TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Maxim Gromov 142686

KÕRGUSINFO MUUTUSTE VISUALISEERIMINE VEEBIPÕHISES KAARDIRAKENDUSES

Bakalaureusetöö

Juhendaja: Juhan-Peep Ernits
Phd

Kaasjuhendaja: Andreas Kiik

Tallinn 2017

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Maxim Gromov

22.05.2017

Annotatsioon

Käesoleva bakalaureusetöö eesmärgiks on luua veebipõhise kaardirakenduse kasutajaliides, mis võimaldab visualiseerida Eesti sildadel interferomeetriliselt mõõdetud deformatsioonipunktide liikumisi.

Autor annab ülevaate veebipõhiste kaardirakenduste arhitektuurist, toob välja rakenduse arenduses kasutatud tehnoloogiate valikut ja kirjeldab valminud rakenduse arhitektuuri. Valminud veebirakenduse nõuded on paika pandud ettevõte Datel AS poolt.

Töö tulemusena valmis kasutajasõbralik veebipõhine kaardirakendus, kus saab sildade deformatsioone jälgida ning analüüsida, kasutades erinevaid rakenduse tööriistu.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 35 leheküljel, 5 peatükki ja 12 joonist.

Abstract

Visualization of height data changes using web-based map application

The main goal of this bachelor`s thesis is implementing web based map application for visualization of interferometrically measured displacement points on Estonian bridges.

Author gives overview of web based map applications, explains the technologies used for application development and describes the finished application architecture. The requirements of application are defined by Datel AS company.

The result of this thesis is user friendly web based map application, where user can follow and analyze deformations on Estonian bridges using different tools.

The thesis is in Estonian and contains 35 pages of text, 5 chapters and 12 figures.

Lühendite ja mõistete sõnastik

JavaScript	Programmeerimiskeel, mida kasutatakse interaktiivsete veebilehtede skriptimiseks [1] .
TypeScript	*JavaScript* järelkäija, loodud Microsoft poolt, lisab juurde tüüpe ja klasse ning on mõeldud suurte rakenduste loomiseks [2] .
Angular	Google poolt arendatav *TypeScript* põhine veebiraamistik, mille abil saab luua skaleeritavaid veebirakendusi [14] .
React	Facebook pool arendatav *JavaScript* põhine veebiraamistik, mille abil saab luua suuri, skaleeritavaid veebirakendusi [4] .
SPA	*Single Page Application* - veebirakendus, mis eksisteerib ühel lehel ilma, et lehte peaks uuendama vaadete vahetamisel [5] .
Google Maps	Google poolt arendatav veebipõhine kaarditarkvara [6] .
OpenLayers	Avatud lähtekoodiga arendatav veebipõhine kaarditarkvara MetaCarta Labs poolt [7] .
EPSG:3301	Eesti kordinaateljestiku süsteem [9] .
WGS84	Geograafilise kordinaatteljestiku süsteemi maailma standart [10] .
MVC	*Model - View - Controller* arhitektuurimuster [11] .
HTML5	*Hyper Text Markup Language 5*. Keel, milles märgendatakse veebilehte [12] .
CSS	*Cascading Style Sheets*. Lihtne keel veebilehtede stiilide kirjeldamiseks [3] .
JSX	*JavaScript*, kus on võimalik kasutada HTML5 komponentide märgistuse kirjeldamiseks React raamistikus [4] .
REST	*HTTP* transpordiprotokollil põhinev veebiteenuste arhitektuur *REST* (*Representational State Transfer*) [15] .

Sisukord

Jooniste loetelu

1 Sissejuhatus

Käesoleva bakalaureusetöö eesmärgiks on luua veebipõhise kaardirakenduse kasutajaliides, mis võimaldab visualiseerida Eesti sildadel interferomeetriliselt mõõdetud deformatsioonipunktide liikumisi. Antud ülesande lahendamisel on tähtis ka samal ajal Datel AS poolt valmiv veebiteenus, millest päritakse mõõdetud andmed vastavalt kasutaja poolt valitud parameetritele kasutajaliideses.

Veebipõhine kaardirakendus valmib eraldiseisva rakendusena AngularJS 4 ja OpenLayers 4 tehnoloogiate baasil. Rakenduse demonstreerimiseks kasutan staatilisi testandmeid, mis on juhuslikult valitud. Rakenduse lähtekood on avalik ning on kättesaadav BitBucket keskkonna vahendusel https://bitbucket.org/maxim_gromov/geodisplacements/src.

Antud töös valmiva rakenduse nõuded on koostatud Datel AS poolt ning need on järgmised:

1. deformatsioonipunktide filtreerimine sillaobjekti ning selle analüüside järgi;
2. filtreeritud andmehulga põhjal kuupäevapõhise graafiku genereerimine;
3. legendi kuvamine ja objektide värvimine skaala järgi;
4. andmepunktide interaktiivne valik kaardil;
5. filtreeritud andmehulga CSV eksport;
6. Maa-ameti Eesti staatiliste kaardikihtide integreerimine.

Lõppkokkuvõttes peaks rakendus võimaldama kasutajal jälgida Eestis ehitatud sildade seisukorda ning tuvastada kriitilisemaid sildade deformatsioone.

Töö sisu on jagatud kolmeks osaks. Esimeses osas tuleb juttu veebipõhiste kaardirakenduste arhitektuuridest üldisemalt. Teises peatükis kirjeldan rakenduse nõuete

implementeerimiseks kasutatud veebitehnoloogiate valikut ning viimases osas kirjeldan rakenduse valmimise protsessi.

2 Veebipõhiste kaardirakenduste arhitektuur

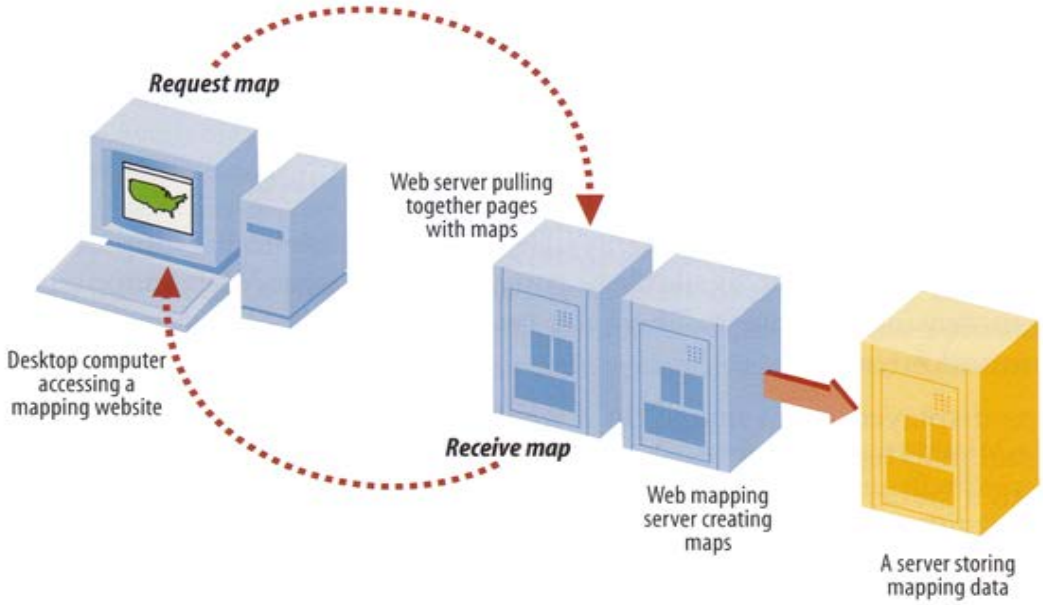
Veebipõhiste kaardirakenduste eesmärk on kujutada ruumilisi andmeid kaardil. Interaktiivsete kaartide puhul on lõppkasutajal võimalik kujutatud andmetega läbi viia erinevaid tegevusi. Kõige levinumad nendest on kaardi peal liikumine ja võimalus kaardi resolutsiooni suurendada või vähendada (*zoom*). Antud peatükk sisaldab ülevaadet veebipõhiste kaardirakenduse arhitektuuridest.

2.1 Kaardirakenduste klassifikatsioon

Veebipõhised kaardirakendused jagunevad staatilisteks ja interaktiivseteks. Staatilistes kaardirakendustes on veebilehel kujutatud pilt kaardist (võivad esineda ka erinevad andmed ja märgistused kaardil), kuid puudub interaktiivsus - kasutaja ei saa kaarti suurendada ega vähendada, rääkimata andmetepõhilistest tegevustest. Interaktiivsed kaardirakendused võimaldavad kasutajal läbi viia igasuguseid tegevusi kaardiga ja selle peal kujutatavate andmetega sõltuvalt rakenduses implementeeritud funktsionaalsusest. Antud töös valmiva rakenduse implementatsiooni raames on kasutatud interaktiivset tüüpi kaarti.

2.2 Interaktiivse kaardirakenduse arhitektuur

Interaktiivne veebipõhine kaardirakendus töötab klient - server arhitektuuri põhiselt. Vastavalt kasutaja poolt valitud kaardiresolutsioonile ning asukohale teeb veebilehitsejas töötav rakendus päringu vastavasse kaardiserverisse, kust tuleb vastuseks kasutaja poolt valitud parameetritele vastav kaardipilt, mida kuvatakse kaardialale eelmise pildi asemel [18] . Joonisel 1 on see lähenemine illustreeritud.



Joonis 1. Interaktiivse kaardirakenduse klient - server arhitektuur (Mitchell, 2005) [19] .

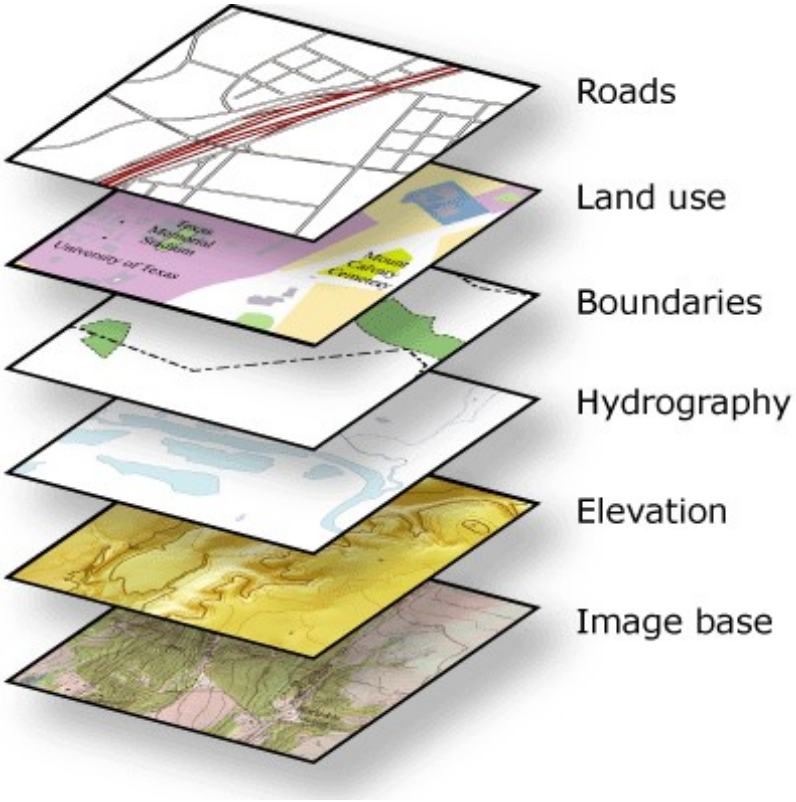
Kaartide veebiserveris hoitakse kaardipilte, mida kliendirakendused saavad pärida WMS või WMTS veebiteenuste kaudu, millest tuleb juttu järgmistes alapeatükides.

2.2.1 Kaardikihid

Interaktiivsete kaartide puhul on tähtis kaardi jagunemine kihtideks. See võimaldab kujutada erinevatest veebiserveritest saadud kaardipildid või ruumilised andmed jagatuna kihtidesse. Selline lähenemine võimaldab eristada erinevaid kaardil kujutatud objekte ja andmeid. Näiteks on võimalik peita mingi konkreetne kiht või eristada eri tüüpi kujutatavad objektid kihtide järgi. Antud rakenduse kaart koosneb mitmest kihist:

1. Maa-ameti *FOTO* kiht;
2. Maa-ameti *HÜBRIID* kiht;
3. sillapolügoonide kiht;
4. kasutaja joonestatud alade kiht;
5. kunstpeegeldajate kiht;
6. andmepunktide kiht;

7. kasutaja joonestatud alade uuritavate väärtuste keskmisi kujutatav kiht



Joonis 2. Interaktiivse kaardirakenduse kihistruktuuri näide. Alt üles: aluskaardi kiht, kõrguste kiht, veekogude kiht, piiride kiht, maakatte kiht, teede kiht [20] .

2.2.2 Projektsioonid

Projektsioonid on kartograafias mõeldud Maa asukohtade sfääriliste koordinaatide transformeerimiseks nende kujutamiseks tasapinnalistel kaartidel. Ilma projektsioonideta poleks võimalik Maa asukohti tasapinnal kujutada [21] . Paljud kaardistamise teegid kasutavad Web Mercator projektisooni (koodiga EPSG:3857) ja toetuvad WGS 84 (EPSG:4326) standardile.

Eestis on kasutusel kaardiprojektsioon EPSG koodiga 3301. Kaardiprojektsiooni parameetrite ja definitsiooni saab leida koordinaatsüsteemide registrist [9] . Antud töö raames valmiva rakenduse põhiliseks projektsiooniks kasutan Eesti kaardiprojektsiooni.

Projektsioonide defineerimiseks JavaScripti kaartide veebiraamistikes on võimalik kasutada selleks spetsiaalset ettenähtud teeki *proj4js* [22] . Selleks tuleb importida

veebirakendusse antud teek JavaScripti failina ning seejärel defineerida nõutud projektsioon. Antud kood on leitav ka koordinaatsüsteemide registrist:

```
proj4.defs("EPSG:3301","+proj=lcc +lat_1=59.33333333333334
+lat_2=58 +lat_0=57.51755393055556 +lon_0=24 +x_0=500000
+y_0=6375000 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m
+no_defs");
```

Joonis 3. Eesti projektsiooni defineerimise näide *proj4js* teegi abil [22] .

2.2.3 Andmed kaardil. GeoJSON

Lisaks aluskaartide kihtidele, kus on kujutatud ühest või mitmest allikast saadud kaardipildid, tuleb ekraanil kuvada muudest allikatest päritud andmepunktid. Nende kuvamiseks peaks igal punktil olema geomeetriline parameeter, mille järgi rakendus otsustab, mis kujuga ja kuhu kohta on vaja antud andmeid paigutada kaardi peal. Kirjeldatud parameetrit võimaldab väljendada spetsiaalne GeoJSON [26] formaat. Kui andmepunktil on olemas mistahes lisaparameetreid, nagu *nimi*, *aasta* või muu, siis sellise objekti tüübiks on *Feature*. Järgmine kood on näide sellisest objektist, millel on lisaparameeter *name*:

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Joonis 4. *GeoJSON* struktuuri näide [26] .

Selliste objektide hulk moodustab objekti tüübiga *FeatureCollection*. GeoJSON toetab erinevaid geomeetrilisi kujundeid nagu *Point*, *LineString*, *Polygon*, *MultiPoint*, *MultiLineString*, ja *MultiPolygon* [26] .

2.2.4 WMS, WMTS teenused

Web Map Service (WMS) on teenus, millelt saab pärida kaardipilte dünaamiliselt, ilma et peaks kaarte endale arvutisse allalaadima. Kaartide pildid ja muud andmed hoitakse ja hooldatakse teenusepakkuja andmebaasides. Piltide saamiseks on vaja teenusel teada küsitava ala piiride punkte ning kihi nime. Tavaliselt ala piiripunktid on genereeritud kaardiraamistiku poolt automaatselt ning arendajal jääb defeneerida, mis kihti teenuselt küsida tahetakse. Antud teenusest kaartide kättesaamine on tavaliselt kiire protsess, kuna ei küsita pilti terve maailma kohta, vaid ainult teatud ala kohta. Siiski tihtipeale võivad avalikud teenused olla koormatud suurema kasutatavuse korral. Tavaliselt hoitakse WMS teenuses kõige värskemaid pilte ja andmeid [23] . Eesti kaartide WMS teenust pakub Maa-amet.

Web Map Tile Service (WMTS) on 2010 aastal *Open Geospatial Consortium* poolt publitseeritud kaarditeenuse standard, mis võimaldab pärida nähtava ala sisse jäävaid kaardipiltide tükke suurusega 256 x 256 pikslit. Võrreldes tavalise WMS teenusega, kus tehakse üks päring, millega tuleb kaasa ekraaniulatusse jääv pilt, tehakse WMTS teenuses korraga kümneid päringuid, et katta ekraanile jääv kaardiala. WMTS teenuse piltide uuenemine toimub paar korda aastas, samas tavalise WMS teenuse poolt saadavad kaardipildid on aga igapäevaselt uuendatud. Kiiruse poolest võib praktikas märgata, et WMTS teenus, vähemalt Maa-ameti puhul, töötab kordades kiiremini [24] [25] .

2.2.5 Maa-ameti teenused

Antud töö rakenduse nõutest lähtuvalt tuleb kasutada Maa-ameti poolt WMS / WMTS teenustena pakutavad kaardipildid. Katsemeetodil järeldasin, et WMTS teenusega edastatud staatilised aluskaardid toimivad palju kiiremini. Samuti soovitab Maa-amet kasutada seda teenust koos selliste tarkvaradega nagu Google Maps, Bing Maps, Open Layers, Leaflet jne [25] . Maa-amet pakub omalt poolt ka näitelahenduse [33] OpenLayers 3 tarkvara baasil. Antud näidet kasutan ositi rakenduse implementeerimisel.

3 Veebipõhiste kaardirakenduste tehnoloogiad

Antud töö eesmärkide saavutamiseks on vaja valida sobivad tehnoloogiad, mis aitaksid luua mugava kasutajaliidesega veebipõhist kaardirakendust. Sellised tuntumad vabavaralised kaardirakenduste tehnoloogiad on näiteks Google Maps ja OpenLayers. Antud teegid võimaldavad luua veebipõhise interaktiivse kaardivaate, millel on võimalik kujutada erinevaid andmekihte.

Üksnes kaardivaatest pole piisav - on vaja luua erinevaid kasutajaliidese komponente, mis võimaldavad kasutajal rakendusega suhelda. Näiteks peaks olema võimalik lõppkasutajal andmeid pärida, filtreerida ja päritud andmete põhjal graafikut kuvada.

Sellist komponendipõhist kasutajaliidese vaadete arendamist võimaldavad mõned tänapäeval laialt levinud avatud lähtekoodiga veebiraamistikud nagu näiteks React ja AngularJS.

Tehnoloogiate valiku tegemiseks leidsin mõned üldtuntumad vabavaralised tehnoloogiad, mida kasutatakse igapäevaselt paljudes IT ettevõtetes. Samuti lähtusin Datel AS ettepanekust kasutada OpenLayers teeki ja oma olemasolevast töökogemusest AngularJS raamistikuga. Järgnevalt põhjendan tehnoloogiate valikut detailsemalt.

3.1 Veebipõhise kaardirakenduse tehnoloogiate valik

Antud peatükis pööran tähelepanu tehnoloogiate valikule ning selgitan välja nende peamised tugevamad ja nõrgemad küljed lähtudes Datel AS poolt püstitatud nõuetest.

Tänapäeval on veebipõhiste kaardirakenduste loomine muutunud arendajate jaoks palju lihtsamaks võrreldes veel mõne aasta taguse ajaga ning Internetis levib sadu erinevaid tasulisi ja ka tasuta raamistike, mis on abiks kaardirakenduste loomisel. Nende seas on kõige populaarsemad ja enimkasutatavad Google Maps ning OpenLayers. Mõlemad teegid põhinevad JavaScript programmeerimiskeelel.

3.1.1 Google Maps

Google Maps pakub suure hulga vahendeid kaardirakenduse implementeerimiseks, kuid kommertseesmärkidel on mõned neist tasulised. Samuti on Google Maps seadnud erinevad piirangud sõltuvalt rakenduse kasutajate hulgast. Näiteks, kui kaarti on laetud üle 25 000 korda 24 tunni vältel, siis rakendub lisatasu kaarditarkvara edaspidiseks kasutamiseks. Üks positiivsematest joontest on aga hea dokumentatsioon ning suure hulga näidiste olemasolu [8] .

3.1.2 OpenLayers

OpenLayers pakub samuti suure hulga funktsionaalsust ja ta on täisulatuses avatud koodiga tasuta JavaScripti teek. Üheks nõrgemaks jooneks on see, et antud tarkvara ja selle dokumentatsioon on pidevas muutumises. Kuna esialgne versioon osutus liiga koodimahukaks ja aeglaseks, siis alates kolmandast versioonist on alustatud selle ümberkirjutamist [8] . Antud töös kasutatakse neljandat versiooni, mis on kõige uuem, mistõttu on keerulisem leida sellele näidislahendusi, kuid samas on täiesti võimalik seda kasutada toetades olemasolevale dokumentatsioonile.

Oluline on mainida, et Datel AS kasutab oma projektides OpenLayers teeki, mis omakorda soodustab antud töö raames implementeeriva rakenduse ülalpidamist Datel AS poolt ka tulevikus.

Antud kaardirakenduse põhiobjektid, mida analüüsitakse, on Eesti sillad, mistõttu üks nõuetest on integreerida maa-ameti kaardikihid, mis toetavad ainult Eesti kordinaatteljestiku süsteemi projektsiooni (EPSG:3301) [9] . OpenLayers omakorda sisaldab vajalike tööriistu, mis võimaldavad transformeerida ruumiliste andmete kordinaadid Eesti kordinaatteljestiku süsteemist standardse Maa geograafilise kordinaateljestiku süsteemi (WGS84 või EPSG:4326) ja vastupidi [10] .

3.1.3 React.js

React on Facebooki poolt arendatud JavaScripti põhine avatud lähtekoodiga veebiraamistik, mis võimaldab arendajal luua mahukaid kasutajaliideseid. Reacti baasil ehitatud rakendus on üheleheline ehk SPA (ingl. k. Single Page Application) rakendus. Võrreldes AngularJS raamistikuga vastutab React MVC (Model - View - Controller)

arhitektuuris ainult vaate eest - andmete uuendamisel toimub lehe ümberjoonistamine automaatselt. Oluliseks omaduseks on see, et komponendile edastatud andmeid, mida vaates kuvatakse, pole võimalik muuta - seda nimetatakse ühesuunaliseks andmevooks (ingl. k. *one-way data flow*). Andmete muutmiseks kasutatakse aga spetsiaalseid toiminguid ehk actions. Komponentide kirjeldamiseks on kasutatud JSX keelt, mis võimaldab JavaScript koodis kasutada HTML keelt [4] .

Suur osa veebiarendajatest on vaielnud selle üle, milline veebiraamistik on parem - AngularJS või React. Kui aga detailidesse laskuda, siis selgub, et React täidab ainult vaate genereerimise eesmärki samal ajal kui AngularJS lahendab palju rohkemaid ülesandeid korraga [13] .

3.1.4 Angular

Angular on Google poolt arendatav TypeScripti põhine avatud lähtekoodiga veebiraamistik , mis võimaldab luua suuri, dünaamilisi, skaleeritavaid ning testitavaid veebirakendusi. Angular on tänapäeval üks populaarsemaid kasutajaliidese raamistike, mis annab arendajale suure koguse võimalusi, millega saab arendada modernseid, dünaamilisi üheleheküljelisi SPA rakendusi. Angular aitab lihtsalt pärida REST veebiteenustelt andmeid, viia neid sobivale kujule, luua erinevate eesmärkidega veebikomponente ning kujundada kasutajaliidest. Angulariga muutub kasutajaliidese koodi testitavus ja taaskasutatavus märgatavalt paremaks [14] .

Alates Angular 2.-st versioonist toetab veebiraamistik TypeScripti, mis teeb koodi kirjutamist veelgi ohutumaks ja testitavamaks. Samuti võimaldab see ehitada kliendirakenduses klassipõhiseid domeenimudeleid, mis hoiavad endas rakenduse andmeid ning äriloogikat. Selline arhitektuur võimaldab arendajal peita kogu äriloogika vastavasse mudelisse ning hiljem kasutada mudelipõhist loogikat erinevates komponendis.

3.1.5 PrimeNG, FontAwesome, Chart.js

PrimeNG on avatud lähtekoodiga tasuta veebikomponentide kogum arendatud PrimeTek Informatics poolt ning on mõeldud AngularJS 2 versioonile. Antud teek sisaldab ligi 100 erineva eesmärgiga konfigureeritavat kasutajaliidese komponenti.

Näiteks selleks, et kasutajale graafik ette kuvada, piisab andmete üleviimisest õigele kujule ja saadud kuju viitamisest graafiku komponendis. Antud teek nõuab tasuta CSS teegi FontAwesome kasutamist, mis võimaldab veebikomponentidele lisada erineva tähendusega ikoone, mida FontAwesome pakub. Samuti selleks, et kasutada graafikuid joonistavaid komponente, on vaja projekti lisada Chart.js, mis on samamoodi tasuta teek, mis sisaldab funktsionaalsust erinevate graafikute joonistamiseks [16] .

3.1.6 Lodash

Lodash on tasuta teek, mis hõlmab endasse suure hulga abifunktsioone tüüpiliste arendaja ülesannete lahendamiseks. Selliste operatsioonide hulka kuuluvad näiteks objekti ülesotsimine listist mingisuguse atribuudi järgi, listi filtreerimine ja sorteerimine. Lodash kasutab funktsionaalprogrammeerimise paradigmat ning pakutavad funktsioonid on hästi optimiseeritud [17] .

3.1.7 Angular Material

Angular Material on tasuta Google poolt arendatav teek AngularJS jaoks. See sisaldab suure hulga testitud veebikomponente, mis järgivad *Google Material Design* spetsifikatsioone [28] . Antud teegist kasutan ainult küljeriba komponenti (*sidebar*).

Antud töö kirjutamise ajal polnud kõik Angular Material veebikomponendid veel täielikult migreeritud toetamaks AngularJS 4.-ndat versiooni, mistõttu eelistan PrimeNG poolt pakutavaid komponente.

3.1.8 Json2csv

Json2csv on lihtne tasuta teek, mis aitab lihtsasti konvertida JSON dokumentide kogumiku CSV formaadis kirjeks, mida saab kasutajale CSV failina ära saata. Saadud CSV dokumendi tulpade pealkirjad on automaatselt genereeritud JSON dokumentide väljade nimedest [31] .

4 Kaardirakenduse implementeerimine

Antud peatükis kirjeldatakse veebipõhise kaardirakenduse implementeerimise protsessi ning tuuakse detailsemalt välja olulisemad käsitletud teemad ja esinenud probleemid. Kaardirakendus on loodud lähtuvalt Datel AS poolt esitatud nõuetest..

4.1 Arenduses kasutatud vahendid

- IntelliJ IDEA 2017.1 - JetBrains poolt loodud arenduskeskkond sisseehitatud *git* versioonihalduse, veebiarenduse ja paljude muude arendajale mugavaks tööks mõeldud tööriistadega [27] .

- Atlassian poolt veebipõhine versioonihalduse süsteem BitBucket. Antud teenus võimaldab hoida koodi igaühele kättesaadavana ja paljugi muud. Antud rakenduse lähtekood asub aadressil https://bitbucket.org/maxim_gromov/geodisplacements.

- Node.js NPM (*Node Package Manager*) - populaarne JavaScripti teekide sõltuvuste haldusvahend [29] .

- Angular CLI - AngularJS meeskonna poolt loodud CLI (*Command Line Interface*) vahend, mis võimaldab käsurealt luua AngularJS projekti, lisada uusi komponente jms [30] .

4.2 Rakenduse arhitektuur

Angular rakendusega töötamiseks on vaja installeerida Node.js ja NPM soovitatavalt viimased versioonid. Seejärel installeerida Angular CLI käsu *npm install -g @angular/cli* abil. Peale seda on võimalik kasutada Angular CLI käsurea vahendit rakenduse genereerimiseks. Antud rakenduse genereeris järgmine käsk: *ng new <rakenduse nimi>*. Edaspidises arenduse faasis on võimalik genereerida antud

vahendiga nii komponente kui ka teist tüüpi Angular rakendusele omaseid osi. Näiteks tüüpilise veebikomponendi genereerimiseks on olemas käsk *ng generate component <komponendi nimi>*. Genereeritud komponent koosneb komponendi nime kandvast kaustast, mille sees on 4 erineva laiendiga faili:

- *<komponendi-nimi>.component.ts* - põhiline komponendi funktsionaalsust kirjeldav TypeScript klass;
- *<komponendi-nimi>.component.css* - komponendi stiile kirjeldav CSS osa;
- *<komponendi-nimi>.component.html* - komponendi HTML märgistus;
- *<komponendi-nimi>.component.spec.ts* - komponendipõhised Unit testid [30] .

Sellisel viisil on genereeritud antud rakenduse erinevad komponendid. Angular CLI võimaldab rakendusega läbi viia mitmeid tegevusi:

- *ng serve* - käivitab rakenduse pordil 4200, failide redigeerimised tuvastatakse automaatselt ja rakendus kompileeritakse uuesti. Antud käsk sobib rakenduse arendamiseks;
- *ng test* - käivitab rakenduse testid ja genereerib tulemused HTML lehe kujul. Reageerib samuti muutustele rakenduse failides;
- *ng build* --<dev | prod> - genereerib lõpliku rakenduse failid etteantud keskkonna konfiguratsiooniga *dist* kausta. Genereeritud failid saab paigaldada mistahes HTTP serverile.

4.2.1 Rakenduse struktuur

Töö raames arendatava rakenduse struktuurist võib välja tuua mõned tähtsamad konfiguratsioonifailid. Näiteks:

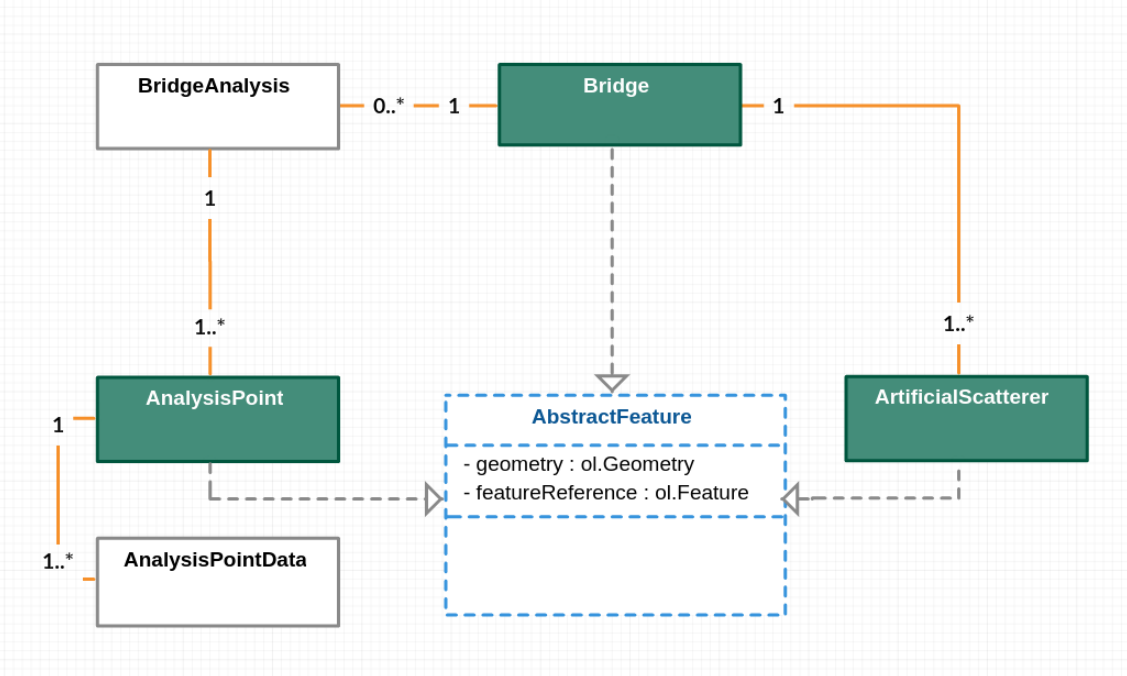
- package.json - sisaldab nimekirja rakenduses kasutatavatest teekidest ja NPM käskude kirjeldusi;

- .angular-cli.json - rakenduse põhiline konfiguratsioonifail, kus on seadistatud enamus rakenduse erinevatest parameetritest. Parameeter *scripts* koosneb erinevatest JavaScript failidest, mille kood lisatakse rakendusele kompileerimise käigus.

Rakenduses arendatud veebikomponendid ja rakenduse põhikood asuvad kaustas src/app. Vaatamata sellele, et tavalehitsejad ei toeta TypeScript'i, koosneb lõplik kompileeritud rakendus ainult JavaScript, HTML ja CSS failidest. Kompileeritud rakendus tekib dist kausta.

4.2.2 Rakenduse domeenimudel

Osa rakenduse funktsionaalsust peitub domeenimudelis, mis paikneb kaustas *domain*. Antud rakenduse põhiobjektiks on sild, millel on 0 või rohkem analüüsi. Samuti võivad sillal olla kunstpeegeldajad, mis hõlbustavad punktide mõõtmist sillal satelliidi abil. Silla analüüs defineerib sillal asuvate punktide mõõtmisi teatud ajavahemikul mõõdetuna teatud nurga all. Analüüsile vastab hulk mõõtepunkte, mille põhilisteks mõõteparameetriteks, mida antud rakenduses visualiseeritakse, on kumulatiivne muutus ja kiirendus.



Joonis 5. Rakenduse domeenimudel.

Antud rakenduses kuvatakse kaardile järgmised objektid (joonisel rohelist värvi):

- polügonikujuline sild – *Bridge.ts;*
- punktikujulised silla kunstpeegeldajad – *ArtificialScatterer.ts;*
- silla mõne analüüsi mõõdetavad punktid – *AnalysisPoint.ts.*

Objektorienteeritud programmeerimine võimaldab antud objektidel pärida mingit ühist klassi, mis sisaldab kõigile kolmele omast funktsionaalsust. Antud rakenduses on sellise klassi nimeks *AbstractFeature*, mille konstruktoris initsialiseeritakse OpenLayers'i tüüpi geomeetriline objekt (sõltuvalt sisendandmetest võib see olla *Point*, *Polygon* või mingi muu geomeetriline objekt) koos geograafiliste WGS84 koordinaatidega, mis on kohe ka transformeeritud Eesti koordinaatteljestiku süsteemi. Lisaks on antud klassiga päritud sellised funktsioonid nagu *initFeature()* ja *getStyleFunction()*. Esimene genereerib objekti väljale *featureReference* OpenLayers'ile omase objekti, mis on valmis kaardile kuvamiseks, kusjuures sellele objektile antakse kaasa ka kõik väljad nii, et hiljem selle leidmisel kaardi pealt on võimalik kätte saada esialgse punkti *AnalysisPoint* tüüpi objekt (*feature.get('pointReference')*). Teisel funktsioonil puudub implementatsioon - seda implementeerib päriv objekt, võimaldades defineerida talle erilise stiilide funktsiooni.

Samuti on rakenduses mõned objektid, mida ei kanta kaardile:

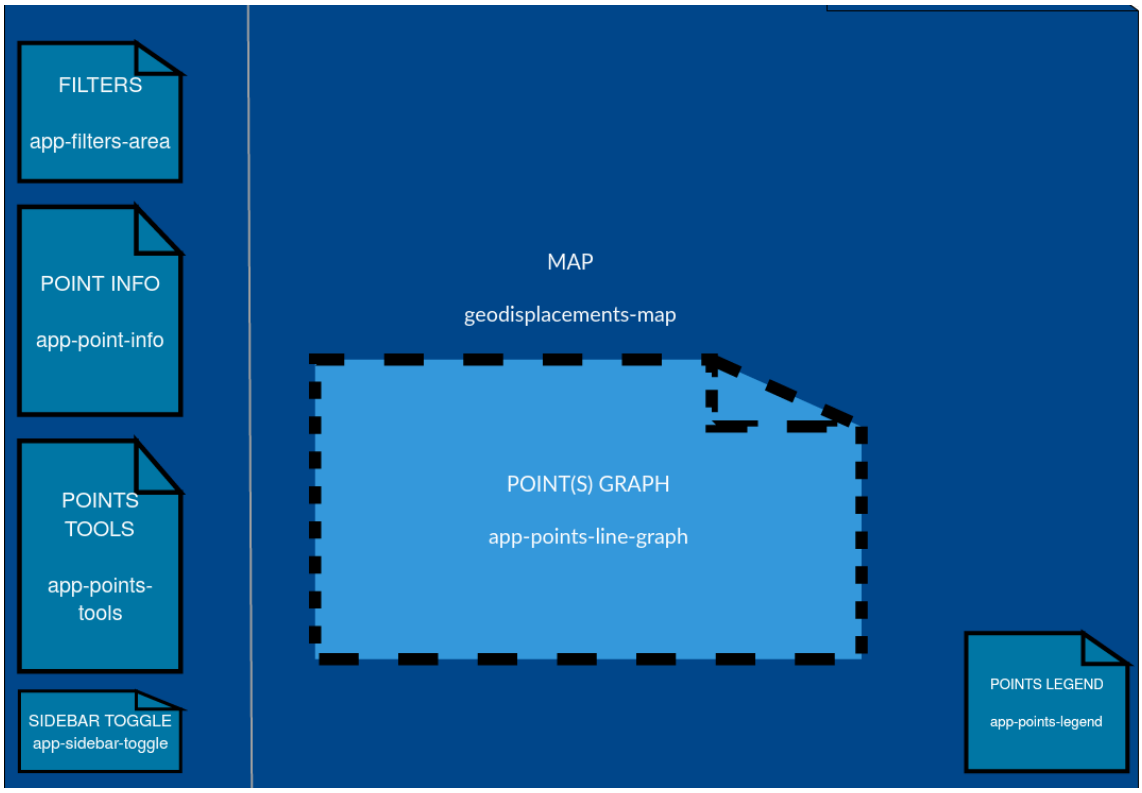
- silla analüüs – *BridgeAnalysis.ts;*
- filtrid – *FeatureFilters.ts;*
- legend – *Legend.ts;*
- punkti detailsemad andmed – *AnalysisPointData.ts;*
- kaardi peal kuvatavate kihtide tüüp – *FeatureType.ts.*

Rakenduses kasutatud domeenipõhine lähenemine aitab elimineerida koodikordusi ja aitab koodi paremini struktureerida. Samuti teeb see arendajale arusaadavaks, mis tüüpi objektiga hetkel tegu on.

4.2.3 Rakenduse veebikomponendid

Valmisrakendusel on 7 erinevat veebikomponenti:

1. *geodisplacements-map* - rakenduse peakomponent, kuhu on paigutatud ülejäänud komponendid. Jagab lehe kaheks - vasakul pool kasutajasõbralik valikute riba ning kaart paremal pool;
2. *filters-area* - võimaldab kasutajal pärida andmeid sildade, sillaanalüüside ja mõõtepunktide kohta;
3. *sidebar-toggle* - nupp küljeriba peitmiseks ja avamiseks;
4. *point-info* - tabel punkti detailsemate andmetega;
5. *points-tools* - võimaldab läbi viia interaktiivseid tegevusi päritud andmetega - sillaobjekti juurde fokuseerimine, kaardi peal punktide valimine, graafiku kuvamine ja eksport CSV faili;
6. *points-legend* - kaardi paremas alumises nurgas asuv legend, mille järgi kaardi peal kujutatud punktid värvitakse;
7. *points-line-graph* - eraldi hüpikaknas andmete põhjal genereeritud graafik.



Joonis 6. Rakenduse veebikomponentide skeem.

Antud joonisel on ülevaade veebikomponentide paigutamisest veebilehel. Roheka värviga komponendid on kasutatud rakenduse põhikomponendis *geodisplacements-map*. Helesinine kast on punktide kohta graafiku kuvamise veebikomponent (modaalaken), mida on kasutatud kasutaja tööriistade komponendis.

Rakenduse avamisel on võimalik näha jooksvaid vihjeid ülemises paremas nurgas, mis hõlbustavad kasutaja kiiret kohanemist antud rakenduse kasutajaliidesega.

4.2.4 Rakenduse teenused

Rakenduse kaustas *services* on implementeeritud kolme teenust:

1. *DataService* - antud teenus pärib andmeid veebiteenusest ning muudab päritud JSON objektid vastavateks domeeni mudeli objektideks;

2. *MapService* - antud teenuses on arendatud mitmeid funktsioone, mis võimaldavad rakenduse komponentidel läbi viia erinevaid tegevusi OpenLayers kaarditarkvaraga. Samuti initsialiseeritakse antud teenuses rakenduse esialgne

kaart koos kõikide kihtidega, lisatakse ka Eesti koordinaateljestiku süsteemi definitsioon;

3. *GraphMathService* - antud teenuses on realiseeritud lineaarse regressiooni ja keskmise joone andmehulkade genereerimise funktsionaalsust. Arvutused baseeruvad andmepunktide deformatsiooni andmetel läbi kogu analüüsi perioodi.

Järgnevates peatükkides on seletatud iga funktsionaalsuse osa detailsemalt ja toodud mõned joonised valmisrakendusest.

4.3 Maa-ameti aluskaardid WMTS teenuse kaudu

Üks põhinõudmisi oli integreerida Maa-ameti aluskaardid. Maa-amet pakub kasutamiseks kas WMS või WMTS teenust, kuid katsetamise käigus selgus, et WMTS teenusega tulevad pildid rakendusse palju kiiremini. Samuti on see ka Maa-ameti soovitus OpenLayers teegiga töötamisel.

Rakenduse tasemel on loodud kaarditarkvaraga suhtlev teenus MapService kaustas *services*, kuhu on paigutatud põhilised kaardiga seotud funktsioonid. Teenuse initsialiseerimisel luuakse OpenLayers tüüpi kaardiobjekt ja paigutakse see teenuse väljale *map*, mis on hiljem mistahes komponendist kättesaadav.

WMTS teenusega sidumiseks on kasutatud OpenLayers'i *source.XYZ* tüüpi objekti, mille parameetrite seas saab defineerida teenuse aadressi.

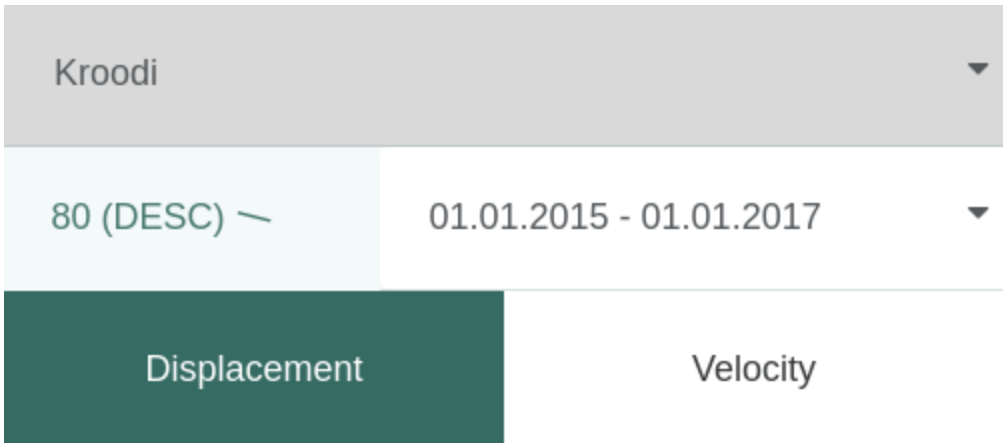
Joonis 7. Maa-ameti Eesti aluskaardi kihid WMTS teenuse kaudu.

Antud rakenduses on kujutatud kaardi peal kahte erinevat kihti:

- HÜBRIID - teenuse aadress http://tiles.maaamet.ee/tm/s/1.0.0/hybriid/{z}/{x}/{-y}.png
- FOTO - teenuse aadress http://tiles.maaamet.ee/tm/s/1.0.0/foto/{z}/{x}/{-y}.jpg

4.4 Andmete filtreerimine ja pärimine

Rakenduse avamisel tehakse koheselt päring, millega saadakse kätte veebiteenusest kõik olemasolevad sillad. Saadud sildadest saab kasutaja teha valiku. Kui sild on valitud, kuvatakse kaardile sillapolügon, silla kunstpeegeldajad ning kasutaja kaardivaade fokuseeritakse valitud sillale. Samal ajal tehakse automaatne päring, et laadida sillale vastavad analüüsid. Seejärel saab kasutaja teha silla analüüside seast valiku. Kui analüüs on valitud, tehakse mõõtepunktide päring antud analüüsi järgi ning saadud punktid kannab rakendus kaardi peale. Sellest hetkest on kasutajal võimalik analüüsida ning uurida valitud mõõtepunkte.



Joonis 8. Mõõtepunktide filtreerimine silla ja analüüsi valikuga.

Joonisel on näha, et on valitud Kroodi silla analüüs kindla ajavahemiku ja orbiidi kohta. Samuti on võimalik määrata, millist punktide näitajat uuritakse.

4.5 Mõõtepunktide uurimine

Antud peatükis on välja toodud erinevad funktsionaalsuse osad, mis võimaldavad punkte uurida ja analüüsida. Kaardi peal punkti valides saab näha küljeribal tabeli, kus on kuvatud punkti detailsemad andmed valitud perioodi kohta.

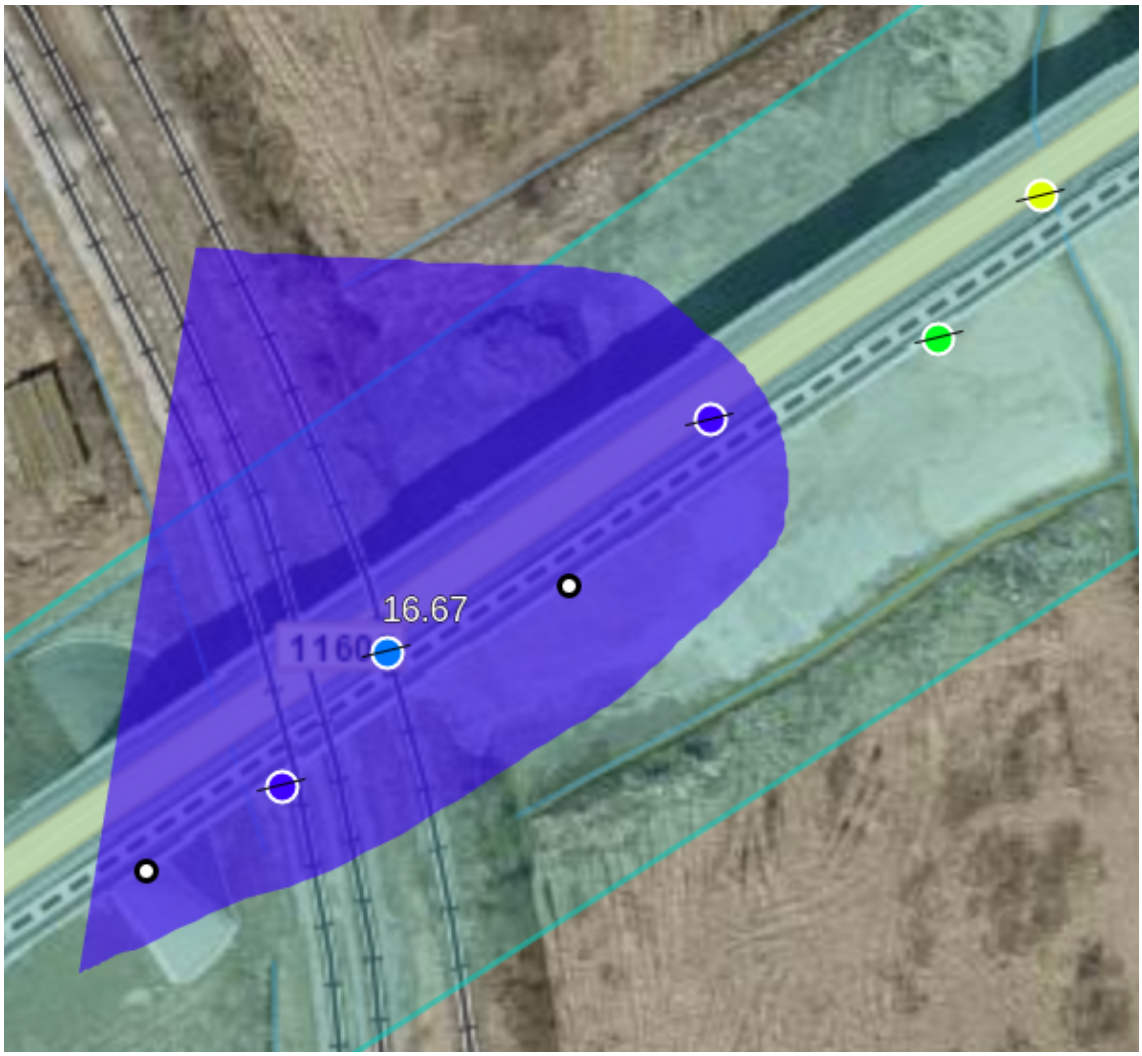
Tabeli all on tööriistade riba, mis võimaldab punkte detailsemalt analüüsida. Nupp *FOOKUS* fokuseerib kaardivaate valitud sillale. Nupuga *CSV* saab alla laadida andmed valitud punktide kohta.

Joonis 9. Mõõtepunkti detailsemad andmed ja tööriistariba.

4.5.1 Punktide alamhulga valimine

Tööriistaribal saab aktiveerida nupu *JOONIS*, mis võimaldab kasutajal valida soovitud punktid kaardil vastavat ala piirjoonega tähistades. Rakendus tuvastab ala sisse jäänud punktid, arvutab alas olevate punktide uuritava väärtuse keskmise ning värvib ala vastavalt legendi skaalale. Samuti kuvatakse keskmine väärtus valitud alale. Ala valimise väljalülitamiseks on vaja uuesti vajutada samale nupule. Seejärel kasutajalt küsitakse, kas kustutada joonisele jäänud alad või mitte.

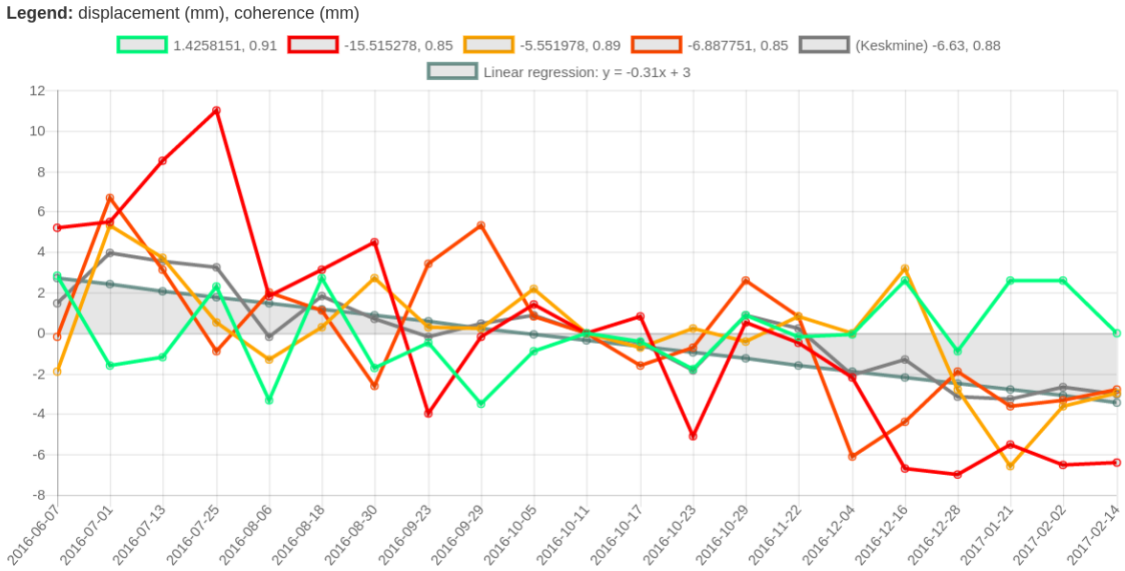
Antud tööriista saab kasutada ka graafikul kuvatavate punktide valimiseks graafikul esitamiseks. Peale punktide valikut tuleb vajutada nuppu *GRAAFIK* (ilma joonestamise režiimi väljalülitamata).



Joonis 10. Punktide alamhulga valimine ja keskmine väärtus.

4.5.2 Punktide nihkumist väljendav lineaarne graafik

Üks nõuetest oli kuvada graafik punkti(de) kohta. Graafikul on võimalik jälgida punktide nihet ajas, punktide keskmist nihet ajas ning lineaarset regressiooni, mille abil on võimalik deformatsiooni käitumisele hinnang anda. Graafikut on võimalik kuvada kas ühe punkti kohta (klikkides punkti peale), kõikide punktide (klikkides nupule *GAAFIK*) või teatud punktide kohta (filtreerides nad punktide valimise tööriistaga).



Joonis 11. Punktide nihe ajas, keskmine ja lineaarne regressioon.

Lineaarse regressiooni arvutamiseks on kasutatud *NPM* teek nimega *regression* [32] .

4.5.3 Legendi kuvamine

Rakenduse kaardiala paremas alumises nurgas asub värviskaalaga legend, mille minimaalset ja maksimaalset väärtust on võimalik suurendada või vähendada vastavate nuppudega. Väärtuste mugavaks muutmiseks on võimalik kasutada hiire ratast. Muutmistega kaasneb kaardi peal olevate punktide ja joonestatud alade ümbervärvimine vastavalt skaala väärtustele.

Joonis 12. Punktide värviskaalaga legend

4.6 Rakenduse testimine

Tarkvara kvaliteedi tagamiseks tuleb see läbi testida. Antud töö raames valmiv rakendus sisaldab hulka automaatteste veebikomponentide kriitilisele funktsionaalsusele. Rakendus tervikuna (sildade valimisest kuni graafiku kuvamiseni) on läbi testitud töö autori ja Datel AS meeskonna poolt ka käsitsi.

Angular raamistik võimaldab implementeeritud komponentide funktsioone ja olekuid testida. Näiteks on võimalik visualiseerida HTML elemendile klikkimist ja seejärel kontrollida, kas antud komponendi teatud atribuut on muutunud õigeks või mitte. Selline testimise viis on töömahukas, sest testi koostamiseks on vaja läbi teha hulk eeltööd:

- veebiteenusega suhtlus on vaja *mock*'ida testandmetega - antud rakenduses on selleks koostatud *DataServiceMock*, mis pärib originaalse teenuse *DataService* funktsioone, ning tagastab sissekirjutatud test andmed;
- enne testimist on vaja algväärtustada komponentide muutujad.

5 Kokkuvõte

Käesoleva bakalaurusetöö eesmärgiks oli luua veebipõhine kaardirakendus, mis visualiseerib Eestis paiknevatel sildadel interferomeetriliselt mõõdetud deformatsioonipunktide liikumisi.

Töö käigus uuriti kaardirakenduste arhitektuure ning toodi välja tähtsamad teemad vastavalt ettevõtte poolt esitatud nõuetele. Töö teises osas on kirjeldatud rakenduse arendamisel kasutatud tehnoloogiaid ja teeke ning põhjendatud nende valikut. Viimases osas on mainitud arenduses kasutatud vahendeid, kirjeldatud rakenduse arhitektuuri peamiseid aspekte ja toodud välja olulisemad funktsionaalsused.

Autori poolt seatud eesmärk luua kaardipõhine punkte visualiseeriv veebirakendus sai teostatud. Arendamisel on jälgitud puhta koodi printsiipe. Rakenduse ühendus ettevõtte poolt loodud veebiteenusega on tehtud võimalikult lihtsaks. Datel AS on loodud rakendusega väga rahul ning kindlasti võtab selle kasutusse.

Kasutatud kirjandus

[1] „ECMAScript® Language Specification“ [WWW]. https://www.ecma-international.org/ecma-262/5.1/. [19.05.2017]

[2] „TypeScript“ [WWW]. https://www.typescriptlang.org/. [01.05.2017].

[3] ”CSS Snapshot 2017“ [WWW]. https://www.w3.org/TR/CSS/. [19.05.2017].

[4] „React (JavaScript library)“ [WWW]. https://en.wikipedia.org/wiki/React_(JavaScript_library). [11.04.2017].

[5] „Single-page applications“ [WWW]. https://www.codeschool.com/beginners-guide-to-web-development/single-page-applications. [01.05.2017].

[6] „Google Maps“ [WWW]. https://developers.google.com/maps/documentation/. [01.05.2017].

[7] „OpenLayers“ [WWW]. https://openlayers.org/. [01.05.2017].

[8] T.Bacinger, „Survey of the Best Online Mapping Tools for Web Developers: The Roadmap to Roadmaps“ [WWW]. https://www.toptal.com/web/the-roadmap-to-roadmaps-a-survey-of-the-best-online-mapping-tools. [11.04.2017]

[9] „Estonian Coordinate System of 1997“ [WWW]. https://epsg.io/3301. [11.04.2017]

[10] “World Geodetic System” [WWW]. https://en.wikipedia.org/wiki/World_Geodetic_System. [11.04.2017]

[11] “Model-View-Controller” [WWW]. https://et.wikipedia.org/wiki/Model-View-Controller. [12.04.2017]

[12] “HTM5 Specification” [WWW]. https://www.w3.org/TR/html5/. [19.05.2017]

[13] Cleveroad. (2017). “React vs Angular: two sides of Javascript.” [WWW]. https://www.cleveroad.com/blog/react-vs-angular-ultimate-performance-research-2017. [19.05.2017]

[14] “Angular” [WWW]. https://angular.io/. [12.04.2017]

[15] “Representational state transfer (REST)” [WWW]. https://en.wikipedia.org/wiki/Representational_state_transfer. [12.04.2017]

[16] “PrimeNG” [WWW]. https://www.primefaces.org/primeng/#/. [13.04.2017]

[17] “Lodash” [WWW]. https://lodash.com/. [13.04.2017]

[18] K.Kommana. (2013) Implementation of a Geoserver application for GIS data distribution and manipulation: Master’s thesis. Stockholm University, Stockholm. [WWW]. https://www.diva-portal.org/smash/get/diva2:640096/FULLTEXT01.pdf. [18.04.2017]

[19] Mitchell, T. (2005). Web Mapping Illustrated. Sebastopol, CA, USA: Oreilly & Associates Inc.

[20] “What are map projections?” [WWW]. http://desktop.arcgis.com/en/arcmap/10.3/guide-books/map-projections/what-are-map-projections.htm [18.04.2017]

[21] “Map projections” [WWW]. https://en.wikipedia.org/wiki/Map_projection [18.04.2017]

[22] “Proj4js” [WWW]. http://proj4js.org/ [19.04.2017]

[23] “WMS - Web Map Services” [WWW]. http://www.mngeo.state.mn.us/chouse/wms/ [19.04.2017]

[24] “Web Map Tile Service” [WWW] https://en.wikipedia.org/wiki/Web_Map_Tile_Service [19.04.2017]

[25] “Maa-amet: TMS, WMS-C ja WMTS teenused” [WWW] http://geoportaal.maaamet.ee/est/Teenused/Avalik-WMS-teenus/TMS-WMS-C-ja-WMTS-teenused-p481.html [19.04.2017]

[26] “GeoJSON” [WWW] http://geojson.org/ [19.04.2017]

[27] “IntelliJ IDEA” [WWW] https://www.jetbrains.com/idea/ [19.04.2017]

[28] “Angular Material - Introduction” [WWW] https://material.angularjs.org/latest/ [19.04.2017]

[29] “Node.JS NPM” [WWW] https://www.tutorialspoint.com/nodejs/nodejs_npm.htm [19.04.2017]

[30] “Angular CLI” [WWW] https://cli.angular.io/ [19.04.217]

[31] “json-2-csv” [WWW] https://www.npmjs.com/package/json-2-csv [01.05.2017]

[32] “Regression” [WWW] https://www.npmjs.com/package/regression [13.05.2017]

[33] “Maa-ameti kaardirakenduse näide OpenLayers 3 baasil” [WWW] http://www.maaamet.ee/rr/ol3/ [14.05.2017]