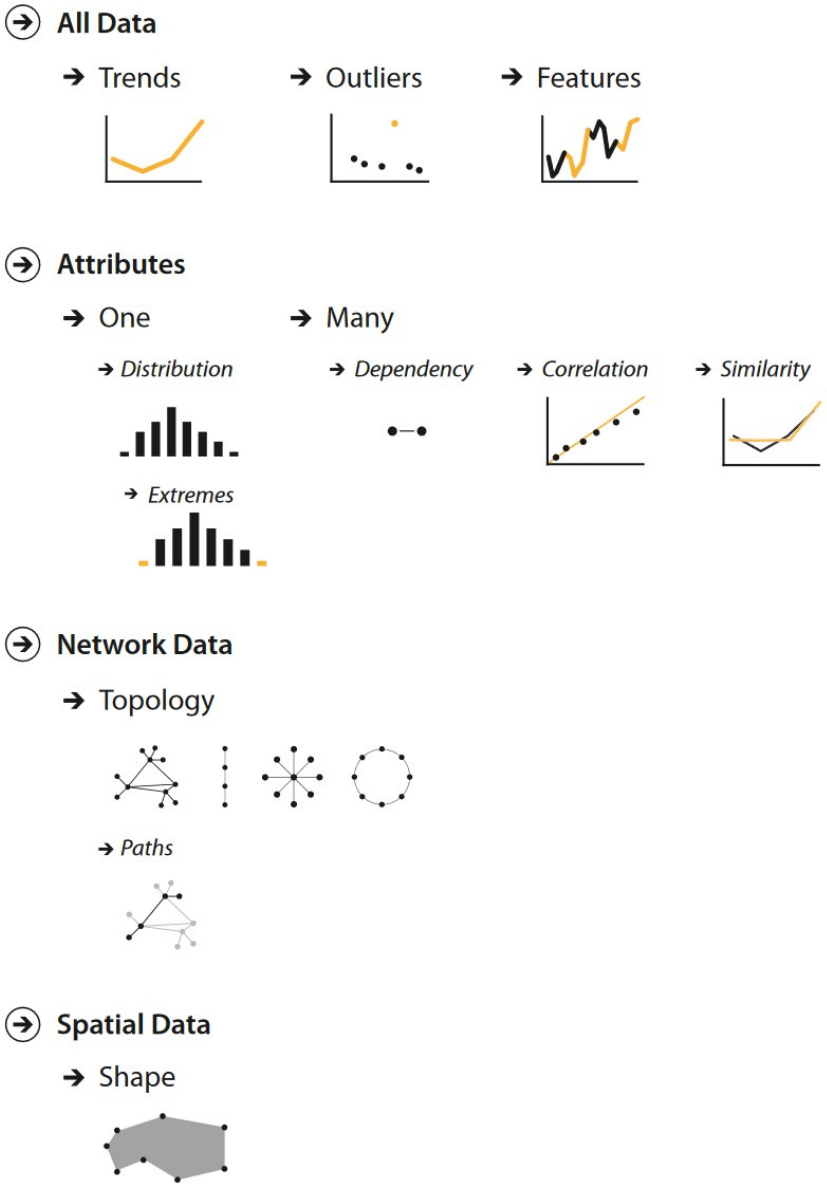
informací, nebo vyhledávání různých časových úseků/primitiv. Nastávají i situace, kdy jsou prováděny i ostatní akce typu analýza, například pro výukové účely nebo produkování dalších částí.

V případě práce s hodnotami atributů a jednotlivými atributy jsou prováděny úlohy pro identifikaci nebo srovnávání těchto hodnot. Zkoumány jsou například ceny určité komodity v daném čase nebo množství určitého zboží. Případně jsou sledovány trendy v naměřených datech. Při porovnávání jsou srovnávány hodnoty dat v různých časových okamžicích, například cena stejné komodity ve dvou různých časových okamžicích, které dávají smysl. Velmi často se provádí porovnání hodnot různých datových položek ve stejném čase, například hodnoty dvou konkurenčních firem. Úlohy zabývající se trendy v datech v různých časových jednotkách, nebo trendy v různých datech při stejné časové časové jednotce. Tak jako u časových jednotek, i zde se najde příležitost k použití obecných dotazů nad identifikací přesné hodnoty/položky[4] či srovnání hodnot. Případně poskytnout přehled nebo shrnutí. Pak je samozřejmostí vyhledávání v datech, kdy uživatel hledá přesné údaje, provádí hledání na základě známých hodnot atributů a tím pak například zjišťuje, kdy se dané měření/události odehrály. V neposlední řadě je pak prováděno prozkoumávání dat a hledání významných částí.

[4]Zde záleží na typu dat, která jsou přidružena k časovému primitivu.



Obrázek 3.1: Přehled různých akcí, které jsou s vizualizací vykonávány [3]



Obrázek 3.2: Přehled cílů, kterých může chtít uživatel dosáhnout [3]

Kapitola 4

Vizuální reprezentace atributů

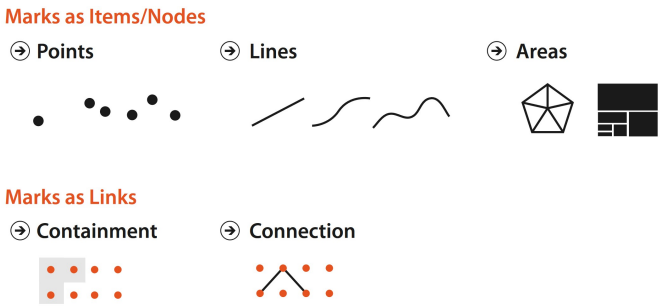
Pokud jsou známá data a další potřebné detaily, pak je nutné zajistit přehlednost a použitelnost dané vizualizace. Vizuální reprezentace je velmi důležitou částí a nesmí být brána na lehkou váhu. Tato část bude obsahovat hlavní principy, které je dobré dodržovat, a k tomu navíc přehled možných vizuálních reprezentací dat různého druhu. Veškeré informace vychází z [3] s doplněním z [8].

Každá vizualizace má jiné požadavky a je nutné vybírat vždy efektivní vizuální kanál pro daná data. Některé kanály jsou vnímány lépe než ostatní, například velikost. Jiné kanály slouží lépe k určení kategorie, než k porovnání. Každý prvek, který je vizualizován, může být rozložen až na prvočinitele, které vytváří danou reprezentaci. Tu je pak možné analyzovat a poté vyhodnotit, zda byly použity vhodné části.

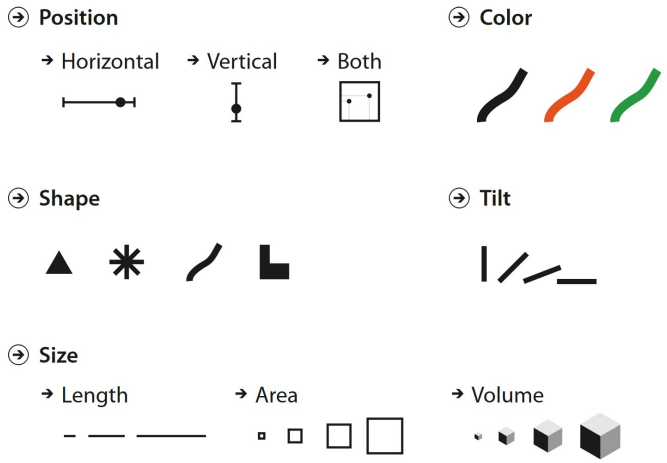
4.1 Základní pojmy a vlastnosti

V [3] jsou rozlišeny dva základní pojmy – značka[1] a kanál[2]. **Značka** se liší v závislosti na dimenzi. Pro nultou dimenzi se jedná pouze o body. Jednodimenzionální jsou pak různé typy přímek a čar, pro dvě dimenze se jedná o plochy a u tří dimenzí o objem. Poslední jmenovaná není příliš často

[1] Mark
[2] Channel



Obrázek 4.1: Ukázka různých typů značek [3]



Obrázek 4.2: Příklady různých kanálu [3]

využívána. **Vizuální kanál** ovlivňuje to, jak vypadají jednotlivé značky, kde jsou, jejich tvar a mnoho dalších vlastností. Jako tyto kanály je možné využít například pozici, barvu, tvar, náklon, délku, objem, tloušťku nebo šířku. Těchto vizuálních kanálů je velké množství a je nutné vědět, jaký je cíl vizualizace, a dodržovat určitá pravidla.

Při vizualizaci je možné využít více vizuálních kanálů a kombinovat různé atributy dohromady. To v některých případech vede ke špatné orientaci v dané vizualizaci. Vždy je nutné zvolit vhodné kanály pro zkoumaná data. Také platí, že některé kanály nemohou být použity ve všech dimenzích. Pokud je předmětem vizualizace například plocha, pak nastává problém s využitím velikosti a tvaru. V tomto příkladě je velikost omezující podmínka daného prvku, a tím pádem ji není možné využít, což platí i o tvaru. Stejně jako u prostorových dat není možné volně mapovat do prostoru, protože data jsou vztažena k určitému místu v prostoru.

4.2 Typy značek a vizuálních kanálů

Lidské vnímání má dva rozdílné senzory vjemu. Prvním je **identita**, která napomáhá rozpoznat informaci o tom, jaké něco je nebo kde to je. Také určuje tvar, tedy zda se jedná o kružnici, puntík, trojúhelník nebo jiné tvary. Mimo to je také dobrým příkladem odstín nebo určení místa, se specifickými vlastnostmi.

Naproti tomu **velikost** udává množství nebo rozměr určité věci. U velikosti je to pak hlavně délka, plocha nebo objem. Na tyto vlastnosti je možné se doptat a porovnat, o kolik je něco delší či větší. Podobné dotazy se týkají i jiných kanálů, například o kolik jsou některé části tmavší nebo světlejší než ostatní, nebo na různé rozdíly úhlů, prostoru, velikosti značek a podobné otázky.

Výše zmíněné poznatky jsou zejména platné pro tabulková data, kde značka je vztažena k jednotlivým položkám. U sítí reprezentuje jedna značka jednu položku (případně uzel) nebo hranu mezi tabulkami. Hrana reprezentuje vztah mezi položkami. Pro tyto hrany jsou dva typy: hrany mezi uzly a obsažení. **Hrany mezi uzly** reprezentují vztah mezi dvěma položkami. Samy o sobě však nemohou být reprezentovány body. Vznikl by chaos při mentálním spojování bodů, což je v některých případech až nemožné. **Obsažení** je pak určitý hierarchický vztah, který využívá plochy a k tomu může využít vnořených značek do sebe na více úrovních. Zatímco vizuální reprezentace značky plochy bývá pouhá lomená čára k zobrazení hranic, obsažení je o využití plochy, která je ohraničena. Zde je vidět, že pro značky je nutné uvažovat, co mají reprezentovat.

4.3 Využití značek a vizuálních kanálů

Z předchozí sekce vyvstává otázka, zda jsou si všechny kanály rovny. Bohužel tomu tak není. Pokud se vezme atribut dat a využije se různého zakódování, pak informace, které budou viditelné, mohou být interpretovány různě. To rozhodně není žádoucí efekt a u každé vizualizace by se na tuto skutečnost mělo dbát. Základní vlastnosti, které by měly pro vytvořené značky platit, je efektivita a expresivita (obě tyto části budou popsány níže). Na základě těchto dvou vlastností je možné sestavit žebříček jednotlivých kanálů pro data, která jsou předmětem vizualizace.

4.3.1 Expresivita

Tento princip říká, že vizuální reprezentace by měla vyjadřovat všechny a pouze ty atributy, které jsou obsaženy ve zkoumaných datech. Nejzákladnějším expresivním principem je přímé vizuální uspořádání[3] u řaditelných dat. U neseřazených dat je tomu naopak. Neřazená data by se neměla zobrazovat tak, aby implikovaly řazení, které neexistuje. Jedná se o aspekt vizualizací, který by neměl být nikdy podceňován a jeho důležitost je vidět již v klasifikaci datových atributů.

4.3.2 Efektivita

Princip efektivity říká, že důležitost má odpovídat význačnosti kanálu, který je použit. Jinak řečeno nejdůležitější atributy je vhodné zakódovat s nejefektivnějšími kanály tak, aby si jich uživatel vždy všiml jako prvních. Poté s klesající důležitostí jsou přiřazeny atributům méně efektivní kanály.

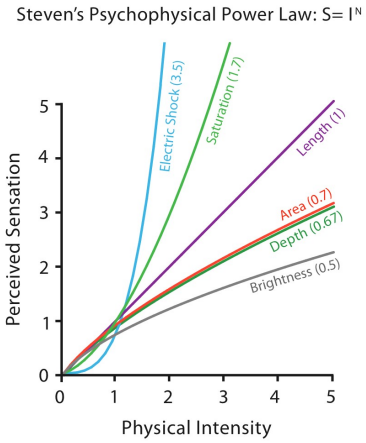
Kapitola 5.5 [3] pojednává o efektivitě jednotlivých kanálů a proč jsou dané kanály tak či onak efektivní, jaké jsou další možnosti a podobné detaily. Zde budou uvedeny a popsány pouze některé aspekty určení, jak je který kanál efektivní.

Přesnost. Efektivitu kanálu kvantifikovatelná pomocí přesnosti. Je však otázka, jak takovou přesnost měřit. Jinak řečeno, jak blízko je usuzovaní z lidského vnímání vůči nějakému objektivnímu měření jevu. Pro řešení tohoto problému je třeba sáhnout do jiného odvětví, a to do psychofyziky, což je odvětví psychologie, která je zasvěcena systematickému měření vjemů člověka. Různé vizuální kanály jsou vnímány s různou úrovní přesnosti a ne všechny kanály jsou stejně rozlišitelné. Zde se uplatňuje Stevensovo pravidlo, které udává vnímavost vůči určitému kanálu. Toto pravidlo lze formulovat následujícím způsobem

$$S = I^n \quad . \tag{4.1}$$

Zde S je míra vnímavosti vůči danému kanálu, I je pak fyzická intenzita

[3] Data jsou vizuálně uspořádána tak, že pořadí je ihned viditelné.

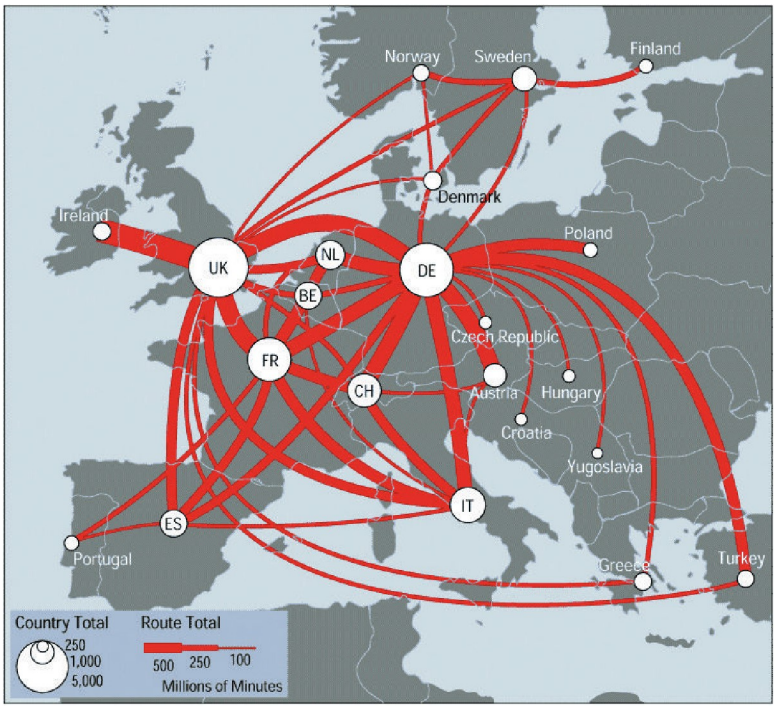


Obrázek 4.3: Dle Stevense je možné ukázat, že relativní velikost všech vnímaných kanálů následuje mocninné pravidlo, kde některé vjemy jsou vnímány silněji, než jaká je jejich objektivní intenzita, a některé jsou naopak potlačeny. Co se týká různých kanálů, tak délka je vnímána přesně, naproti tomu plocha je potlačena a barevná saturace je zesílena [3]

kanálu a n je exponent. Tento exponent může získat hodnoty od sublineárních (0.5) až po superlinearní (3.5), kde každý kanál má svou hodnotu n, což je vidět na grafu 4.3. Hodnota n říká, jak moc je vnímání citlivé vůči danému kanálu. Kanály, které jsou blízko jedničky, člověk vnímá jako téměř správné hodnoty. Pokud je n menší než jedna, pak hodnota s takovým kanálem bude vnímána jako menší než reálná hodnota. Opačně pak pro n větší než jedna, kde vnímaná hodnota bude větší, než ve skutečnosti je. Zdroj [3] pak obsahuje další pojednání o různých měřeních, porovnání jednotlivých kanálů a jejich přesnosti, případně chybovosti.

Rozlišitelnost. U tohoto aspektu se řeší otázka, zda jsou data využívající určitý vizuální kanál dostatečně rozlišitelná pro lidské vnímání tak, jak bylo zamýšleno. Vizuální kanál je tedy charakterizován určitým počtem přihrádek, do kterých jsou data rozdělitelná. Každá přihrádka musí být jasně rozlišitelná od ostatních. Zde je dobré zvolit rozumné rozdělení do jednotlivých rozmezí a dávat pozor, aby nedošlo k záměně vizuálních kanálů. Například pokud se bude zvětšovat šířka vizuálního kanálu úsečky, pak dojde při určité šířce k záměně s plochou. Proto je nutné přemýšlet i nad možnou záměnou tvarů.

Oddělitelnost. Zde přichází ke slovu propojení jednotlivých kanálů. Obecně nelze brát všechny vizuální kanály jako naprosto nezávislé na ostatních, zejména kvůli interakci s ostatními. Pro tento aspekt je nutné uvažovat potenciální interakce mezi kanály pro každý pár od ortogonálních a nezávislých kanálů k nerozdělitelným kombinacím shrnujících kanálů. Vizuální reprezen-

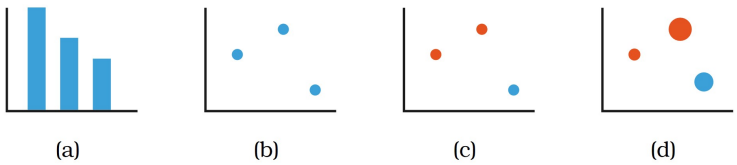


Obrázek 4.4: Zde je ukázka špatné rozlišitelnosti. Uživatel nemusí být schopen určit jakou velikost má která z hran. Ty mohou být jen o malý kousek užší než největší ukázaná data a tím pádem se přiřadí hodnota do špatné přihrádky [3]

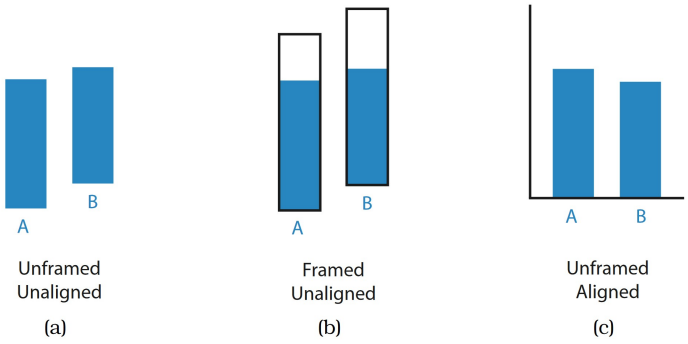
tace je přímočará s dobře oddělitelnými kanály, ale pokus o reprezentaci různých informací v shrnujícím kanálu selže. Jinak řečeno pokud se bude dávat dohromady mnoho vizuálních kanálů, pak může dojít k tomu, že uživatel nebude schopen rozlišit jednotlivé informace. Místo toho budou přijaty nepravdivé nebo zkreslující informace o daném atributu.

Shlukování. Efekt shlukování může vzniknout správným použitím různých vizuálních kanálů či hran mezi jednotlivými položkami. Tyto vztahové vlastnosti bývají reprezentovány mnoha způsoby, například prostorem, ve kterém se určitá data nachází, nebo samotnými hranami. Z těchto dvou prezentovaných možností je lépe vnímána prostorová souvislost (obsažení v nějaké skupině) a po ní teprve hrany.

U kategorických dat je možné využít pro označení skupin kanálů identity, pro připomenutí poslouží obrázek 4.9. Zde je vnímání přenášeno mezi jednotlivými kategorickými atributy, pokud všechny sdílejí prezentovaný atribut. Člověk není nucen vynakládat velké mentální úsilí k pochopení dané vizualizace, a tím pádem zjednodušuje její studování. Tato reprezentace není tolik vhodná jako spojování hranami, ale je zde možné využít snadného přístupu s ohledem na přeplnění vizualizace. Nevzniká zde zbytečný vizuální



Obrázek 4.5: V tomto obrázku je možné vidět různé spojení značek a kanálů. Obrázek (a) udává spojení dvou atributů, kde vertikální je kvantitativní a horizontální kategorický. Obrázek (b) pak ukazuje dva kvantitativní atributy, kde u obou je využito prostoru. Obrázek (c) přidává další atribut s využitím barevného kanálu. Obdobně v obrázku (d), kde je přidán další kanál, a to velikost. Je vidět, že je možné za určitých podmínek spojovat různé kanály a atributy tak, aby pomohly dokreslit příslušnou vizualizaci [3]



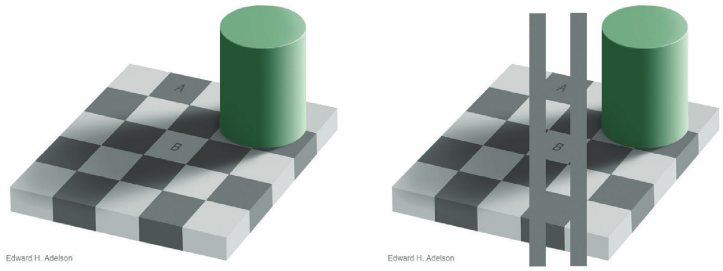
Obrázek 4.6: (a) Znázorňuje dva sloupce, které nejsou zarovnány ani ohraničeny s mírně jinou velikostí, a je tedy těžké je porovnat. (b) Pak přidává rámečky, které umožňují zlepšit porovnání jednotlivých částí a přesněji určit jejich vztah. (c) Při zarovnání je pak velmi snadné porovnat prezentované hodnoty. [3]

nepořádek[4].

Dalším velmi dobrým přístupem k vizualizaci kategorických dat je **blízkost**. To znamená, že položky, které mají stejné nebo podobné hodnoty, jsou prostorově umístěny do podobného místa. V této myšlence je viditelné, že prostorové shlukování je jedním z nejlépe vnímaných vizuálních kanálů pro kategorická data. **Podobnost** je další kanál, který je vhodný k využití u kategorických dat. Zde dle obrázku 4.9 je vhodné využít například podobnost v pohybu, odstínu nebo tvaru[5]. V [3] je dále uvedeno, že z logického pohledu má poslední zmiňovaný kanál vlastnosti podobné blízkosti. Nicméně z perceptuálního pohledu na tuto problematiku má vnímání prostoru lepší efekt než podobnost nebo ostatní kanály.

[4]Přítomnost hran v jiných případech.

[5]Tvar musí být pečlivě zvolen, respektive s dobrou mírou rozlišitelnosti jednotlivých tvarů.



(a) (b)

Obrázek 4.7: Vjem osvětlení je založen na relativním posouzení. Tedy dva čtverce A a B se mohou jevit jako různé (a), ale po přidání jednolitě šedých proužků je vidět, že se jedná o stejné barvy (b). [3]

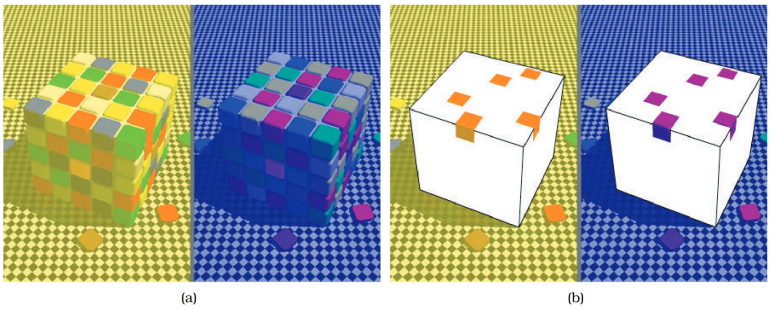
4.3.3 Relativní vs. absolutní posouzení

Lidské vnímání je založeno na relativním vnímání a posuzování vnímaných objektů. Při absenci měřítka každý uvede svou odhadovanou délku nebo velikost nějakého prvku a zde je vidět ona relativnost. Tato skutečnost je známá jako Weberovo pravidlo[6]. Například jak velké množství rozdílu v délce je člověk schopen vnímat, je jen procentuální část absolutní délky.

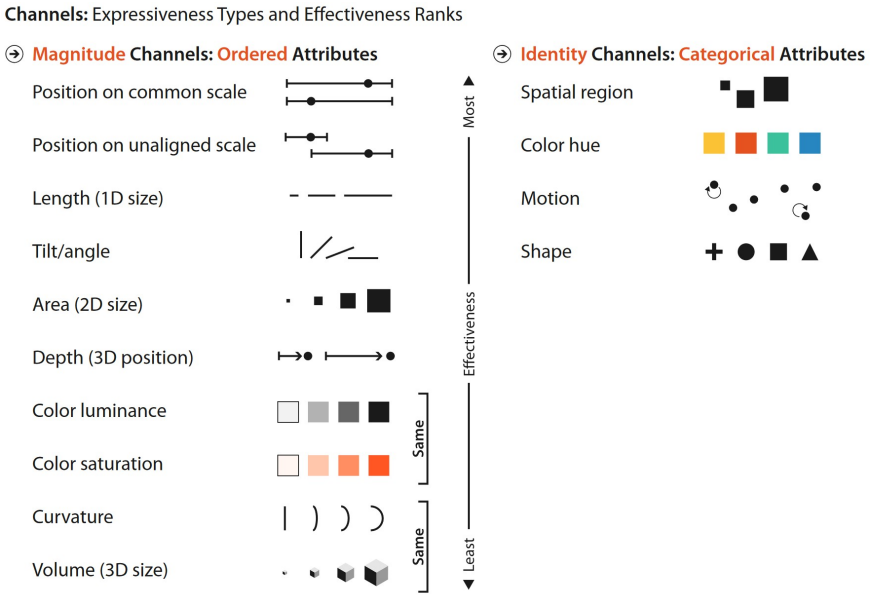
Fakt, že člověk vnímá relativně a ne absolutně, má dalekosáhlé dopady. Navíc se tento fakt odráží v mnoha senzorických vjemech. Zde opět přichází ke slovu přesnost a rozlišitelnost jednotlivých kanálů lidským vnímáním. Je tedy nutné rozlišovat mezi absolutním a relativním posouzením daného kanálu. Příklad těchto vjemů je vidět na obrázku 4.6, kde je zřetelné, že mnohem přesněji lze vnímat objekty, které jsou vedle sebe a zarovnány, než objekty, které jsou dále od sebe a různě v prostoru. Případná dodatečná informace v podobě ohraničení pomáhá se zpřesněním vjemu. Na příslušném obrázku je také příklad Weberova pravidla, kde je vidět důležitý poznatek. Pokud je u délek a velikostí uvedeno měřítko, pak je možné tyto velikosti mnohem snáze určit a porovnat.

Relativní porovnání se netýká pouze velikostí, ale i barev a osvětlení, kde tyto části jsou kompletně kontextuální, například na kontrastu s okolními barvami. Tuto skutečnost ukazuje obrázek 4.7. Zde je zdá, že čtverce A a B jsou rozdílné stupně šedi, ale při přidání masky je vidět, že tyto čtverce jsou stejné barvy. To je následek relativního vnímání osvětlení a stínu. Podobně je tomu u obrázku 4.8, kde je pro změnu sledována barevnost jednotlivých částí. Při pohledu na červené kousky v části a, je vidět stále stejná barva (nebo při

[6]Formálně se jedná o zaznamenatelný rozdíl v intenzitě vjemu I jako část K velikosti objektu, tedy $\delta I/I = K$.



Obrázek 4.8: Vjem barev je také relativní vůči okolí a závislý na kontextu. (a) Obě kostky mají části, které vypadají červeně. (b) Pokud se však odstraní určitá část vjemu, pak začnou působit jako jinak barevné dílky.[3]



Obrázek 4.9: Řazení kanálů dle jejich efektivity podle typu data a kanálu [3]

nejmenším podobná). Při pohledu na část b je však barva naprosto jiná.

4.4 Řazení vizuálních kanálů

Obrázek 4.9 ukazuje pořadí vizuálních kanálů dle dvou expresivních typů – pro řaditelná data a kategorická data. Pořadí je zde uvedeno od nejefektivnějších po ty nejméně efektivní.

Pokud se pracuje s řaditelnými atributy, pak by měly být zobrazeny jako

velikostní kanály. Nejefektivnější je zarovnaná prostorová pozice následovaná nezarovnanou. Další je délka a úhel. Poté následuje plocha a hloubka. Podobnou vypovídající hodnotu pak má svítivost a sytost barev. Nejméně vypovídající kanály pro řazená data jsou zakřivení a objem.

Kategorické atributy je dobré reprezentovat kanály, které umožní identifikovat tyto kategorie. Nejefektivnějším kanálem je prostorová oblast, které daný atribut patří. A je následována barevným odstínem elementů. Zde je zajímavé, že využití dynamického prvku má stejný efekt. Pohyb je též velmi efektivní kanál, jak ukazuje následující příklad: auta Enyaq, Yeti a Roomster se pohybují po kružnici proti směru hodinových ručiček a Mondeo, Mustang a Focus se budou pohybovat nahoru a dolu o malý kousek. Zde je z pohybu vidět, že první tři patří k sobě a další tři patří také k sobě. Posledním kanálem vhodným pro kategorické atributy je tvar.

Teoreticky je možné využít jmenované kanály i obráceně. Ale efektivita a expresivita by špatným použitím velmi utrpěla. Je zajímavé si povšimnout, že v obou případech na obrázku 4.9 jsou prostorové kanály na prvním místě. Toto je také jediný kanál, který je použit v obou případech bez ztráty efektivity. Opět zde záleží na dimenzi prostoru, ve kterém se uvažují. Mysl uživatelů je velmi často fixována na pozici, a tím pádem velmi často přitahuje pozornost a občas uživatelé podvědomě hledají nějaké souvislosti s prostorem.

Kapitola 5

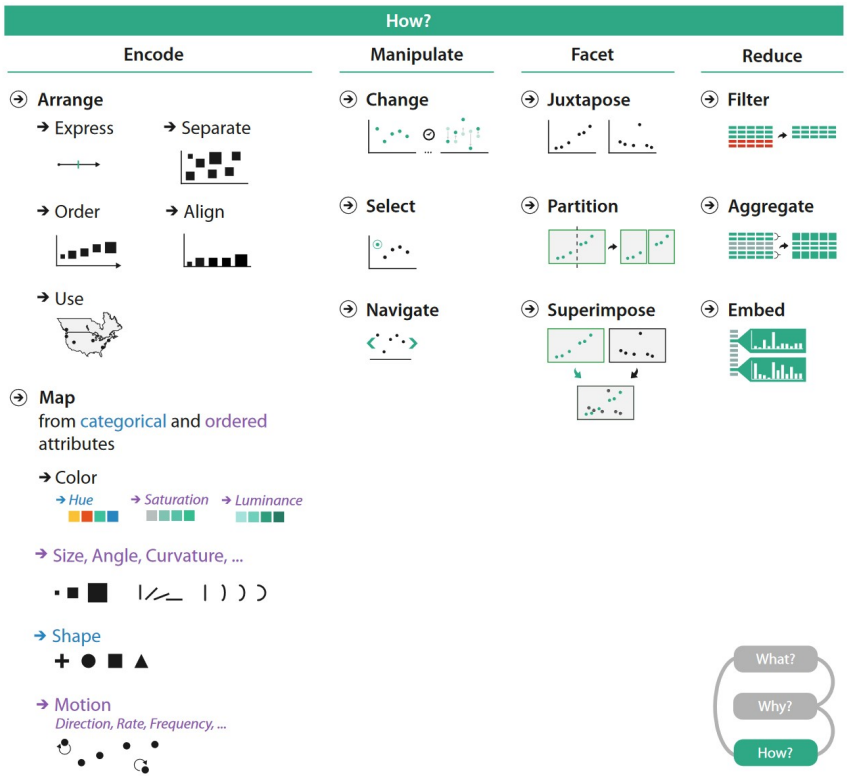
Interakce

V předchozích kapitolách bylo probráno mnoho informací o datech, o jejich zařazení, o úlohách, které je možné s danými daty řešit a které se nabízí uživateli. Poté byly probrány možnosti vizuální reprezentace atributů, jejich vlastnosti a naznačení dopadu na samotnou vizualizaci. Tedy již bylo debatováno, co se vizualizuje, jak se vizualizuje a co je možné řešit. Zatím však nepadlo jediné slovo o interakci s vizualizacemi. Následující sekce vychází z [3] a jako doplňující materiál [9]. Pro specifické informace bylo využito zdrojů [10, 11, 12].

Analýza vizualizace je proces, který se větví, má různé odbočky a zajímavé objevy. Velmi často je interaktivní a může být i animovaný. Samotný prvek interakce je klíčový k umožnění lepšího pochopení komplexních dat. Interakce samotná pak také pomáhá k průzkumu dat, když není jasné, co je cílem analýzy, nebo když se v datech obecně hledá nějaký zajímavý jev. Interakce jako taková rozlišuje vizuální analýzu (průzkum dat) od statické vizualizace. V případě použití interakcí s danou vizualizací je nutné poskytnout přiměřený čas odezvy vizualizace, který se odvíjí od požadavků na vizualizaci.

Pro samotnou interakci je v [3] popsána vizualizační mantra, která je dobrým návodem, jak vytvořit smysluplnou vizualizaci. Samotná mantra je následující – nejdříve přehled, pak filtr a přiblížení, detail na vyžádání.[1]. Ve vizualizaci je vždy dobré snažit se poskytnout uživateli přehled o všech datech, která jsou mu prezentována, trendy a podobné statistické veličiny. Následně umožnit výběr určité podmnožiny dat, se kterou bude dále pracovat. Tento výběr dále přiblížit a zobrazit menší podmnožiny zkoumaných dat. Nakonec

[1] V angličtině pak „Overview first, then filter and zoom, details on demand.“



Obrázek 5.1: Přehled toho, co je vhodné řešit při vytváření vizualizace [3]

zobrazit detaily, tedy jednotlivé hodnoty, pokud jsou vybrány. Mimo základní části je možné dodat navíc některé další interakce. Jako například vztahy mezi zobrazovanými daty a umožnit tak jejich srovnání. V případě provádění více akcí je dobré umožnit návrat ve vykonaných krocích a poskytnout informace o datech, které uživatel mezi jednotlivými kroky mohl přehlédnout nebo zapomenout. Velmi dobrým prvkem interakce je pak označení nebo zaznamenání dat pro případné znovupoužití.

5.1 Úrovně interakce

Ve vizualizacích existují dvě základní úrovně manipulace. První je označována jako **nízkoúrovňová**, což je například mačkání kláves, pohyb myší, kliknutí na určitý prvek nebo chycení a tažení. **Vysokoúrovňové** manipulace jsou např. manipulaci s parametry vizualizace, výběr, přiblížení nebo různé rotace. Dalším nástrojem vysokoúrovňové manipulace je filtrace, kde se provádí výběr podmnožiny dat nebo využití agregace. Interakce na vyšší úrovni pak také zahrnuje vytvoření různých pohledů jako je *juxtaposition*, *superimposition*

nebo *embedding*, případně *distribuce dat v pohledech*, o kterých bude řeč později. [3, 9]

Fittovo pravidlo. Toto pravidlo udává, jak rychle je uživatel schopen vybrat konkrétní prvek. Jedno z možných znění je následující „Fitts' law states that the amount of time required for a person to move a pointer to a target area is a function of the distance to the target divided by the size of the target," [10]. Jinak řečeno čím delší je vzdálenost k požadovanému elementu a čím menší daný element je, tím déle bude trvat, než uživatel zvládne na daný element kliknout. Toto má velký dopad nejen ve vizualizacích, ale i v obecném návrhu aplikací. V těch je snaha sdružit podobnou funkcionalitu nebo spolu související funkcionalitu v malém prostoru u sebe tak, aby se minimalizoval pohyb myší. To samé platí pro vizualizace – při snaze vybrat vzdálený malý element bude celkový čas výběru delší než blízký a větší element. Celý tento zákon byl získán empiricky a jeho matematické vyjádření je následující

$$T = a + b \cdot log_2\left(2 \cdot \frac{D}{W}\right) \quad . \tag{5.1}$$

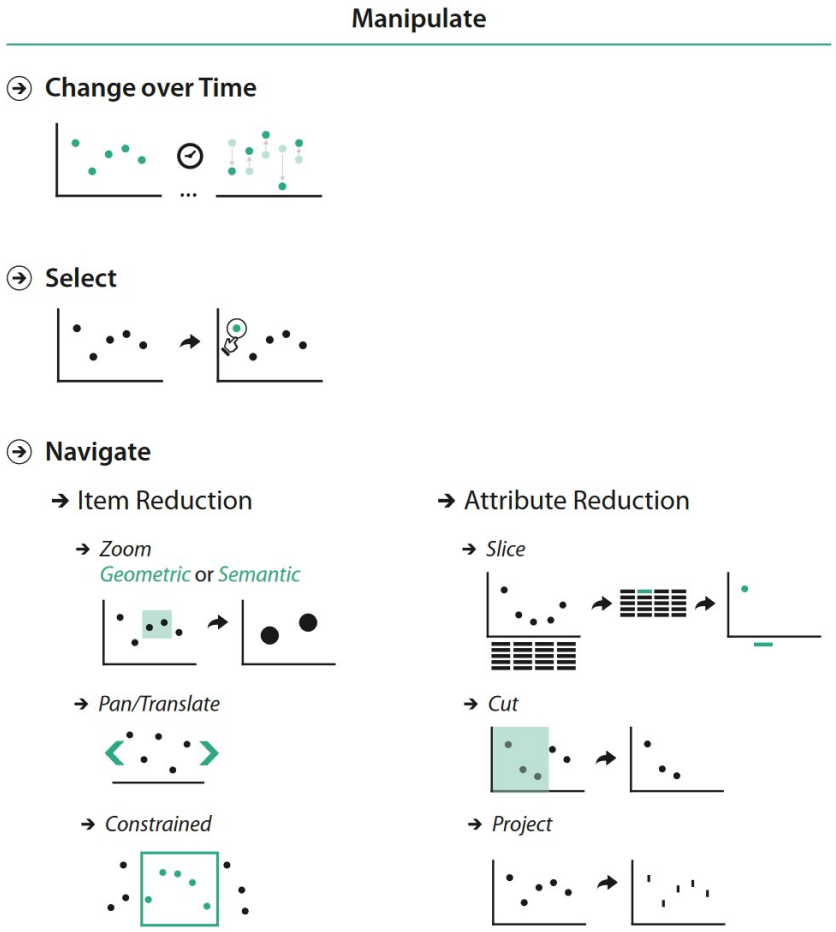
Zde T značí celkový čas na dokončení pohybu. Koeficienty a a b jsou empirické regresní koeficienty. D pak značí vzdálenost z výchozí pozice k cílové poloze. W značí šířku daného objektu. [10]

5.2 Manipulace s daty

Typickým základním prvkem manipulace je pohyb myší, který umožňuje průzkum dat. Tím je myšlen například výběr a následný průzkum podmnožiny dat. Tato sekce se zaměří právě na výběr dat, zkoumání jejich hodnot a různé techniky výběru.

Výběr lze provádět v mnoha ohledech, například vybírat pomocí hodnot, či ukázáním na určité elementy vizualizace. Tyto typy výběru je v různých případech dobré rozšířit o další techniku, jako třeba vyhledání v blízkosti elementu nebo „obarvení" části vizualizace, která je předmětem zkoumání.

Výběr hodnotou. Jedná se o pouhé zadání hledané hodnoty a získání elementů, které mají tuto hodnotu. Tento koncept je velmi podobný jako hledání pomocí zkratky Ctrl + f, kde se zadá požadovaný výraz a ten se zvýrazní. Zde je jediná změna, a to, že se nemusí jednat o text, ale i o různé další datové typy, například čísla. Specifikace hledané hodnoty bývá zadána



Obrázek 5.2: Přehled možných manipulací [3]

přesnou hodnotou[2] nebo rozsahem hodnot[3]. Výběr samotný je pak hojně využíván ve filtraci a organizaci dat, která bude probrána později.

Označování. Jedná se o techniku výběru podmnožiny dat, kdy výběr neprobíhá na základě hodnot, ale na základě vizuálního výběru. Každý element může být vybrán pomocí kurzoru myši kliknutím. S tímto výběrem je následně prováděno mnoho úkonů i úloh. Vybraný prvek lze například zvýraznit, ukázat jeho detail nebo daný element přiblížit. Další možností u tohoto výběru je, že se ke zvýrazněnému elementu ukáží další prvky, které s ním mají nějaký vztah. Tento vztah je různého druhu, například stejná hodnota, stejná kategorie nebo nějaký jiný vztah. Tento způsob interakce je v některých případech velmi náročný. Speciálně v případech, kdy je v dané vizualizaci mnoho malých

[2]Číslo nebo text s rozlišováním velkých a malých písmen.
[3]Interval čísel nebo určitý druh regulárního výrazu.

a překrývajících se prvků. Zvýraznění je velmi často záměnný termín pro označování a samotné zvýraznění bývá provedeno různými způsoby, například barvou, velikostí nebo tvarem. Velmi zajímavý experiment provedli Ware a Bobrow [11], kteří experimentálně zjistili, že využití pohybu pří zvýraznění velmi často překonává tradiční přístupy, jako jsou barvy nebo velikosti. [3, 11]

Vylepšené označování. Technika, kdy se samotné označování upraví, aby se s ním lépe pracovalo. Při označování vyvstává otázka, jak pomoci uživateli s výběrem tak, aby vybral to, co v danou chvíli potřebuje, a ne prvek, který je vedle. K tomuto zlepšení je využíváno různých technik vylepšeného výběru. Mezi tyto techniky patří **Voroného označování**. Toto označování využívá Voroného diagramy, kde každý element má svou vlastní oblast, která je jemu nejblíže. Proto při posunutí kurzoru do dané oblasti dojde ke zvýraznění daného elementu, případně kliknutím proběhne výběr. Vzdálenost, ve které dojde k samotnému výběru, lze omezit, aby nedocházelo k nechtěnému výběru. Další technikou je **bublinový kurzor**, který je podobný Voroného označování, ale s vizuální odezvou v podobě kruhu okolo kurzoru. Velikost kurzoru je nastavena tak, aby vždy obsahovala nebo protínala pouze jeden element. Tyto techniky však neposkytují pohled na detail vybraných dat, pouze usnadňují jejich výběr. Detaily jsou pak vidět s pomocí techniky **excentrického označování**, kdy všechna data v určitém okruhu okolo kurzoru mají zobrazené své hodnoty. Což pomáhá šetřit mysl a paměť uživatele a ulehčuje porovnání většího množství prvků. Poslední zde zmíněnou technikou je tzv. **brushing**[4]. Jedná se o proces výběru elementů z vizuální reprezentace a dalo by se přirovnat k malování přes jednotlivé elementy. Tento pomyslný štětec „obarví“ požadované prvky, které jsou předmětem zkoumání v jiných náhledech nebo kontextech. Často se využívá i několik barev a tím se vytvoří podskupiny výběru. Tato technika je také využívána ke zvýraznění položek ve vizualizacích, které mají více než jeden pohled nebo atribut. Díky tomu je pak snadnější sledovat interakce s ostatními položkami a také sledovat vývoj v datech. [9]

5.3 Navigace v datech

Navigace v datech je způsob, jak si uživatel prohlíží data nebo určuje, která data jsou mu ukazována na obrazovce. Množství dat, které je třeba vizualizovat, je stále větší a obecně dochází k limitaci v rozlišení zobrazovacího média. Tedy není možné zobrazit všechna data z určité množiny na jednu

[4]Způsob označovaní dat, která zajímají uživatele. Opět je to pro vizualizace hojně používaný pojem, a proto bude zachován.

Obrázek 5.3: Příklad s použitím techniky přehled a detail. Celý prostor zabírá mapa a v pravém dolním rohu je pak celkový přehled o daném místě. [3]

obrazovku s veškerým detailem. Vyvstává tedy otázka, jak se v dané vizualizaci pohybovat. Navigace je zde myšlena jako změna pohledu, ze kterého se vykreslují aktuální data. Navigaci jako takovou lze rozdělit na tři složky. **Zoom**, který umožní vizualizovaná data přiblížit nebo oddálit. **Panning**[5], který zastupuje pohyb pohledu[6] paralelně k vizualizační ploše. A poslední je **rotace** která, jak se dá očekávat, otáčí danou vizualizaci okolo osy kamery. Ve dvou rozměrech není tak významná při porovnání s prostorovým využitím rotace. Následující techniky jsou pouze ukázky a mohou být kombinovány dohromady.

Přehled a detail. První z technik, které umožňují jednodušší navigaci v datech. Tuto techniku si lze představit jako mini mapu ve strategických počítačových hrách. Zde je pohled shora na mapu veden jako detail a mini mapa v rohu ukazuje celkový pohled na data (mapu), čímž umožňuje snazší orientaci. Pokud se tento konkrétní příklad trochu upraví a přenese na vizualizace, pak hlavní část obrazovky zabírají detaily aktuálního pohledu na data. Kromě tohoto pohledu je v obrazovce přítomna i část, kde je hrubý přehled dat. V něm je možné sledovat, jaká data jsou kde a kudy musí uživatel jít, aby se na určitý detail mohl podívat. Jinak řečeno, obě části jsou zobrazovány paralelně. Zde se objevuje problém prostorového oddělení, což znamená, že uživatel dané vizualizace musí měnit záběr své pozornosti mezi jednotlivými pohledy. Výhodou této techniky je, že nevyžaduje uživatelovu paměť a usnadňuje tak práci.

[5]V následujících sekcích nazváno jako manipulace pohledem. V literatuře je však tento termín běžně užíván, proto se může v některých případech objevit.

[6]V prostoru lze připodobnit k translaci kamery.

Manipulace pohledem a zoom. Tento přístup umožní uživateli interagovat s vytvořenou virtuální plochou. K tomu slouží právě manipulace pohledem a zoom[7]. **Manipulace pohledem** je interakční technika, která pomáhá procházet data v daném záběru. Typicky zde není přítomna žádná hranice, což znamená, že pohyb v datech není omezen v žádném směru. Nejjednodušší pohyb je chycení a potažení pohledu, případně posun pomocí šipek. Mimo to se také řeší omezení nebo neomezení pohybu po virtuálním plátně s daty. Oba přístupy mají své plus i minus a je nutné s nimi pracovat již při návrhu. **Zoom** je pak nejzákladnější interakční technika při vizualizacích – vzhledem k omezenému rozměru média je přiblížení vhodné pro odstranění tohoto problému. Zde vzniká další z problémů, který je nutné vyřešit. Jedná se o časové oddělení, kdy v okamžiku manipulace nebo zoomu je nutné, aby si uživatel pamatoval předmět svého zkoumání. Tedy vzniká zde požadavek na paměť uživatele, která může být velmi zrádná. Tento nedostatek bývá částečně odstraněn pomocí výběru a zobrazení jen určitého počtu detailních částí. [3, 9]

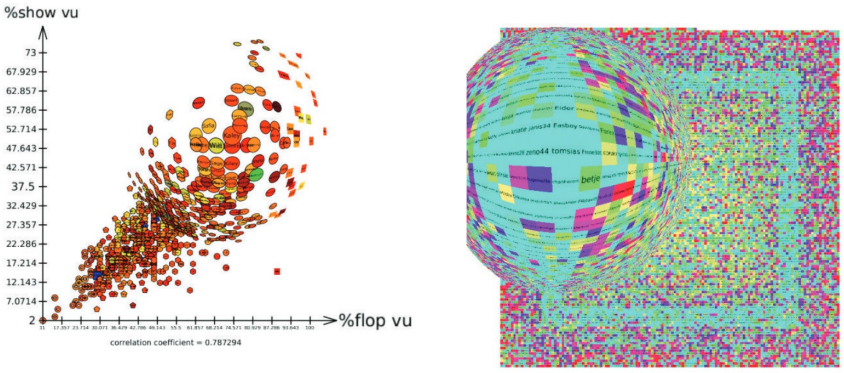
Pro zoom dále platí, že existuje více způsobů, jak ho provést. Existuje **geometrický zoom**, kterým uživatel specifikuje velikost dané vizualizace, tedy jak se má daná vizualizace zvětšit nebo zmenšit. Tento typ zoomu umožňuje zaměření se na specifickou oblast, ale také dobré a rychlé navigování ve velkém množství dat. Vše, co je mimo hranice, je pak zahozeno a zobrazuje se tedy jen existující část. Geometrický zoom je vidět například v prohlížečích obrázků.

Dalším typem je **sémantický zoom**. Sémantický zoom bývá konceptuálně protiklad k redukci pixelů. Pro lepší představu následuje příklad tohoto zoomu. Lze ho najít v mapách, kde dochází k postupnému načítání přesnějších dat a změnám obsahu oblastí. Tento typ zoomu také mění tvar nebo kontext informací, které jsou zobrazovány. Zde je opět krásný příklad s mapami, kde na vyšší úrovni je možné vidět státy, při přiblížení pak jednotlivé kraje nebo menší vnitrostátní celky a při dalším přiblížení je možné vidět další detaily.

Poslední zde zmíněný typ zoomu bude **Fisheye zoom**. Ten se soustředí na bod nebo položku v daném pohledu, ale nepřidává žádnou okolní informaci. Se zvětšující se vzdáleností od místa, které je sledováno, se ostatní položky zobrazují s menším detailem. Tento typ zoomu je také zástupcem následující techniky.

Focus & Context. Technika, kdy se zobrazí malé množství dat s větším detailem a zbytek dat pak s menším množstvím informací, jinak řečeno

[7] Občas bude označeno jako přiblížení, obecně je tímto slovem myšlena změna měřítka.



(a) (b)

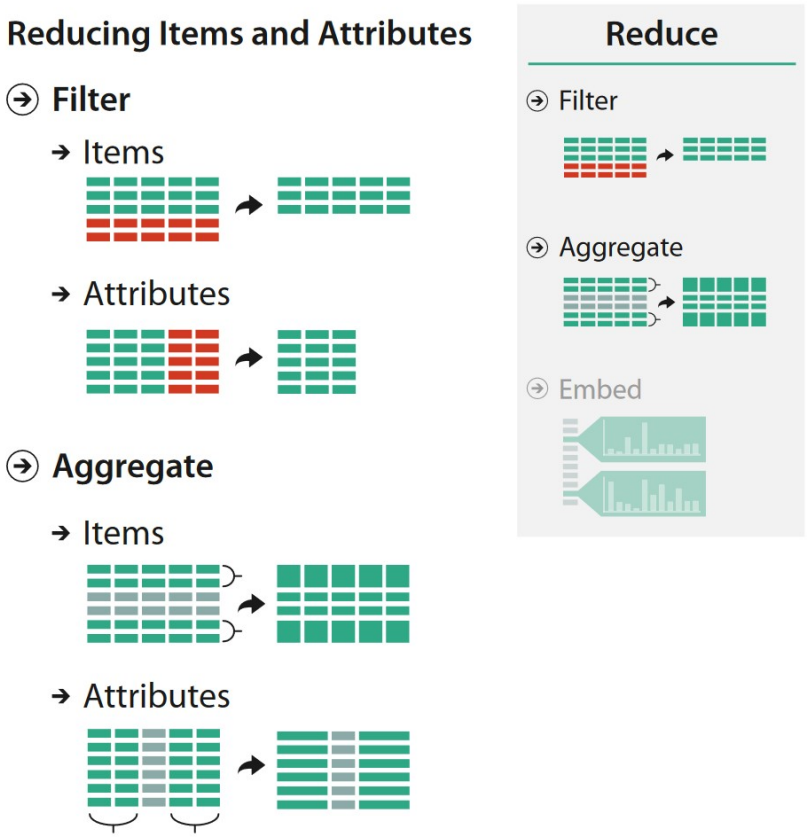
Obrázek 5.4: Ukázka techniky Focus&Context FishEye Zoom [3]

bez detailu. Hlavní oblast sledování se pak nazývá anglicky focus. Zde je pak zobrazena většina nebo všechny informace o daných položkách. Toto vše je zobrazeno v jednom pohledu. Je tedy vidět i určitý přehled okolí (kontext, anglicky context) a zároveň podstatné informace z místa soustředění. Tato technika má mnoho podob a některé s sebou přináší neduhy v podobě deformace. Může dojít k ovlivnění už tak pokřiveného vjemu korelace a dalších statických veličin. Případně celkově může být horší na interpretaci. Příkladem této techniky je již zmíněný Fisheye zoom. Informace zde uvedené jsou velmi stručné, neboť je této problematice věnováno mnohem více prostoru v kapitole 14 [3].

5.4 Redukce dat

Redukce dat probíhá ve dvou smyslech – filtrace a agregace. Oba tyto smysly jsou více použitelné v tabulkových datech, ale mají své uplatnění i u jiných typů dat. K nim jsou pak přidruženy další techniky, jako například přihrádkování, shlukování nebo třeba statistické distribuce.

Filtrace je využívána u prostorových dat, kdy se do měření mohla dostat chyba v podobě vysoké hodnoty, nebo je třeba zredukovat hodnoty na ty, které jsou v mezích požadavků. Také slouží ke zpřehlednění komplexních vizualizací. Pokud se pro příklad vztáhne filtrace na tabulková data, pak se dá chápat jako odstranění atributů nebo dimenzí z vizualizace. Což vede ke snížení dimensionality a jednoduššímu pochopení. Velmi často bude uživatel hledat ve vizualizaci nějaká zajímavá data, tuto funkcionalitu je možné reprezentovat například pomocí dvou políček, která udávají minimální/maximální rozsah vyhledávaných hodnot. V případě průzkumu dat bude uživateli takto

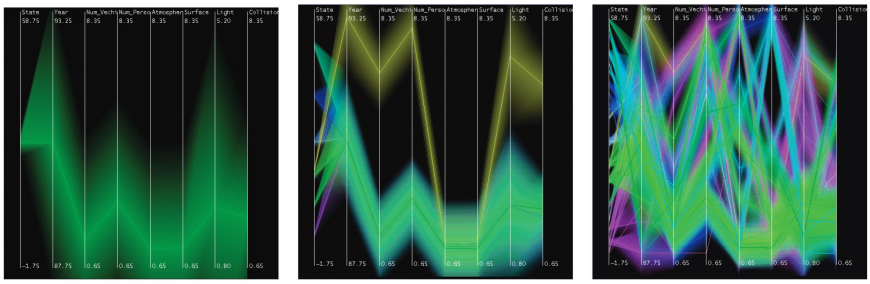


Obrázek 5.5: Ukázka možností filtrace a agregace dat. [3]

prezentovaná funkcionalita dělat problémy, protože neví, jaké hodnoty zadat. To se může zlepšit například pomocí posuvníků, díky kterým lze interaktivně sledovat změny. Mimo klasické statické dotazy bývají také uplatňovány tzv. dynamické dotazy. U těchto dotazů je úzké spojení mezi vizuální reprezentací a interakcí s daty, která jsou následně filtrována a ukázána. Dynamické dotazy pak pracují velmi rychle a uživatel může využít různých doprovodných prvků k další interakci s vizualizací. Pro zadání těchto dynamických dotazů je dobré splnit určité požadavky, ty však zde nebudou dále rozebírány, nicméně je lze dohledat v [12]. [3, 9, 12]

Agregace pak slouží k vytvoření atributů, které reprezentují sloučení několika atributů do jednoho. Díku této operaci lze snížit počet dimenzí dané vizualizace a odstraňuje vizuální odpad[8]. Hlavním cílem je tedy snížit počet atributů na rozumné a dobře reprezentovatelné množství. Pro agregaci samotnou se nejvíce využívá podobnost v datech, která jsou potenciální kandidáti na spojení. Ačkoliv se agregace tváří jako vysněná metoda, opak je

[8]Nepotřebné části, které odvádí pozornost nebo představují jiné problémy ve vnímání vizualizace.



(a) (b) (c)

Obrázek 5.6: Příklad vizualizace, která využívá hierarchické paralelní souřadnice. (a) Obsahuje pouze nejvyšší úroveň dat. (b) obsahuje několik menších shluků dat a (c) zobrazuje mnoho shluků. [3]

pravdou. Je možné s ní lépe reprezentovat data v menším množství, nicméně cena za tuto výhodu bývá ztracení zajímavých částí dat. Typicky se využívají jednoduché operace jako minimum, maximum, průměr a tak podobně. Ty však mohou způsobit právě ono ztracení zajímavých míst. Proto je velmi těžké nalézt adekvátní agregaci, která zachová i zajímavá místa v datech. I přes tento nedostatek je to velmi silný prostředek při vytváření vizualizací a může být ještě lepší, pokud se uživateli předá možnost měnit styl agregace nebo jakým způsobem se bude agregovat. Pokud se bude zkoumat, jak je možné takovou agregaci provést, pak existuje mnoho možností. Zde budou uvedeny jen tři nejčastější. [3, 9]

Přihrádkování[9] je jednoduchá a snadno proveditelná metoda, kdy se data rozdělí do jednotlivých přihrádek, které reprezentují určité intervaly. Pro každou přihrádku jsou udržovány počty prvků v nich a tento počet je pak mapován na další vizualizační prvky. Jednoduchým příkladem je histogram, který obsahuje sloupce udávající počty výskytu dané hodnoty. Tato metoda pomůže v případě, kdy je množství dat příliš velké.

Dále je tu **shlukování**[10], kdy se data shlukují na základě podobnosti v metrikách. Podobnost samotná je vyjádřena různými vizualizačními technikami. Jednou z nich je spočtení průměrné hodnoty pro každou skupinu a mapování průměrné hodnoty na jednotlivé vizuální prvky.

Poslední zde zmíněnou metodou jsou **statistické distribuce**. Statistické hodnoty jsou reprezentovány různými styly, například box ploty nebo konfidenčními intervaly. V datech se spočítají statistické informace pro každou položku a tyto spočtené informace se následně zobrazí.

[9] Binning
[10] Clustering

5.5 Organizace pohledů na data

Poslední zde probranou částí budou pohledy na data. Pojmy, se kterými se při navrhování vizualizace pracuje, jsou například juxtaposition, superimposition, koordinované pohledy, multiformní pohledy, sdílené kódování a velmi často se využije brushing. V několika krátkých odstavcích budou probrány jednotlivé typy pohledů. [3, 9]

Juxtaposition je typ zobrazení, kdy se využívá několika pohledů na data najednou a všechny jsou vedle sebe. Je vyžadováno naznačení propojení ať už koordinací nebo propojeným pohledem[11]. Dále platí, že s narůstajícím počtem pohledů jsou jednotlivé pohledy menší, a tím pádem pro každý pohled je méně a méně pixelů.

Superimposition funguje velmi podobně jako vrstvy v grafických editorech, kde v jednotlivých vrstvách jsou různé části obrázku a při pohledu shora skládají celkový pohled. V tomto zobrazení je to stejné, jednotlivé vrstvy obsahují různá data ze zkoumané oblasti. Obsahují různé vizualizace[12] a v místech, kde nejsou žádná data, jsou průhledná místa. Toto zobrazení pak umožňuje lepší srovnání mezi daty (seřazení) a také je velmi užitečná pro zvýraznění zajímavých míst.

Usazení[13] umožní spojení dvou pohledů na data v jeden. Typicky není možné vidět přechod mezi pohledy příliš jasně a jako příklad poslouží Focus & Context technika navigace v datech, přesněji Fisheye. U tohoto zobrazení je vidět celkový pohled na data se zobrazením malého množství dat uprostřed s dodatečným detailem. Nicméně přechod mezi detailem a přehledem není jasně vyhrazen.

Propojený nebo koordinovaný pohled umožňuje sledovat interaktivní změny z jednoho pohledu ve zbývajících pohledech. Propojení několika pohledů skrze interaktivitu poskytuje více informací než při vyhodnocování jednotlivých pohledů nezávisle na sobě.

Multiformní a násobný pohled zobrazuje podmnožiny nebo všechny atributy/data ze zkoumané oblasti v několika pohledech. Tyto pohledy zobra-

[11] Všechny pohledy budou například označovat stejná data, která byla vybrána v jednom z pohledů.

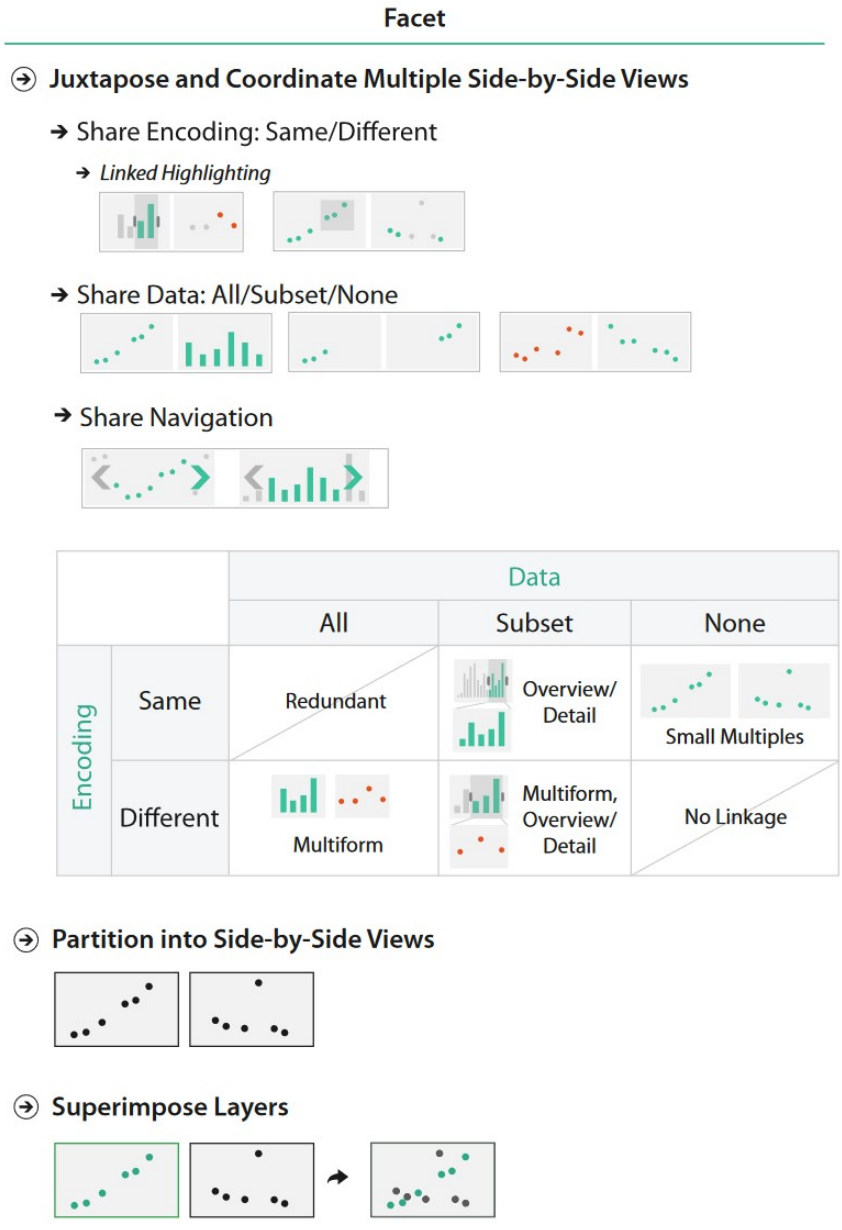
[12] Například pro snímek města je možné mít vrstvy znázorňující hluk, znečištění, provoz, atd.

[13] Embedding

zují vždy stejnou množinu dat, nicméně v každém pohledu bývají zakódovány jiným způsobem. Jinak řečeno je možné mít několik vizualizací jedné množiny dat, kde každá využívá jiné vizualizační techniky nad stejnou množinou dat. Tímto způsobem je možné v některých případech odstranit nebo alespoň zmírnit problémy některých technik.

Sdílené kódování pohledu říká, že atributy jsou zakódovány stejně ve všech pohledech na data. Nicméně každý pohled obsahuje jiné atributy/data. Je nutné rozdělit data mezi jednotlivé pohledy, což se označuje jako *Faceting* nebo *Trellis* [3]. Díky tomuto pohledu je možné snížit složitost zobrazení dat, a to tím, že v jednotlivých pohledech jsou vidět různé atributy/data se stejnou vizualizační technikou. Umožní tedy odlehčení vizualizace.

Brushing a propojování jsou dvě části zobrazení, kdy je možné snáze vidět specifikovaná data. Pomocí brushingu je vybrána podmnožina datových položek, které jsou prezentovány. Typicky pak dojde ke zvýraznění vybraných dat. Následně jsou tato vybraná data zobrazena v doprovodných oknech vizualizace (pokud jsou) pomocí propojení. Což má za následek, že vybrané datové položky jsou vidět i v ostatních pohledech na data.



Obrázek 5.7: Ukázka možný zobrazení dat [3]

Kapitola 6

Metody vizualizace časových dat

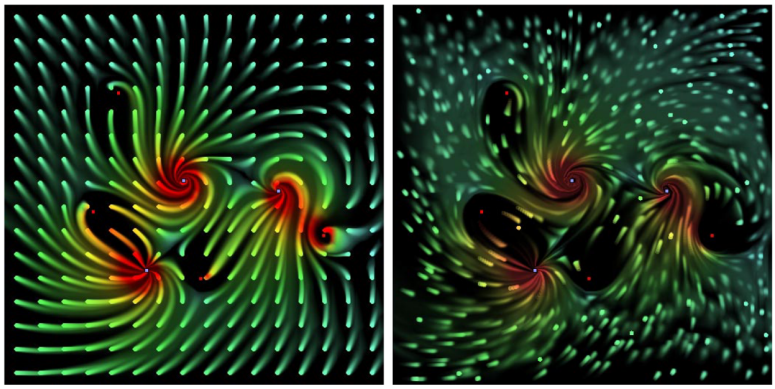
Tato kapitola obsahuje metody, které jsou spjaty s časově orientovanými daty. Hledány jsou zejména metody, které zvládnou vynášet data na časovou linii a zároveň pracovat s více liniemi najednou[1]. Sekce 2.5 obsahuje přiblížení časově orientovaných dat a na tomto základě bude dále stavěno. Použitelnost jednotlivých metod je následně diskutována v sekci 7.5.

6.1 Obrazově založená vizualizace toků

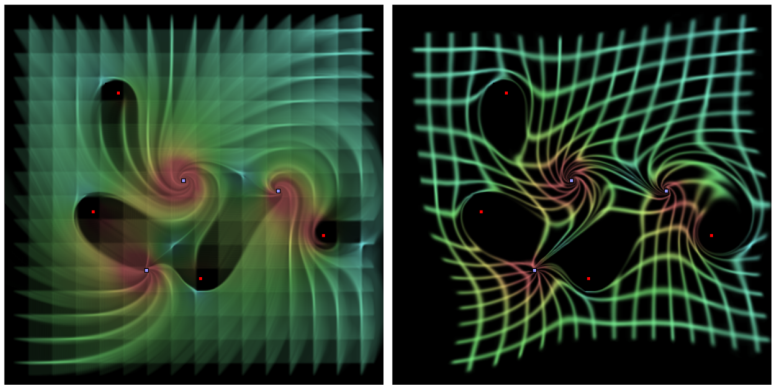
Tato metoda je vhodná pro vizualizace dvou dimenzionálních toků. Jedná se tedy o metodu, která pracuje s prostorovými daty. V každém vrcholu mřížky[2] se nachází hodnota měření a ty jsou pomocí obohacení rozšířena a připravena ke zpracování. Na základě této obohacené reprezentace se následně pomocí doprovodných algoritmů vypočítá pohyb v mřížce. Tato reprezentace je dále využita jako podklad pro vykreslení, případně znázornění pohybu. Tato metoda je také schopná vytvářet animace těchto toků. Animace se vytváří přímo pomocí střídání jednotlivých reprezentací a je vyvolán dojem změny v pohybu. Případně se do reprezentací pouští částice, které využívají aktuální reprezentací a pohybují se dle spočtených hodnot v reprezentaci. Další zajímavou reprezentací je využití deformace prostoru, kde je reprezentace postupně brána a následně komponována k vytvoření výsledku. Přesnější popis lze nalézt v [13]. Obrázky 6.1, 6.2 ukazují možné výsledky.

[1] To vyplývá z požadavků zadavatele.

[2] Může být 2D, nebo 3D.



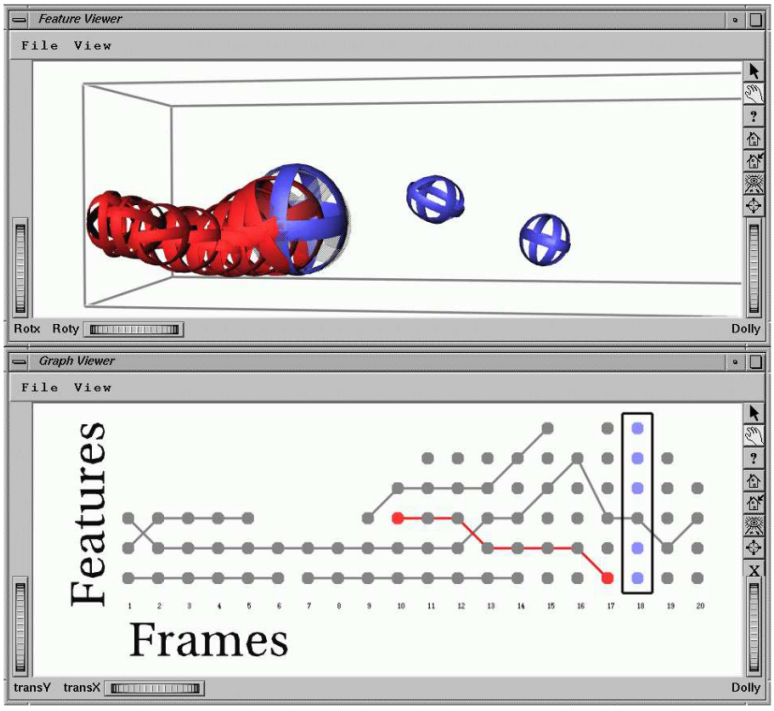
Obrázek 6.1: Ukázka možného výstupu obrazově založené metody [13]



Obrázek 6.2: Jiná reprezentace výsledků obrazově založené metody [13]

6.2 Vizualizace založená na událostech a sledování specifických vlastností

Tato metoda má velmi široký rozsah, jak je vidět v [14]. Už z popisu a prvních částí tohoto zdroje je vidět, že se jedná o komplexní metodu, která využívá několik úrovní. První popisovaný krok je nalezení zajímavých vlastností dat, kde se v každém kroku získávají popisy a atributy. Dalším krokem je spojení částí, které spolu souvisí, aby bylo možné sledovat jednotlivé kroky. Kromě toho mohou nastat specifické události jako nečekané změny, určité stavy vývoje v datech nebo interakce s daty. Následně se vytvoří vizualizace, kde je vidět interakce v jednotlivých částech. Jednotlivé kroky zde pak znamenají snímky animace, kde je možné sledovat vývoj v datech celkově nebo jen částí, které jsou určitým způsobem zajímavé pro uživatele. Díky tomu, že se jedná o animaci, je možné sledovat celý vývoj od začátku do konce, sledovat životnost různých jevů a tak dále. Pro lepší představu je z [14, 7] převzata ukázka 6.3.



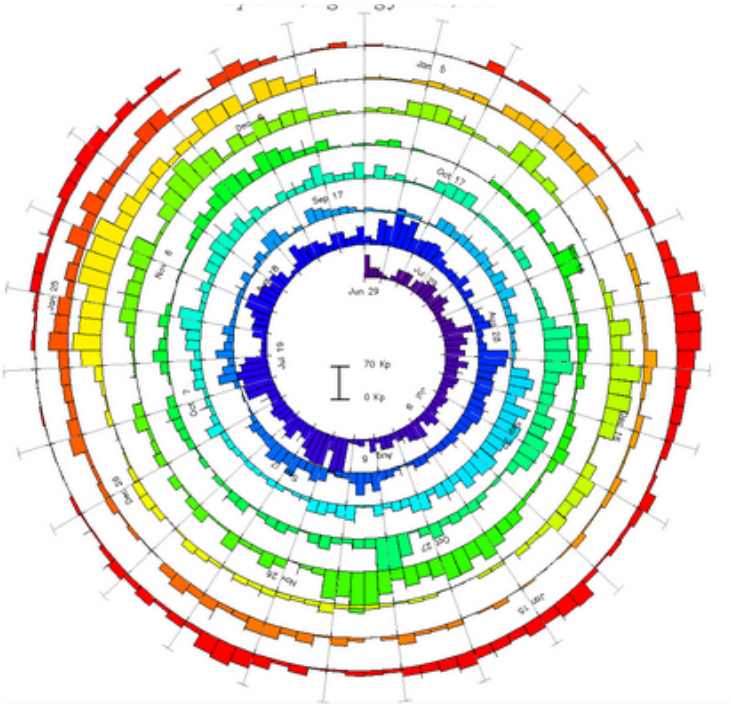
Obrázek 6.3: Ukázka sledování vlastností [14]

6.3 Spirálový graf

Jedná se o graf, kdy časová data jsou mapována na spirálu od středu ke vnějšku. Postupným odmotáváním se také „odmotává“ čas. Jedno otočení zde znamená jeden cyklus, což je vhodné pro odhalování různých sezónně závislých jevů. Velikost jednoho cyklu je velmi často možné měnit a díky tomu je snadnější vidět zmíněné sezónní efekty, případně je zvýrazňují. Na spirále je typicky přítomen jeden atribut, ale také je možné využít více atributů současně. U této metody je také důležité stanovit, co je jeden cyklus a co všechno je vidět, jinak dojde ke zmatení uživatele. U této metody jsou hlavně využívány abstraktní datové typů s číselnými hodnotami. [15]

6.4 ThemeRiver

Je metodou se statickou reprezentací změn v tématech dokumentů. Z toho je vidět, že metoda pracuje s abstraktními datovými typy. Tato metoda pomáhá identifikovat časově vztažené vzory, trendy a vztahy napříč velkým množstvím dokumentů. Samotná vizualizace se podobá řece, která má v některých místech

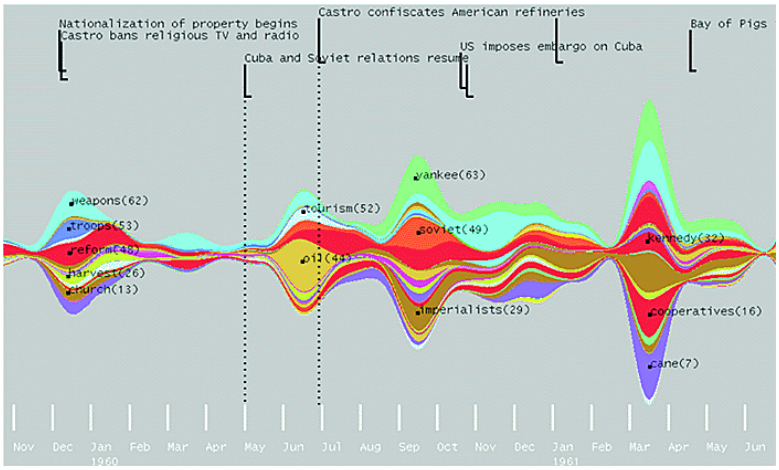


Obrázek 6.4: Ukázka možné reprezentace spirálového grafu [15]

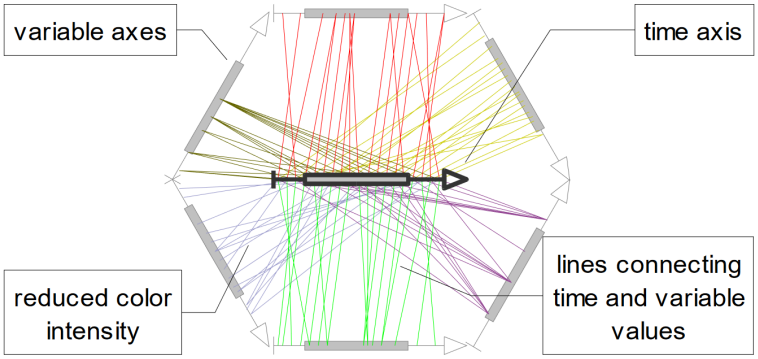
větší koryto a v některých zase menší. To odpovídá síle daných témat v různých časových okamžicích. Čím širší je „řeka“ v daném místě, tím silnější dané téma bylo. Vzhledem k přítomnosti časové osy je vidět i časový kontext, který může vysvětlovat oblíbenost určitého tématu. Dokumenty, které jsou prezentovány, nemusí být pouze knihy nebo odborné články, ale i noviny nebo jiné typy zpravodajství. [16, 15]

6.5 TimeWheel

Jedná se o vizualizaci, která je založena na osách, kde jsou vynesena data. Hlavní zaměření této metody je na časové závislosti. Tato vizualizace má dvě části, časovou osu, která udává sledovaný časový rozptyl, a jednotlivé osy sledovaných atributů. Hodnoty přítomné na ose atributu jsou napojeny na časovou osu a tím značí, kdy nastaly. Na každé z os je navíc přítomno okno, které udává, kde se aktuálně pohybuje čas. Při pohybu v čase je možné na jednotlivých osách sledovat naměřené hodnoty. Přesnější popis lze získat z [17]. Obrázek 6.6 ukazuje možný výsledek této metody.



Obrázek 6.5: Ukázka vizualizace ThemeRiver [16]

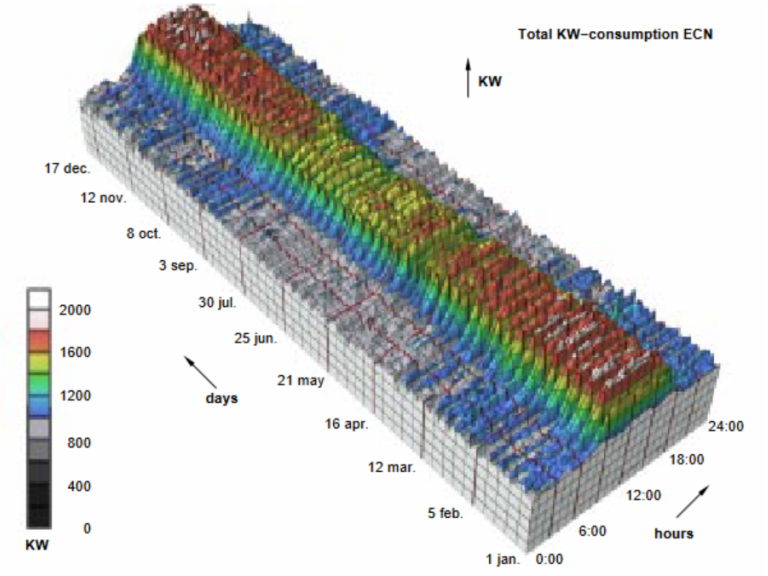


Obrázek 6.6: Ukázka vizualizace TimeWheel. [17]

6.6 Shlukování a kalendářová vizualizace

Časově vztažená data jsou všudypřítomná a jsou zaznamenána v různých stylech. Cílem analýzy časové posloupnosti je získat vhledy do různých fenoménů, objevit opakující se jevy a trendy, předpovídání budoucího vývoje a tak podobně. Zde je lepší začít s příkladem, než s popisem metody. Pro příklad bylo provedeno měření hodnot spotřeby energie, znečištění ovzduší nebo podobné veličiny. Tato data jsou naměřena v krátkých intervalech v průběhu jednoho roku a z těchto dat je nutné získat nějaký přehled. Ten pak může vypadat jako několik kalendářů, kde každý zobrazuje jednu z naměřených hodnot. Ty jsou pro lepší přehlednost rozlišeny barevnými odstíny. Podrobnější popis a více informací lze nalézt v [18].

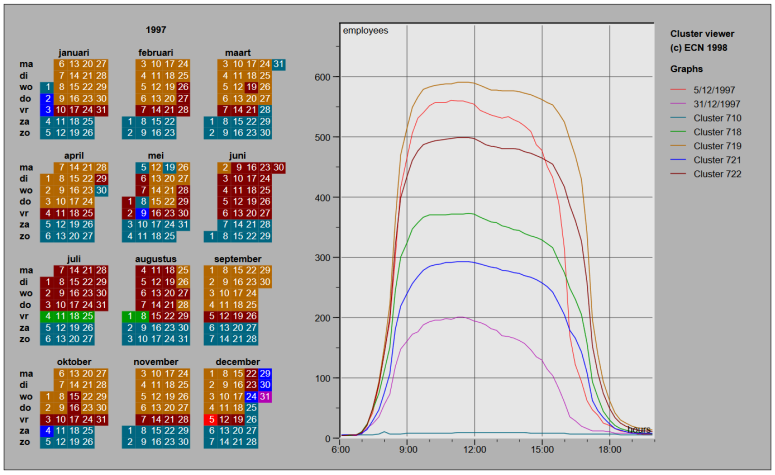
V následujícím popisu metody bude využito dat ze zdrojového článku, kde jsou zaznamenány příchody do práce. Základním krokem je vytvoření shluků



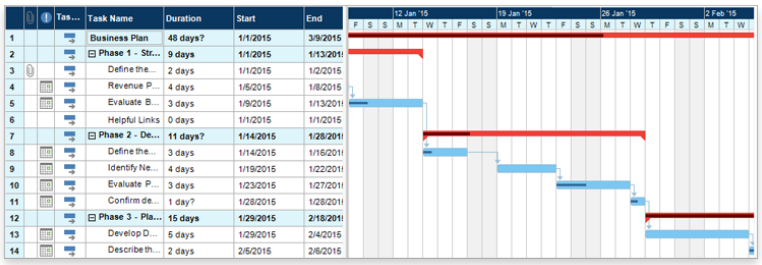
Obrázek 6.7: Metoda využívající uspořádání do kalendáře [18]

naměřených dat, článek uvádí jako možnost výpočet rozdílů. Ty pak udávají podobnost dat – minimální rozdíly znamenají podobnost, maximální naprosto odlišná data. Nejvíce podobná data se spojují do větších celků a opět se provede hledání nejvíce podobných dat a jejich slučování. Tímto postupem vznikne binární strom, ve kterém se snadno hledají jednotlivá data.

Pokud by samotná vizualizace proběhla pouze pomocí stromů, pak by byl vznikl problém s instancemi, které mají velké množství dat. Proto je nutné tato data zobrazit jiným způsobem. Tato reprezentace má navíc i další problémy, jako je například špatné procházení různé granularity v časových jednotkách. To lze velmi dobře omezit pomocí jednoduchého nástroje, jakým je kalendář. U něj se využívá zaměření na jednotlivé dny, týdny, měsíce a tak dále. Zdroj 6.8 obsahuje příklad s možným výsledkem vizualizace. V levé části jsou vidět časové jednotky podbarvené podle toho, k jakému shluku patří. V pravé části jsou pak průměrné hodnoty pro jednotlivé shluky. V dané ukázce je vidět jen sedm nejvýznamnějších shluků. Jinak řečeno vizualizace bere výhodu rozdělení časových údajů v kalendáři a slučuje je pomocí dostatečně silných vizuálních kanálů. Samotné průměrné hodnoty jsou viditelné v pravé části, ze které se získá vhled do dat. [18]



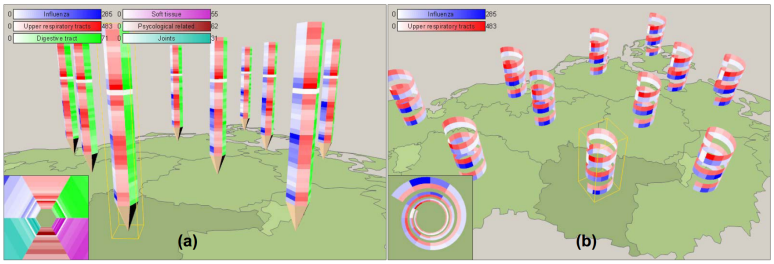
Obrázek 6.8: Metoda využívající uspořádání do kalendáře [18]



Obrázek 6.9: Znázornění Ganttova diagramu [19]

6.7 Plánování Gantt chart

Tato metoda je velmi často využívána v plánování projektů, nebo je velmi užitečná pro sledování různých aktivit vůči času. Provedení bývá různé, například zobrazení aktivních úloh a vedle jejich naplánovaná doba trvání. Zde jsou vidět zejména ukotvené intervaly, které jsou oporou této metody. V grafu je možné vidět tok času, aktuální den, postupy v jednotlivých úlohách, různé fáze, časové úseky, které signalizují různé věci, a další kaskáda zákonitostí. Pracuje se hlavně s časovým aspektem dat, ke kterému je přidána nějaká událost, akce nebo kontrola, které se provedou. Velmi stručný přehled je uveden v [19], robustnější zdroj je pak [20]. Příklad je uveden na obrázku 6.9.



Obrázek 6.10: Ukázka obou možností této vizualizace, (a) obsahuje Lexis pencil a (b) pak Helix glyph [21]

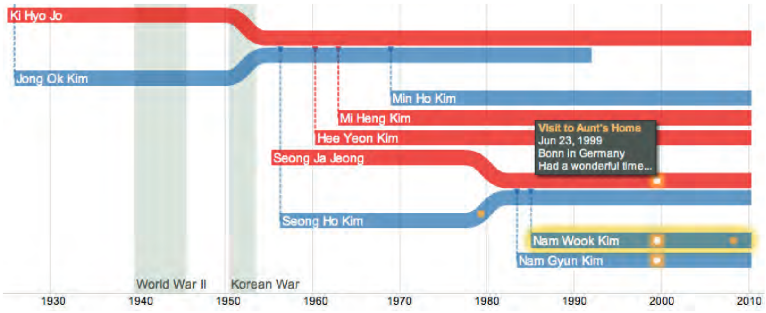
6.8 Helix / Lexis pencil glyfy na mapě

Tyto dvě metody vizualizace využívají jako jedny z mála prostorové[3] vizualizace. Pro **Lexis pencil** je využito elementu, který reprezentuje veškerá data, a zároveň je jasná i kontextová návaznost na mapu. Jinak řečeno, samotné reprezentace dat ukazují do prostoru, například do mapy, a udávají, jaká data jsou ve specifickém místě[4]. Zkoumaným tématem jsou metody pro časově orientovaná data, ale o časovém aspektu nebylo zatím nic řečeno. Postup času je vidět podél elementu této vizualizace a tím je tužka. Ta je hrotem namířená na místo, kde proběhlo měření. Její kolmé stěny jsou pak jednotlivé atributy, které se odvíjí od hrotu ke konci, což symbolizuje čas. Pro přesnější představu je tato vizualizace vidět na obrázku 6.10. Varianta s **Helix** elementem umožňuje stejné lokalizace jako Lexis pencil. Rozdíl je hlavně v elementu, na kterém jsou vynesená data. Ten je velmi podobný DNA šroubovici. Dle počtu atributů obsahuje jednotlivé „pásky“, čas pak plyne od spodní části šroubovice k vrcholu. Obě tyto metody v sobě mají zaneseny různé vizualizační strategie, u Lexis pencil je to Focus+Context, což je vidět na tužce, kde kromě zkoumané stěny je vidět na vývoj dat na sousedních stranách. U Helix reprezentace je pak vidět opakující se vzor v datech. Zde je vidět podobnost se spirálovým grafem. Přesnější popis lze nalézt v [21].

6.9 TimeNet genealogický pohled

Článek [22] obsahuje upravenou metodu vizualizace příbuzenských vztahů, která bude dále označována jako genealogická. V článku je prezentována vylepšená vizualizace, která napomáhá s vnímáním sociálních vztahů. Největším

[3] Prostorové vizualizace je vhodné využít jen s velmi dobrým odůvodněním.
[4] V závislosti na granularitě to může být stát, kraj, město nebo podobné celky.



Obrázek 6.11: Příklad s vizualizací genealogických vztahů [22]

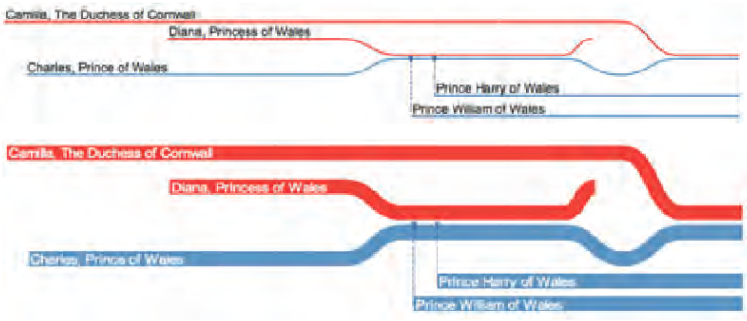
problémem genealogů je správné rozeznání rodinných a časových vztahů, k čemuž se typicky využívá rodinných stromů. Tento strom zajišťuje snadný popis příbuzenských vztahů, ale už ne přesnějších časových ukotvení nebo uvádění v kontext s dobou. Vzhledem ke složitosti skládání těchto vizualizací vznikají chyby, které vedou až ke špatnému složení rodinného stromu. Přesnější popis problémů je uveden ve zmíněném článku. Systém navrhovaný v tomto článku je schopen ošetřit mnoho problémů, které se obvykle neřeší, případně přidat některé komplexnější vztahy.

Hlavním cílem designu dané metody je umožnit současné sledování mnoha různých aspektů. Ať už příbuzenské, časové nebo jiné atributy či vztahy. Zdroj také zmiňuje, že díky uzpůsobení je možné zobrazit moderní jevy, jako jsou rozvody, případně zvyky některých jiných zemí, např. polygamie.

Základem celého článku jsou klasické lineární osy, které postupují zleva doprava. Každá linie představuje individuální časovou osu jedné osoby od narození po smrt. Kromě tohoto základního nástinu je v článku uvedeno rozdělení linií dle pohlaví[5]. Navíc u každé linie může být přítomný popisek, případně se dá upravit tloušťka a jiné parametry dané linie pro snadnější sledování.

Dalším krokem je zavedení prvních vztahů mezi liniemi, jako je rozvod nebo svatba. Tato vlastnost lidského života je zakódována ve vertikálním přiblížení dvou linií, které pak znamenají svatbu nebo určitý partnerský vztah. Tedy při přiblížení dvou os vznikne událost (například svatba) a od této události jdou obě linie spolu. V určitém bodě však může dojít k další události (protiklad svatby je rozvod) a v takovém případě se osy od sebe vzdálí. Tento přístup přirozeně značí významné body v životě a jednoduše ukazuje změny. Pro tyto dva prvky se nejlépe hodí křivky. Článek pak přesně uvádí, který typ křivky byl použit k vytvoření konvergujících nebo divergujících linií, jedná se o Bézierovy křivky.

[5]Pro úplnou korektnost by v současné době bylo potřeba více barev než modrá a červená.



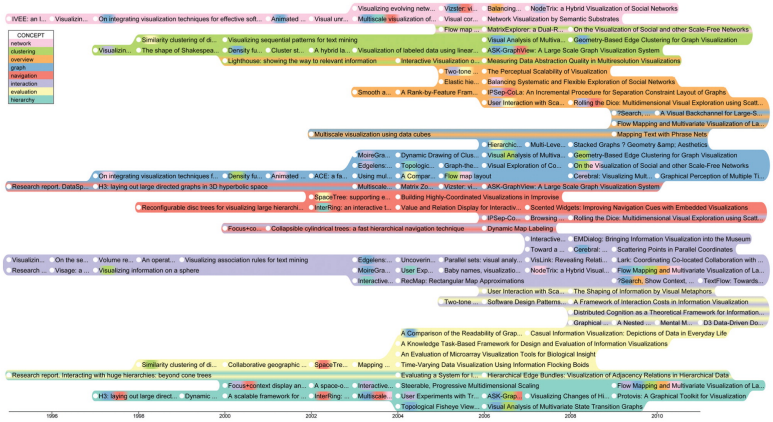
Obrázek 6.12: Další příklad s vizualizací genealogických vztahů [22]

Důležitým tématem je vyřešení pokrevních vztahů typu rodič-dítě, které byly dle [22] řešeny několika způsoby, kde se jednodušší strategie ukázala jako lepší. Původní přístup byl, že život potomka začal na linii manželství a pokračovalo níže na svou vlastní osu. To však vedlo ke značnému vizuálnímu zastínění a nepořádku. Navíc mohlo dojít k překrytí rozvodu a tím menší přehlednosti celé vizualizace. Jednodušším přístupem bylo vytvoření samostatné linie rovnou a spojení přerušovanou čarou potomka a rodičů. Nevýhodou tohoto přístupu je však větší kognitivní zátěž uživatele. Ale také má své výhody a to, že při více dětech je možné je seřadit v takovém pořadí, že se minimalizuje překřížení. Dále jsou ve zdrojovém článku diskutovány jednotlivé možnosti řazení, s ohledem právě na křížení.

Další aspekty budou pouze shrnuty, mezi ně patří nejistota, vzory a atributy. Pro údaje, u kterých není jistota, zda jsou správně, volí článek několik přístupů, v první řadě na nejisté časové údaje upozorní. Následně jsou upraveny časové linie pomocí gradientu. Tím, že jsou daná místa zvýrazněna, si může uživatel lépe všimnout, co chybí, a údaje doplnit. Ostatní vzory a atributy mohou být reprezentovány různě, v článku je pak zmíněna hlavně barva samotné linie, která se může měnit vzhledem ke zvoleným atributům. Takto upravená linie může obsahovat informaci například o nemocích, cestování a podobné aspekty života. Zdrojový článek také zmiňuje techniku focus&context, která umožňuje lépe skrývat, nebo naopak zobrazovat určité prvky, které jsou zajímavými objekty pro uživatele. [22]

6.10 TimeSet

Článek [23] se zaměřuje na další vizualizaci časových dat, která využívá shlukování podle souvislostí a je označena jako TimeSets. Už v začátku článek naznačuje, jaký typ dat se s tímto přístupem dá řešit, a udává první

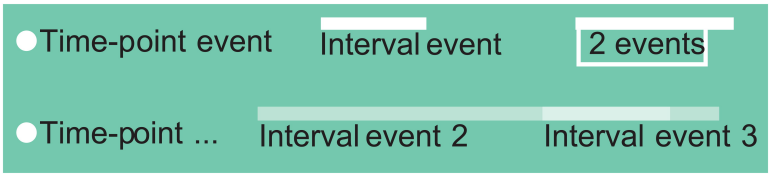


Obrázek 6.13: Ukázka celých TimeSets [23]

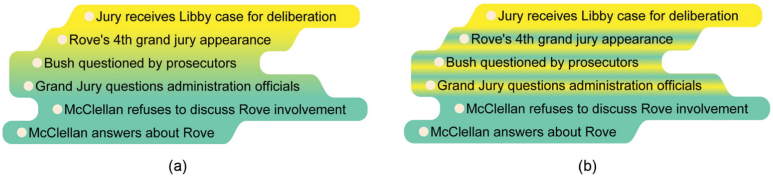
odkazy do historie na tuto metodu. V tomto pohledu na data se opět řeší intervaly i časové body, které značí významné události. Pro vytvoření vztahů lze využít například barev nebo různých tvarů, které budou naznačovat vztahy. To ovšem s sebou nese další problémy, jak naznačuje [23]. Dále je v článku zmíněno několik metod, které pomáhají s řešením komplexních vztahů v časových datech, jako jsou mezilidské vztahy nebo společné oblasti. Pro množinové vztahy v datech bylo článku nalezeno několik dalších prací, které prezentují metody pro spojení elementů ve stejné množině. Mezi tyto metody patří **Bubble Sets**, **LineSets** nebo **KelpFusion**. Předchozí příklady jsou pouze ilustrativní. Hlavním cílem zdrojového článku je vizualizace, která bude schopna vytvořit vztahy v množině informací na časové ose. Zmíněné podporované body jsou přehled o distribuci dat v množině, identifikace trendů a srovnání množin v průběhu času. Prezentovaná metoda následuje dva principy, a to blízkost a spojitelnost dat. Související data jsou blízko sebe, to je navíc podpořeno podbarvením.

Tento článek pracuje s událostmi, které jsou vizualizovány jako řádka textu, popisující danou událost. Samotný indikátor je buď kolečko nebo protáhlý obdélník. Texty bývají různého druhu – plné, useknuté s výpustkou, nebo spojené se zapsaným počtem událostí. Obrázek 6.14 pak ukazuje, s jakými typy událostí celá vizualizace pracuje, kde je vidět poslední nezmíněný prvek, a to překrývající se interval.

Hlavními využitými principy jsou propojení a blízkost v datech. Blízkost je v článku vyřešena pomocí umístění stejných událostí blízko sebe a jejich podbarvení pro zapojení dalšího vjemu uživatele. Vzhledem k využití času na horizontální ose nejde provést prostorové přiblížení v tomto rozměru. Proto tento článek využívá vertikálního poskládání do vrstev, které pak spojují související události. Co se dále týká souvisejících událostí, článek nabízí dvě možnosti vyřešení přítomnosti událostí ve více množinách najednou. Buď je



Obrázek 6.14: Ukázka různých prvků dle [23]



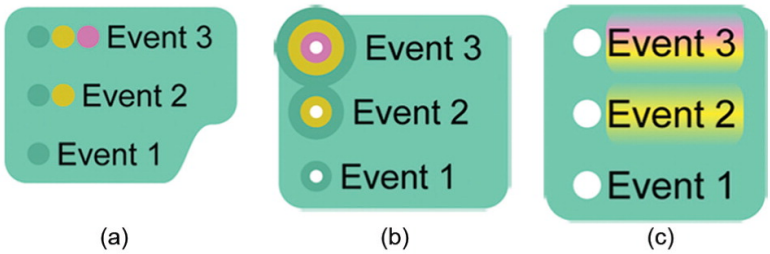
Obrázek 6.15: Ukázka zvažovaných barevných vyznačení [23]

možné je propojit (což může vést k většímu vizuálnímu nepořádku, ale trochu jasnější návaznosti), nebo je duplikovat, což byla zvolená cesta v článku. Zprvu může působit trochu zmateně, ale ulehčuje vizuálnímu kanálu.

Specifický tvar vizualizace je vytvořen pomocí obálek. Tyto obálky jsou buď přesně okolo jednotlivých událostí nebo jako přibližný tvar. Ve zdrojovém článku využívají druhého přístupu s možností úpravy některých parametrů uživatelem. Tento přístup byl zvolen zejména z výkonnostních důvodů. S hrubějším obalem se pracuje snáze. Zjednodušení se týká hlavně horní a dolní části, které jsou zarovnány do roviny a pouze ze stran se nechají „zubaté“ konce, které je možné upravit, jak již bylo řečeno. Tato obálka je pak konvertována do křivek a zaoblena, to je odůvodněno snadnějším sledováním hladkých křivek.

Co se týká barevného zpracování, jsou v článku navrhovány dvě cesty. Označení podbarvením buď intervalu / časového bodu, nebo celého intervalu. Článek využívá druhé možnosti, která se snáze sleduje, a zároveň se snaží nepřidávat žádné další prvky. Tím pádem zbývá vyřešit jediný problém, a to, jak správně obarvit události tak, aby dávaly smysl. Nabízí se dvě možnosti, jak je vidět na obrázku 6.15. První využívá jednoduchého gradientu od jedné barvy ke druhé. U této volby vzniká problém rozeznání kde začíná jedna událost a končí druhá. To je vidět na obrázku 6.15 (a). Obrázek 6.15 (b) pak vyobrazuje výsledný zvolený přístup, který jasně rozlišuje jednotlivé události i společné.

Pro události, které se vyskytují v několika množinách událostí, je velmi těžké vytvořit uspokojivou vizualizaci se zachováním přehlednosti. Což je dle zdroje problém i u mnoha moderních metod. Jak již bylo zmíněno v této části, pro více množin událostí bude využita duplikace tam, kde to dává smysl.



Obrázek 6.16: Ukázka zvažovaných možností znázornění vztahů [23]

Obrázek 6.16 nabízí přehled tří možností, jak naznačit spojitost v množinách událostí. Dle článku se nejlépe jeví právě poslední způsob, tedy podbarvení, které je navíc i vhodné pro intervalové události. [23]

6.11 Focus-Context pro více časových os

Článek [24] popisuje specifickou techniku focus&context, kde se dávají do souvislostí různé časové osy a různé typy událostí. Zde nejsou události chápány jako jeden specifický bod v čase, ale i jako časově delší úsek, například výstavba Brooklynského mostu. V této vizualizaci se na jednu časovou osu vynáší různé typy událostí, jak je vidět na obrázku 6.17. Zároveň je možné je dát do kontextu různých časových událostí. Zde je tedy využito výběru určitého zkoumaného tématu, například čínská imigrace na Brooklyn, s ohledem na historický kontext výstavby Brooklynského mostu. Dochází k propojení kontextu a zkoumané události. Také je zajímavé, jakým způsobem se přistupuje ke granularitě jednotlivých událostí. Hlavním nástrojem je zde jediná časová osa, na kterou se pak vynáší „menší časové osy“, které reprezentují události. Specifičnost techniky focus&context je, že se nezobrazují detaily jednotlivých událostí. Spíše jsou vytaženy tak, aby bylo jasné, co se zkoumá, a k nim je přidán na opačné straně kontext dané doby.

6.12 Vizualizace komplexních sémantických časových os

Zdroj [25] poskytuje několik nástrojů pro vytváření a manipulaci s časovými osami. Z článku vyplývá, že tento nástroj poskytuje několik technik pro zobrazení časových os. Jako první je prezentováno časově nezávislé skládání. Kde se nad sebe neskládají jednotlivé časové osy. Ty na sobě nejsou závislé

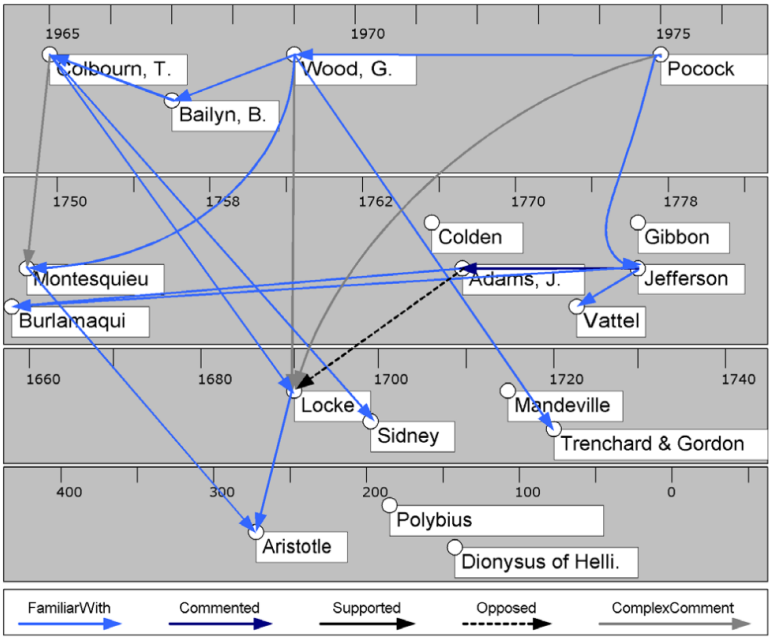
Obrázek 6.17: Ukázka techniky Focus&Context[24]

čímž vytváří vrstvy. Každá vrstva může mít vlastní granularitu a co je podstatný bod této vizualizace, je možnost propojení jednotlivých událostí, které jsou umístěny ve vrstvách. Tím pak vzniká kontext z různých časových úseků, zmíněný kontext je zejména využitelný pro různé publikace a odkazy na tyto publikace. Vizualizace tímto způsobem připomíná nemocniční monitory na které jsou napojené různé životní hodnoty[6], což se také považuje za jednu z vizualizačních technik. Kromě hierarchického složení os je také možné dodat i další sémantické informace o zobrazovaných datech. Jak je vidět na obrázcích 6.18 a 6.19, kde šipky představují další informaci. Tyto vztahy jsou různého typu například: sourozeneckého, podporovatel, nepřítel a další. Zajímavou vlastností uvedenou v článku je možnost definovat jednotlivé události jako další menší časovou osu, která je skrývatelná. Případně tyto menší části vynést na samostatnou osu pro lepší sledování.

6.13 Aktuální verze

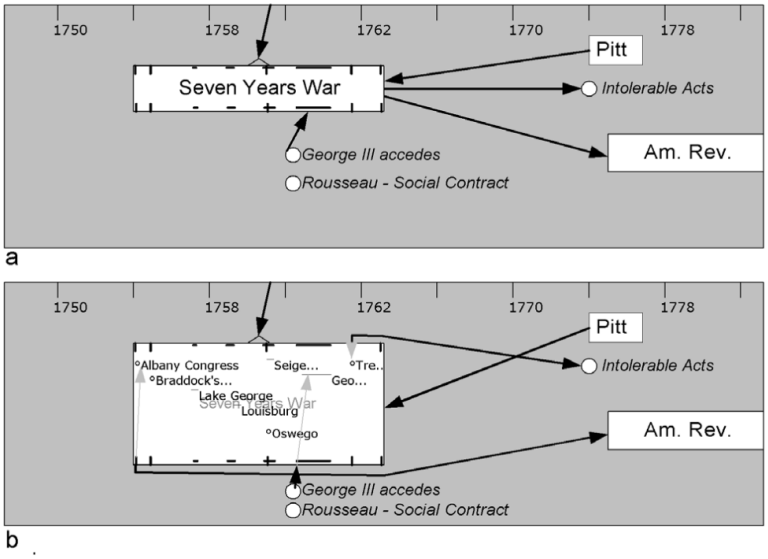
Původní návrh [2] je založen na několika zkoumaných aplikacích a jejich stylech vizualizace. Všechny aplikace využívají jednu časovou osu a události s ní spjaté jsou viditelné. V této verzi se zobrazují pouze události, které se týkají určitého tématu, což je například život jedné postavy nebo vývoj územního celku. Tato práce poukazuje na nevhodnost myšlenky se zobrazením všech částí na jednu časovou osu, případně všech os najednou. To by vedlo k velkému vizuálnímu nepořádku a z dané vizualizace by se nedalo téměř nic vyčíst. Dále uvádí řešení a to vytvoření kategorií a přidání příslušných klíčových slov.

[6] LifeLines



Obrázek 6.18: Ukázka vizualizace komplexních sémantických dat [25]

Takto vytvořený systém pak umožňuje filtraci a zobrazení částí, které jsou spojené určitým významem. Mimo to tento zdroj také využívá více časových os najednou, což vede k vytvoření lepšího přehledu souběžných událostí, případně lepší zasazení do kontextu doby. Pokud je předmětem zkoumání časová osa určité postavy a její působení v různých místech. V případě že jsou některé události společné dvěma zkoumaným osám, pak jsou v tomto zdroji spojeny jednoduchou přímkou. Což při zobrazení velkého množství časových os najednou vede k problémům. Obrázek 7.6 pak ukazuje, jak taková vizualizace vypadá. Jednotlivé události jsou označeny na základě kategorie do které patří pomocí ikonky. [2]



Obrázek 6.19: Větší detail na komplexních sémantická data [25]

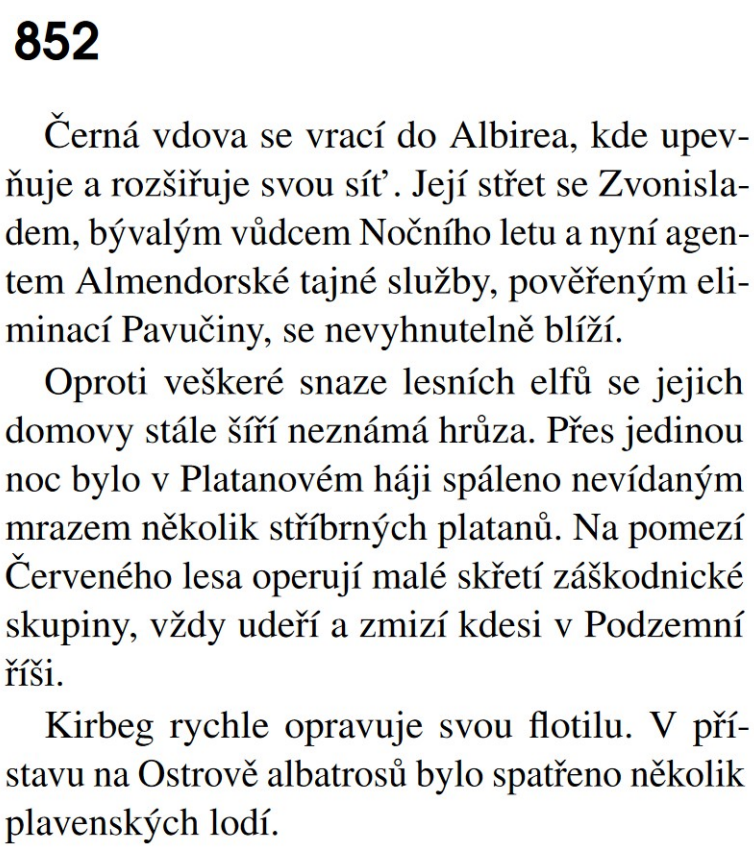
Kapitola 7

Data Asterionu

Tato kapitola obsahuje základní popis a definice dat světa Asterion dle [2]. Za tímto popisem následuje shrnutí náhledu do databáze. Následně jsou zde uvedené typické úlohy, kterých chce uživatel dosáhnout při používání vizualizace. S úlohami se také pojí různé typy interakce, které následují hned po úlohách. V obou případech se bude opět porovnávat s [2] a jednotlivé aspekty vizualizační teorie [3, 5, 8, 9]. Nakonec zde bude analýza zvolené vizuální reprezentace a její dopady na vizualizaci, posledním tématem bude výběr vizualizační techniky dat.

7.1 Popis dat

Data světa Asterion jsou ve své surové podobě zápisy různých událostí tak, jak je napsali autoři jednotlivých knih. Data samotná jsou zapsána v kronikách světa, kde většina zaznamenaných událostí nastala mezi roky 846 a 873. Zdroj [2] dále uvádí další vlastnosti z kroniky. Každý rok je samostatná kapitola a všechny události jsou chronologicky poskládány. Každá událost je jeden odstavec, který může obsahovat i úzce spojené události. Zdrojová práce pak také uvádí, že v některých případech je zde uvedena i lokace, kde k dané události došlo. Dle ukázky 7.1 u událostí nemusí být přítomny přesné časové údaje, kdy daná událost nastala, v takovém případě tento zdroj přiřazuje náhodné datum s podmínkou zachování pořadí událostí. [2]



852

Černá vdova se vrací do Albirea, kde upevňuje a rozšiřuje svou síť. Její střet se Zvonisladem, bývalým vůdcem Nočního letu a nyní agentem Almendorské tajné služby, pověřeným eliminací Pavučiny, se nevyhnutelně blíží.

Oproti veškeré snaze lesních elfů se jejich domovy stále šíří neznámá hrůza. Přes jedinou noc bylo v Platanovém háji spáleno nevídaným mrazem několik stříbrných platanů. Na pomezí Červeného lesa operují malé skřetí záškodnické skupiny, vždy udeří a zmizí kdesi v Podzemní říši.

Kirbeg rychle opravuje svou flotilu. V přístavu na Ostrově albatrosů bylo spatřeno několik plavenských lodí.

Obrázek 7.1: Ukázka dat z kroniky pro rok 852 [26]

Z tohoto popisu jednotlivých událostí se pak vytváří datové položky. Výsledek ukázky 7.1 může vypadat jako na obrázku 7.2. Jednotlivé události tedy mají svůj název, popis, datum a tagy. Popisná část není tolik zajímavá, jedná se pouze o shrnutí a popisy. Důležitější je informace o tom, kdy daná událost nastala a pak také tagy. **Tagy** samotné jsou klíčová slova, která s danou událostí souvisí. Ta pak krátce a výstižně pojmenovávají různé typy kategorií, které jsou jednoznačné. Příklad by měl vše vyjasnit. Existuje událost „Karel IV. nechal postavit hrad Karlštejn“, k této události se vytvoří dva tagy. Prvním bude „osoba“/„panovník“ „Karel IV.“, druhým je pak „Karlštejn“, který je v kategorii „místo“/„hrad“. Více možností v kategoriích je odůvodněno různou granularitou. Kromě základních tagů také existují **metatagy**, které mohou zastupovat větší celky. [2]

Podstatnou částí dat je časové ukotvení záznamu. To je v datech řešeno na úroveň roku, měsíce a dne. Nicméně Kalendářní systém Asterionu je jiný než gregoriánský nebo podobné kalendářní systémy z reálného světa. Zdroj [2] uvádí, že je složitější[1]. Z popisu v kapitole 4.1.1 tohoto zdroje vyplývá, že jeden rok má délku dvanácti měsíců. Každý měsíc má přesně třicet dní

[1] Spíše závisí na úhlu pohledu

První událost

- **Název události:** Černá vdova se vrací do Albirea
- **Popis:** Černá vdova se vrací do Albirea, kde upevňuje a rozšiřuje svou síť. Její střet se Zvonisladem, bývalým vůdcem Nočního letu a nyní agentem Almendorské tajné služby, pověřený eliminací Pavučiny, se nevyhnutelně blíží.
- **Datum:** 3. 2. 852
- **Tagy:** Černá vdova (osoba), Albireo (země), Zvonislad (osoba), Pavučina (organizace), Almendorská tajná služba (organizace)

Druhá událost

- **Název události:** Lesní elfové jsou napadání skřety
- **Popis:** Oproti veškeré snaze lesních elfů se jejich domovy stále šíří neznámá hrůza. Přes jedinou noc bylo v Platanovém háji spáleno nevídaným mrazem několik stříbrných platanů. Na pomezí Červeného lesa operují malé skřetí záškodnické skupiny, vždy udeří a zmizí kdesi v Podzemní říši.
- **Datum:** 17. 6. 852
- **Tagy:** lesní elfové (rasa), Platanový háj (les), Červený les (les), skřeti (rasa), Podzemní říše (země)

Třetí událost

- **Název:** Kirbeg opravuje svou flotilu
- **Popis:** Kirbeg rychle opravuje svou flotilu. V přístavu na Ostrově albatrosů bylo spatřeno několik plavenských lodí.
- **Datum:** 22. 8. 852
- **Tagy:** Kirbeg (osoba), Ostrov albatrosů (ostrov), Plavena (země)

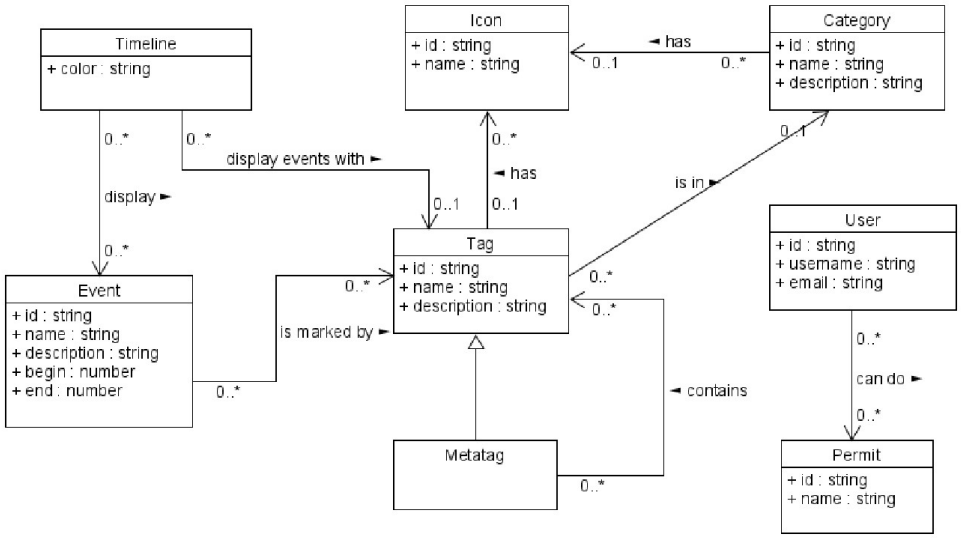
Obrázek 7.2: Interpretace dat z ukázky 7.1. [2]

a jeden týden[2] má deset dní. Mimo tento základní systém pak existují svátky, které nepatří do žádného týdne a stojí na hranici měsíců. Těchto svátků je deset jen není zřejmé, zda jsou tyto svátky pohyblivé či nikoliv a kdy přesně se odehrávají[3]. Zde by tedy mohla být jediná komplikace a to zanášení dat na časovou osu. Nicméně zvolené zanášení dat ve zdroji je dobrý nápad. Každá událost se měří jako vzdálenost od určitého dne, přesněji prvního dne roku nula. Tím pádem je snadné data seřadit, protože počet dnů dle popisu nekolísá a vše lze dopočítat. [2]

Na obrázku 7.3 je vidět schéma uložení dat. Dále pak obrázek 7.4 ukazuje část databáze, z obou obrázků vyplývá, že se bude jednat o relační data a tedy abstraktní datový typ. Tato data jsou navíc ukotvena v čase, a nejedná se o čistě relační data, ale o sloučení relačních a časově proměnných dat. Časová složka zde reprezentuje historický vývoj. Ten je srovnatelný s jakoukoliv reálnou časovou linií, která popisuje válečný vývoj. Na časové linii jsou zaznamenány body, které obsahují události, ty nesou svůj popis a další informace.

[2] Asterionský týden se nazývá **alden**.

[3] Po rozhovoru se zadavatelem se tato část vyjasnila. Každý rok je stejný, tedy neexistují výjimky jako v reálném kalendáři s přestupnými roky.



Obrázek 7.3: Schéma uložení dat Asterionu. [2]

Celá práce [2] se zabývá pouze jednoduchými událostmi[4]. To však není přesný popis všech dat, jelikož některá data mohou být i intervalová[5], hierarchická[6] nebo časové kontexty. To je nejsnáze představitelné na příkladu postavy. Každá postava se někdy narodila, což značí začátek a někdy umřela, tedy konec. Mezi těmito dvěma body je možné vést linku, která přeskakuje mezi osami politických celků na znamení toho, že se daná postava pohybovala v určitou dobu na určitém území nebo pro dané území vykonávala úkoly. Tyto události bude vhodné zohlednit a případně pro ně připravit zázemí.

7.2 Úlohy

Sekce 3.7 popisuje možné úlohy s časově proměnnými daty. Ve zkratce řečeno, časová data jsou složená[7] a tím pádem jsou dotazy děleny pro jednotlivé části. Nad samotnou časovou osou se typicky provádí dotazy na existenci událostí, kdy probíhali dlouhotrvající události nebo jaké bylo pořadí událostí. Další úlohy s časovou osou jsou například pohled do dat, nalezení specifické události, prohlížení časové osy a hledání různých událostí a jejich vztahů, případně i anotace nových událostí. Je také možné zjištění prezentovat někomu jinému nebo čistě objevovat, co data skrývají.

[4] Jednorázové okamžiky v čase.
[5] Například války, dohody, spolky atd.
[6] Kaskáda různých událostí, které zapříčiňují jinou událost v řadě.
[7] Tím je myšleno, že máme časový aspekt a samotnou informaci.

Kromě časové osy se studují i data, která jsou přidružena časovému primitivu. Opět jsou prováděny dotazové akce s identifikací událostí a čeho přesně se týkali. V obecné rovině by se zde mohlo objevit i porovnání, které poslouží ke srovnání států, ztrát/nákladů války a tak podobně. Dále úlohy poskytnutí přehledu, případně shrnutí událostí. Při prohledávání je možné podívat se na podrobnosti události o které se ví, kdy byla. Dále je možné vědět o nějaké události, ale je třeba zjistit kdy nastala, případně průzkum všech událostí s hledáním nějaké závislosti nebo určité procházení a hledání na základě popisu. U samotných záznamů také dává smysl zkoumání pro možnou prezentaci, vyvozování závěrů, vlastní užití nebo například prezentaci či anotaci.

S připomenutím je tedy možné přesněji určit, jaké úlohy je možné řešit s daty. V datech Asterionu bude vhodné provádět zejména úlohy na vyhledávání různých událostí. Vyhledávání může být prováděno jak nad časovou osou tak nad samotnými událostmi. Typický příklad proč je vhodné podporovat zejména vyhledávání je třeba, když uživatel chce sestavit určitý příběh a hledá místo, kam daný příběh zařadit. Prvním krokem bude poskytnutí přehledu o dané časové ose pro požadované území. Zde je viditelná úloha dotazu na celkový přehled. Následuje průzkum dat, hledání konkrétních událostí nebo s myšlenkou konkrétní události, prohledat její okolí. Se získanými daty pak uživatel dále pracuje dle potřeby. Mimo zmíněné úlohy je také možné dostat se k následující úloze: uživatel se dozvěděl o nějaké postavě z Asterionu, ale nemá přístup ke psanému zdroji, aby zjistil, kde se daná postava pohybovala. Ve vizualizaci je dle [2] možné zobrazovat nejen státy, ale i osoby, známé němé tváře, národy, rasy nebo třeba pohromy. Tím pádem se takový uživatel podívá do vizualizace a zjistí informace, které mu chybí. To přináší kromě úlohy identifikace také verifikaci[8] informací, které se uživatel mohl dozvědět.

Celá vizualizace však nebude jen o zobrazování jednotlivých časových os. Vizualizace má také posloužit jako nástroj, který umožní porovnávat více časových linií a dále s nimi pracovat. Dokáže ukázat kontext okolních království, vůči jednomu vybranému a tak podobně. Bude se tedy typicky pracovat s více než jednou časovou osou a budou se mezi sebou porovnávat[9]. Pro více časových os budou vykonávány úlohy týkající se dotazů nebo hledání. Jeden z možných dotazů je identifikace událostí, které jsou ve více liniích nebo jejich souvislost s jinými událostmi. Pokud by existovaly i číselné údaje k osám pak by bylo možné je na základě různých atributů porovnávat. Například během války provádět porovnání investic, nasazených vojáků, ztrátách na životech a podobně. Hlavní bod pro dotazy je pak přehled, který časové linie posazené blízko sebe mohou poskytnout. Vyhledávání umožňuje provést všechny jeho typy – nahlédnutí, kdy proběhla válka mezi dvěma královstvími, kdo se účastnil nějaké války, kdy se odehrála tragická/šťastná/významná

[8] Čili analytické úlohy jsou také přípustné.
[9] Hledat souvislosti mezi událostmi.

událost, prohlížení dat se znalostí některých kousků informací nebo čistě při bádání v datech. Kromě toho také slouží k některým analytickým účelům jako například ověřování domněnek. Jak již bylo naznačeno na začátku této sekce šířka záběru vizualizace je veliká. Lze zde najít analytické, dotazové nebo vyhledávací úlohy a je nutné s tím počítat. Hlavním cílem celé vizualizace je pomoci nalézt různé souvislosti mezi časovými osami a jejich daty, ze kterých se poté vyvodí závěry.

7.3 Interakce

Zdroj [2] uvádí v kapitole šest různé možnosti interakcí. Jako první je nastavení časových os, které jsou zobrazovány. To přesněji znamená změny barev případně posunutí osy výše či níže dle potřeby. Další interakcí je výběr buď samotných časových os nebo jednotlivých událostí a sledování jejich detailů. Kromě těchto interakcí je také možné měnit aktuálně viditelné tagy a vybírat jiné, které chce daný uživatel prozkoumat, tato část zastupuje filtraci či agregaci.

Z literatury [3] se nabízí široká paleta možných interakcí. Jak již bylo dříve zmíněno v 5.1, interakce mají více úrovní. Při porovnání s aktuálním stavem [2] je vidět mnoho prvků, které jsou použity[10]. Je zde zastoupena manipulace i změna měřítka. Dále jsou vidět také některé prvky z vyšší úrovně interakce, jako je změna dané vizualizace pro potřeby uživatele. Vyhledávání tagů zastupuje možnost interakce výběru hodnot, které budou vidět ve vizualizaci a v samotné vizualizaci je možné vybírat jednotlivé části[11]. U těchto částí je pak zobrazen detail. Je zde i zastoupena forma techniky focus&context. Kde je na přehledové ose vidět celkový časový úsek s naznačenými oblastmi různých událostí. V hlavním okně je pak vidět zvětšená podoba. Manipulace s pohledem je, jak již bylo zmíněno, prováděna pomocí přehledové lišty a tím se i posouvá vizualizovaná část dat. Filtrace dat je provedena na základě aktuálně vybraných dat, jak může být vidět na obrázku 7.5 a 7.6. Vizualizaci pak tvoří celkově dva pohledy, přehled a hlavní okno s vizualizací. [2]

Na základě zjištěného typu dat zde bude dále rozebráno jaké jsou vhodné interakce na základě [3]. Z nižší úrovně interakcí je možné použít tažení myší nebo mačkání kláves pro posun časové osy. Případně kliknutí na prvky pro vyvolání nějaké další funkcionality. Dále budou popsány vyšší úrovně interakce. Z manipulace s daty je pro data Asterionu dobrá interakce výběru jednotlivých prvků vizualizace ať už se bude jednat o události nebo o samotné

[10] A také mnoho, které by bylo možné zlepšit

[11] Osy nebo události.

osy. Pro více os se také nabízí využití posunu samotných os pro zlepšení přehlednosti. Jelikož se jedná o časové osy, tak z [3] vyplývá jako dobrý styl navigace **manipulace s pohledem a zoom** na přiblížení překrývajících se dat(událostí). Technika **přehled a detail** také najde své uplatnění po vzoru [2], nicméně s menší obměnou. Vzhledem k množství dat bude vhodné využít určitou formu **filtrace nebo agregace**, kdy se do hlavního pohledu umístí jen uživatelem vybraná data. Případně by dávalo smysl provést agregaci některých částí, například při spojení království. To se tedy týká hlavně časových os. Pro události by agregace dávala smysl při sdružení několika událostí, které se staly ve stejné době na jednom místě. Při analýze různých zobrazení se zdá jako nejlepší volba již vytvořené rozmístění z [2]. Přesněji menší okénko s hrubým přehledem a hlavní okno s vybranými daty.

7.4 Vizuální reprezentace

Zdroj [2] uvádí, že pro zobrazení více os najednou bude dobré využít minimalistické zobrazení jednotlivých os. To znamená, že každá osa bude tvořena přímkou, na kterou se budou vynášet jednotlivé události. Každá osa má podle [2] své specifické označení pro snadnější orientaci. Což s menšími odchylkami odpovídá optimální reprezentaci na základě literatury.

Co se pak týká událostí [2] uvádí, že jednotlivé události budou reprezentovány pomocí malého obrázku(ikonky) na časové ose. Dále také zmiňuje, že každá událost bude mít přiřazenou svou ikonku, která patří do určité kategorie. Tato část také odpovídá zjištění z literatury, navíc jak z tohoto zdroje tak ze jištění vyplývá, že bude vhodné sdružit podobné události pod jednu reprezentaci(značku). Vytvořené značení událostí je tedy možné použít. [2]

Jak již bylo řečeno jedná se o časová data, která na sobě mají navázané další informace. Je tedy nutné uvažovat dvě části vizualizace. Časovou osu bude dobré reprezentovat pomocí přímky nebo křivky[12], pokud by docházelo k přibližování jednotlivých časových os, které spolu nějakým způsobem souvisí. U os tedy pozbývá smyslu řešit rozdílné šířky, délky nebo tvary, zároveň je potřeba dle [3] dodržet například přehlednost s co nejmenším zastíněním. Pokud by se ve vizualizaci vyskytovalo pouze velké množství os, pak by taková vizualizace nebyla příliš dobře použitelná. Hlavním důvodem by byla špatná rozlišitelnost/oddělitelnost jednotlivých elementů. Proto je vhodné podpořit

[12]Křivka bude pravděpodobně trochu lepší nástroj, důvodem je možnost jejího přesměrování na určitou dobu do jiné části. To pomůže při znázorňování válečného konfliktu, spojenectví a podobných dlouhodobých událostí.

tento aspekt s kanálem, který má velmi dobrou rozlišitelnost a to barevnými odstíny. Při zobrazení pak bude jednoduché sledovat jednotlivé časové osy a lépe se držet sledované linie. U barevných odstínů je nutné dbát na jednu podstatnou vlastnost a to rozlišitelnost. Pokud budou odstíny příliš blízko sebe, pak si je uživatel začne plést. Z toho plyne, že bude nutné správně vzorkovat barvy, které se použijí.

K časovým osám také bude dobré přiřadit, koho přesně se týkají. Vzhledem k povaze dat[13] lze použít například znaky království, erby měst, obecné ikony pro přírodní lokace spolu s názvem dané lokace. Tato reprezentace umožní jasné rozlišení ke komu daná časová linie náleží. Při sledování časové linie bude také dobré zvýraznit jednotlivá časová období, což při jednobarevném pozadí může být problém. Pokud se však každý rok/desetiletí/století[14] podbarví jinak pak bude snazší sledovat jednotlivá období na dané ose nebo pro vybranou část dat[15].[2, 3]

Když je podchycena samotná osa, na kterou se budou vynášet data, je dobré přesunout pozornost k datům, která jsou přidružena k časovým jednotkám. Jednotlivé události jsou označeny tagy/metatagy a jsou přiřazeny k odpovídající časové ose nebo osám. Těmto událostem bude vhodné přiřazovat tvary nebo obecně ikony. Vzhledem k velké rozmanitosti událostí je jasné, že bude existovat i velké množství jednotlivých značek, které budou reprezentovat specifické události. Tato rozmanitost pramení z faktu, že v kronikách jsou i data na úrovni různých postav, které vykonaly nějaký skutek tak i události postihující celá království. Samozřejmě je zde početná část událostí, které budou popsány synonymy a tím pádem nebude nutné přidávat jejich reprezentace. Nicméně pro větší část bude nutné vytvářet specifické značky. Tento počet je možné zmenšit, pokud se podaří nalézt šikovné rozdělení na kategorie, tím pádem se dle [3] využije i určitého typu shlukování. Respektive vytvořit „slovník“, který by sdružoval podobná slova a dával jim stejné značky. Některá data, jako například války, se mohou týkat více než jedné časové osy a tím pádem je vhodné tyto související části nějakým způsobem spojit. Spojení lze provést buď na úrovni os, které se k sobě přiblíží a po dobu konfliktu budou vedle sebe, nebo jen určitou spojnicí, která bude reprezentovat tento vztah. Druhá možnost však povede k velkému vizuálnímu nepořádku, který bude nutné redukovat nebo jiným způsobem odstranit. Kromě toho je také možné sledovat určité události, které jsou následky jiných událostí. Tato část nemusí být triviální a může vést na mnoho problémů.

[13] Data fantasy světa Asterion.
[14] Zde závisí na zvolené časové granularitě.
[15] Například stát nebo několik států pod sebou.

7.5 Vizualizace dat Asterionu

Mezi zkoumanými metodami bohužel nebyla nalezena taková, která by přesně odpovídala potřebám vizualizace dat Asterionu. Po prozkoumání dříve zmíněných metod jde vyřadit ty, které nejsou příliš použitelné. **Obrazově založená metoda** (6.1) a **sledování významných bodů** (6.2) nebudou vhodné pro data Asterionu. Obrazově založená metoda je vhodná pro prostorová data a sledování významných bodů je vhodnější pro sledování různých vlastností a jejich vývoje, což by bylo použitelné například pro jedno království a sledování vývoje pod různými panovníky. Nicméně pro sledování více částí najednou by mohl nastat problém.

Metody **spirála** (6.3), **TimeWheel** (6.5), **ThemeRiver** (6.4) a **kalendářová metoda** (6.6) jsou použitelné na některé aspekty[16], nikoliv však pro větší vizualizace a porovnávání různých os mezi různými entitami jako jsou království, postavy nebo jiná data. Nejméně použitelnou metodou je kalendář, který je schopen ukazovat jen omezenou část dat, jako jsou například společné události. Spirála je vhodná pro zkoumání jedné entity, nikoliv však pro více entit najedou, zejména kvůli nepravidelnosti událostí a také kvůli různým typům událostí. Metoda TimeWheel by trpěla podobnými nedostatky jako spirála, ale při rozdělení na jednotlivé události by mohla trochu pomoct, protože by vytvářela celkový přehled o jednotlivých typech událostí. Metoda ThemeRiver by posloužila k obdobnému účelu jako TimeWheel, ale pro porovnání různých částí dat by nebyla příliš dobře použitelná.

Metoda plánování pomocí **Ganttova diagramu** (6.7) také nebude příliš dobře použitelná. Nicméně má v sobě potenciál doprovodné metody. Tato metoda by se dala využít pro vykreslování jednotlivých dlouho trvajících událostí jako jsou například války. Ve kterých vykreslí veškeré potyčky, které na sebe chronologicky navazují s lepší časovou granularitou.

Dále tu jsou metody **Focus&Context** pro více časových os (6.11) a **komplexních sémantických časových os** (6.12), které jsou již vhodnější pro některé úlohy. Zejména při zobrazování několika os najednou. Dále také pro vytváření logických návazností a souvislostí v časových datech. Z ukázek sémantických časových os je vidět značný vizuální nepořádek v samotné vizualizaci a proto by bylo vhodné tuto vadu zmírnit. U metody Focus&Context s více časovými osami je problémem nejasné granularity událostí. Přesněji řečeno zda je možné měnit časové osy dle potřeb na dny a měsíce, nebo je to práce navíc. Nicméně se stále jedná o více použitelné metody než předcházející.

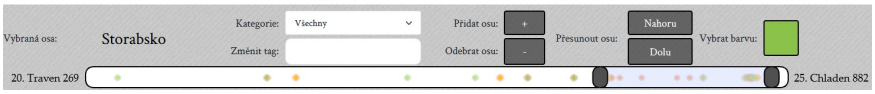
[16] Pro některé úlohy.

Poslední nerozebrané metody v rámci této části jsou **helix/lexis glyfy** (6.8), **genealogické metody**(6.9) a **TimeSet** (6.10), které mají z prozkoumaných metod největší potenciál předat zamýšlenou informaci. Metody helix/lexis glyfů jsou z posledních tří asi nejméně použitelné ve své originální formě. Při úpravě by vznikla zajímavá vizualizace časového postupu na jednom území, kde by se sledoval vývoj na časové ose s pohybem po mapě. Což by pomáhalo utvořit celkový náhled na vývoj událostí v rámci jednoho území nebo postavy. Avšak u porovnání několika časových os by opět nastal problém. Tedy tato metoda je využitelná v určitých aspektech, ale ne vždy. To je trochu rozdíl u zbývajících dvou metod. Metodu TimeSet je možné využít tak, jak je viditelné na odpovídajících obrázcích. Ovšem s plynulejší návazností a lepší možností spojovat události, které spolu souvisí[17]. Genealogická metoda bude nejlépe využitelná v mnoha ohledech. Umožňuje snadné porovnávání dvou os, které je i možné k sobě přiblížit a vyznačit na ně důležité události, případně jinak upravit. Jedná se pouze o nadstavbu vůči současnému stavu, ale zdá se být dobrým odrazovým můstkem pro další zkoumání a s menšími úpravami bude použita. Při využití některých aspektů z metody TimeSet a jejich opatrném zapracování do genealogické metody by vznikla požadovaná vizualizace, nebo alespoň její dobrý základ.

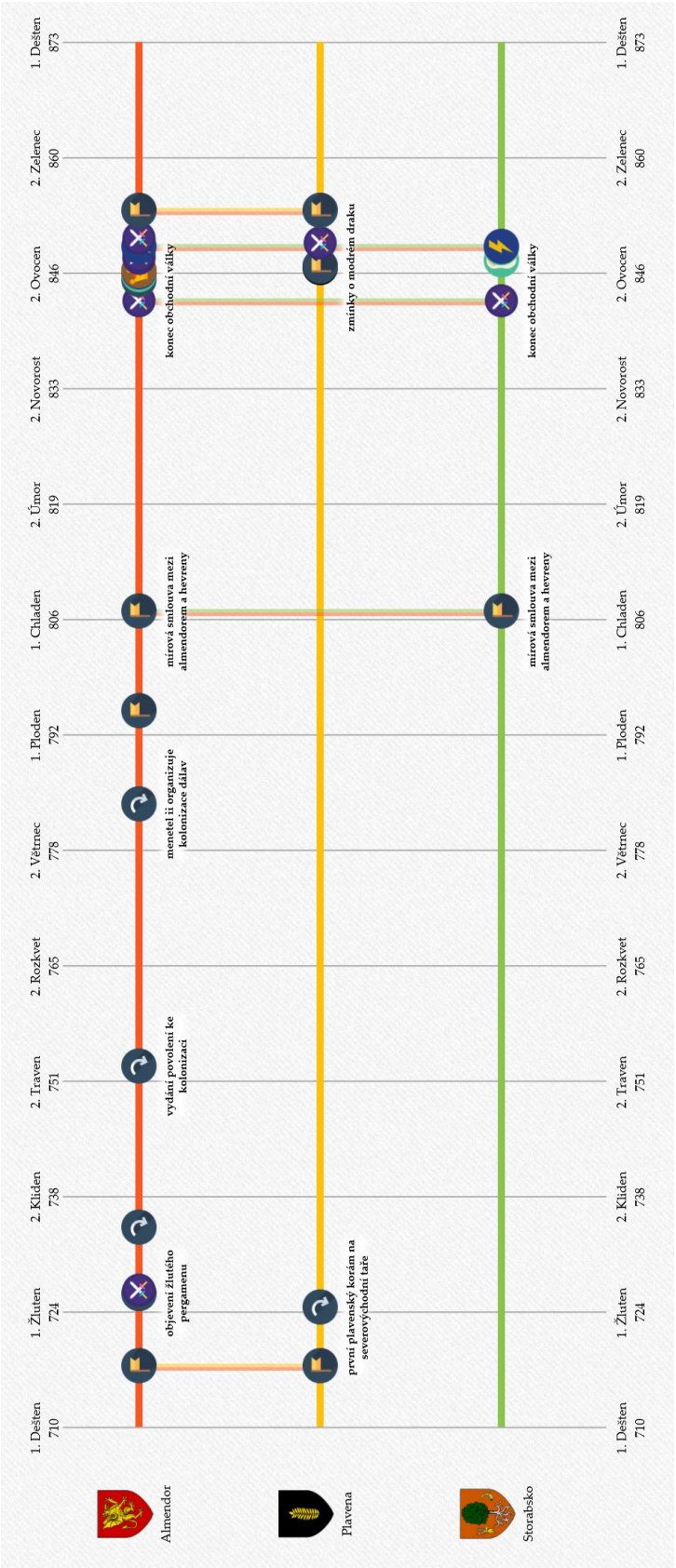
[17]Což by byl ideální stav vůči hlavní úloze/úkolu, který má vizualizace podpořit.

event_id	old_	event_ye	event_	event_title	event_description
100	88	846	0	objevení se vyšších harpyjí v Jarlebském knížectví	V Jarlebském knížectví v Západní Dálavě se objeví hejno harpyjí. Lovec přízraků je prohlásí za neobyčejně nebezpečné jedince vyš
101	32	846	0	Orlí poutníci přebírají Taros	Orlí poutníci zřídí stálou základnu v Tarosu a poklidnou cestou zde nahradí politické struktury Východního království.
102	135	846	0	první zmínky o skřetech u Barbarských ostrovů	Dva šťastnou náhodou zachránění trosečníci ze sarindarské obchodní lodi viděli těsně před jejím potopením zástupy vodních skře
103	71	846	0	sucha na Lendoru	Na Lendoru propuknou mimořádná sucha, na některých místech nezaprší i čtvrt roku. V Keledoru a Almendoru, které jsou postiže
104	105	846	0	zmizení vesnice Dubeneč	Vesnice Dubeč, ležící hluboko v Lese padajících stínů, nejzasší výspa civilizace v této oblasti, beze stopy zmizí, stejně jako její obyv
105	66	847	0	bouře nad Yrakanem	Přes jezero Yrakan se přežene silná a ničivá bouře. Ztráty na životech budou malé (jak to již v Umrlčím království bývá), ovšem na
106	82	847	0	gilda Varemar najde cestu přes Zelany	Hevrenka Ebelita provede pro gildu Varemar přes Zelanské vrchy k Durenu karavanu vezoucí železo k dalšímu zpracování. Stezka v
107	8	847	0	nástup Waldena er Gwendor na trůn	Rioden II. odstoupí z trůnu Východního království a přenechá vládu svému synovi Waldenovi.
108	33	847	0	ovládnutí gildy Kalista Hesterily	Kupecký rod Hesterilů z Gondromu, známý svými kontakty na nejvyšší kruhy Mořského císařství, ovládne pomocí obchodních i jin
109	61	847	0	Rebenocovo neštěstí	Ve Vřesovém údolí za záhadných okolností vyhoří dům kupce Rebenice, který zde ovšem již dva roky nepobývá. Ostatky jeho ženy
110	72	847	0	Telurian dokončuje zkoumání Koule vědění	Velmistr Telurian odhalí poslední vlastnost Koule vědění - spojení s obdobným zařízením v Silviandu - a začne přesně zaměřovat p
111	103	847	0	výprav Doreka Straky do Džungle padlých stromů začíná	Do Džungle padlých stromů se vydá ze Severní dálavy výprava storabského hrdiny Doreka Straky.
112	18	847	0	zmínky o modrém draku	Nad Plavenou bude již poněkolikáté za posledních pár let spatřen obrovský modrý drak.
113	106	847	0	ztrácí se Firunag	V Khelegových horách se ztratí trpasličí děvčátko Fiunrag. Veškeré pátrání po ní bude bezvýsledné.
114	75	847	0	zvěsti o Inace	Nejprve na Taře a posléze i na Lendoru se mezi lidmi rozšíří pověst, že sucha z minulého roku nebyla dílem Aurionovým, ale způso
115	81	848	0	exulanti Finwalurovi	Téměř všichni kněží Poutníka ze Storabska přejdou k temnému bohovi chaosu a vrtkavého štěstí Sarapisovi.
116	118	848	0	hevrenská odplata	Jeden z almendorských jízdních pluků bude překvapen přesilou hevrenů, která na něj zaútočí jednoho dne za svítání koncem roku
117	2	848	0	narozen dědic Storabska	V Manře oslaví narození královského dědice. Králi Krutji počal syna Barina.

Obrázek 7.4: Ukázka dat v databázi [26]



Obrázek 7.5: Ukázka aktuální verze vizualizace, pouze hlavička[2]



Obrázek 7.6: Ukázka těla vizualizace [2]

Obrázek 7.7: Návrh kolegyně Zimmermannové na ikonky několika různých událostí [2]

Kapitola 8

Analýza technologií

Tato kapitola obsahuje informace ohledně použitelných technologií pro realizaci vizualizace. Základní informace jsou převzaty z [2], které jsou v některých místech doplněny. Zároveň je přidáno několik dalších knihoven, které mají potenciál vytvořit požadovanou vizualizaci. Na konci této kapitoly bude vybrána nejlépe se hodící knihovna. Vzhledem k tomu, že se jedná o frontend webové vizualizace[1], pak nemá smysl zaměřovat se do hloubky na jazyky/knihovny, které řeší backend[2]. Proto zde budou rozebrány hlavní části frontendu.

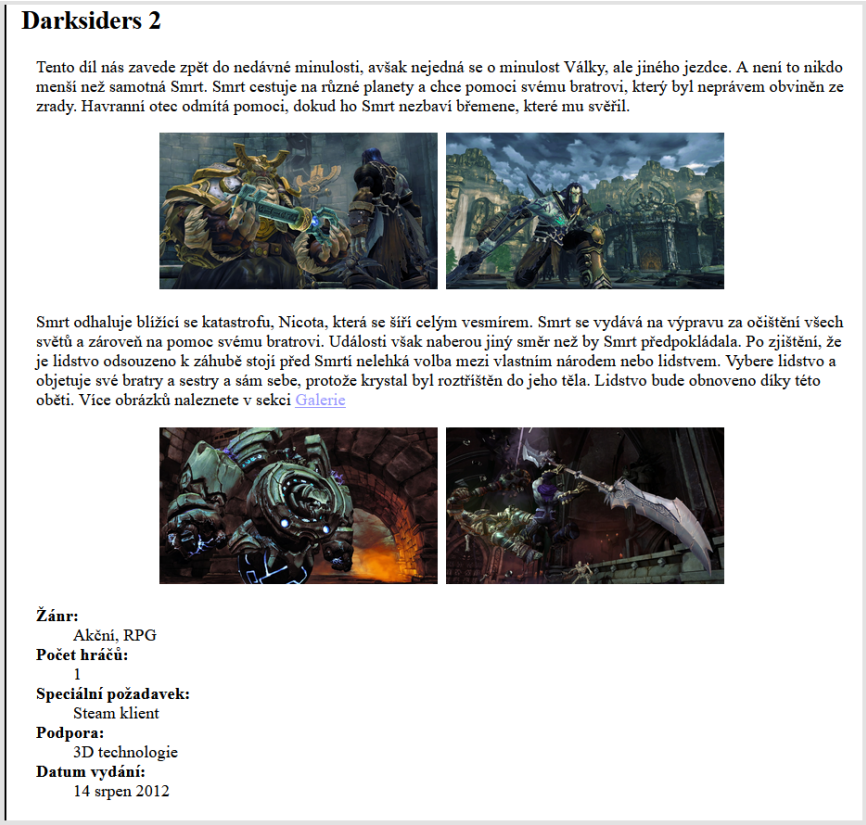
8.1 Webové stránky

Obecně platí, že vývoj webových stránek je rozdělen do dvou částí. **Frontend** je část webových stránek, které prezentují data uživateli, zpracovávají jeho dotazy a zobrazují odpovědi z backendu. **Backend** se pak typicky zabývá zpracováním dotazů a jejich vyhodnocení, správou a zpracováním dat. V této práci jde zejména o frontend spojený s vizualizací. Z toho vyplývá, že některé programovací jazyky/knihovny zde nebudou vůbec zmíněny.

Po prozkoumání několika webů s tématikou webových stránek vyplynulo, že nejčastěji používané technologie pro vývoj frontendu jsou **HTML**, **CSS** a **JavaScript**. U zbývajících technologií informace o použití nebyla přímo

[1] Stanoveno v zadání.

[2] Stejně jako frontend i zde se jedná o běžně používaný výraz u webových aplikací a tak bude použit i zde.

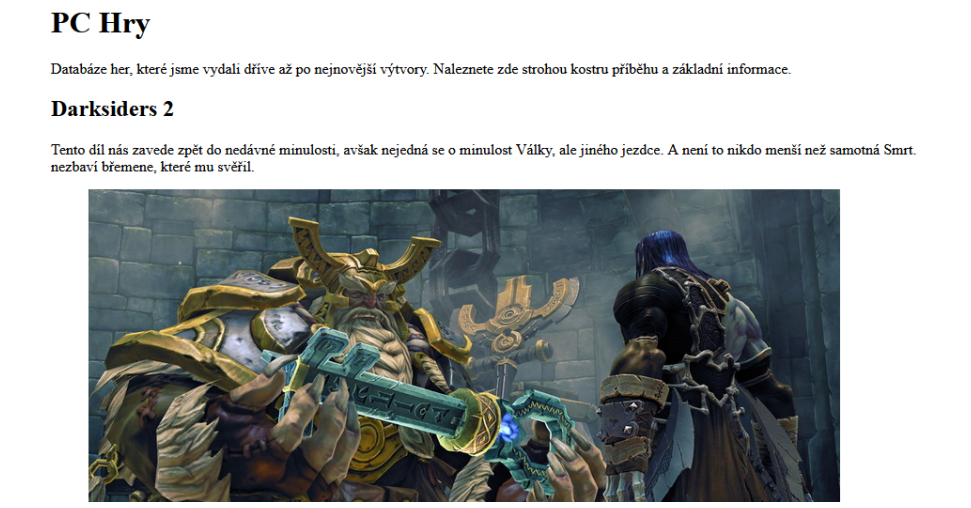


Obrázek 8.1: Ukázka spojení HTML a CSS pro statickou webovou stránku

uvedena, nebo bylo uvedeno, že se využívají zejména na backend daného webu. Výjimku tvoří například Angular [27] či PHP [28]. Angular je sám o sobě **framework**[3] pro frontend a PHP je spíše serverový jazyk, nicméně je možné v něm vše připravit a odeslat uživateli. Vzhledem k výběru technologie v [2] bude učiněn stejný krok a budou debatovány pouze jednotlivé frameworky, které je možné využít. [29, 30, 31, 32]

O **HTML** se nedá říct, že se jedná o programovací jazyk, jde o určitý druh popisu struktury. Ta je popisována pomocí **tagů**[4], které jsou ve většině případů párové, ale existují i nepárové. Každý z těchto tagů může mít k sobě další parametry, pomocí kterých je možné jej snáze nalézt nebo určitým způsobem upravit. Tagy samotné jsou pak uzavřeny symboly větší a menší než. Jako příklady poslouží *<body>*, *<ol>* nebo * *, který spolu s tagem ** definují menší celky. Tyto celky lze dle libosti upravovat. Úpravy mohou být zcela jiné než u jiného tagu typu * *. Všechny zmíněné tagy jsou párové, ty v kódu mohou vypadat například takto * Loren ipsun dolor sit anet, consectetur adipisci elit, sed eiusnod tenpor incidunt ut labore et dolore nagna aliqua. *. Čistý HTML kód se využívá zejména na

[3] Běžně používaný výraz i mimo webové aplikace.
[4] Neplést s tagy světa Asterion.



Obrázek 8.2: Ukázka převodu HTML kódu do vizuální podoby

```
<div id="content">
  <h1>PC Hry</h1>
    <p>Databáze her, které jsme vydali dříve až po nejnovější výtvory. Naleznete zde strohou kostru příběhu a základní informace. </p>
    <!-- položka //-->
 <div class="kariera_polozka">
    <h2 id="darksiders2">Darksiders 2</h2>
      <p>Tento díl nás zavede zpět do nedávné minulosti, avšak nejedná se o minulost Války, ale jiného jezdce. A není to nikdo menší než samotná Smrt. Smrt cestuje na
      <ul>
          <li><img src="./images/Darksiders_2/154313.jpg" alt="Klíč konstruktů" title="Smrt získává dar od staršího Tvořitele" /></li>
          <li><img src="./images/Darksiders_2/154310.jpg" alt="Příprava na boj" title="Smrt v akčním postavení" /></li>
      </ul>

      <p>Smrt odhaluje blížící se katastrofu, Nicota, která se šíří celým vesmírem. Smrt se vydává na výpravu za očištění všech světů a zároveň na pomoc svému bratrovi
      Více obrázků naleznete v sekci <a title="">Galerie</a></p>
      <ul>
          <li><img src="./images/Darksiders_2/148840.jpg" alt="Konstrukt" title="Jízda na konstruktovi" /></li>
          <li><img src="./images/Darksiders_2/144622.jpg" alt="Souboj" title="Souboj Smrti s jedním z mnoha nepřátel" /></li>
      </ul>

      <dl>
         <dt>Žánr:</dt>
          <dd>Akční, RPG</dd>
         <dt>Počet hráčů:</dt>
          <dd>1</dd>
         <dt>Speciální požadavek:</dt>
          <dd>Steam klient</dd>
         <dt>Podpora:</dt>
          <dd>3D technologie</dd>
         <dt>Datum vydání:</dt>
          <dd>14 srpen 2012</dd>
      </dl>
 </div>
```

Obrázek 8.3: Ukázka čistého HTML kódu bez interpretace

statických webových stránkách, kde se obsah téměř nemění. HTML samo o sobě vypadá jako prostý text 8.2. K tomu aby bylo možné dát textu nějaké zajímavé uspořádání nebo vzhled je potřeba další díl a tím jsou kaskádové styly neboli CSS. [2, 33]

Kaskádové styly (CSS) slouží zejména k tomu, aby přidali nějaký poutavý design ke kódu HTML. Styly lze definovat přímo v kódu HTML nebo v externím souboru, který je pak jednoduše načten pro daný HTML kód. Je i možné pro jeden HTML soubor mít načtených více souborů se styly a z nich čerpat. Tuto část webového vývoje je snadné se naučit, ale vzhledem k obsahu je velmi těžké znát veškeré podrobnosti. Pro každý HTML tag existují až stovky různých vlastností CSS. Samozřejmě existují i knihovny, které mohou s vytvářením stylů pomoci a tím pádem není nutné mít větší znalost v této části. Zatím bylo pouze naznačeno, co CSS umí a podtrženo, že to není jednoduchá část vývoje na ovládnutí. Co přesně je tedy možné

s CSS vytvořit? Na obrázku 8.2 je vidět základ statické webové stránky, která je definována kódem z obrázku 8.3. Vzhled je pak upraven podle pravidel stanovených v CSS souboru. Tento soubor je poměrně rozsáhlý a tím pádem jsou zde ukázány jen čtyři výřezy ze souboru se styly, obrázek 8.4. Výsledná aplikace pravidel CSS na elementy HTML kódu jsou vidět na obrázku 8.1. [2, 33]

HTML poskytuje základní funkcionalitu textových polí a tlačítek. Pokud je však cílem dodat další funkcionalitu pak je nutné sáhnout k programovacím jazykům. Jak již bylo zmíněno, pro vývoj webových stránek se nejčastěji používá **JavaScript**. Jedná se o skriptovací jazyk, který umožňuje ve velké míře rozšiřovat funkcionalitu webových stránek. S jeho pomocí je možné dynamicky měnit různé prvky dané webové stránky, například ovládání multimédií nebo animace. Vzhledem ke svému velkému zastoupení na webových stránkách vzniklo mnoho různých knihoven, které usnadňují práci s různými částmi HTML kódu a také vytvářet různé verze grafiky (jak 2D tak 3D). Tento jazyk tedy může být použit na téměř cokoliv, co si člověk může vymyslet. Zde je hlavním cílem použití na frontendu, nicméně Javascript je možné využít i na backendovou část webu a to v podobě Node.js. Kromě tohoto frameworku existují ještě další jako je například Vue, React nebo Angular, které jsou velmi poptávané. Obrázek 8.5 ukazuje velmi jednoduchý příklad funkce pro úpravu obsahu tagu. [2, 33]

8.2 XML/SVG

Tato sekce krátce shrne dvě hojně využívané technologie v rámci webů. Jedná se o XML a SVG. Opět se jedná o značkovací jazyky s tím rozdílem, že v XML je možné přidávat nové tagy. SVG je pak nadstavbou nad XML. Tato technologie je zmíněna zejména z toho důvodu, že byla uvedena v posudku oponenta bakalářské práce [2]. Navíc se jedná o jiný přístup než je použit ve zmíněné bakalářské práci. Navíc přinést zajímavé výkonnostní výsledky. Proto byla tato technologie vybrána jako stěžejní prvek, který musí hledané knihovny umět ovládat.

8.2.1 XML

Jedná se o rozšiřitelný značkovací jazyk, což jinak řečeno znamená, že se podobá HTML s tím rozdílem, že nejsou předdefinovány tagy. Tedy každý

programátor si vytvoří vlastní specifikace, podle svých potřeb. Je to mocný nástroj, jak uložit data v požadovaném formátu. Takto definovaná data je snadné prohledávat a sdílet. Vzhledem ke své standardizaci je pak možné přenášet XML soubory různými způsoby a na jiná zařízení a tam je opět zobrazit ve stejném rozložení i formátu. Nad touto technologií bylo vystaveno několik dalších jazyků jako například XHTML, SVG nebo RSS a také mnoho dalších. [35]

Na první pohled by se mohlo zdát, že HTML a XML jsou si velmi podobné a mohla by vyvstat otázka proč se nevyužívá pouze XML. Důvod je, že každá z těchto technologií byla vytvořena s jiným cílem. HTML bylo vytvořeno tak, aby zobrazovalo data, která mu jsou předána, je tedy zaměřen na to, jak jsou data reprezentována. Naproti tomu XML bylo vytvořeno k přenosu dat, tedy se zaměřením na to, co jsou data. Jak již bylo zmíněno, XML nemá proti HTML pevně stanovené tagy. [34]

Dále je možné o XML říci, že zjednodušuje sdílení, přenos, změnu platformy nebo dostupnost. Sdílení je možné nalézt například pokud je cílem přenos dat mezi systémy, ty typicky nemusí být kompatibilní. Taková výměna bývá časově náročný úkon, proto je dobrým krokem data konvertovat a sdílet pomocí XML. Data samotná jsou uložena jako text. To umožňuje softwarově i hardwarově nezávislý transport a sdílení. Při využití XML jsou data snadněji dostupná pro všechny možné typy čtenářů[5]. [34]

8.2.2 SVG

Jedná se o jazyk, který je zaměřený na vektorovou grafiku a je přímým rozšířením standardu XML. Jedná se o textově založený webový standard pro popis obrázků, které jsou zobrazeny čistě v jakékoliv velikosti a je specificky vytvořen pro dobrou spolupráci s ostatními webovými technologiemi[6]. SVG grafika je vztažena ke XML textovým souborům, což znamená, že opět mohou být prohledávány, indexovány nebo komprimovaný dle potřeby. Navíc je možné tyto soubory upravovat pomocí kteréhokoliv textového nebo kreslicího editoru. Při porovnání s klasickými formáty pro grafický výstup jako jsou JPEG nebo PNG, formát SVG reprezentuje jiný typ obrazu[7]. SVG formát umožňuje bezztrátové vykreslení v jakékoliv velikosti. Což je rozdíl oproti např. PNG, které se při velkém zvětšení začne rozdělovat do menší regionů[8]. Navíc vektorový obrázek formátu SVG je možné lokalizovat na základě jeho

[5] Jak pro strojové tak pro lidské.
[6] CSS, JavaScript, atd.
[7] SVG – vektorová vs PNG/JPEG – rastrová.
[8] Pixelů

úprav bez potřeby grafického editoru. Se správnými knihovnami je možné upravovat SVG obrázek za běhu. [37]

8.3 Knihovny JavaScript

Tato sekce obsahuje různé zkoumané knihovny, které jsou použitelné pro vykreslování různých objektů s pomocí JavaScriptu. Z těchto knihoven bude nakonec jedna vybrána, která bude využita pro vytvoření vizualizace dat Asterionu. V úvahu jsou brány i již zjištěné části z [2].

8.3.1 PixiJS

Knihovna založená na WebGL a Canvas, která umožňuje vykreslovat různé typy grafiky. Kromě dvourozměrné grafiky je možné s rozšířením vytvářet i trojrozměrnou grafiku, nicméně k tomu je nutné vzít Pixi3D. PixiJS jako takové má spoustu zajímavé funkcionality a aspektů. Podporuje například vytváření multiplatformní vizualizace, organizaci objektů do hierarchie scény, práci se sprity[9] nebo z podstaty WebGL různé druhy filtrování. Dále je dle [38] intuitivní a jednoduché na použití a zároveň dostatečně silné. Ukázkou možného použití je výsledná vizualizace práce [2], kterou je vidět na obrázku 7.6.

PixiJS byla v práci [2] upřednostněna z důvodu dřívější práce s ní. Na základě dokumentace lze říci, že obsahuje vše, co bude třeba k vytvoření vizualizace. Z hledání vyplývá, že největší problém nastane při pokusu o požití SVG grafiky. Dle [39] je to způsobeno tím, že se jedná o rastrové vykreslování s použitím WebGL. Z ukázek na oficiálních stránkách PixiJS vyplývá, že v rámci rastrové grafiky bude možné vytvořit téměř cokoliv. Z příkladů přiložených na hlavní stránce se potvrzuje bohaté rozhraní i široká použitelnost.

Po důkladném přečtení [39] bylo zjištěno, že funkcionalita okolo SVG skutečně v době položení dotazu (2014) nebyla a bylo nutné ji provést pomocí různých parametrů či si vytvořit vlastní funkcionalitu. Postupně vznikala menší řešení tohoto problému, ale není jasné, zda bylo některé z nich přijato jako oficiální řešení. V roce 2021 byl uveřejněn první větší doplněk pro PixiJS, který dle [40] umožňuje mnohem lepší práci s SVG grafikou.

[9] Dvou dimenzionální bitmapa, typické zejména pro 2D video hry.

8.3.2 Phaser

Jedná se o knihovnu, která slouží zejména pro snadnou tvorbu her. Tedy podporuje fyziku, sprity, částice, různé druhy vstupů, ovládání zvuku a mnohé další. Opět využívá WebGL a Canvas. Tato knihovna umí dvou i tří rozměrnou grafiku a zaměřuje se pouze na Webové technologie(HTML5). [41]

Phaser bude vzhledem ke své povaze obsahovat mnoho funkcionalit pro snadné vytvoření uživatelského rozhraní. Co se týká možností zobrazených částí, pak z ukázek na hlavní stránce [41] je vidět, že knihovna umožňuje implementaci velkého množství funkcionalit jak rozhraní, tak mechanik či vzhledu. Vzhledem k využití WebGL bude opět problém s interpretací SVG grafiky [43], které budou při špatném nastavení rozmazané. Jinak má podobný rozsah jako PixiJS. Což není žádným překvapením, vzhledem k tomu, že využívá právě PixiJS jako vykreslovací knihovnu. [41, 44]

8.3.3 D3.js

D3.js se na první pohled zdá jako velmi solidní knihovna, která má bohaté portfolio možných vizualizací. Z oficiálních stránek [45] vyplývá, že tato knihovna pracuje s HTML, SVG a CSS pomocí manipulace s jednotlivými prvky dokumentu[10]. Využívá silnou kombinaci vizualizačních komponent a datově řízeného přístupu k **DOM**[11]. Na oficiálních stránkách je dále uvedeno, že D3.js umožňuje aplikovat datově řízenou transformaci dokumentu. To si lze představit jako vygenerování HTML tabulky z čísel nebo k vytvoření interaktivního SVG náhledu.

Tento framework se nesnaží nalézt funkcionalitu pro každý myslitelný problém. Spíše se snaží vyřešit jádro daného problému pomocí efektivní manipulace s prvky dokumentu na základě dat. To pak umožňuje dostatečnou flexibilitu a odemyká plný potenciál webových standardů. Tento framework má minimální režii a je dostatečně rychlý. Umožňuje práci s velkými datasety a vytvářet dynamické chování v podobě interakcí nebo animací. [45]

Při zkoumání ukázek bylo nalezeno několik velice zajímavých a dobře použitelných typů vizualizací. S jejich pomocí bude implementace ulehčena a rozšířena o zajímavé interaktivní prvky. To samozřejmě platí v případě, že

[10] Webové HTML stránky.

[11] Document Object Model, to znamená, že programy mohou přistupovat ke struktuře dokumentu a k jeho jednotlivým částem jako je například styl nebo obsah [46].

se využije tento framework. Obrázky 8.11 a 8.10 znázorňují různé využití této knihovny. Zdroj [2] dochází ke stejným závěrům, ty jsou vidět hlavně v kapitole závěr technologické rešerše. Samotný framework není příliš rozsáhle popsán.

8.3.4 Two.js

Tento framework je v [2] letmo zmíněn a nejsou u něj řečeny žádné případy použití či ukázky možných vizualizací. Oficiální stránky [49] obsahují na první pohled podobné množství informací. Hlavním rysem tohoto frameworku je dvou dimenzionální vykreslování jednotlivých objektů a to pomocí moderních webových technologií. Umožňuje také vykreslování několika kontextů zároveň v SVG, Canvas nebo WebGL. Zaměření tohoto frameworku je spíše na vektorovou grafiku a tvary. Tím pádem zjednodušuje vytváření animací pomocí těchto objektů. Pro reprezentaci jednotlivých částí využívá scénový graf. Po vytvoření objektu je možné na něj aplikovat jakoukoliv transformaci a tím upravit co je nutné. Velmi zajímavým prvkem je možnost propojit Two.js s jinou animační knihovnou. Vzhledem ke svému založení na SVG obsahuje příslušný interpretr a tím pádem je možné využívat výstupy z různých programů na vektorovou grafiku. Navzdory hlavnímu záběru v SVG je možné využít i bitmapové obrázky. [49]

Nutno podotknout, že u tohoto frameworku je velmi těžké nalézt příklady pokročilejších vizualizací a tím pádem je otázka, zda je dobré vůbec uvažovat o jeho využití. Nicméně nenalezení dobrých příkladů neznamená, že daný framework není možné použít.

8.3.5 Processing

Processing je samostatný nástroj, který je volně ke stažení a použití. Oficiální stránka nepopisuje příliš obecné možnosti. Nicméně z příkladů jsou vidět některé zajímavé případy použití jako třeba fraktály, L-systémy nebo celulární automaty. Příklady z oficiální stránky neukazují příliš hledanou funkcionalitu. Ukázka možností [51] však už naznačuje, že je samotný nástroj silnější než se může na první pohled zdát. Populární youtube kanál The Coding Train [52] tento nástroj využívá na většinu svých výtvorů, záznamy tohoto kanálu mohou být zdrojem mnoha užitečných informací. Zde by se jeden příklad dobrého využití programu processing hledal velice těžce, jelikož celý kanál má mnoho výborných ukázek, jak vytvořit různé algoritmy nebo vizualizace.

Seznam různých příkladů pro processing je vskutku velký, nicméně žádný se nezaměřuje vyloženě na vizualizaci časových dat. V příkladech je možné najít již zmíněné fraktály nebo L-systémy. Mimo to je možné vytvářet různé druhy pohledů, jak dvou tak tří dimenzionálních, manipulace s prvky, barevné manipulace, různé druhy ovládání a vstupu, matematické operátory, osvětlení nebo práce s daty. Samozřejmě stránka s příklady ukazuje mnohem více různých možností, ale ty by bylo zbytečné zde uvádět. Je ale zřejmé, že je tento nástroj velmi silný a s vhodnými kombinacemi prvků bude možné se dostat k zajímavým vizualizacím. [53]

8.3.6 p5.js

Opět se jedná o volně dostupný nástroj, po vzoru processing. To je zejména z důvodu, že tyto dva programy jsou od stejných autorů. Jedná se o nástroj, který umožňuje kreativní psaní kódu pro kohokoliv. Tento nástroj využívá metaforu pro skicy a tedy využívá plně kreslící funkcionality. Dále je na oficiální stránce uvedeno, že uživatel není limitován pouze na **canvas**[12], ale může využívat celou stránku jako skicu i s jednotlivými HTML5 objekty a dalšími prvky jako například text, vstup, video, kamera nebo zvuk. [54]

Podobně jako u processing i tento nástroj má mnoho možností využití. Opět je možné využívat různé struktury, typografie, tvary, interakce jak s uživatelem, tak s barvami, různé vstupní prvky, obrazy, pohyb nebo matematické objekty. Kromě toho je opět možné využívat různé druhy simulace, kdy jsou uvedeny celulární automaty nebo L-systémy a mnoho dalšího. Při průzkumu stránky s příklady bylo zjištěno, že pro tří dimenzionální vykreslování je využíváno WebGL, což dále prohlubuje možnosti použití tohoto nástroje. Obrázek 8.14 ukazuje možné využití p5.js, kde jsou vizualizována data kanálu The Coding Train. [56]

8.3.7 yWorks

Zdroj [2] uvádí jako další zkoumaný nástroj program yWorks. Jedná se o hotový produkt na vizualizaci dat různými způsoby. Umí využít několik přístupů pro zpracování, úpravu a následné zobrazení vizualizace. V příkladech vizualizací pomocí tohoto programu je mnoho zajímavých výsledků. Bohužel u žádného z nich není možné dostat se ke zdrojovým kódům, aby bylo možné zjistit, která část je automatická a která je napsána programátorem. Podpůrná

[12] Místo, kam je možné vykreslovat různé tvary.

videa k úvodu do tohoto programu také nebyla příliš nápomocná. Zdá se, že hlavní síla tohoto programu je v načtení a vytvoření vizualizace na základě dat, která mu byla zadána. Dále při průzkumu příkladů na oficiální stránce nebyl nalezen jediný příklad s časově orientovanými daty. Ve výsledku by se mohlo ukázat, že to je velmi složitý úkol. Obrázek 8.15 ukazuje možnou vizualizaci dat se síťovým tokem při použití yWorks. [58]

8.4 Závěr

Mezi zmíněnými knihovnami bude nutné vybrat jednu, která nejlépe odpovídá požadavkům a je dostatečně flexibilní. Knihovna PixiJS je využita ve verzi vizualizace [2]. Vzhledem k využívání WebGL nejsou některé části příliš dobré. Například využití SVG formátu by mohlo být lepší. Jinak obsahuje vše potřebné pro tvorbu vizualizací. Vzhledem k použití v práci [2] bude lepší ji využít spíše v momentě, kdy ostatní knihovny selžou. V podobném duchu pak pracuje knihovna Phaser, která je založena na PixiJS a tedy má velmi podobnou funkcionalitu, ale využívá se spíše ke tvorbě her než vizualizací. Phaser je nejméně vhodnou volbou pro vytvoření vizualizace.

Nástroj yWorks je z ukázek použitelný na mnoho vizualizací, ale spíše se jeví jako hotový produkt, který umožňuje vytvářet různé vizualizace. Ty by dle jednotlivých ukázek mělo být možné dále upravovat nebo rozšiřovat pomocí kódu, ale nebylo nalezeno dost informací podporujících tuto informaci. Jak již bylo řečeno jedná se o hotový nástroj a ten nemusí obsahovat vše, co bude potřebné. Tedy jeho použití by bylo možné, ale nebude použit.

Two.js ze svých příkladů nevypadá příliš použitelně a nalézt alespoň nějaké vizualizace bylo opravdu těžké. Z toho vyplývá, že tato knihovna není jako vizualizační nástroj hojně využívána. Proto nebude ani vhodné ji využít pro vizualizaci dat Asterionu.

D3.js je rozsáhlá knihovna se spoustou ukázek a její použití by bylo optimální. K dispozici jsou různé ukázky i tutoriály a tím pádem je možné v případě stagnace vyhledat pomoc online bez většího strachu. Navíc je i rovnou zmíněno, že umožňuje využívat SVG a další zajímavé prvky.

Poslední dvě neshrnuté knihovny jsou Processing a p5.js. Oba nástroje budou mít stejnou možnost použití jako D3.js. Nicméně ze čtených zdrojů není zcela zřejmé, jak umí pracovat s vektorovou grafikou. Na druhou stranu existence kanálu The Coding Train je jeden z důvodů, proč bylo zvažováno

jejich použití. Celý kanál obsahuje velké množství videí a určitě bude možné nalézt řešení různých problémů které by mohly v průběhu vývoje nastat.

Tedy nejlépe se jeví poslední tři knihovny/nástroje D3.js, Processing a p5.js. Z těchto tří byla vybrána D3.js. Tato knihovna bude využita k vytvoření prvotních prototypů i dalších verzí vizualizace. Nicméně během implementace může nastat problém s funkcionalitou a bude nutné změnit knihovnu/nástroj.

```css
#footer{
  color:white;
}

#footer li{
  list-style: none;
  padding: 0rem 1rem;
}

#nav ul, .podkategorie ul{
  text-align: center;
  margin: 0rem;
  font-size: 0rem;
}

#nav li{
  width: 10rem;
  line-height: 3rem;
  display: inline-block;
  font-size: 1.25rem;
}

#nav li a{
  text-decoration: none;
  color: white;
  display: block;
}
```

```css
.podkategorie li {
  display: inline-block;
  color: gray;
  line-height: 2rem;
  width: 9rem;
}

.podkategorie li a{
  color: black;
  text-decoration: none;
  display: block;
}

.button{
  text-align: center;
  border: 0.1rem solid rgba(0,0,0, 0.19);
  width: 10rem;
  line-height: 3rem;
  background-color: rgb(255,255,255);
  color: rgba(0,0,0, 0.4);
  margin-bottom: 2rem;
  box-shadow: 0.2rem 0.2rem 0.2rem gray;
  margin-top: 1rem;
}

.button a{
  color: black;
  text-decoration: none;
  display: block;
}
```

```css
  .prebal{
    display: none;
  }

  #drobecky ol{
    margin-left: 0rem;
    padding-left: 1rem;
  }
}

@media print{
  .button {
    display: none;
  }

  .kariera_polozka dl:after{
    clear: both;
    content: "";
    display: block;
  }

  .prebal img, .kariera_polozka dl{
    page-break-inside: avoid;
  }
}
```

```css
ul{
  padding: 0rem 1rem;
  text-align: center;
}

.polozka{
  padding-left: 0.8rem;
  padding-bottom: 0.2rem;
  border-left: 0.14rem solid rgb(176,23,181);
  background-color: white;
  margin-bottom: 0.2rem;
}

.aktivni{
  background-color: rgb(176,23,181);
  color: white !important;
}

.kariera_polozka{
 padding-left: 1rem;
 padding-bottom: 0.2rem;
 border-left: 0.14rem solid black;
 margin: 0rem auto;
  background-color: white;
}
```

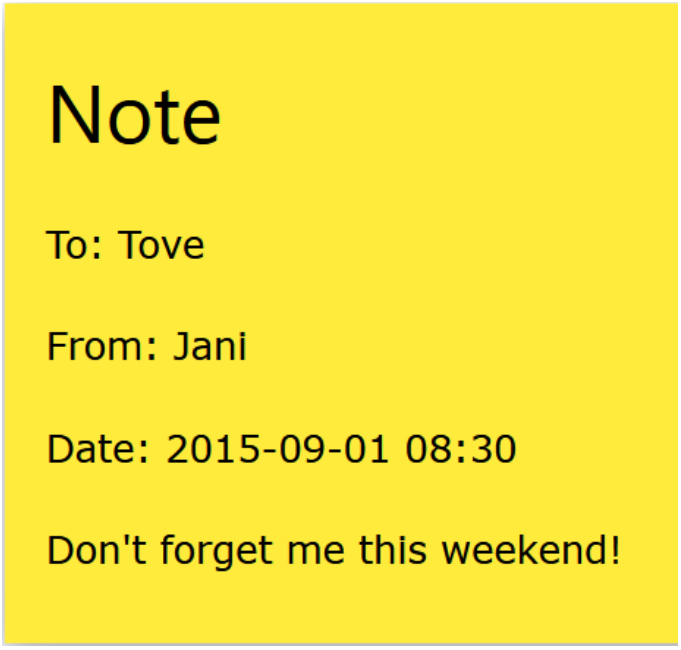
Obrázek 8.4: Ukázka stylů pro stránku 8.1

```
function changeText() {
    let headerTitle = document.getElementById('hello');
    headerTitle.textContent = 'Hello you ';
}
```

Obrázek 8.5: Ukázka jednoduché funkcionality realizované pomocí JavaScriptu

```
<note>
  <date>2015-09-01</date>
  <hour>08:30</hour>
  <to>Tove</to>
  <from>Jani</from>
  <body>Don't forget me this weekend!</body>
</note>
```

Obrázek 8.6: Ukázka prostého XML souboru bez interpretace [34]

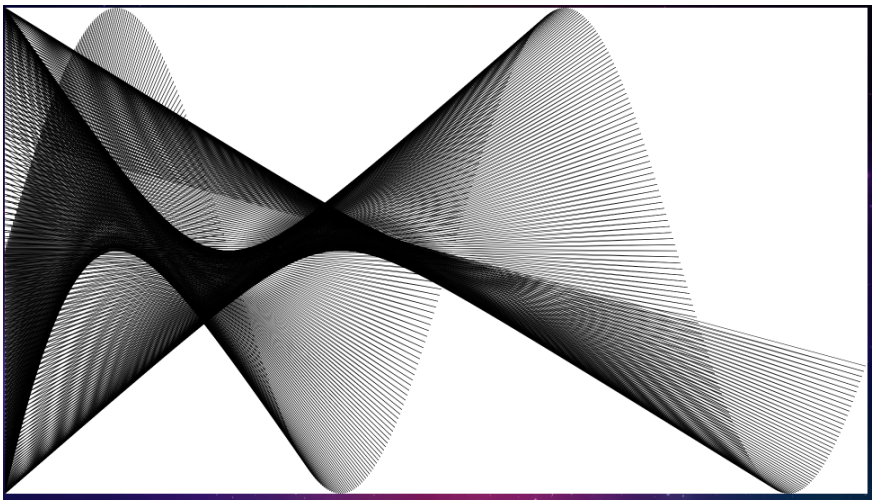


Obrázek 8.7: Interpretované XML z obrázku 8.6 [34]

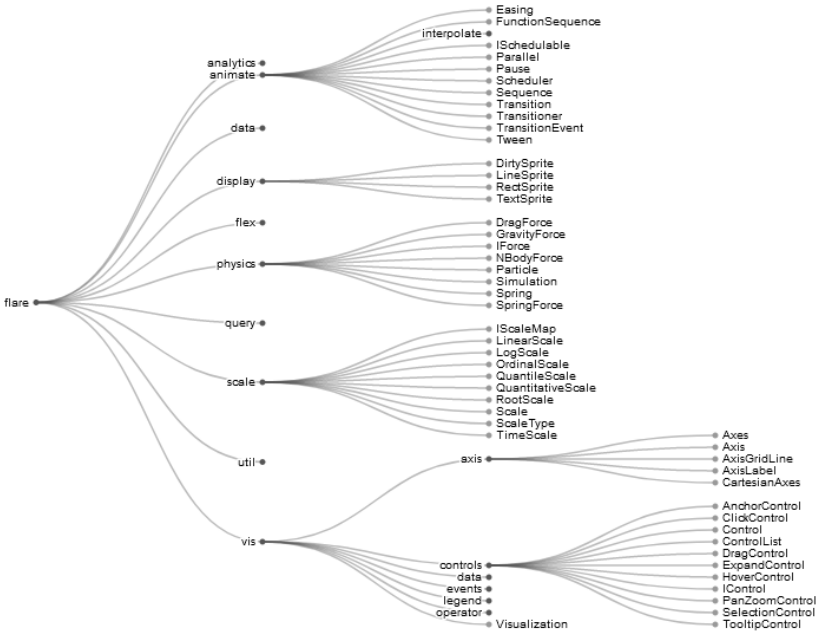
```
<!DOCTYPE html>
<html>
<body>

<h1>My first SVG</h1>

<svg width="100" height="100">
   <circle cx="50" cy="50" r="40" stroke="green" stroke-
width="4" fill="yellow" />
   Sorry, your browser does not support inline SVG.
</svg>

</body>
</html>
```

My first SVG

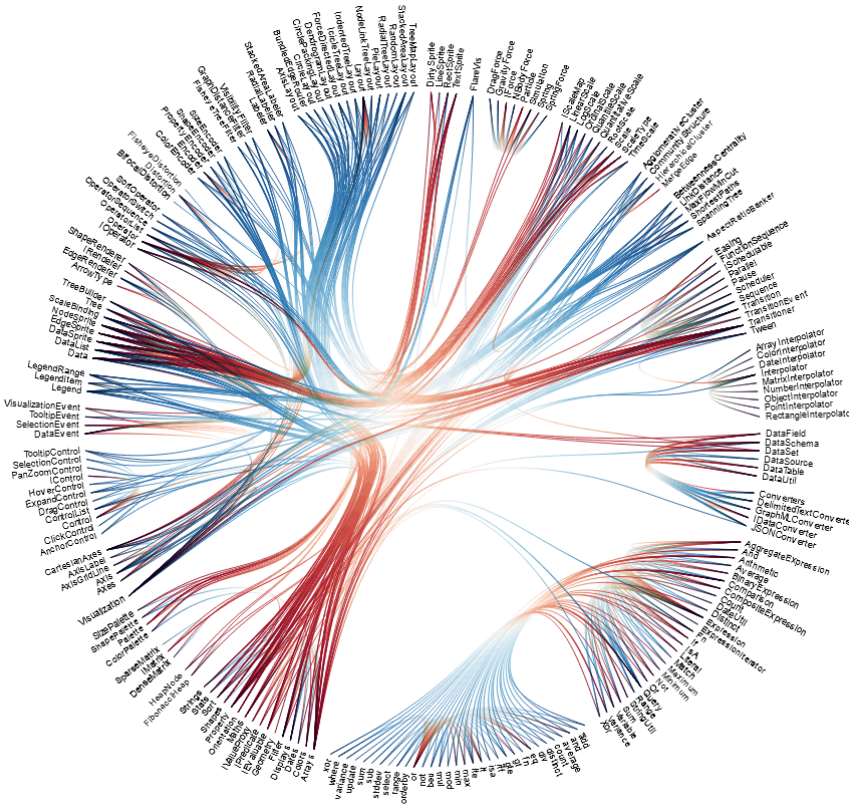
Obrázek 8.8: Ukázka textové a grafické reprezentace vektorové grafiky [36]



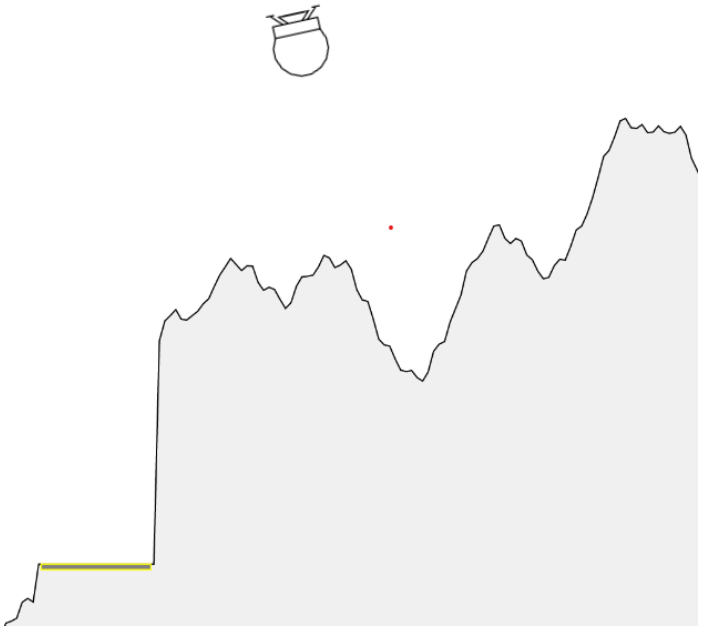
Obrázek 8.9: Ukázka možného použití frameworku Phaser [42]



Obrázek 8.10: Ukázka možné vizualizace sbalitelného stromu [47]

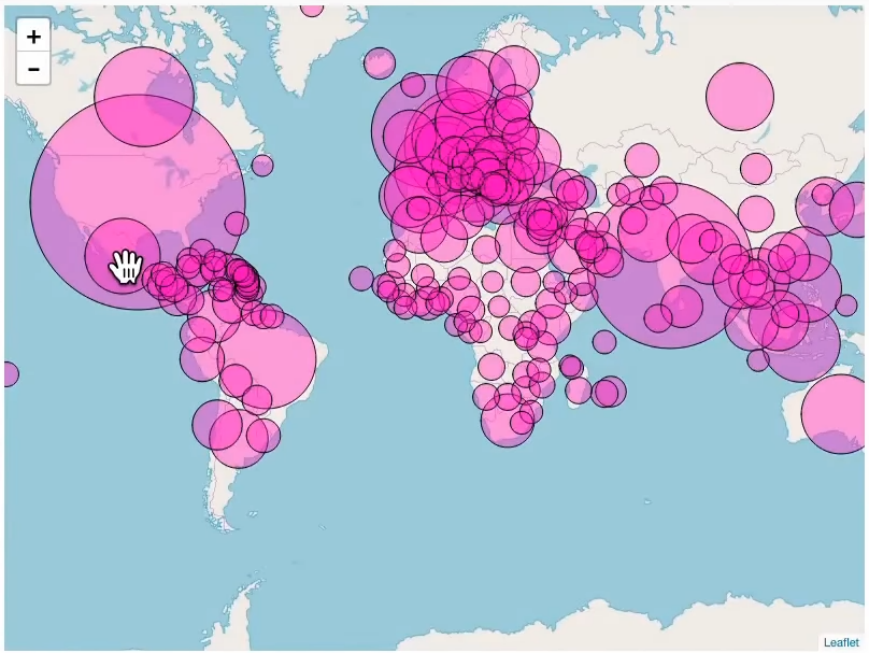


Obrázek 8.11: Hierarchický svazování hran [48]

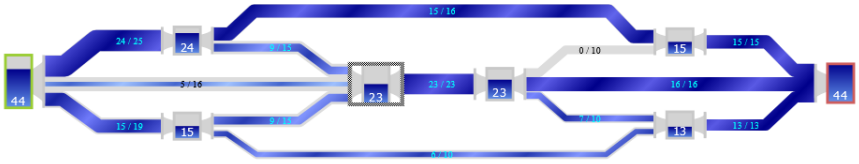


Obrázek 8.12: Možný interaktivní výstup s použitím knihovny Two.js [50]

Obrázek 8.13: Ukázka vizualizace dat s programem Processing [51]



Obrázek 8.14: Ukázka vizualizace předplatitelů kanálu The Coding Train s programem p5.js [55]



Obrázek 8.15: Ukázka výstupu programu yWroks, vizualizace maximalizace toku v síti [57]

Kapitola 9

Návrh

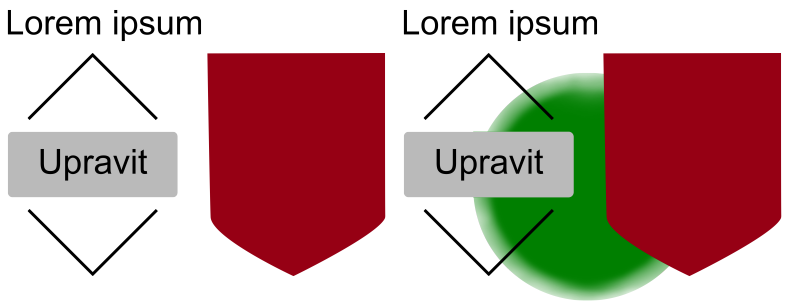
V této kapitole bude naznačen přístup ke všem dílčím částem, které je nutné vyřešit, případně další návrhy pro rozšíření. Postupně budou rozebírány jednotlivé části stávající vizualizace. Ty budou následně upraveny dle literatury tak, aby měli co nejlepší vypovídající hodnotu. Tato kapitola bude tedy rozdělena do několika částí a to ovládací prvky, značení událostí a samotné linky.

9.1 Ovládací prvky

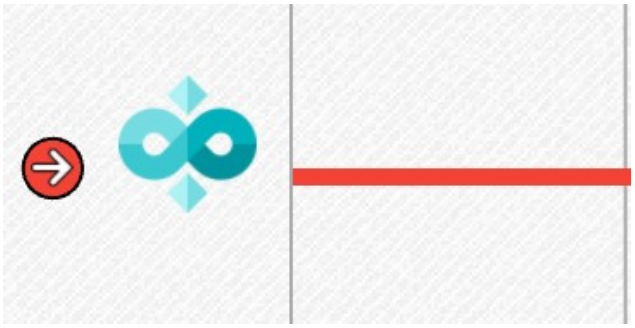
V současné verzi[1] vizualizace zabírá větší část obrazovky právě ovládání celé vizualizace, jak je vidět na obrázku 7.5. Vzhledem k velkým vzdálenostem mezi osami a ovládáním hrozí, že uživatel netrefí tlačítko, které chce využít, nebo v průběhu přesunu kurzoru zapomene jaký je jeho cíl. Některé tyto části by bylo možné přesunout přímo k jednotlivým osám. S touto změnou se pak redukuje možnost zapomenutí cíle.

Pokud bude chtít uživatel posunout nejspodnější časovou osu, bude muset cestovat s myší po obrazovce velké úseky. Tuto část je možné vyřešit přesunutím funkcionality posunu na jiné místo. Vedle označení osy se nachází pouze šipka aktuálního výběru, jak je vidět na obrázku 9.2. Ta se dá změnit na podbarvení dané položky, nebo jiné označení. Poté místo šipky výběru

[1] Verze z bakalářské práce [2], typicky se bude mluvit o současné nebo aktuální verzi.



Obrázek 9.1: Tyto dva obrázky reprezentují navrhovaný vzhled zobrazovaných tagů. V levé části je možné vidět rag bez výběru a v právé části pak s výběrem.

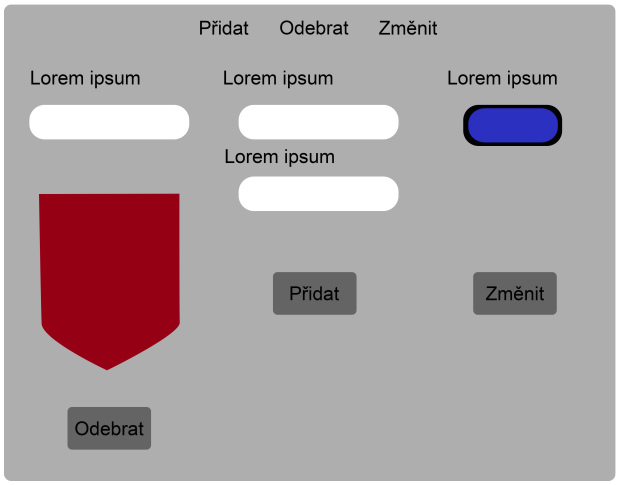


Obrázek 9.2: Aktuální podoba výběru.

přidat šipky nahoru a dolů, které umožní posun zvýrazněné osy. Tento návrh je vidět na obrázku 9.1.

Mimo to je možné dané osy měnit, přidávat nebo ubírat. Tyto funkcionality jsou opět přítomny ve vrchní části, kterou je možné využít jinak. Tlačítko pro změnu vybrané osy může být vloženo právě mezi šipky posunu. Tato část je vidět na obrázku 9.1. V obou obrázcích je pouze jedno tlačítko, to je v této části pouze jako nástin, výsledná podoba se bude lišit. Co se pak týká přidání/odebrání osy. Tuto funkcionalitu bude vhodné přidat do kontextového menu, které by mělo několik položek. Například na změnu barvy, změnu významu osy, přidání nové osy na místo aktuálně vybrané osy nebo na konec. Případně na odebrání vybrané osy, všech os nebo nějakého výběru. Kontextové menu je viditelné na obrázku 9.3.

V tento moment celková vizualizace vypadá tak jako na obrázku 9.4. Je vidět čistější vrchní část, ve které bude možné přidat nějaký smysluplný prvek jako například kontext ke studované části. Co se týká posuvníku s přehledem o časové ose, tato část je naprosto vyhovující a dobře využitá. Zanesení principu přehled a detail (focus&context) může uživateli hodně pomoci při hledání. Pravděpodobně bude vhodné provést menší rozšíření a přidat například desetiletí na dané ose tak, aby se mohl uživatel rychleji zorientovat jaký rok přesně hledá a kam se musí posunout.

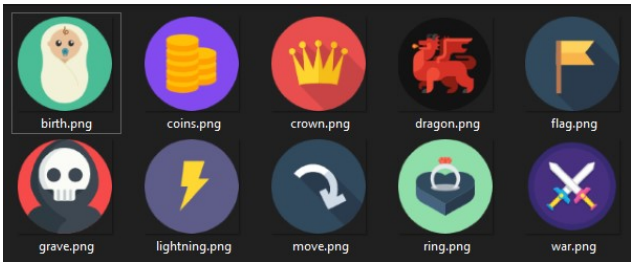


Obrázek 9.3: Navrhovaný vzhled kontextového/modálního menu.

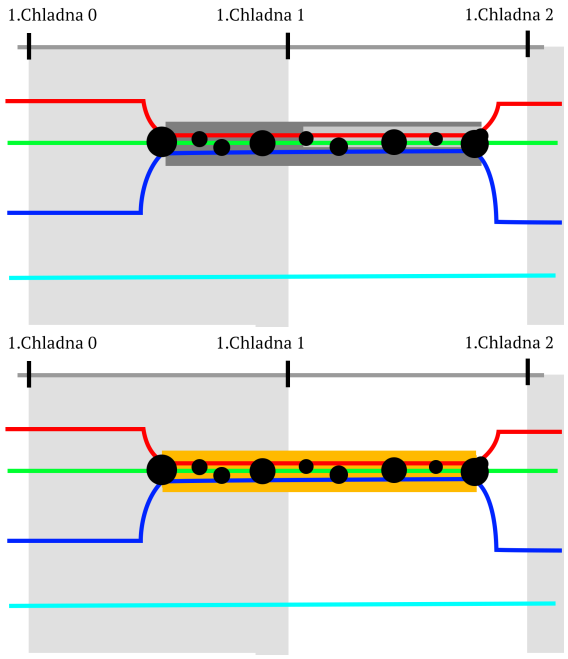
Obrázek 9.4: Přepracovaná hlavička vizualizace. Ve vrchní části jsou vidět tlačítka, kde první zobrazí kontextové menu 9.3. Zbývající tři jsou odkazy na nápovědu a na externí stránky.

Část, která naprosto chybí v aktuální verzi vizualizace je ovládání pomocí klávesnice a klávesových zkratek. V momentě, kdy se některý z uživatelů naučí velmi dobře ovládat navrhovanou vizualizaci, pak nastane problém efektivity práce. Neustále klikání myší, hledání v kontextovém menu atd. může vést k neefektivní práci. Proto je dobré využít klávesových zkratek pro jednotlivé akce. Jako příklad lze uvést jednoduché posouvání aktuálního výběru. K tomuto úkonu bude nutné využít tlačítek šipky nahoru nebo dolu, které posunou výběr o jednu pozici. Možnost výběru je vidět na obrázku 9.1 vpravo. Pro posun v čase bude vhodné zavést podobnou funkcionalitu a to s pomocí šipek vlevo a vpravo, které by posouvali celou vizualizace v daných směrech o rozumně velký krok. To samé může platit pro přidávání nebo ubírání os například pomocí klávesových zkratek Ctrl + a, Ctrl + d. Pro změnu osy pak například Ctrl + e. Je vidět, že po přidání těchto zkratek se zmenší čas potřebný pro vykonání některého z úkonů myší a umožní hladší práci s vizualizací.

Po přidání ovládání klávesami také vyvstává otázka, jak pracovat s výběrem a možnosti pohybu mezi osami pomocí šipek. Návrh řešení pro tento problém je vcelku jednoduchý. V momentě, kdy bude chtít uživatel vybrat některou z os a pracovat s ní, tuto osu kliknutím myši vybere, případně se k ní dostat pohybem výběru. V případě potřeby posunutí vybrané osy bude k dispozici



Obrázek 9.5: Ukázka všech typů událostí, které aktuálně jsou k dispozici

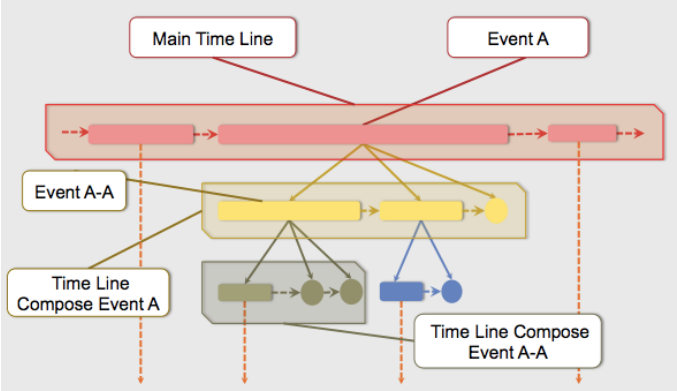


Obrázek 9.6: Obrázek nahoře znázorňuje možnou reprezentaci války, ve které vzniklo další společenství. Obrázek dole pak spojenectví.

další zkratka. Pro prohození os stačí využít klávesové zkratky Shift + šipka nahoru, která prohodí aktuální osu a osu, která je nad ní. Pro šipku dolů je zkratka analogická, Shift + šipka dolů a prohození os směrem dolů.

9.2 Značení událostí

Jednotlivé události v aktuální verzi jsou značeny různým způsobem. Obrázek 9.5 znázorňuje většinu kategorií, které jsou aktuálně k dispozici v databázi. Tyto ikonky typicky znázorňují jednoduché události, které mají typickou dobu trvání jednoho dne. Tedy staly se v zadaný den. Zmíněné ikonky kategorií je ovšem možné změnit a případně doplnit. To však není hlavním cílem této



Obrázek 9.7: Ukázka hierarchické události. [59]

práce a proto bude ponechána na doplnění při zbývajícím čase nebo pro další rozšíření. Události jako takové budou převzaty beze změny. Pouze bude přidána funkcionalita pro déletrvající události.

Dále zde budou uvedena některá značení, která aktuálně chybí. Označení pro dlouhotrvající události není nikde zmíněno. Respektive je zavedeno velmi zvláštním způsobem, který bude vhodné změnit. V databázi jsou uložený záznamy o různých bitvách a válkách, avšak vše jako jednoduchá událost. Z lidských dějin je známo, že války jsou typicky dlouhotrvající a mají v sobě mnoho dalších kritických událostí. U bitev pak dává smysl jednoduchá událost, ale u konfliktů[2] nikoliv. V aktuální verzi jsou tato data spojena tzv. párovými tagy, kde jeden značí začátek a druhý konec. Jednodušší a snáze zpracovatelný způsob je následující. Místo dvou událostí jedna, která má začátek a trvání. Takový typ událostí může být označen jiným způsobem, který budou doprovázet i další znaky, jako například podbarvení. Symbolický nástin dvou různých situací je možné vidět na obrázku 9.6

Další chybějícím prvkem je značení hierarchických událostí, které vzniknou činností různých obchodních, politických nebo jiných spolků. Tyto skupiny mohou mít vliv na kritické události a vyvolávat je. Tyto události je možné si představit jako několik os, které se všechny týkají jednoho období, jen každá může mít jinou granularitu. Ukázka je vidět na obrázku 9.7. Kde je několik menších os s dalšími informacemi a událostmi delšího trvání. Co ovšem nechybí a je očekávaným prvkem je možnost přečíst si detail o zobrazované události. Kde po interakcí s vybranou událostí se zobrazí její detaily. Tuto funkcionalitu bude dobré přenést i dále. Značení událostí bude zachováno, protože se jeví jako dobře zvolené, ale bude třeba doplnit některé části zmíněné výše.

[2]Konfliktem je myšleno delší období bitev a střetů.

9.3 Časové osy

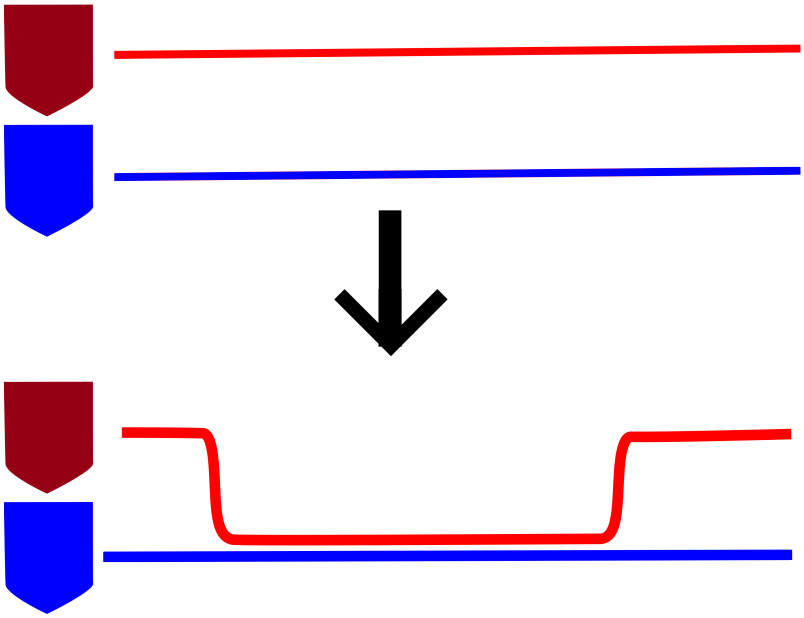
Pro začátek je dobré popsat vzhled časových os v aktuální verzi. Ty jsou reprezentovány s pomocí nekonečných časových os pro všechny typy tagů[3]. Takto zvolená reprezentace je minimalistická a tím pádem vhodná, pro připomenutí je dobré podívat se na obrázek 7.6. Menší problém pak nastává s tagy typu postava, předmět nebo zvíře, protože ty mají zpravidla omezenou životnost, tím pádem není nekonečná osa přesný popis.

S časovými osami souvisí dříve zmíněné události a jak je vidět na obrázku 7.6, události jsou vynášeny na tyto osy. Některé události souvisí s více osami a to je zde naznačeno pomocí spojnic. V některých případech to nemusí být šťastné řešení, zejména při větším počtu os a spojnic. Na již zmíněném obrázku jsou vidět tři osy, kde druhá spojnice zprava spojuje první a poslední osu. Již u první osy je problém se shlukem událostí, kde není jasné k čemu přesně spojnice patří, ale to vyřeší zoom. Trošku závažnější je pak situace, kdy spojnice protíná prostřední osu a není jasné, zda událost, která je velmi blízko spojnice s ní souvisí. V horším případě bude událost přesně na průsečíku a vůbec nemusí souviset s danou spojnicí.

Tento problém má řešení v podobě lepšího pořadí os tak, aby spojnice co nejméně protínali ostatní osy. Toho lze docílit manuálně, ale uživatelé nebudou chtít trávit svůj volný čas hledáním optimálního pořadí os. Spíše budou chtít vidět jistým způsobem přehlednou vizualizaci ze které bude snadné číst. Proto bude vhodné vytvořit metody pro výpočet optimálního pořadí os. Tyto metody určí na základě svého výpočtu optimální rozestavení os pro minimalizaci křížení a toto pořadí je dále předáno k vyobrazení. Tedy uživatel si pouze vybere, co chce vidět, a zobrazí se mu výsledek s optimálním rozložením.

I samotné osy bude vhodné upravit. Místo nekonečných přímek bude použit jiný prvek a to cesty. Nejjednodušší cesta je definována počátečním a koncovým bodem, což odpovídá aktuální verzi. Zde bude využito více bodů a to ve stylu přesměrování os k souvisejícím osám. To je vidět na obrázku 9.8. Toto přesměrování nastane právě v místech, kde by jinak byli spojnice. V ideálním případě nedojde k žádnému křížení, protože se osy budou rozestavovat v optimálním pořadí. Tímto způsobem bude možné zanést déletrvající události do databáze i jejich zobrazení. S touto úpravou nebudou potřeba spojnice a veškeré spolu související informace budou hned u sebe. Obrázek 9.6 ukazuje pokročilý návrh situace, kdy je vedena válka a v ní se

[3]Pro připomenutí, tagem je myšleno např.: království, spolek, významná postava, předmět atd.



Obrázek 9.8: Ukázka změněného vzhledu časových os

dějí další déle trvající události.

Kromě toho budou osy dále upraveny tak, aby dávaly smysl. Tím je myšleno, že království musela vzniknout z některého jiného královského celku a tím pádem v něm bude mít začátek své cesty, případně se sloučí s jiným královstvím a tím zanikne nebo vznikne zcela nové království. Tyto části bude dobré podchytit, ale nebude to lehký úkol. Vzhledem k tomu, že zatím nejsou rozlišená území a království jako taková. Kromě toho nedává nekonečná osa smysl u významných osobností nebo zvířat, tyto osy budou mít zaručeně začátek a konec v narození a úmrtí daných postav, pokud k takové události může vůbec dojít[4]. Jediné místo, kde bude trochu problém aplikovat minimalizaci křížení, je právě sledování postav. Ty se typicky pohybují po světě a mohou navštívit různá území, účastnit se různých akcí. Zde bude vznikat trochu více křížení, ale vzhledem k povaze řešeného problému je to očekávaný výsledek.

V aktuální verzi se postupně přidávaným osám přiřazují různé barvy tak, aby vždy bylo jasné co je co. V případě spojnic jsou pak využity obě barvy souvisejících os. Vynášeným událostem jsou přiřazovány značky v mini mapě. Ty mají barvu časové ose ke které patří. Tato funkcionalita je velmi dobře zpracována bude přenesena i do nově vytvářené verze. Ale odpadne nutnost dvojího barvení spojnic.

[4]Některé rasy jsou dlouhověké až nesmrtelné.

U takto změněných os je pak nutné najít způsob, jak spočítat jejich vhodné uspořádání. K tomu bude vhodné vytvořit metody pro řešení pořadí. Vytvořené metody jsou následující **hrubá síla**, **frekvenční tabulky** a **silové metody**.

9.3.1 Hrubá síla

Tato nejjednodušší metoda pouze vezme časové osy a spočítá všechny možnosti rozestavení zadaných os. U každé si bude pamatovat ke kolika křížením došlo a tato informace bude následně využita pro výběr pořadí os. Jak je zřejmé, složitost této metody se odvíjí od počtu os se kterými pracuje. Platí zde, že čím více os, tím více možností, jak tyto osy seřadit a tím více času na výpočet.

9.3.2 Frekvenční tabulky

Další metodou pro určení vhodného pořadí jsou frekvenční tabulky. Ty obecně pomáhají analyzovat získaná data a následně o nich sdělit určitá zjištění. Jednoduše řečeno, frekvenční tabulky ukazují četnost výskytu dat v množině. Umožňuje nalézt vzorce v datech a také provést statistická měření v datech. Tato metoda tedy pomáhá pouze zorganizovat data. Další úkony jako například statistické testy nebo analýzy jsou až pozdějším krokem. Jak přibližně může vypadat frekvenční tabulka, je ukázáno na obrázku 9.9.

Využití této metody pro časové osy je následující. Po získání všech aktuálně požadovaných dat se vezmou jednotlivé osy a propočítá se vzájemná četnost událostí mezi osami. Následně se vezme osa s nejvíce společnými událostmi vůči ostatním osám. Okolo této osy se následně budou umísťovat ostatní osy v závislosti na počtu společných událostí s ostatními osami. Nejlépe i s ohledem na to, kam byly umístěny osy sousedící s krajními, aby nedošlo k překřížení celé vizualizace. Jinak řečeno, bude nutné vyzkoušet několik možností umístění os na stejné úrovni[5] a vybrat z nich tu nejlepší variantu.

Tento výpočet bude pravděpodobně také časově náročný. Frekvenční tabulky jako takové budou vyplněné rychle, nicméně druhý krok je časově náročnější. Os na stejné úrovni může být více a tím pádem je více možností, kde by mohlo dojít ke snížení počtu křížení. Nová osa se jednoduše umístí na

[5] Úrovní je myšlena spočtená frekvence.

	A	B	C	D	E	F	Celk.
A	xxxxxxxx	4	6	0	6	0	16
B	4	xxxxxxxx	3	0	4	0	11
C	6	3	xxxxxxx	0	3	0	12
D	0	0	0	xxxxxxxx	0	0	0
E	6	4	3	0	xxxxxxxx	0	13
F	0	0	0	0	0	xxxxxxxx	0

Obrázek 9.9: Ukázka možné frekvenční tabulky, pro několik vybraných os.

horní nebo dolní konec vizualizace. Spočítá se množství křížení a porovná se s aktuálním počtem. Pokud se nic nezmění, tak se pokračuje s další osou, při změně se vyzkouší prohazování nové osy s předchozími osami stejné úrovně. Zde se pak hledá pořadí, které bude mít nejmenší počet překřížení a takové pořadí se využije pro další iteraci.

9.3.3 Silové modely

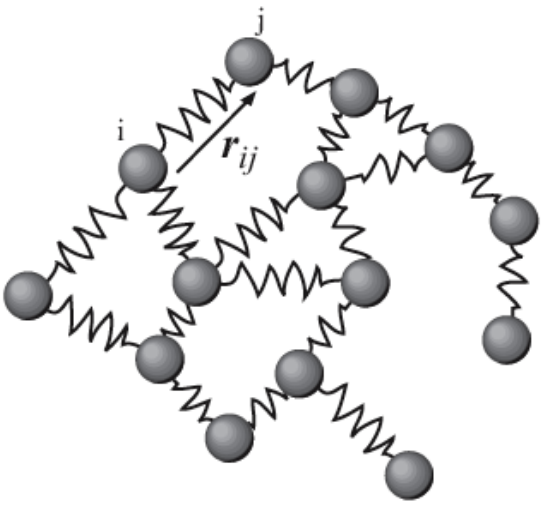
Další možností, jak získat pořadí os je využít silových modelů[6]. Při použití této metody bude nutné využít simulace os jako bodů. Ty spolu budou provázány vzájemnými vztahy, jinak řečeno událostmi, které jsou zastoupeny v obou osách. Takto vytvořené vztahy udávají síly, kterými se jednotlivé osy přitahují/odpuzují. Po ustálení budou mít jednotlivé body své určené pozice. Pro určení pořadí os bude stačit využít souřadnici y. Podle ní pak seřadit aktuální osy a takové pořadí předat na další zpracování. Ilustrativní obrázek pro připomenutí pružinových modelů je možné vidět na obrázku 9.10.

Tuto část velmi usnadní využívaná knihovna D3.js, která tyto simulace umožňuje provádět. Je však otázkou do jaké míry bude tato knihovní část použitelná.

9.4 Struktura webové aplikace

V této části bude dále rozebrána struktura vytvářené aplikace. Ta se bude skládat z několika málo tříd, které budou zajišťovat různé funkcionality. Mezi ty patří uchování aktuálního stavu aplikace, komunikace se serverem nebo výpočet vhodného pořadí os pomocí již dříve zmíněných metod.

[6] Také označované jako pružinové modely.



Obrázek 9.10: Připomínka vzhledu pružinových modelů. [60]

9.4.1 Kontext

Tato třída bude v sobě uchovávat veškerá data, která zatím byla získána ze serveru. Mimo to bude také využívat ostatní třídy pro získání dat ze serveru,k výpočtu pořadí os či celkové výstavbě. Na data se bude celá aplikace doptávat postupně v momentě, kdy budou potřebné. Pro následující urychlení se data nebudou zapomínat, protože opětovné dotazy na server mohou být velmi pomalé obzvlášť, pokud má uživatel slabé internetové připojení nebo nedostatečné pokrytí ve svém okolí.

Třída kontext bude tedy obsahovat pole objektů, které bude držet veškerá data. Mimo to bude také obsahovat seznam aktuálně zobrazovaných os, cestu k obrazovým zdrojům, které se využívají pro naznačení o jakou událost na časové ose se jedná. Limity pro různé části aplikace, jako například aktuální výběr, odkazy na jednotlivé části vizualizace, konstanty pro přidávání a změnu obsahu, případně další potřebné proměnné, které se mohou objevit v průběhu vývoje.

9.4.2 Komunikace

Třída určená pro komunikaci bude zajišťovat přenos informací mezi serverem a kontextem. Zároveň bude umožňovat předzpracovat a vytřídit potřebné

části pro zobrazení ve webové aplikaci. Pomocí této třídy se aplikace doptá serveru na informace, přeloží a uloží si odpověď pro další zpracování a vytřídí jen to, co je třeba. Tím se opět ušetří nějaký čas, protože z uložené odpovědi nebude nutné se znovu doptávat serveru na informace. Ovšem nastane problém v momentě, kdy se bude uživatel dožadovat čerstvých informací, ale aplikace mu již běží delší dobu v prohlížeči, pak bude dobré umožnit smazání uložených informací a opět je získat ze serveru. Což by se také dalo vyřešit pravidelným promazáváním uložených informací a jejich opětovné načítání.

Jediné části, které bude mít tato třída jsou nezbytné adresy pro získání informací a jejich případné uložení v kontextu nebo u sebe. Pro vyplnění příslušných polí ve vizualizaci (při výběru zobrazovaného obsahu) bude využita třída pro výstavbu. Ta bude také obsahovat metody na aktualizaci obsahu.

9.4.3 Pořadí os

Pořadí os je jeden z největších a nejtěžších problémů, které bude nutné vyřešit. Toto pořadí bude s pomocí dříve zmíněných metod řešit tato třída. Kromě implementace těchto metod bude také obsahovat nutné proměnné pro usnadnění mezivýpočtů případně udržení si aktuálně nejlepší konfigurace nebo vytvoření pohledu na aktuálně nejlepší konfiguraci.

9.4.4 Výstavba

Tato třída bude obsahovat veškeré metody pro výstavbu základní stránky, ale i pro aktualizace jednotlivých částí. Kromě základní části bude také obsahovat informace nutné pro výstavbu kontextové nabídky na úpravy. Pokud to bude třeba, pak tato třída bude také obsahovat odkazy na jednotlivé části celé vizualizace. Dále také nástroje na správné mapování zobrazovaného obsahu (propojení mini mapy a plátna). Metody této třídy budou využity pro doplňování nových informací do vizualizace a i jejich aktualizace.

9.4.5 Interakce

Velkou částí samotné vizualizace je i jednoduchost ovládání a změn. Proto se veškeré ovládání uloží v jedné třídě. Z té se po výstavbě budou vybírat

jednotlivé metody, které budou navázány na tlačítka, ikony a jiné části, se kterými bude možné interagovat. Mimo to bude tato třída také odpovědná za ovládání z klávesnice pro všechny části vizualizace.

9.5 Organizace dat

Data v jednotlivých částech je nutné uspořádat do přehledného formátu, pokud se toho podaří docílit, pak bude snadné provádět různé operace. Pozornost si zejména zaslouží dvě části kontext a data ze serveru. Tyto dvě položky jsou spolu do určité míry propojeny. Data uchovávaná v kontextu jsou více detailní než ta u serverové komunikace. V kontextu jsou uchována data, která již byla někdy zobrazena, u komunikace je pak jejich menší podoba, která slouží k rychlému vyhledání aktuálně zadávanému textu.

Kontext bude rozebrán za chvíli, nejdříve zde bude rozebrána podoba uložení dat v komunikaci. Jak již bylo zmíněno výše, toto úložiště slouží ke zrychlení při vyhledávání. Tedy klíčem zde bude text (název) dané události nebo osy. Jako klíč bude zvolen identifikátor dané události/osy. Tím pádem musí existovat další tabulka, která spojuje identifikátor a událost/osu. Tím dochází k dalšímu hledání a porovnávání. Kromě uložených dat z hledání v různých kategoriích bude dobré využívat i kontextu. Pro urychlení a případné přeskakování delších dotazů na server bude komunikační třída nahlížet do již získaných a uložených dat v kontextu.

Informace připravené v kontextu budou mít složitější podobnu, než ta uložená v komunikaci se serverem. Identifikátorem každého záznamu bude opět text, kde nejlepší volbou bude název. Každá položka pak bude mít zapsán svůj identifikátor a cestu ke své ikoně. Mimo to také dva další seznamy – společné události, kde bude seznam identifikátorů daných událostí a osy se kterou souvisí. Druhý seznam pak obsahuje výčet všech událostí, které jsou opět rozlišeny pomocí identifikátoru, dále obsahují název, popis, počátek a konec dané události, cestu k ikonce, která danou událost reprezentuje a filtry, které v sobě nesou další informace. Tyto dodatečné informace bude možné použít při vyhledávání určitých položek v seznamu.

9.6 Závěr

V této kapitole byl popsán návrh se všemi plánovanými detaily. Je ovšem možné, že se jednotlivé části budou ve finálním produktu lišit kvůli zpočátku neviděným chybám a detailům.

Kapitola 10

Implementace

Tato kapitola obsahuje veškeré implementační detaily z nově vytvořeného frontendu vizualizace. Témata rozebírána v kapitole 9 budou znovu otevřena a okomentována. Tím je myšleno, zda nastaly změny oproti původnímu návrhu, či nikoliv. V závěru pak budou zmíněna rozšíření, která budou postupně dodávána do vizualizace tak, aby bylo zpříjemněno ovládání, rozšiřování nebo úpravy, a to jak na úrovni kódu, tak obsahové.

10.1 Organizace dat

Styl ukládání dat popsán v 9.5 byl použit ve statických návrzích. V pozdějších verzích však tento návrh nebyl zcela uskutečnitelný z různých technologických důvodů. Proto byl tento aspekt návrhu několikrát přepracován až dospěl k finální podobě. Třída zajišťující komunikaci se serverem má uložen pouze výběr, kterým je možné určit, která data zajímají uživatele. Zbytek dat je pak uložen na jednom místě, a to ve třídě kontext. Do ní se ukládají informace o již dotazovaných pojmech.

Třída kontext je přístupná všem ostatním třídám, které z ní získávají data nebo je do ní naopak ukládají. Třída samotná pak nastavuje své vnitřní proměnné jako je mapování, rozsah vizualizace (minimální a maximální datum), proměnné pro ovládání zoom a brush. Jednou z nedílných součástí je barevná paleta. Ta obsahuje 30 barev, které jsou dle nástroje [61] dostatečně rozlišitelné pro zdravé oko. Mimo to obsahuje metody, které pomáhají získávat

```
export default class Context {
// web url
url = ...;
// Array containing all the data fetched from the server
data = [];
// Data showed at the moment in the viz
active = [];
// Look up table for easier timelines switching
activeOrder = [];
// Min and Max dates
minDate = 0;
maxDate = 0;
// Constants for content adjustement
lowerEnd = -100;
upperEnd = 50;
// Color pallete represenatation
colorPallete = [{color: "#d25800", users: 0},{...}, ...];
...
}
```

Listing 1: Nástin řešení organizace dat.

nevyužité barvy, měnit části vizualizace, jako je ovládání zoom a brush nebo upravovat nastavení škálovacích nástrojů. Část třídy kontext je možné vidět v kódu 1.

Následuje vnitřní organizace důležitých proměnných ve třídě kontext. Těmi jsou *activeOrder*, *active* a *data*. Nejjednodušší na popis je první zmíněný, *activeOrder*, který obsahuje pouze identifikátory jednotlivých os v pořadí v jakém mají být vykresleny ve vizualizaci. Toto pole slouží zejména pro správné mapování pohybu os. Struktura zbývajících dvou je možné vidět v kódu 2. Poslední část, která není rozebrána je, jak vypadají události uvnitř. V ukázce 2 je pouze naznačeno, že je to další objekt. Jeho přesná podoba je ukázána v kódu 3.

10.2 Struktura webové aplikace

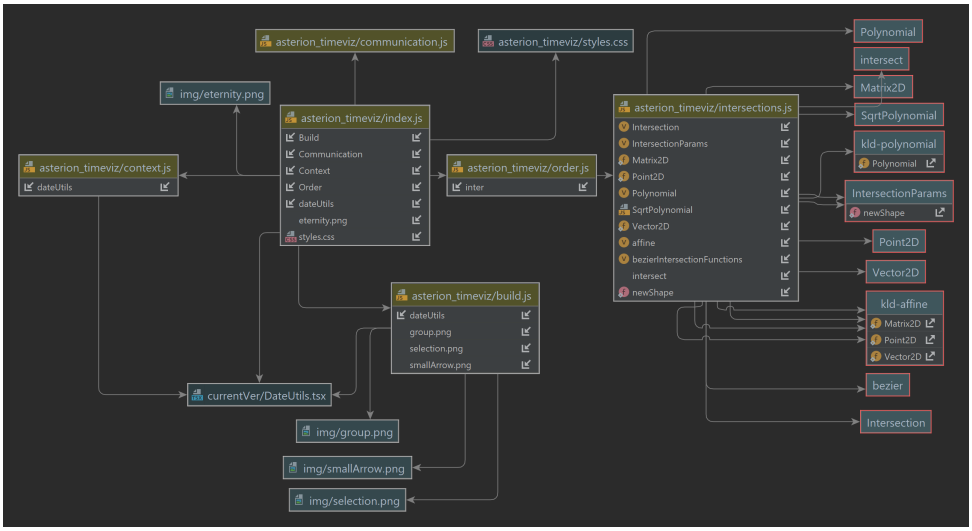
Podobně jako v 9.4 budou i zde rozebrány jednotlivé třídy i s jejich funkcionalitou. U každé třídy budou popsány výsledné změny a jejich funkcionality. Obrázek 10.1 pak ukazuje diagram tříd.

```
data = [{
    name: Almendor,
    clsName: Almendor,
    id:1,
    category:{id: 1, icon: "/img/tags/misto.png"},
    icon: "img/tags/almendor.png",
    events:undefined
}, {...}, ...]

active = [{
    name:Almendor,
    clsName: Almendor,
    id:1,
    events:[{...},...],
    color: "#d25800",
    iconPath: "img/tags/almendor.png"
}, {...}, ...]
```

Listing 2: Ukázka reprezentace *active* a *data*. Data jsou vložena tak, jak je ukázáno u příslušné proměnné. S tím, že pokud není definována ikonka daného tagu, tak ji nahradí výchozí, která reprezentuje kategorii ke které daný tag přísluší. Položka *events* je z počátku prázdná dokud uživatel nezobrazí daný tag. Poté dojde k naplnění daty o událostech, které s příslušným tagem souvisí. *activeData* obsahuje aktuálně zobrazovaná data. Jejich podoba je vidět u příslušné proměnné.

```
events: [
    {
        id: 46,
        name: "Rozdělení velkého království",
        description: "Sioval rozděluje Velké království na
        ↪  čtyři části země - Severní, Jižní, Východní a
        ↪  Západní.",
        begin: 158257,
        icon: {id:5, name:"flag", path:"img/events/flag.png"},
        filters: [...]
    },
    ...
]
```

Listing 3: V této ukázce je možné udělat si obrázek o přibližném vzhledu dat, která jsou zasílána ze serveru. Tato data jsou pak uložena a použita při zobrazování. Položky *id*, *name*, *description* by měly být vcelku jasné. Položka *begin* obsahuje počet dnů od prvního dne roku nula. Položka *icon* udává cestu, na které leží ikona k příslušné události. Položka *filters* obsahuje odkazy na další tagy, které s ním souvisí.



Obrázek 10.1: Diagram tříd webové aplikace. Hlavní třídy jsou *communication.js*, *context.js*, *order.js*, *build.js*, které využívá *index.js*. Zbývající entity jsou pouze doplňkové.

10.2.1 Kontext

Vnitřní uspořádání této třídy bylo rozebráno v předchozí sekci, proto zde bude pouze doplněn komentář, který podchytí změny oproti návrhu. Hlavní změna je, že tato třída je spíše pasivní úložiště, které něco dělá pouze v případě, kdy dojde ke změně aktivní množiny os. S tím, jak uživatel vyhledává různé termíny, tak dochází k ukládání různých výsledků hledání na serveru. Tyto mezivýsledky se ukládají a v případě potřeby je možné je rozšířit o další informace. Obrazový materiál, jako jsou ikonky, je zatím nutné získávat přímo ze serveru z příslušné složky.

10.2.2 Komunikace

Návrh této třídy byl promítnut do implementace téměř celý. Zatím nevyužitou částí návrhu je možné promazávání informací z kontextu. Tato možnost bude znovu zvážena až proběhne další, důkladnější testování. Další část, kterou bylo nutné přesunout, bylo uložení adresy, ze které si tato třída bude brát své informace. Ta je v aktuální chvíli uložena v kontextu.

Obrázek 10.2: Finální vzhled hlavičky vizualizace bez zaplněných událostí a bez aktivního výběru pro nástroj brush.

10.2.3 Pořadí os

Tato třída obsahuje pouze metody nutné k výpočtu pořadí aktuálně vybraných os. Obsahuje všechny tři metody z návrhu a několik dalších konstant, které umožní vytvořit lepší vzhled vizualizace. Nejdůležitější metodou zde je *makeView*, která vezme již seřazená data z proměnné *active* třídy kontext a vytvoří náhled těchto dat. Tento náhled je následně předán dále k zobrazení. Komentáře k jednotlivým metodám je možné dohledat v příslušné sekci implementace.

10.2.4 Výstavba

Implementace třídy pro výstavbu proběhla téměř beze změny. Tak jak je stanoveno v návrhu, tato třída pouze vykresluje nebo překresluje obsah vizualizace. V návrhu je přítomna jedna poznámka, která hovoří o spravování mapování zobrazovaného obsahu. Mapování má pod kontrolou kontext a tím pádem nebylo potřeba jej umístit ve výstavbě.

10.2.5 Interakce

Soubor *index.js* obsahuje vše tak, aby celá vizualizace mohla plynule fungovat. Vytváří instance jednotlivých tříd, vytvoří mezi nimi spojení a následně po vykreslení doplní všem prvkům příslušné metody. Například aby tlačítko posunu skutečně posunula vybranou osu. Zároveň je tento soubor zodpovědný za iniciaci komunikace, jelikož zpětná volání pro získání dat ze serveru jsou uložena zde. V tomto souboru je také možné nalézt ovládání pro zoom, brush nebo ovládání z klávesnice. Zde může vyvstat otázka proč zrovna soubor a ne třída. Při prvních pokusech o spojení jednotlivých tříd vznikl problém s klíčovým slovem *this*, které bylo očekáváno v určité podobě, nicméně díky zařazení ve třídě obsahovalo něco jiného. Proto byla možnost využití třídy zatím zavrhnuta.

Obrázek 10.3: Ukázka finální podoby tagu. Levý obrázek neobsahuje výběr, prostřední a pravý ano ano. Poslední obrázek také ilustruje jak vypadá tag, pokud na něm nestojí kurzor.

10.3 Ovládací prvky

Celková změna ovládání proběhla přesně dle 9.1. Většina funkcionality byla přemístěna a schována, případně doplněna o další části. Hlavním rozšířením jsou klávesové zkratky, díky kterým je možné zefektivnit práci. Mimo to došlo k propojení nástrojů zoom a brush, které je možné vytvořit s pomocí knihovny D3.js. Propojení těchto dvou prvků umožňuje vytvořit interaktivní mini mapu v záhlaví. To ale není vše, pokud uživatel najede na oblast s časovými osami, tak může využít myš. S pomocí kolečka myši je možné rychle měnit aktuální přiblížení a stisknutím a tažením se pohybovat v celé vizualizaci, případně zobrazit části, které se ve výchozím postavení nevejdou na monitor.

Mimo to bylo zachováno zobrazení informací o jednotlivých událostech. Po najetí na některou z událostí se zobrazí prvek se všemi důležitými informacemi. V případě, že by mělo dojít k překrytí událostí, se vytvoří skupina, která v sobě bude skrývat všechny kolidující[1] události. Finální podoby různých částí je možné vidět na obrázcích 10.2, 10.3 a 10.4. V případě kolidujících událostí vypadá shluk podobně jako na obrázku 10.5 vlevo. Ten bude následně rozbalen do své plné šíře a ukáže skryté události. Tak, jak je vidět na obrázku 10.5 vpravo.

10.4 Metody výpočtu

V kapitole 9.3 byly rozebrány tři metody, které je možné využít pro stanovení pořadí os. Mezi tyto metody patří hrubá síla, frekvenční tabulky a silové metody. Všechny tři byly implementovány,odzkoušeny a testovány na

[1] Jedná se o více událostí svázaných se stejným datem.

datech. Jejich rozbor je uveden níže s popisem jejich implementace a možnými úpravami za účelem zlepšení výsledků nebo zrychlení výpočtu. Všechny tyto metody jsou zakončeny vykreslením výsledku. Jedná se o posloupnost *b.drawRes(this.makeView());*, kde vnitřní metoda vytvoří pořadí na základě výpočtů. Vnější metoda pak vezme tento výsledek a zaktualizuje vizualizaci. To může znamenat přidání nebo smazání os nebo jejich přeskupení.

10.4.1 Hrubá síla

Implementace této metody byla nejvíce přímočará. Získalo se výchozí rozestavení os (takové jaké dostane metoda od uživatele), na základě tohoto pořadí je pak spočteno nové nejlepší pořadí, které má minimální počet průsečíků jednotlivých os. Již při třech osách je tato metoda pomalejší a při zvyšování počtu os její běh kolikrát přesáhne dobu, kterou je uživatel ochoten čekat. Tato metoda funguje dle očekávání, spočítá veškerá možná pořadí a u těchto pořadí určí počet překřížení. Pro zjednodušení je v paměti uložena pouze nejlepší varianta. Proti této doposud nejlepší variantě jsou porovnávány nové výsledky a případně nahrazují doposud nejlepší výsledek. Tuto metodu je možné vidět v ukázce kódu 4. Již z principu je jasné, že tato metoda bude velmi pomalá, proto bylo uvažováno i nad jejím zrychlením, to však nebylo zatím implementováno. Zrychlení spočívá v jednoduché optimalizaci. Pokud bude možné zjistit, které osy nemají vliv na křížení, pak je možné tyto osy umístit na okraj a tím pádem klesne počet zkoumaných možností. Jinak řečeno, pokud by bylo možné vyřadit určité možnosti, tak bude možné získat lepší časovou složitost.

10.4.2 Frekvenční tabulky

Návrh této metody byl téměř celý převeden do výsledné implementace. Jedinou výjimkou je zatím nezrealizovaná část s ověřením zda je dané pořadí skutečně optimální z pohledu počtu překřížení. Tato část je zatím ponechána jako rozšíření v budoucnu. Metoda sama o sobě dává rozumné výsledky. Na začátku této metody jsou opět získány aktivní prvky vizualizace (aktuální zobrazované osy) a s jejich pomocí je vytvořena tabulka. Do té jsou pak napočítány společné události pro každou dvojici os. Takto získaná tabulka je pak seřazena dle celkového součtu událostí pro jednotlivé řádky. V případě shody jsou tyto řádky dále porovnávány na jednotlivé dílčí společné události a dle nich řazeny. Z takto seřazené tabulky je získáno pořadí a to aplikováno na aktivní prvky vizualizace. Ty se seřadí dle předaného pořadí a vykreslí. Tato metoda je popsána kódem 5.

```
// b is the build instance, which takes care of redrawing the canvas with the result
bruteForceOrder(b) {
    // update necessary variables
    this.context.updateScales();
    const entriesCount = this.context.active.length;
    const idxs = [...Array(entriesCount).keys()];
    const permutations = this.permutate(idxs, entriesCount);
    // best permutation with the following structure
    // #intersections, perm, paths, events
    let bestPerm = undefined;
    // find out wich permutation is the best in a sense of minimal number of intersections
    for (let perm of permutations) {
        // rearrange active paths
        this.context.active = perm.map(r => this.context.active[r]);
        // get path points for the current order
        let tmpPaths = this.makeView(this.context.active);
        // count the number of intersections
        let numInt = this.getNumberOfIntersection(tmpPaths.paths);
        // is this permutation the best?
        if (bestPerm === undefined || bestPerm.numInt > numInt)
            bestPerm = {numInt: numInt, perm: perm, res: tmpPaths};
    }
    // set correct order and redraw canvas
    this.context.active = bestPerm.perm.map(r => this.context.active[r]);
    b.drawRes(bestPerm.res);
}
```

Listing 4: Metoda s použitím hrubé síly vezme indexy aktuálně zobrazovaných os a ty postupně permutuje a ukládá tato pořadí. Ta jsou následně prověřena na počet křížení a vyhodnocena zda jsou lepší nebo horší než aktuálně nejlepší pořadí.

```
// b is the build instance, which takes care of redrawing the canvas with the result
frequencyTable(b) {
    this.context.updateScales();
    const names = this.context.active.map((item) => item.name);
    // 2d array of frequencies created from mutual events
    let freqData = this.makeFrequencyTable(names);
    freqData.sort((a,b) => {
        if (a.freq[names.length] < b.freq[names.length])
            return 1;
        else if (a.freq[names.length] === b.freq[names.length]) {
            for (let idx = 0; idx < names.length; idx++)
                if (a.freq[idx] !== b.freq[idx] &&
                    freqData.indexOf(a) !== idx &&
                    freqData.indexOf(b) !== idx)
                    return a.freq[idx] > b.freq[idx];
        }
        else
            return -1
    });
    const order = freqData.map( (item) => item.idx);
    this.context.active = order.map(i =>
    this.context.active[i]);
    // create view and draw it
    b.drawRes(this.makeView());
}
```

Listing 5: Frekvenční metoda načte aktivní prvky, nechá vytvořit frekvenční tabulku. Ta je následně seřazena a dle ní je stanoveno vhodné pořadí os.

10.4.3 Silové metody

Tato metoda využívá dostupné funkcionality z knihovny D3.js a to *d3.forceSimulation*. S pomocí této a několika dalších metod bylo dosaženo vyhovujícího výsledku. Kód 6 ukazuje implementaci této metody. Jejím základem je sestavení sítě, kterou je nutné stabilizovat. Síť je vytvořena pomocí frekvenční metody, díky které je možné sestavit hrany mezi jednotlivými uzly a nastavit jejich sílu. Takto vytvořená síť je přitahována k určité y souřadnici, kde je nutné příslušné uzly srovnat do rovnovážného postavení. Nalezení rovnovážného stavu trvá různě dlouhý časový úsek a v některých případech je třeba trocha trpělivosti. Nicméně výsledek, který lze z této metody získat má dobrou vypovídající hodnotu a může ukázat osy v jiných pořadích oproti frekvenční metodě. Informace k této metodě byli čerpány z [62].

```
// b is the build instance, which takes care of redrawing the canvas with the result
forceMethod(b) {
    this.context.updateScales();
    // get "nodes" which are active in current viz
    const names = this.context.active.map((item) => item.name);
    // Nodes are real nodes, not like the line above
    let nodes = [];
    // from names get random position for all nodes
    names.forEach(item => nodes.push({"id":item.index, "x": 150, "y":this.getRndNumber(200,600)}));
    // get "strength" of connections
    let freqData = this.makeFrequencyTable(names);
    let links = [];
    for (let i = 0; i < names.length; i++)
        for (let j = i + 1; j < names.length; j++)
            if (freqData[i].freq[j])
                links.push({source: i, target: j, weight: freqData[i].freq[j]});
    // create scaler for connection strength from min and max value to 0.1 => 1.0
    let weightScale = d3.scaleLinear()
        .domain(d3.extent(links,  (d) => { return d.weight }))
        .range([.1, 1])
    // let the simulation decide the order of lines
    d3.forceSimulation().nodes(nodes)
        .force("link",d3.forceLink(links).strength((e) => weightScale(e.weight)).distance(0))
        .force("x", d3.forceX(150))
        .force("collide", d3.forceCollide().radius(10))
        .force("charge", d3.forceManyBody().strength(1))
        .on("end", () => {
            // when the simulation ends, create the correct order and draw the result
            nodes.sort((a,b) => a.y - b.y);
            const order = nodes.map(item => item.index);
            this.context.active = order.map(i => this.context.active[i]);
            b.drawRes(this.makeView());
        });
}
```

Listing 6: Ukázka výsledné podoby silové metody, která se částečně opírá o frekvenční metodu. K tomu navíc využívá silové simulace nad získanými daty. To vede k odlišnému výsledku oproti frekvenční metodě.

	#Prvků					
Metoda	10	50	100	150	200	250
Frekvenční metoda	1,0	3,4	5,6	6,9	16,9	16,5
Silová Metoda	1817,0	1894,6	1883,7	1865,7	1842,3	1850,2
	3	4	5	6	7	8
Hrubá síla	42,1	1275,6	5838,2	50453,2	550217,7	xxx

Tabulka 10.1: Tato tabulka obsahuje všechny naměřené časy u jednotlivých metod. Tabulka uvádí dva počty časových os, to z důvodu rychlého nárůstu složitosti u poslední metody. Časy v řádcích jsou měřeny v milisekundách. V těchto hodnotách je dobré poukázat na skutečnost, že silová metoda se s dostupným množstvím vzorků chová téměř konstantně, pokud se zanedbává čas potřebný na vytvoření a nastartování simulace. Při pohledu na na frekvenční metodu je vidět určitý nárůst potřebného času na zpracování. Vzhledem k fungování příslušné metody, se pravděpodobně bude jednat o kvadratický nárůst a tedy odpovídající časovou složitost.

10.4.4 Měření rychlosti výpočtu

Po dokončení implementace a odstranění všech zásadních chyb byla změřena rychlost výpočtu jednotlivých metod. Jak se dalo očekávat neoptimalizovaná varianta metody hrubou silou velmi brzy překročila práh několika minut. Frekvenční metoda se oproti tomu nijak výrazně nezadrhla ani při využití všech dostupných časových os. U této metody je však stále vidět postupný nárůst zpracování. Toto souvisí s využitím tabulky, ve které jsou příslušná data zaznamenána. Největším překvapením byla stabilita silové metody, která měla v průměru stále stejný výsledek, navzdory větší režii. Pro lepší porovnání by bylo třeba získat větší vzorek. Tabulka 10.1 obsahuje naměřená data.

10.5 Značení událostí

Značení událostí bylo zachováno z předchozí verze a na určitou dobu takto zůstane. V databázi bude nutné udělat různé změny a jejich rozsah zatím není jasný. Bude nutné vyhledat párové tagy válek a ostatních podobných událostí a nahradit je navrhovanou jednou událostí s dobou trvání nebo konečným datem. Kromě této úpravy je v plánu také přidat několikrát zmiňované hierarchické události. Ty budou vyžadovat další přepracování databáze a velmi pravděpodobně i serverové části, která je zatím využívána v původní verzi. Jediný nový prvek, který musel být zaveden s ohledem na možná překrývání

událostí je shlukování do skupin. Jedná se čistě o záležitost frontendu. Vzhled byl uveden u interakce, jelikož tyto skupiny mají za následek další interakční rozšíření. Pro úplnost se jedná o dva obrázky z 10.5. Zmíněné okna s detailními informacemi je možné vidět také na obrázku 10.5. Toto menší okno obsahuje Název dané události, její popis a časový údaj, kdy k ní došlo. V pravé části je pak seznam všech dalších tagů, kterých se daná událost týká.

10.6 Časové linky

Změna časových linek proběhla v souladu s 9.3, kde opět proběhlo doplnění nebo je teprve plánováno. Časové osy je možné vidět na obrázku 10.6, kde je vidět stažení jedné osy k jiné. V závislosti na úrovni přiblížení/oddálení se však mohou ukázat detaily tohoto stažení osy. Jako podklad slouží nekonečná osa. Zde může vyvstat otázka, proč byla původní, ne příliš kladně zmiňovaná, osa použita. Odpověď se skrývá u událostí, které se nachází v prázdném prostoru nad přitaženu osou. V momentě, kdy se dané osy přiblíží vznikne u jedné z nich prázdné místo v jejím původním běhu. V tomto prázdném místě se mohou vyskytovat další události, které by visely v prázdném prostoru a nebylo by jasné komu patří. Tím, že se přidá původní nekonečná osa se tomu dá jednoduše zabránit na úkor většího počtu křížení. Samotné vybočení osy je v několika bodech zanesený oblouk dokreslený s pomocí metod knihovny D3.js.

10.6.1 Barvy časových os a událostí

Tak jak bylo naznačeno již v návrhu, tato část původní vizualizace byla velmi přínosná. Největší zásluhu na tom má automatické přiřazování barev časovým osám. Jinými slovy tato funkcionalita dávala rozumné barvy uživateli automaticky a jejich změna proběhla jen na přání uživatele. V první části implementace bylo barevné označení os ponecháno na uživateli, ale po konzultaci daného prvku, byl změněn do původní automatické podoby. Vzhledem k tomu, že je u vizualizace nutné zachovat přehlednost a zřetelnost, musely být vybrány barvy, které budou těžko zaměnitelné. K tomuto srovnání bylo využito zajímavého nástroje Viz-Palette [61]. Ten umožňuje vytvořit barevnou paletu a zvýrazňuje u ní v jakých případech budou dvě barvy zaměnitelné. Tuto aplikaci je možné vidět na obrázku 10.7. Takto vytvořená paleta obsahuje třicet barev, které jsou používány jednotlivými osami. Tyto barvy jsou pak dodržovány i jednotlivými značkami v mini mapě, kde dle zbarvení osy odpovídají příslušné značky. Tento nástroj by obecně mohl pomoci při

výuce vizualizací a proto je pod [63] uveden ve své původní podobě tak, aby bylo snadné jej dohledat.

10.7 Problémové části předchozí vizualizace

V předchozí verzi bylo několik problémových částí, které nebyly řešeny nebo byly vyřešeny velmi zvláštním způsobem. Mezi tyto části patří například spojování stejných událostí, překrývání událostí, nedostatečné přiblížení, velké vzdálenosti mezi jednotlivými částmi ovládání nebo neuniformní rozdělení časové osy. Ne všechny zmíněné části se podařilo v nové vizualizaci vyřešit, jak je vidět na obrázcích v této sekci. Mezi vyřešené problémy je možné počítat snížení obrazového znečištění vlivem spojnic. Ty jsou v momentální chvíli řešeny jako přitažené části os k sobě. Čímž vznikl jiný problém – co když bude více událostí ve stejný den a bude se týkat stejných/podobných os? Pak vzniknou skupiny, které tu již také byly zmíněné. Ty následně shlukují různé události k sobě a zajišťují čistější obrazovku. Překrytí událostí bylo odstranění díky odstranění horního limitu na přiblížení a zlepšení ovládacího schématu. To se rozšířilo o možnost užívat myš k pohybu přímo v zobrazovacím prostoru. Velké vzdálenosti k ovládacím prvkům os pak byly přesunuty hned k popisu os a také přidáním ovládání z klávesnice. Problém, který přetrvává, je schován v uniformním rozdělení. Zatím nebyla nalezena cesta, jak donutit knihovnu D3.js k tomu, aby rozdělovala prostor dle časových a ne prostorových vzdáleností.

10.8 Rozšíření vizualizace

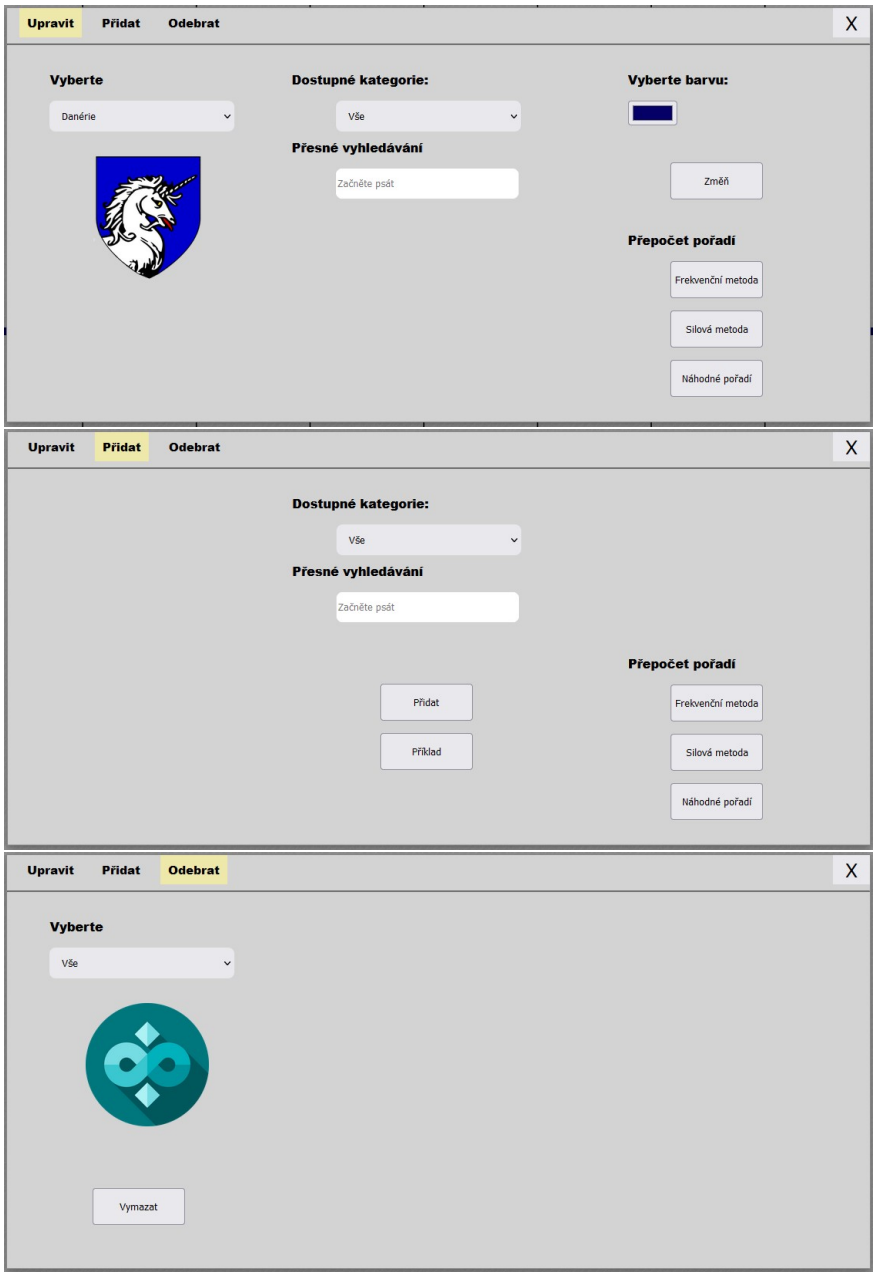
Celou vizualizaci je možné ještě zlepšit a přidat další ovládací prvky, které jsou populární, nebo mohou být požadovány na základě testování. Mezi tyto prvky patří například možnost drag&drop, která umožní lepší přemístění jednotlivých os. Kromě ovládání bude určitě nutné přidat další zmiňované typy událostí jako jsou hierarchické nebo déle trvající. Rozšíření událostí je vcelku přirozeným krokem a je s ním počítáno. Zejména v oblasti lepšího rozlišení kterých os se daná událost týká. Jinak řečeno, po najetí na událost zůstanou zbarvené pouze ty osy, které s danou událostí souvisí, zbytek se zbarví do šedivé nebo jiné barvy, která nebude rušit soustředění uživatele. Kromě barevného rozlišení je dobrou myšlenkou obecné rozlišení kategorií jednotlivých os,

například v podobě délky daných os[2]. Dalším dobrým rozšířením je vyzkoušet další metody výpočtu pořadí. Co se týká optimalizace, tak kromě vytvořených metod bude dobré optimalizovat i některé podčásti. Mezi ty patří například optimalizace vyhledávání zadávaných pojmů.

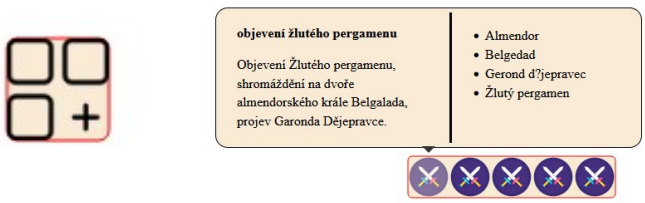
10.9 Závěr

Tato kapitola shrnula implementační část práce. Zmínila změny oproti návrhu a některá doplnění. Části, které byly navrženy a nedostali se do implementace nejsou zahozeny. Mnoho z nich má potenciál stát se dobrým prvkem v celé vizualizaci a tak bude dobré si je později připomenout a zakomponovat pokud jejich použití bude stále relevantní. Ke konci této kapitoly byly také zmíněny některé problémy předchozí verze a jak si s nimi tato nová verze poradí. Testování ukáže smysluplnost provedených změn.

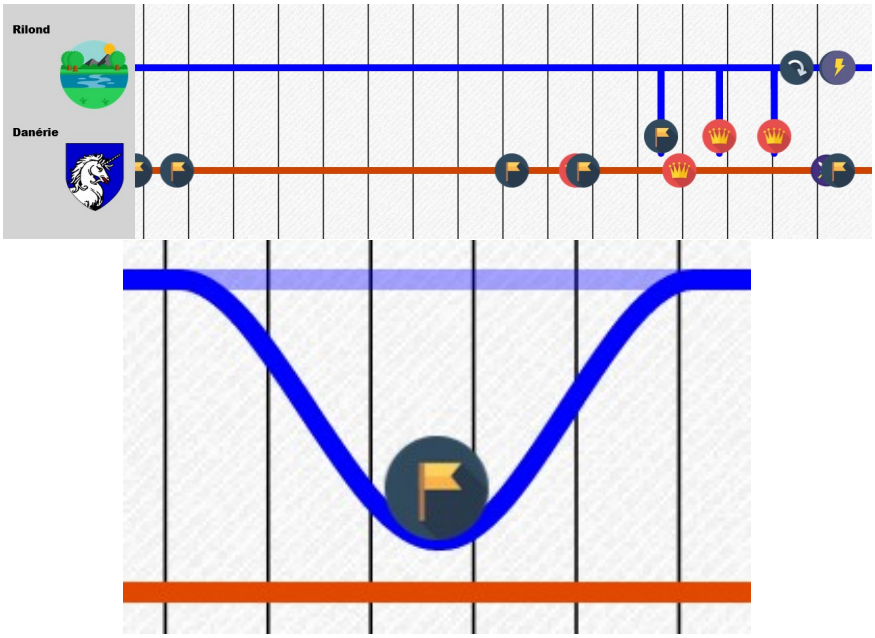
[2]Území bude možné nalézt v mnohem delším měřítku než například jednotlivé postavy nebo předměty.



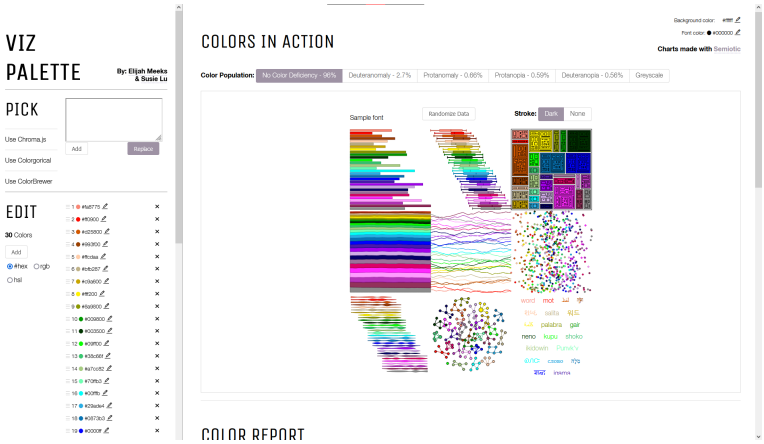
Obrázek 10.4: Tyto tři obrázky ukazují podobu jednotlivých záložek s jejichž pomocí je možné upravovat vizualizaci.



Obrázek 10.5: Ukázka reprezentace v případě kolizí několika událostí. Obrázek vpravo obsahuje pouze ilustrativní text a obrázky.



Obrázek 10.6: Aktuální vzhled časových os v nejmenším přiblížením, kdy je vidět celá osa a v detailu.



Obrázek 10.7: Tento obrázek ukazuje náhled stránky viz-palette se zvolenými barvami, s jejíž pomocí byla vytvořena paleta pro časové osy a události na minimapě.[61, 63]

Kapitola 11

Testování

V této kapitole bude shrnuto testování. Dle zadání mělo být provedeno kvalitativní testování s alespoň šesti uživateli. Kromě toho bylo těmto uživatelům představeno několik alternativ některých prvků vizualizace a tito uživatelé pak vybírali, které varianty se jim líbily nejvíce. Výsledky této části jsou v sekci 11.2.

11.1 Kvalitativní testování

Testování proběhlo, jak již bylo řečeno, s šesti uživateli. U každého uživatele byly využity stejné scénáře a po každém scénáři proběhlo s uživateli krátké zhodnocení a debata nad daným scénářem či testovanou funkcionalitou, v jednom případě dokonce k lepšímu návrhu celého rozhraní. Dle slov jednoho testera vypadá daná stránka jako z brzkého přelomu století. To je způsobeno tím, že přesný design měl být rozhodnut později. Nejdříve budou rozepsány obecné problémy, které byly nalezeny a poté zmíněný nový návrh rozhraní, jelikož to si zaslouží vlastní část.

Obecné připomínky k celé vizualizaci byly následující. V některých částech není jasné, co je myšleno různými slovními spojeními, to může být způsobeno buď neznalostí slovníku, nebo špatně zvolenými výrazy. Například nebylo hned zřejmé, že se jednotlivé časové osy řadí do různých kategorií. Jedním z chybějících prvků byla možnost vycentrovat vizualizaci a vrátit se zpět na střed, pokud došlo k tomu, že uživatel cestoval příliš vzhůru nebo dolů.

Část modálního okna s možností přidávat, upravovat a mazat byla přehlcena funkcionalitou a tím pádem méně srozumitelná. Zejména pak nebylo jasné, co znamená podčást změna pořadí os. Zde bylo zjištěno, že kdyby uživatel nevěděl, že daná funkcionalita umožní zpřehlednit vizualizaci, tak by ji nepoužil. U metod na výpočet pořadí os bude vhodné přidat náznak postupu práce, případně načítací kolečko, aby bylo jasné, že se na pozadí něco děje. Velmi nepříjemná situace pro testery nastávala, když došlo k přepočtu pořadí a jejich výběr na mini mapě se zrušil a opět byla vybraná celá vizualizace. Pro lepší představu bude uveden příklad. Uživatel si vybere na mini mapě určitý úsek který ho zajímá, ale ten není dost přehledný. Využije jednu z metod na seřazení os a jeho pracně vytvořený pohled na data je ztracen. Očekávané chování bylo, že pohled zůstane tam, kde byl a ke změně dojde pouze v pohledu na data. Další připomínkou bylo, že upravovaná osa by měla zůstat vybraná i po změnách, které na ní proběhnou, třeba výměna za jinou.

Testeři měli dále tendence využívat některé ovládací prvky, které zatím nebyly doplněny jako například drag&drop, potvrzení výběru a přidání pomocí klávesy Enter. Oba tyto ovládací prvky budou přidány, aby měli uživatelé větší pohodlí při manipulaci.

Další funkcionalita, kterou bude třeba doplnit je posouvání celé vizualizace pokud výběr vyskočí mimo zobrazovanou část. To je možné provést dvěma způsoby, buď posouvat celou vizualizaci s výběrem nebo aktuální výběr zamrzne na místě a vše ostatní bude posouváno okolo tohoto výběru.

Velmi často docházelo ke klikání na jednotlivé události s cílem zobrazit detaily o daných událostech. Zde bude možné danou funkcionalitu doplnit o nějaký rozumný prvek. U událostí pak byla zaznamenána snaha nechat určitou událost viditelnou delší dobu, pro možnost porovnání s jinou. Tato část nejspíše nenajde případ užití, nicméně je možné ji využít. K událostem je také vztažena jedna velmi dobrá připomínka. Vyhledávání je jedním z prvků, které pokrývají většinu akcí. Jinak řečeno dva testeři ve scénáři s vyhledáváním událostí zkoušeli objevit lištu pro zadání hledaného výrazu. Tento prvek nebyl zvažován, nicméně je to velmi dobrá připomínka a bude dále zapracována. Kromě vyhledávání byla také naznačena možnost přidání filtrování událostí. To si lze představit tak, že si uživatel bude chtít zobrazit například všechny války, úmrtí, významné korunovace a tak dále. Co se týká velikosti jednotlivých událostí, tak bylo zjištěno, že by byla uvítána možnost změnit velikost jednotlivých událostí s cílem zmenšit překrytí událostí. Tato myšlenka je přínosná a bude zařazena do budoucího rozšíření vizualizace.

Co se týká modálního okna, tak posloupnost akcí není hned zřejmá, je tedy nutné provést změny v úrovních fontů a okolí tak, aby bylo jasné, co daná část dělá. To také souvisí s volbou vhodného popisku jednotlivých částí.

U mini mapy v horní části bude třeba změnit velikosti krajních madel, jsou příliš malá na chytnutí a táhnutí. K tomuto prvku byla zaznamenaná ještě jedna velmi dobrá poznámka. Tečky v mini mapě jsou vrstvené na sebe a tím pádem dochází k velkému překryvu. Aby bylo jasné, kde má vybraná osa události je možné zbývající tečky skrýt a nechat viditelné pouze ty, které přísluší vybrané ose.

11.1.1 Testování s UX testerem

Jeden z testerů se aktivně zabývá UX/UI, tato skutečnost byla velmi přínosná a ukázalo se, že bude třeba přepracovat větší část rozhraní a udělat tutoriál, který bude ulehčovat seznámení s celou vizualizací. Opět budou postupně probrány jednotlivé části, tak jako v předchozí sekci.

První poznámka, která se uplatňuje na celé rozhraní je rozlišení úrovní textů. Tím je myšleno, že drtivá většina textů je stejného významu a tím pádem není jasné, která část má větší význam. Další následek je, že některé části zanikají a uživatel je nemusí nalézt. Jako příklad lze uvést datumy okolo mini mapy, okolní text tyto datumy přehlušuje a tím pádem je snadné je přehlédnout. Tlačítka v záhlaví rozhraní jsou také jednolitá a není jasné, které z nich má přednost. Zde by měla být posloupnost „Upravit“, „Nápověda“, „Asterion“, „Discord“. Vzhledem ke stejnému fontu a nepoužití jiného podbarvení se tyto informace ztratí a není jasná primární akce. To samé platí v modálním oknu, které má svá tlačítka avšak tato tlačítka příliš splývají s pozadím.

Při otevření přivítá uživatele prázdná obrazovka, což nemusí být nutně špatné, nicméně je lepší navrhovat takovéto části s myšlenkou „*Empty state message*“, která ukáže uživateli, jak má začít s interakcí. Velmi dobrou myšlenkou je zde vytvoření tutoriálu pro návštěvníka, který se dostal na tuto stránku poprvé. Pro další návštěvy bude dobré nechat nasměřování na přidání časových os.

Některé použité pojmy jsou zavádějící a nemusí být hned jasné, co znamenají. Výrazy jsou často příliš generické a je nutné je nahradit konkrétními slovy nebo slovními spojeními. Jako příklad postačí samotné slovo příklad. V modálním okně v záložce pro přidání os je možnost zobrazit si připravený příklad. Tato funkcionalita se skrývá právě pod tlačítkem „Příklad“. Pokud však uživatel netuší, co se na dané stránce dělá, pak mu ani nemusí dávat smysl samotné slovo příklad. Přesněji řečeno čeho je to příklad? Je to příklad několika os? Je to příklad vztahů mezi osami? Příklad událostí a jejich vztahů? Takovýchto otázek může vyvstat mnoho.

Obrázek 11.1: Tento obrázek znázorňuje nové rozložení záhlaví stránky s časovými osami. S úpravami dle zpětné vazby z testování.

Správně by celé rozhraní mělo uživatele vést či svádět k provedení některých úkonů. Aktuální rozhraní nic takového nedělá a je vyžadována velká pozornost uživatele. Dále už budou jen zmíněny některé opakující se části jako například vyhledávání v událostech, větší interakční prvky pro mini mapu, posun výběru, přidání filtrů, různé styly ikonek.

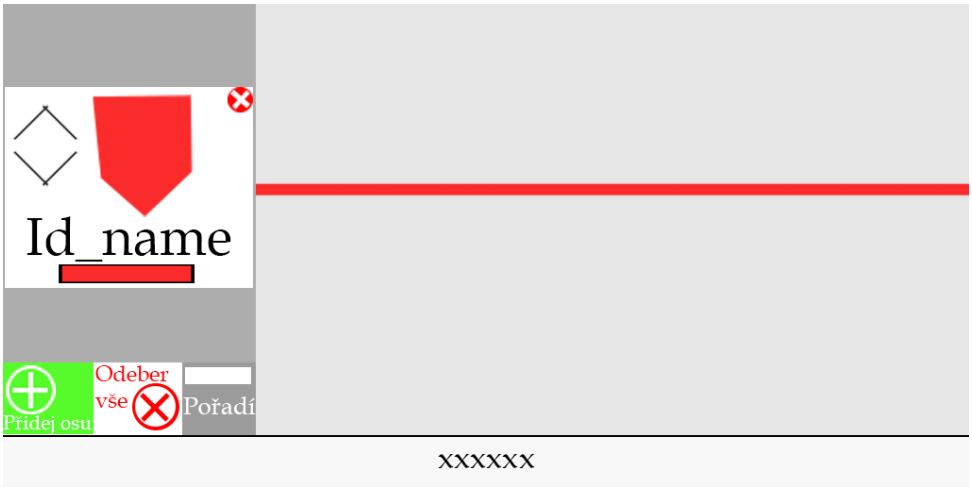
Největší problém dělá samotné modální okno, u kterého není zcela jasné interakční schéma. A to ani v jedné ze záložek. Dalším problémem je, že dvě z prezentovaných záložek obsahují více než jednu funkcionalitu. Záložka „Přidat“ a „Upravit“ obsahuje navíc funkcionality pro pořadí os. Zde tedy bude vhodné vytvořit vlastní záložku nebo tyto metody vystavit do jiné části rozhraní. V těchto dvou záložkách bylo navíc velmi matoucí výběr kategorie, který by v drtivé většině případů zůstal na výchozí hodnotě „Vše“.

Při následné debatě bylo rozebráno několik možností pro většinu částí, případně doptání se na konkrétní, lepší příklady jednotlivých částí. Ve většině případech bylo nutné zjistit trochu více o daném problému a spolu s návrhy pro UX vytvořit lepší vzor pro rozhraní. Tímto se zabývá následující podsekce.

11.1.2 Navrhované změny

Zde budou rozebrány jednotlivé návrhy a ty nejpodstatnější rozebrány podrobněji. První částí zde je předělání textů a rozdělení jednotlivých akcí na primární, sekundární a další. Předělání textů proběhne jednoduše změnou fontů a jejich velikostí tak, aby se srovnala důležitost prvků na nižší úroveň. Následně budou vytaženy prvky které jsou důležité, např. primární akce přidání osy. Toto rozlišení je možné provést například barevným odlišením. S tímto problémem se také pojí symbolika, tlačítka pro odkazy je možné změnit dle jejich příslušnosti. Například místo textu „Nápověda“ je možné použít pouze ikonku s písmenem i v kroužku nebo pro odkaz na server Discordu ikonku této platformy. Nový návrh je možné vidět na obrázku 11.1.

Dalším velmi podstatným prvkem je změna používaného slovníku. ten je třeba upravit tak, aby každé slovní spojení nebo slovo dávalo smysl a uživatel si za ním bude schopen představit konkrétní věc. Jako příklad je možné uvést slovíčko „Přidat“, které značí přidání časové osy po jejím výběru. Toto slovo



Obrázek 11.2: Zde je možné vidět úpravy rozhraní dle zpětné vazby. Barvy v této ani v 11.1 nejsou finální a bude třeba sestrojit v hodnou paletu.

je příliš generické a nemusí být jasné, co se do vizualizace přidá. Zde bude nejspíše lepší využít slovního spojení „Přidat časovou osu“ nebo „Přidat příběh“, protože časová osa popisuje nějaký příběh nebo cestu, historií. Tedy konkrétně pojmenovat, co se přidává.

Hlavním bodem tohoto návrhu je pak úprava rozhraní a odstranění modálního okna, respektive jeho redukce na jednu funkcionalitu. Rozhraní záhlaví bude upraveno tak, aby tlačítka zbytečně nelákala uživatele a tyto akce byly až jako vedlejší. Jak již bylo řečeno ze tří textem naplněných tlačítek se stanou pouze dvě ikonky a jeden menší text s možností ikonky. Tlačítko „Upravit“ bude nahrazeno menším panelem, který bude mít výrazné tlačítko přidat a pak méně výrazné tlačítko na odstranění všech časových os ve vizualizaci. Tato dvě tlačítka budou v levé spodní části místa s identifikací časové osy.

Modální okno samotné bude redukováno pouze na vyhledávání v databázi a přidání nalezeného pojmu nebo k načtení příkladu. S tím pak souvisí i identifikátory jednotlivých os. Zatím bude nutné zachovat možnosti pohybu nahoru a dolů a tím pádem zůstanou jak klávesové zkratky, tak šipky v rozhraní. Změní se ovšem jejich poloha, budou usazeny k sobě v jedné z krajních částí. Samotný identifikátor, např. erb Almendoru, bude na středu zatím používaného prvku. V pravém horním rohu bude křížek, který značí možnost odstranění dané osy. Text s názvem dané osy je možné nechat tam, kde se momentálně nachází nebo ho přesunout pod identifikátor. Změna barvy bude také přístupná rovnou z této části rozhraní. Prvek pro tuto interakci bude uložen na spodku elementu s možností okamžité změny barvy příslušné osy. Přesný vzhled je možné vidět na obrázku 11.2.

Tímto je většina funkcionality modálního okna odebrána a přesunuta na jiné části. Tím pádem se modální okno zredukuje na již zmíněnou jedinou funkcionalitu. Co se týká záměn časových os za jiné, tento případ užití je třeba podrobněji prodebatovat a prozkoumat. Zatím by mohlo být možné kliknout na název v příslušném elementu a zde by se rovnou rozbalila možnost změny. Po výběru by stačilo potvrdit pomocí klávesy Enter a změny by se hned projevily. Nicméně prvním krokem je zjištění, zda daná funkcionalita je vůbec potřebná.

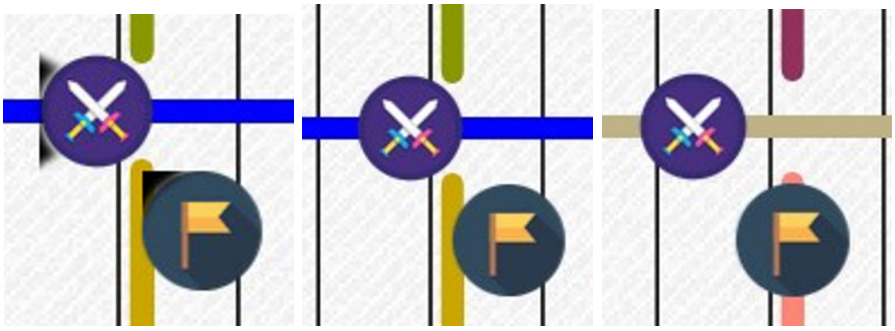
Poslední část, která nebyla vyjmuta z modálního okna je pořadí os. To by bylo možné umístit do vrchní části rozhraní případně na jiné vhodné místo. Navíc bude vhodné vždy zobrazovat uživateli již spočítané pořadí, tak aby dostával, co nejlepší výsledek. Pokud bude uživatel chtít využít jiného pořadí, může ho zvolit a nechat přepočítat aktuální vizualizaci či přednastavit pro teprve vznikající. Jinak řečeno spočítané pořadí bude upřednostněno před uživatelským zadáváním.

Z uživatelského hlediska také dává smysl zavést drag&drop, tato funkcionalita byla testery zkoušena. Tím pádem bude vhodné v nejbližších dalších iteracích tuto funkcionalitu přidat.

11.2 Porovnání variant

Po hlavním testování proběhla diskuze nad možnostmi zobrazení jednotlivých prvků. To se týkalo zejména časových os a událostí. Bylo nutné zhodnotit různé reprezentace a vybrat z nich ty, které dávali největší smysl pro uživatele. Obrázky 11.3 ukazují možnosti mezi kterými testeři vybírali nejvhodnější reprezentaci. V této části nejlépe dopadla možnost vycentrovaných událostí. Nicméně bylo poukázáno na to, že varianta s šipkami také není k zahození, pokud se využije jiného rozložení událostí. Místo toho aby měla událost náležící jedné ose zvláštní černou zarážku u levé strany, tak by bylo vhodné udělat posunutí dané osy výše nebo níže a přidat šipku k hornímu nebo dolnímu levému rohu. Prozatím bude ponechána varianta s vycentrovanými událostmi.

Druhým dotazovaným prvkem byly samotné osy. Ty, jak již bylo řečeno, v dřívějších částech této práce, jsou reprezentovány pomocí cest. Tím pádem se mohou k sobě přibližovat a zase se od sebe oddalovat. Pokud se k sobě přiblíží dvě, či více os, tak nastává otázka, co udělat s místem, které je v linii s původní pozicí. Obrázky 11.4 naznačují tři možnosti. Buď nebude přítomna



Obrázek 11.3: Tyto obrázky ukazují možnosti zobrazení událostí na osách. Levý ukazuje uchycení levým okrajem a přidání šipkou, prostřední pak vzhled bez šipek a pravý vycentrované události.

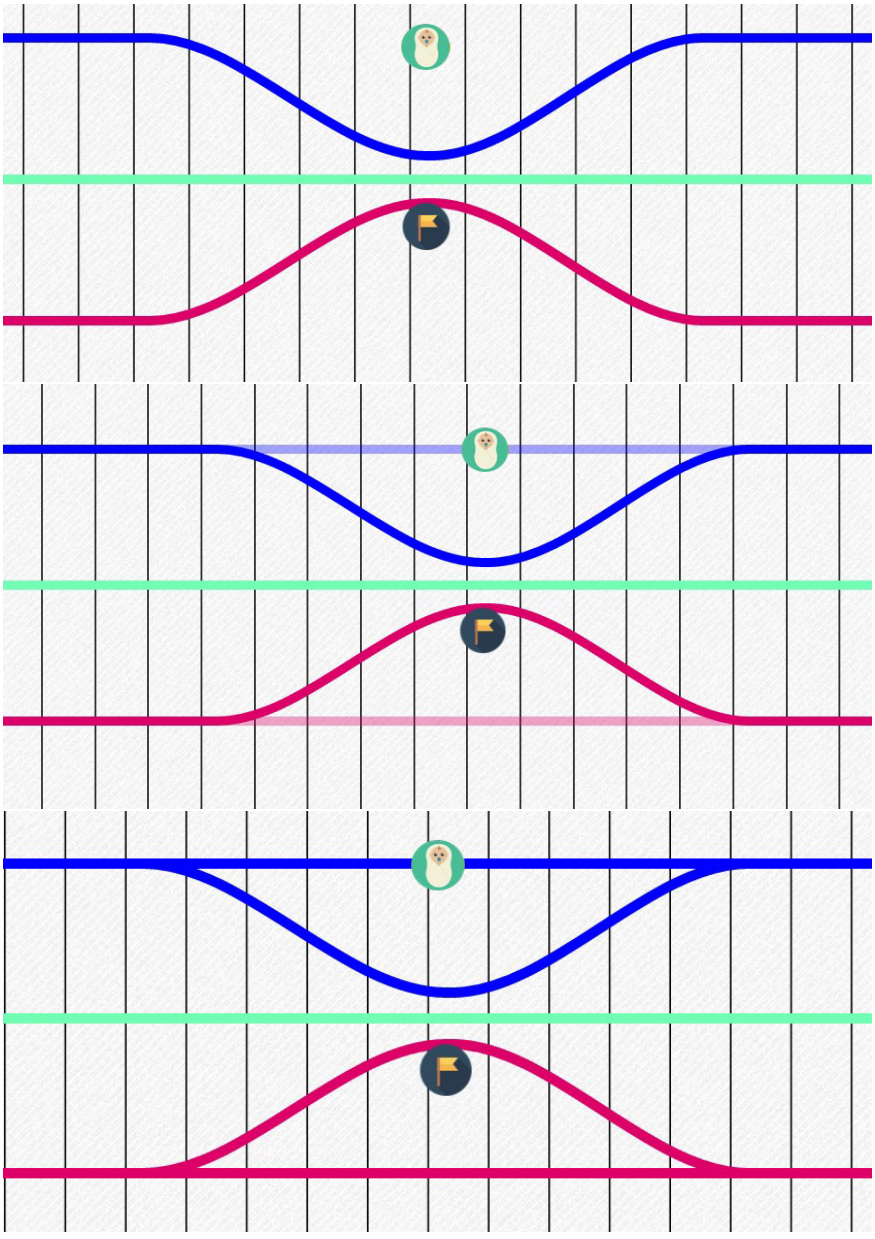
žádná osa, což může vést ke ztrátě kontextu, ke které ose daná událost patří. Tato možnost byla jednomyslně zamítnuta. Zbývající dvě možnosti byly přijímány podobně, avšak větší a o něco lepší efekt vytvářela možnost s poloprůhlednou původní osou. Je dobré zmínit, že jeden z testerů také uvedl možnost přerušované čáry. Tato možnost je také proveditelná a pro další testovaní bude zařazena do možných výběrů.

11.3 Porovnání metod

Většinový názor prezentovaných metod řazení časových os byl, že frekvenční metoda dává přehlednější výsledek, ačkoliv může být zhuštěna okolo jedné osy. Pro silovou metodu se kladně vyjádřil jen jeden tester s komentářem, že daná metoda rozloží události více a tím pádem jsou přehlednější a lépe se v nich orientuje.

11.4 Závěr

Z testování vyplývá mnoho ovládacích prvků, které bude třeba doplnit a zlepšit používaný slovník u jednotlivých částí. Kromě toho je třeba přepracovat navrhovaný vzhled na základě testování s UX zdatným testerem. Toto testování odhalilo mnoho slabých míst a je třeba je odstranit. Testování také přineslo mnoho nových pohledů na problematiku uživatelského rozhraní, které budou dále zapracovávány a využívány.



Obrázek 11.4: Tyto obrázky reprezentují možnosti vzhledu os. Horní znázorňuje možnost bez doprovodné linie. Zbývající dvě pak tuto linii mají s různou průhledností.

Kapitola 12

Závěr

Cílem této práce bylo vytvořit novou verzi vizualizace časových os světa Asterion. Tato nová verze vychází z předchozích prací a snaží se rozšířit dosavadní fungování o lepši rozhraní a možnosti automatického řazení časových os. Toto řazení má pomoci zpřehlednit vizualizaci a tím i ulehčit uživatelům orientaci v datech plus zvýraznit některé zajímavé skutečnosti. Cíl nové verze rozhraní dopadl dobře, stejně tak i vytvoření možnosti různého řazení časových os, kde nad očekávání dopadla zejména metoda řazení s frekvenční tabulkou.

V této práci lze najít průchod základní teorií k vizualizacím a snahu o jejich uplatnění na již existující vizualizaci. Dále analýzu jednotlivých částí předchozí verze oproti zjištěným teoretickým informacím a doplnění rozboru v místech, kde naprosto chybí, například jaké jsou jiné možnosti vizualizace časových dat. Důležitou částí této práce je vytvoření metod pro vhodnou změnu pořadí tak, aby vznikla přehledná vizualizace. Celkově byly navrženy a zrealizovány tři metody, které byly později rozšířeny o náhodné pořadí. To může poukázat na některé zajímavé skutečnosti. Celá implementace byla zakončena testováním a jeho vyhodnocením s novým návrhem do další iterace projektu časových os Asterionu.

Výsledné vizualizace časových os podávají pěkné a ve většině případů i přehledné výsledky, nicméně stále nepodávají přesný pohled do dat a celé problematiky. To je způsobeno absencí některých podpůrných elementů či dat v databázi. O tyto části bude postupně celá databáze doplněna a s tím i vizualizace.

Tato práce má a bude mít velké uplatnění při tvorbě příběhů pro RPG hru zasazenou do světa Asterion. Při vytváření příběhu bude dobré mít přehled o událostech, které se v průběhu dějin stávaly a tento nástroj umožňuje jejich snadný náhled. V pozdějších rozšířeních bude nabízet i možnost rozšiřování celé databáze či vytváření vlastních časových os. Nejprve však bude nutné provést další iteraci na základě zpětné vazby z testování.

Bibliografie

1. RŮŽIČKA, Tomáš. *Asterion - backend vizualizace časových os.* Praha, 2022.
2. ZIMMERMANNOVÁ, Michaela. *Asterion - frontend vizualizace časových os.* Praha, 2021.
3. TAMARA, Munzner. *Visualization Analysis and Design.* A K Peters/CRC Press, 2015. A K Peters Visualization Series. ISBN 9781466508910. Dostupné také z: `https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=861951&site=ehost-live&scope=site`.
4. TELEA, A.C. *Data Visualization: Principles and Practice, Second Edition.* Taylor & Francis, 2014. ISBN 9781466585263. Dostupné také z: `https://books.google.cz/books?id=2WoLBAAAQBAJ`.
5. ČMOLÍK, Ladislav. *VIZ 02 DATA AND TASK CLASSIFICATION* [online]. 2021. [cit. 2022-01-02]. Dostupné z: `https://docs.google.com/presentation/d/1zoUGLhDQqKKyRCnmEzaZpwsCeAIKdrzGNbx-yWspeSc/edit#slide=id.p3`. [Soubor přístupný po přihlášení do sítě ČVUT].
6. SIMON EVANS, Rosamund Pierce. *10 examples of interactive map data visualizations.* [B.r.]. Dostupné také z: `https://www.tableau.com/learn/articles/interactive-map-and-data-visualization-examples`.

7. AIGNER, Wolfgang; MIKSCH, Silvia; MÜLLER, Wolfgang; SCHUMANN, Heidrun; TOMINSKI, Christian.
Visualizing time-oriented data—A systematic view.
Computers & Graphics. 2007, roč. 31, č. 3, s. 401–409. ISSN 0097-8493.
Dostupné z DOI: https://doi.org/10.1016/j.cag.2007.01.030.

8. ČMOLÍK, Ladislav.
VIZ 03 VISUAL ENCODING OF ATTRIBUTES [online].
2021. [cit. 2022-01-02]. Dostupné z: https://docs.google.com/presentation/d/1IHgzFdA5Vtm4hMmUiMTkTK_2ZdQMcO5xApPZhB8tB_0/edit#slide=id.p3.
[Soubor přístupný po přihlášení do sítě ČVUT].

9. ČMOLÍK, Ladislav.
VIZ 04 INTERACTION IN VISUALIZATION [online].
2021. [cit. 2022-01-02]. Dostupné z: https://docs.google.com/presentation/d/1PuwpFzJqtNmkTfgv80ms8xZBe2X7iMZ_orZ2gtWdlqs/edit#slide=id.p3.
[Soubor přístupný po přihlášení do sítě ČVUT].

10. PURWAR, Sourabh. *Breaking down Fitts law for UX designers* [online].
[B.r.]. [cit. 2022-01-02].
Dostupné z: https://uxplanet.org/breaking-down-fitts-law-for-ux-designers-542cabb48f9.

11. WARE, Colin; BOBROW, Robert.
Motion to Support Rapid Interactive Queries on Node–Link Diagrams.
ACM Trans. Appl. Percept. 2004, roč. 1, č. 1, s. 3–18. ISSN 1544-3558.
Dostupné z DOI: 10.1145/1008722.1008724.

12. AHLBERG, Christopher; WILLIAMSON, Christopher; SHNEIDERMAN, Ben. Dynamic Queries for Information Exploration: An Implementation and Evaluation. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.
Monterey, California, USA: Association for Computing Machinery, 1992, s. 619–626. CHI '92. ISBN 0897915135.
Dostupné z DOI: 10.1145/142750.143054.

13. WIJK, Jarke J. van. Image Based Flow Visualization. In: *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*. San Antonio, Texas: Association for Computing Machinery, 2002, s. 745–754. SIGGRAPH '02. ISBN 1581135211.
Dostupné z DOI: 10.1145/566570.566646.

14. REINDERS, Freek; POST, Frits; SPOELDER, Hans. Visualization of Time-Dependent Data with Feature Tracking and Event Detection.
The Visual Computer. 2001, roč. 17, s. 55–71.
Dostupné z DOI: 10.1007/PL00013399.

15. ČMOLÍK, Ladislav.
VIZ 13 VISUALIZATION OF TIME-ORIENTED DATA [online].
2021. [cit. 2022-01-02]. Dostupné z:
`https://docs.google.com/presentation/d/1x1LbvhLxCTo9T3lPeZ-87XQP_UfYZl25wDOswCBwr_E/edit#slide=id.p1`.
[Soubor přístupný po přihlášení do sítě ČVUT].

16. HAVRE, S.; HETZLER, E.; WHITNEY, P.; NOWELL, L.
ThemeRiver: visualizing thematic changes in large document collections.
IEEE Transactions on Visualization and Computer Graphics.
2002, roč. 8, č. 1, s. 9–20. Dostupné z DOI: `10.1109/2945.981848`.

17. TOMINSKI, Christian; ABELLO, James; SCHUMANN, Heidrun.
Axes-Based Visualizations with Radial Layouts. In:
Proceedings of the 2004 ACM Symposium on Applied Computing.
Nicosia, Cyprus: Association for Computing Machinery, 2004,
s. 1242–1247. SAC '04. ISBN 1581138121.
Dostupné z DOI: `10.1145/967900.968153`.

18. VAN WIJK, J.J.; VAN SELOW, E.R.
Cluster and calendar based visualization of time series data. In:
Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis'99). 1999, s. 4–9.
Dostupné z DOI: `10.1109/INFVIS.1999.801851`.

19. GANTT.COM. *What is a Gantt Chart?* [online].
[B.r.]. [cit. 2022-01-02]. Dostupné z: `https://www.gantt.com/`.

20. AIGNER, W.; MIKSCH, S.; THURNHER, B.; BIFFL, S.
PlanningLines: novel glyphs for representing temporal uncertainties and their evaluation. In:
Ninth International Conference on Information Visualisation (IV'05).
2005, s. 457–463. Dostupné z DOI: `10.1109/IV.2005.97`.

21. TOMINSKI, C.; SCHULZE-WOLLGAST, P.; SCHUMANN, H.
3D information visualization for time dependent data on maps. In:
Ninth International Conference on Information Visualisation (IV'05).
2005, s. 175–181. Dostupné z DOI: `10.1109/IV.2005.3`.

22. KIM, Nam Wook; CARD, Stuart K.; HEER, Jeffrey.
Tracing Genealogical Data with TimeNets. In: *Proceedings of the International Conference on Advanced Visual Interfaces*.
Roma, Italy: Association for Computing Machinery, 2010, s. 241–248.
AVI '10. ISBN 9781450300766.
Dostupné z DOI: `10.1145/1842993.1843035`.

23. NGUYEN, Phong H; XU, Kai; WALKER, Rick; WONG, BL William.
TimeSets: Timeline visualization with set relations.
Information Visualization. 2016, roč. 15, č. 3, s. 253–269.
Dostupné z DOI: `10.1177/1473871615605347`.

24. ALLEN, Robert B. A Focus-Context Browser for Multiple Timelines. In: *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*. Denver, CO, USA: Association for Computing Machinery, 2005, s. 260–261. JCDL '05. ISBN 1581138768. Dostupné z DOI: 10.1145/1065385.1065445.

25. JENSEN, Matt. Visualizing complex semantic timelines. *Derived from the World Wide Web: http://newsblip. com/tr*. 2003. Dostupné také z: https://www.researchgate.net/publication/2560605_Visualizing_Complex_Semantic_Timelines.

26. BUCHTA, Jiří; MAKOVSKÝ, Karel; ČECH, Vladislav. *Asterion: svět pro Dračí doupě*. Praha: Altar, 1999. ISBN 80-85979-28-4.

27. PATEL, Jeel. *10 Reasons to Consider Angular Framework For Web Development* [online]. [B.r.]. [cit. 2022-02-28]. Dostupné z: https://www.monocubed.com/blog/angular-framework-for-web-development/.

28. KENNETH(ANSWER), Richard. *Can PHP be used in front-end?* [online]. [B.r.]. [cit. 2022-02-28]. Dostupné z: https://www.quora.com/How-do-you-use-PHP-in-the-front-end.

29. BLOGGER, GMI. *15 Best Programming Languages for Web Development in 2022* [online]. [B.r.]. [cit. 2022-02-28]. Dostupné z: https://www.globalmediainsight.com/blog/programming-languages-web-development/.

30. TEAM, Manifera offshore. *Best programming languages for web development in 2020* [online]. [B.r.]. [cit. 2022-02-28]. Dostupné z: https://www.manifera.com/best-programming-languages-for-web-development-in-2020/.

31. TUAN(QUESTION), Bui Quang. *What are the best programming languages for building a website?* [online]. [B.r.]. [cit. 2022-02-28]. Dostupné z: https://www.quora.com/What-are-the-best-programming-languages-for-building-a-website.

32. JAVINPAUL. *Top 5 Programming languages for Web development in 2022* [online]. [B.r.]. [cit. 2022-02-28]. Dostupné z: https://medium.com/javarevisited/top-5-programming-languages-for-web-development-in-2021-f6fd4f564eb6.

33. BACKES, Marc. *HTML, CSS and JavaScript Explained For Beginners* [online]. [B.r.]. [cit. 2022-03-01]. Dostupné z: https://www.codewall.co.uk/html-css-and-javascript-explained-for-beginners/.

34. W3SCHOOLS. *Introduction to XML* [online]. [B.r.]. [cit. 2022-03-20]. Dostupné z: `https://www.w3schools.com/xml/xml_whatis.asp`.

35. MOZILLA.ORG CONTRIBUTORS. *XML introduction* [online]. [B.r.]. [cit. 2022-03-20]. Dostupné z: `https://developer.mozilla.org/en-US/docs/Web/XML/XML_introduction`.

36. W3SCHOOLS. *SVG Tutorial* [online]. [B.r.]. [cit. 2022-03-20]. Dostupné z: `https://www.w3schools.com/graphics/svg_intro.asp`.

37. MOZILLA.ORG CONTRIBUTORS. *SVG: Scalable Vector Graphics* [online]. [B.r.]. [cit. 2022-03-20]. Dostupné z: `https://developer.mozilla.org/en-US/docs/Web/SVG`.

38. PIXIJSTEAM, Mat Groves. *PixiJS* [online]. [B.r.]. [cit. 2022-03-03]. Dostupné z: `https://pixijs.com/`.

39. CONFILE. *SVG Image on Pixi.js renders very blurry* [online]. [B.r.]. [cit. 2022-03-03]. Dostupné z: `https://github.com/pixijs/pixijs/issues/936`.

40. PAL, Shukant. *Vector rendering of SVG content with PixiJS* [online]. [B.r.]. [cit. 2022-03-03]. Dostupné z: `https://javascript.plainenglish.io/vector-rendering-of-svg-content-with-pixijs-6f26c91f09ee`.

41. STORM, Photon. *Phaser* [online]. [B.r.]. [cit. 2022-03-03]. Dostupné z: `https://phaser.io/`.

42. DAVEY, Richard. *5500* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: `https://phaser.io/examples/v3/view/dwitter/5500`.

43. FSELCUKCAN. *Can we use SVG?* [online]. [B.r.]. [cit. 2022-03-03]. Dostupné z: `https://phaser.discourse.group/t/can-we-use-svg/4570`.

44. WIKIPEDIA COMMUNITY. *Phaser (game framework)* [online]. [B.r.]. [cit. 2022-03-03]. Dostupné z: `https://en.wikipedia.org/wiki/Phaser_(game_framework)`.

45. BOSTOCK, Mike. *Data-Driven Documents* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: `https://d3js.org/`.

46. MOZZILA. *Introduction to the DOM* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: `https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction`.

47. BOSTOCK, Mike. *Collapsible Tree* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: `https://observablehq.com/@d3/collapsible-tree`.

48. BOSTOCK, Mike. *Hierarchical Edge Bundling* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: `https://observablehq.com/@d3/hierarchical-edge-bundling/2`.

49. YUIN. *Two.js* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: https://two.js.org/.

50. HALAPROP. *Luniland* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: https://luniland.halabe.com/.

51. LINKEDIN LEARNING. *Data Visualization Tutorial - Overview of data visualization in Processing* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: https://www.youtube.com/watch?v=T5lRLA_Vn7o.

52. THE CODING TRAIN. *The Coding Train* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: https://www.youtube.com/channel/UCvjgXvBlbQiydffZU7m1_aw.

53. PROCESSING TEAM. *Welcome to Processing!* [online]. [B.r.]. [cit. 2022-03-15]. Dostupné z: https://processing.org/.

54. MCCARTHY, Lauren Lee. *p5.js* [online]. [B.r.]. [cit. 2022-03-16]. Dostupné z: https://p5js.org/.

55. THE CODING TRAIN. *Coding Challenge #109: Visualizing 500,000 Subscribers* [online]. [B.r.]. [cit. 2022-03-16]. Dostupné z: https://www.youtube.com/watch?v=Ae73YY_GAU8&t=547s.

56. MANY AUTHORS. *p5.js Examples* [online]. [B.r.]. [cit. 2022-03-16]. Dostupné z: https://p5js.org/examples/.

57. YWORKS. *Network Flows Demo* [online]. [B.r.]. [cit. 2022-03-20]. Dostupné z: https://live.yworks.com/demos/analysis/networkflows/.

58. YWORKS. *yWorks* [online]. [B.r.]. [cit. 2022-03-16]. Dostupné z: https://www.yworks.com/.

59. INOUE, Y.; TSURUOKA, K.; ARIKAWA, M. Spatio-Temporal Story Mapping Animation Based On Structured Causal Relationships Of Historical Events. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. 2014, roč. XL-4, s. 101–103. Dostupné z DOI: 10.5194/isprsarchives-XL-4-101-2014.

60. USER:S. *A 2-dimensional spring system.* [online]. [B.r.]. [cit. 2022-10-10]. Dostupné z: https://en.wikipedia.org/wiki/Spring_system#/media/File:Elastic_network_model.png.

61. SUSIE LU, Elijah Meeks. *Viz Palette* [online]. [B.r.]. [cit. 2022-10-10]. Dostupné z: https://projects.susielu.com/viz-palette?colors=[%22#fa8775%22,%22#ff0900%22,%22#d25800%22,%22#993f00%22,%22#ffcdaa%22,%22#bfb287%22,%22#c9a600%22,%22#fff200%22,%22#8a9800%22,%22#009800%22,%22#003500%22,%22#09ff00%22,%22#38c66f%22,%22#a7cc82%22,%22#70ffb3%22,%22#00fffb%22,%22#29ade4%22,%22#0873b3%22,%22#0000ff%22,%22#4b19a9%22,%22#a400ec%22,%22#d0ade7%22,%22#050067%22,%22#9b709a%22,

%22#dc0068%22,%22#ff29ff%22,%22#ff9bff%22,%22#b535ba%22,
%22#92315b%22,%22#8c8c8c%22]&backgroundColor=%22white%22&
fontColor=%22black%22&mode=%22normal%22.

62. D3.JS. *d3-force* [online]. [B.r.]. [cit. 2022-10-11].
Dostupné z: https://github.com/d3/d3-force.

63. SUSIE LU, Elijah Meeks. *Viz Palette* [online]. [B.r.]. [cit. 2022-10-10].
Dostupné z: https://projects.susielu.com/viz-palette.