KARELIA-AMMATTIKORKEAKOULU
Tietojenkäsittelyn koulutusohjelma

Peltola Ville-Santeri

RESPONSIIVISEN KÄYTTÖLIITTYMÄN SOVELTAMINEN LIFERAY-YMPÄRISTÖSSÄ

Opinnäytetyö
Huhtikuu 2015

Karelia AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Huhtikuu 2015
Tietojenkäsittelyn koulutusohjelma

Karjalankatu 3
80200 JOENSUU
013 260 600

Tekijä
Ville-Santeri Peltola

Nimeke
Responsiivisen käyttöliittymän soveltaminen Liferay-ympäristössä

Toimeksiantaja
Pohjois-Karjalan Tietotekniikkakeskus Oy

Tiivistelmä

Responsiivisella web-suunnittelulla tarkoitetaan sitä, että sivusto ottaa huomioon sitä selaavan laitteen näytön koon. Sivuston elementtien asemointia muuttamalla ja kuvia pienentämällä parannetaan käyttäjäystävällisyyttä eri alustoilla, eikä jokaiselle laitetyypille tarvitse tehdä omaa sivua. Responsiivisuus web-kehityksessä on kasvattanut suosiotaan 2010-luvulla. Älypuhelinten kasvava markkinaosuus voidaan yhdistää responsiivisuuden kysynnän kasvuun.

Opinnäytetyö toteutettiin toiminnallisena ja siinä etsittiin keinoja muuttaa toimeksiantajan Pohjois-Karjalan Tietotekniikkakeskus Oy:n asiakkaiden sivustot toteuttamaan responsiivisen verkkosuunnittelun periaatteita. Tärkeänä toimeksiannon kriteerinä pidettiin syntyvien ratkaisujen kustannustehokkuutta.

Työn tieto-osuudessa tarkastellaan Liferay-portaalia ja sen rakentumista sekä responsiivisen websuunnittelun yleisiä periaatteita. Opinnäytetyön toiminnallisessa osuudessa kuvataan toimeksiantona saadun projektin eteneminen ja tulokset.

Työn aikana löydetystä ratkaisusta esitellään malli PTTK Oy:n demosivustossa. Toimeksiantajalle tehtiin myös kirjalliset ohjeet ratkaisun saavuttamiseksi myöhemmissä tuotantoratkaisuissa.

Kieli
suomi

Sivuja
43

Asiasanat
Liferay, responsiiviset käyttöliittymät, mobiili

Karelia
UNIVERSITY OF APPLIED SCIENCES

Thesis
April 2015
Degree Proragramme in Business Information Technology

Karjalankatu 3
FI 80200 JOENSUU
FINLAND
013 260 600

Author
Ville-Santeri Peltola

Title
The Application of Responsive Design in Liferay Environment

Commissioned by
Pohjois-Karjalan Tietotekniikkakeskus Oy

Abstract

Responsive web design means that the website, that uses the design, will scale according to the device screen size. Better usability is achieved by changing the layout of the page and re-sizing pictures of the website. This also means that there is no need to make multiple sites for different device types. This design-philosophy has been on the rise since 2010 and it can be linked to the rise of the smart phones.

The thesis was done according to practice-based-thesis implementation. The aim of the thesis is to find ways to convert the Pohjois-Karjalan Tietotekniikkakeskus Oy's customer sites to follow the principles of responsive web design. One commissioner based criteria for the results of the thesis was cost-efficiency.

In the theoretical part of the thesis, Liferay portal and the way it is built are described. The basic principles of responsive design are also presented. The progression and results of the project are explained at the end of the thesis.

The found results of the thesis are presented in the PTTK Oy's demo-website. Furthermore, written instructions regarding how to achieve the solution for later production usage were given to commissioner.

Language
Finnish

Pages
43

Keywords

Liferay, responsive design, mobile

Sisältö

1 Johdanto

Internetin käyttö on yleistynyt kotitalouksissa 2010-luvulla ja jopa 86 prosenttia 16-89-vuotiaista käytti internetiä vuonna 2014. Suomessa internetiä monta kertaa päivässä käyttävien määrä on myös noussut kaikissa ikäryhmissä vuonna 2014. 16-89-vuotiaiden internetin käyttö matkapuhelimella muualla kuin kotona tai työpaikalla nousi vuonna 2014 seitsemällä prosentilla 54 prosenttiin. (Suomen virallinen tilasto(SVT) 2014.) Älypuhelimet ja niihin saatavat nopeat internetyhteydet lisäävät internetin käyttöä entisestään. Mobiilialustat aiheuttavat uusia ongelmia sivujen kehittäjille: pienentynyt näyttökoko sekä alustan hiirettömyys aiheuttavat haasteita sivustojen käytettävyyttä suunniteltaessa. Käytettävyydeltään puutteellinen sivusto voi johtaa jopa asiakkaiden vähenemiseen tai sivuston kävijämäärien laskuun.

Opinnäytetyön tarkoituksena on selvittää Pohjois-Karjalan Tietotekniikkakeskus Oy:lle mahdollisuuksista muuttaa asiakkaiden nykyiset Liferay-portaalipohjaiset sivut responsiivisen web-suunnittelun periaatteita noudattaviksi. Asiakkaiden sivustot ovat jo lähes kaikilla valmiit ja useimpia voidaan jo selata. Responsiivisuuden mahdollisuutta on mietitty jälkikäteen. Koska sivuston tietovarannot ovat laajat, täytyy sen tarjota selkeä käyttöliittymä asiakkaalle tietojen saavuttamiseksi ja hyödyntämiseksi. Kuntien sivut useasti tarjoavat kävijöilleen niin tietoa kunnasta kuin ohjeet erilaisiin kunnan sisäisiin käytäntöihin. Ihmiset myös voivat odottaa tietojen olevan helposti saatavissa.

Pohjois-Karjalan Tietotekniikkakeskus Oy, PTTK Oy, tarjoaa omistajillensa yhteisien ja yhteiskäyttöisten tietoviestintätekniikoiden ratkaisuja sekä myös ylläpitää ja kehittää niitä (Pohjois-Karjalan Tietotekniikkakeskus Oy 2015). Yhtiön omistajia ovat Pohjois-Karjalan alueen kunnat ja kaupungit. Muun muassa Joensuun kaupunki sekä Rääkkylän kunta kuuluvat Pohjois-Karjalan Tietotekniikkakeskus Oy:n asiakkaisiin.

Opinnäytetyön kehittämistehtävänä on tutkia, miten PTTK Oy:n asiakkaiden sivut muutetaan responsiiviseen muotoon mahdollisimman yksinkertaisesti ja

vähin resurssein. Asiakkaiden sivut on toteutettu Liferay Community Edition 6.1 -portaalilla ja useimmat asiakkaat käyttävät samaan layoutiin pohjaavaa ratkaisua.

Opinnäytetyössä esitellään Liferay-julkaisualustaa yleisesti ja pohditaan, kuinka responsiivisuus on saavutettavissa portaalissa. Opinnäytetyön tieto-osuudessa tarkastellaan yleisesti, kuinka näitä keinoja voidaan käyttää toimeksiannon tapauksessa. Luvussa kolme esitellään responsiivisuuden perusperiaatteita ja avataan niitä hieman tarkemmin. Myös yleisimmät CSS-mallit, joiden avulla responsiivisuus voidaan saavuttaa, esitellään luvussa kolme. Luvussa neljä käydään läpi opinnäytetyön teknistä puolta, tutkimus- ja testausvälineistöä, kehittämisympäristöä sekä toimeksiantoprojektin työnkuvaa. Opinnäytetyön lopussa esitellään tutkimustulokset ja opinnäytetyön prosessin arviointi sekä pohdinta opinnäytetyöstä ja sen onnistumisesta.

Opinnäytetyön tuotoksena syntyi demosivu, jossa löydettyjä ratkaisuja ja sovellettuja kokonaisuuksia voidaan tarkastella käytännössä. Sen lisäksi luotiin yleiset ohjeet siitä, kuinka ratkaisuja voidaan hyödyntää myöhemmässä tuotantovaiheessa, jos PTTK Oy päätyy tarjoamaan responsiivisia sivuja asiakkailleen. Ohjeita ja demosivun ratkaisua voivat hyödyntää PTTK Oy:n työntekijät toteuttaessaan mahdollista siirtymistä responsiiviseen malliin, minkä yhtiö tulee arvioimaan vuosien 2015 ja 2016 välissä.

2 Liferay

Liferay-portaali tarjoaa erilaisia toimintoja käytettäväksi sivuston ylläpitäjille ja käyttäjille (Liferay Inc. 2015a). Keskeisiksi Liferayn portaalitoiminnoiksi voidaan katsoa web-alusta (Web Platform), web-sisällönhallintajärjestelmä (WCM System), integrointialusta (Integration Platform), yhteistyöalusta (Collab Platform) sekä sosiaalinen alusta eli Social Platform (Liferay Inc. 2015b).

Web-alustaan kuuluu muun muassa ominaisuus, joka näyttää käyttäjälle sivuston erilailla riippuen tämän oikeuksista, roolista sekä statuksesta. Liferay myös tarjoaa helpon tavan sivuston ylläpitäjälle hallinnoida portaalin sivujen eri kieliversioita. Portaaleissa sivustot rakentuvat useasti erilaisista upotettavista elementeistä kuten portleteista tai plugineista sekä muokattavista ulkoasullisista piirteistä. Näitä yhdistelemällä sivuston kehittäjät voivat käyttää vanhoja elementtejä uusien sivustojen luomiseen. (Liferay Inc. 2015b.)

Web-sisällönhallintajärjestelmässä portaalissa voidaan jakaa roolituksia, joiden mukaan käyttäjien toimintoja voidaan rajoittaa. Käytännössä tämä voi esimerkiksi tarkoittaa sitä, että sisällöntuottajien on näytettävä sisältöön tekemänsä muutokset korkeamman tason omaavalle henkilölle, jolla on oikeudet julkaista sisältö. (Liferay Inc. 2015c.) Näin voidaan varmistaa sivustolla julkaistavien elementtien tyylin pysymisen samana sekä jakaa käyttäjien taitotason mukaan vastuuta sivuston päivittämisestä. Web-sisällönhallintajärjestelmässä toisena tärkeänä ominaisuutena on, että portaali voi toimia säilytyspaikkana materiaaleille, dokumenteille ja tiedostoille (Liferay Inc. 2015c).

Integraatioalusta keskittyy sivujen ja toimintojen keskittämiseen yhteen portaaliin. Organisaatiolla voi olla monia sivustoja ja ohjelmia, joille pitäisi tehdä omat sivunsa. Portaalin avulla nämä voidaan keskittää järkevästi. Organisaation sivujen alla voi olla asiakkaan oma sivustoportaali sekä työntekijöiden oma portaali. (Liferay Inc. 2015d.)

Yhteistyöalustassa portaali tarjoaa tavan eri henkilöille luoda omia yhteisöjä, jotka voivat toimia yhteistyössä keskenään. Yhteisöt voivat lisätä yhteistyöhön liittyviä välineitä omalle web-alueelleen. Tällaisia ovat esimerkiksi blogit, wikitoiminnot ja kalenteri. Välineistöä voidaan jakaa myös eri projektien ja tiimien kesken, jolloin yhteistyötoimintoja voidaan organisoida yhtiön vaatimusten mukaan. (Liferay Inc. 2015e.)

Sosiaalinen alusta Liferay-portaalissa tarkoittaa portaalin tarjoamia sosiaalisia applikaatioita. Ne voivat hyödyntää sosiaalista identiteettiä, tietoja ja toimintoja

palveluissaan. Liferay-portaali voi jakaa käyttäjälle virallisen identiteetin sekä sosiaalisen identiteetin. Käyttäjällä voi olla Liferay-portaalissa organisaation virallisempi roolipohjainen ympäristö, sekä sosiaalisen identiteetin perusteella luotava epävirallisempia suhteita omaava ympäristö. (Liferay Inc. 2015f.)

Liferaysta on kaksi erillistä versiota: Enterprise Edition ja Community Edition. Versioiden ominaisuudet vaihtelevat hieman, vaikka peruspiirteet ovat samat. Enterprise Edition on maksullinen versio portaalista ja tarjoaa erilaisia lisäominaisuuksia, kuten bugien korjauksia Liferay Oy:n puolelta, Liferay kehitysstudion (Liferay Developer Studio) ja muita käyttäjää helpottavia ominaisuuksia. Community Edition on ilmainen GNU LGPL:n (GNU Lesser General Public License) alaisuudessa jaettava versio. Lisäksi se on erittäin yhteisöpainotteinen. (Liferay Inc. 2015g.)

Liferay-sivusto koostuu useasti erilaisista portleteista, joita voidaan hyödyntää sivustoja rakentaessa (Liferay Inc. 2015h). Liferay tarjoaa monia portletteja jo heti asennuksen jälkeen. Nämä portletit jakaantuvat kolmeen pääkategoriaan: sisällönhallintaan ja web-julkaisuun sekä yhteistyö- ja sosiaalisiin portletteihin. (Liferay Inc. 2015i.)

Kirjoitushetkellä sekä Community Editionin että Enterprise Editionin versionumero oli 6.2. Opinnäytetyö keskittyy kuitenkin tarkastelemaan Liferayn versiota 6.1, koska tarkasteltavat tuotantoratkaisut on tehty tähän versioon, eikä päivitystä versioon 6.2 ollut vielä suunniteltu.

2.1 Portlet

Portletit ovat Javalla kirjoitettuja käyttöliittymän osia, joissa portaalin toiminnot sijaitsevat (Liferay Inc. 2015j). Liferayn asennuksen jälkeen se sisältää valmiiksi yli 60 portlettia, ja nämä voidaan jakaa kolmeen pääkategoriaan: sisällönhallinta ja web-julkaisut-, yhteistyö- sekä verkostoitumisportletteihin. Yhteistyökategoriassa sijaitsevat keskustelufoorumit, blogit ja wikit. Siellä sijaitsevat myös portletit, jotka mahdollistavat käyttäjän oman työn ja ajatusten

ilmaisemisen muille. Verkostoitumiseen Liferay tarjoaa viestien lähettämismahdollisuuden sekä aktiviteettien seurannan. Se tarjoaa myös keinon kehittää omia sovelluksia sekä liittää kolmannen osapuolen sosiaalisia ominaisuuksia sivustolle. (Liferay Inc. 2015i.) Monissa tapauksissa sivusto näyttää käyttäjälle useita portletteja kerrallaan. Portaali muodostaa sivun kutsumalla siihen sijoitettuja portletteja ja lisäämällä HTML-koodia niiden ympärille. Kun käyttäjä hyödyntää jotakin portletin toimintoa sivustolla, kaikki sivuston portletit renderoidaan uudestaan. Portleteilla on kaksi vaihetta: toimintovaihe sekä render-vaihde. (Liferay Inc. 2015k.)

Liferay on helpottanut sisällöntuotantoa portaalissa tarjoamalla web-sisällölle rakenteita ja pohjia, joita käyttämällä voidaan luoda rakenteisia portletteja. Ne määrittelevät, minkä muodon julkaistava web-sisältö voi saada. Sisällölle voidaan määrittää myös ajankohta, jolloin se on käyttäjille näkyvissä ja milloin se poistuu näkyvistä käyttämällä hyväksi aikataulutusominaisuutta. Sisällölle on mahdollista määrittää Liferay-portaalin Workflow-järjestelmä, jolloin voidaan tarkasti asettaa sisällön tuottamisprosessit. Sekä rakenteille että pohjille voidaan määrittää muokkaamisoikeuksia käyttäjäroolien mukaan, jolloin voidaan varmistaa, että ainoastaan oikeutetut henkilöt voivat muokata niitä.(Liferay Inc. 2015l.)

Rakenteet (structures) ovat sisällön ”luut”. Siinä päätetään minkälaisia kenttiä, tekstejä ja erilaisia tietoja web-sisällöllä voi olla. Sisällöntuottajalle, joka aikoo tehdä rakenteisen portletin, näytetään mahdollisina ainoastaan ne tietotyypit, jotka rakenteeseen on merkitty. Kentät voidaan asettaa toistuviksi, jolloin niitä saadaan monistettua helposti, sekä antaa kentille oletusarvoja. Rakenteilla on mahdollista rajoittaa web-sisältö muotoon, joka on yhtenäinen sivustolla. Niiden avulla saadaan tehtyä sivuston sisällöntuottajille rakenteet, joiden avulla he tietävät minkälaisia kenttiä heidän tulee täyttää, jotta portlet toimii oikein. Rakenteita suunnitellessa on tärkeää asettaa kentille nimet, sillä pohjissa voidaan kohdistaa toimintoja kenttien nimien mukaan. Jos kentille ei anneta erikseen nimiä, Liferay generoi automaattisesti niihin arvon, jota on hyvin vaikea ennustaa. Rakenteet voidaan tehdä suoraan XML-skeeman koodilla tai käyttämällä hyväksi Liferayn editoria. (Liferay Inc. 2015l.)

Pohjat (template) avaavat portletin kehittäjälle tavan vaikuttaa rakenteissa annettuihin materiaaleihin. Sillä voidaan muun muassa antaa rakenteissa oleville kentille luokkanimi tai id-tunnus. Näin voidaan esimerkiksi helpottaa sivuston tyylittämistä, koska tyylitiedostoon voidaan tehdä kaikille samaa rakennetta ja pohjaa käyttäville web-sisällöille tyylitykset yhdellä kertaa. Jos sivustolla käytetään rakenteisia portletteja, täytyy rakenteella olla myös pohja, koska ilman pohjaa Liferay ei voi tietää kuinka rakenteisen portletin tietoja tulisi esittää. Pohjan muokkaukseen voidaan käyttää neljää eri kieltä: Velocity Macro (VM), Extensible Style Sheet (XSL), FreeMarker Template Language (FTL) sekä Cascading Style Sheet (CSS). (Liferay Inc. 2015l.)

2.2 Teema

Liferayssa teemalla tarkoitetaan erilaisia mekanismeja, joiden avulla voidaan muuttaa portaalin yleisilmettä ja toimintoja. Mekanismit, joiden avulla tämä saavutetaan, ovat CSS, Velocity, JavaScript sekä XML. (Liferay Inc. 2015m.) Kaikki Liferay 6.1 -teemat rakennetaan kerroksittain eli teemojen muutokset tulevat voimaan aina tietyissä osissa. Myöhemmin tuleva muutos päällekirjoittaa edeltäjänsä.

Liferayssa käytetään yleisesti kahta perusteemaa, kun luodaan omia teemoja. Aluksi teema nimeltä _unstyled lisätään ytimeksi. Se sisältää portaalin ”luuston”. Seuraavaksi lisättävä _styled-teema antaa käyttöön peruselementit. Jos halutaan käyttää vielä jotakin teemaa pohjana omalle teemalle, se lisätään näiden perusteemojen päälle. Käyttäjän omat muutokset ja tyylitykset lisätään näin kerroksissa päällimmäiseksi ylikirjoittaen aikaisemmat arvot. (Liferay Inc. 2015n.)

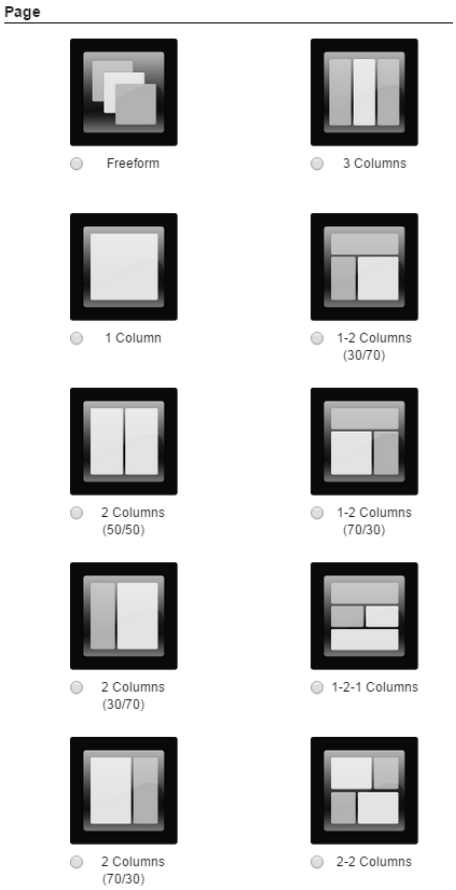
Omat tyylitykset sekä muutokset sijoitetaan _diffs-kansioon, joka sijaitsee docroot-kansiossa. Liferayn sipulimaisen rakenteen ansiosta käyttäjän ei tarvitse muuttaa aikaisempien teemojen tyylityksiä, vaan tämän tarvitsee vain luoda ylikirjoittavat tiedostot kansioon. Yksi Liferayn parhaista käytännöistä on, että teeman tyylitysten muutokset tulisi tehdä ainoastaan _diffs-kansioon

sijoitettuun custom.css-tiedostoon. Käytännöllä on myös konkreettista hyötyä: se selkeyttää yleistä organisointia, koska muutokset ovat ainoastaan yhdessä kansiossa (Liferay Inc. 2015o.)

2.3 Layout Templates

Layout template määrittelee, missä järjestyksessä portletit näytetään sivustolla. Layout templaten kirjoituskieli on Velocity (Liferay Inc. 2015p). Sisällöntuottaja voi listätä portletteja sivustolle yksinkertaisella drag-and-drop-mahdollisuudella (Liferay Inc. 2015q). Liferayssa on myös oletuksena monia erilaisia layout-malleja, kuten kuvassa 1.

Projektissa on käytössä demosivu, jossa käytetään samoja layout templateja kuin tuotannossa. Ulkoasu voi vaihdella, sillä palstamäärät layouteissa ovat yhdestä kolmeen. Myös etusivulle on tehty oma layoutinsa. Projektin mahdolliseen syvällisempään tarkasteluun käytettävässä ympäristössä ei täysin samoja layout templateja ole käytettävissä. Mahdollista on myös rakentaa uusi layout, mutta parasta olisi jos ratkaisut saataisiin toimimaan suoraan demosivustossa olevien layoutien kanssa, jolloin niitä ei tarvitsisi vaihtaa.



Kuva 1. Layoutteja esimerkkiprojektista.

2.4 Liferay ja CSS

Liferay käyttää monia tyylitiedostoja teemassaan. Muokatussa (Custom) teemassa käytetään erityistä CSS-tiedostoa, joka ladataan viimeisenä ja näin ollen siinä olevat tyylit jäävät voimaan. Myös muita tyylitiedostoja voidaan muokata, mutta ongelmaksi saattaa tällöin muodostua muutosten mahdollinen hajanaisuus. (Liferay Inc. 2015o.) Jos tietty tyylityskomento halutaan ehdottomasti säilyttää alemmalta tyylitystasolta, voidaan käyttää CSS-komentoa !important halutun arvon perässä. !important-komennon käyttöä ei yleisesti suositella, sillä se voi sekoittaa myöhempiä tyylityksiä. Lisäksi sen käyttö on useissa tapauksissa vältettävissä kunnollisella tyylityssuunnittelulla. (Mozilla Foundation 2014a.)

Sivuston tyylityksiä muokatakseen ei välttämättä tarvitse muokata sivuston tyylitiedostoja. Liferayn hallintapaneelissa sijaitsee Hallinnoi sivua -osio, jossa

voi sivustokohtaisesti vielä vaikuttaa ulkoasuun. Ulkoasun voi määrittää jäljittelemään jotakin toista sivua tai sitten sille voidaan luoda aivan omat tyylitykset. Myös portleteille voidaan määrittää erikseen omat ulkoasumuutokset portlet kerrallaan portletin omista asetuksista.

2.5 JavaScript

JavaScript on suhteellisen kevyt olio-ohjelmointiin pohjautuva skriptauskieli (Mozilla Foundation 2015a). Kieli kehitettiin vuonna 1995 ja julkaistiin Netscape 2 -selaimelle vuonna 1996. JavaScriptillä ei ole paljoakaan tekemistä Java-kielen kanssa, vaikka nimi voi näin antaa olettaa. (Mozilla Foundation 2015b.) Sitä käytetään yleisesti web-sivuilla ja joskus myös muissa ympäristöissä. JavaScript on pyrkinyt pitämään syntaksinsa lähellä Javan sekä C++:n syntaksia. Yksi suurimmista eroista näihin kuitenkin on, ettei JavaScript sisällä luokkia. (Mozilla Foundation 2015a.)

Liferay 6.1 sisältää oman JavaScript-kirjastonsa nimeltään AlloyUI, joka on laajennus Yahoon YUI3-kehykseen. Molempien tarjoamia ominaisuuksia voidaan hyödyntää käyttäjän luomissa teemoissa.(Liferay Inc. 2015r.) Yahoo on kuitenkin lopettanut YUI:n kehittämisen vuonna 2014 (Lecomte 2014). Liferay siirtyy myöhemmissä versioissaan käyttämään jQuery-pohjaisia ratkaisuja. Liferay vakuuttaa tulevien versioiden olevan yhteensopivia myös YUI3-pohjaisen AlloyUI:n versioiden kanssa. (Cavanaugh 2014.) Nykyisiä AlloyUI-ratkaisuja voidaan siis vielä käyttää, mutta opinnäytetyössä pyrittiin niitä välttämään, jotta varmistettaisiin mahdollisimman suuri päivitettävyys asiakkaiden sivustoissa.

JavaScript-koodia voidaan lisätä portaaliin hallintapaneelin kautta. Tätä keinoa hyödynnettiin opinnäytetyössä, kun toiminnallisuutta haluttiin lisätä elementteihin. Ratkaisu tarjoaa helpotusta testaukseen, jos sivuston rakenteeseen, kuten header-osioon, halutaan lisätä jotain. On kuitenkin muistettava, että JavaScriptin käyttö voi johtaa ongelmiin. Käyttäjällä voi olla skriptinesto-ominaisuus selaimellaan, ja tällöin JavaScript ei toimi.

2.6 Liferay Mobile Rules

Liferay 6.1 Mobile Rules tarjoaa keinon kohdistaa erilaisia toimintoja eri laitteille, kuten iPhonelle sekä Android-puhelimille. Säännöillä voidaan esimerkiksi muokata sivun layout-muutos tai luoda uudelleenohjaus, jos mobiilisäännöissä on laitteelle kohdistettu sääntö. Yksittäisellä sivulla portaalissa voi olla omat sääntönsä, jotka poikkeavat yleisistä säännöistä. Liferay Mobile Rules tarvitsee Device Recognition Provider -pluginin toimiakseen. Liferay Marketplacessa on saatavilla APGLv3 lisenssin alainen WURF-tietokantaan pohjaava ratkaisu. Plugin antaa Liferay-portaalille tavan tunnistaa tietokannassa olevat laitteet. (Liferay Inc. 2015s.) Sen avulla voidaan identifioida myös laitteen käyttöjärjestelmä (Liferay Inc. 2015t).

Liferay Device Detectionin avulla sivustosta saadaan skaalautuva melko helposti, mutta plugin on maksullinen ja saatavilla Liferay Marketplacessa ainoastaan Enterprise Editioniin. Tästä johtuen applikaation toimivuutta ei voitu testata Community Edition versiolla. Maksullisuus on myös erittäin merkittävä seikka kustannuksia harkittaessa.

Mobiilisääntöjä voi tehdä Liferayn hallintapaneelissa ilman tietokantaa, mutta tällöin ne eivät toimi käytännössä (Liferay Inc. 2015s). Tietokannan rakentaminen olisi periaatteessa mahdollista omatoimisesti, mutta sen tekeminen olisi resursseja ja aikaa vievä prosessi. Tätä tietokantaa pitäisi myös aika-ajoin päivittää itse, jotta se ymmärtäisi uusimmat laitteet.

3 Responsiivinen web-suunnittelu

Maailman ensimmäinen web-sivu julkaistiin elokuussa 1991. Se oli yksinkertainen ja yksisarakkeinen tekstidokumentti, johon oli upotettu muutamia linkkejä. Sitä seuranneet sivustot noudattivat samaa web-suunnittelun periaatteita. Yksinkertaisuus johtui ajan HTML:stä, joka salli ainoastaan hyvin yksinkertaisen sisällöntuotannon. HTML:n myöhemmät versiot lisäsivät kielen

monimuotoisuutta muun muassa kuvilla sekä tuella taulukoihin. (Chapman 2009.)

Taulukkopohjaiset ratkaisut nousivat suosituksi valinnaksi web-kehittäjien keskuudessa, kun HTML oli lisännyt <table> komennon syntaksiinsa. Kehittäjät löysivät tavan käyttää taulukoita koko sivun kattaviksi monisarakkeisiksi layout-ratkaisuiksi, vaikka niiden oli alun perin tarkoitus olla vain keino näyttää dataa taulukossa. (Chapman 2009.) Taulukot saattoivat paisua useita sarakkeita ja rivejä sisältäviksi supertaulukoiksi, joihin sivuston elementit sijoitettiin. Oli myös melko tavallista, että taulukon sisään sijoitettiin toinen taulukko ja näin kasvatettiin sivuston monimutkaisuutta entisestään.(Ruluks 2014.)

Flash kehitettiin vuonna 1996. Se kehittyi vuosien saatossa keinoksi rakentaa sivut ennennäkemättömillä tavoilla (Chapman 2009). Web-sivustoissa alkoi tapahtumaan merkittäviä muutoksia. Flashin avulla voitiin tehdä sivustolle animointia ja interaktiivisuutta. (Carlson 2011.)

Vaikka Cascading Style Sheets, eli CSS, oli ollut olemassa jo aikaisemmin, sai se suosiota kehittäjien keskuudessa vasta 2000-luvun alussa. Yhtenä syynä myöhäiseen käyttöönottoon oli huono selaintuki. (Chapman 2009.) Ennen CSS:iä kehittäjät joutuivat muokkaamaan jokaisen sivun koodia yksi kerrallaan tapauksesta riippumatta halutessaan päivittää sivustonsa ulkoasua. CSS:n avulla tyylityksiä voitiin nyt kohdistaa yhdeltä tiedostolta moneen eri web-sivuun. (Carlson 2011.) CSS:ä käyttävät sivustot olivat useasti myös pienempiä kuin taulukkoperiaatetta käyttävät (Chapman 2009).

JavaScriptin suosio alkoi nousemaan 2000-luvun puolivälissä, noin kymmenen vuotta kielen kehittämisen jälkeen. Näihin aikoihin alkoivat myös taulukkopohjaiset internetsivut vähenemään. JavaScript onnistui muun muassa ensimmäisenä animoimaan valikoita ilman Flashia. (Carlson 2011.)

Responsiivisuuden määritteen isänä pidetään Ethan Marcottea. Hän otti kantaa artikkelissaan ”A List Apart” web-kehityksen laitteistopohjaisiin haasteisiin: erikokoisten internetiä käyttävien laitteiden yleistyminen loi paineita sivuston

kehittäjille, kun haluttiin tarjota paras käyttökokemus. Responsiivisen käyttöliittymän periaate yksinkertaisimmillaan on se, että yksi sivu toimii kaikilla alustoilla laitteesta riippumatta. (De Graeve 2011.) Responsiivisuuden saavuttamiseksi noudatetaan yleensä kolmea pääperiaatetta: fluid grids (joustava ruudukko), fluid images (dynaamiset kuvat) ja media queries (media haut) (Pettit 2014).

3.1 Joustava ruudukko

Internetsivusto rakennetaan useasti suunnitellun asettelun mukaan, jossa määritetään eri osasten ja rakennepalasten paikka sivustolla. Joustava ruudukko tarkoittaa sitä, että CSS-tyylitiedoston avulla annetaan niin sanottuja tyylityksiä, jotka muokkaavat sivuston rakennepalasten asemointia niin, että ne voivat liikkua tarpeen vaatiessa (De Graeve 2011).

Staattisessa ratkaisussa sivuston ulkoasu ei muutu ja näyttää samalta näytön koosta riippumatta. Joustava ratkaisu antaa kehittäjälle mahdollisuuden määrittää sivustoa koskevaa tyyliä. Sivuston elementtien sijainti on siis muuttuva, koska niitä ympäröivä rakenne voi muuttua.

3.2 Dynaamiset kuvat

Sivustojen kuvat voivat olla suuria kooltaan. Joustavan ruudukon muuttuessa sivustolla on tarve myös mahduttaa nämä sivuston suuret elementit uuden koon mukaan. Dynaamisissa kuvissa niihin kohdistetaan joko CSS-tyylitys, joka pienentää kuvaa kun ympäröivä elementti pienenee tai sitten piilottamalla näytöstä ylimenevä osa (De Graeve 2011). Kuvat ovat myös haaste suunniteltaessa responsiivisia web-sivuja. Responsiivisten sivujen on tarkoitus käyttää samaa sivustoa eri laitteilla, mikä tarkoittaa yleensä samojen kuvien käyttämistä alustasta riippumatta. Niiden lataaminen vie loppukäyttäjän kaistaa, ja pahimmassa tapauksessa ei välttämättä edes näy kunnolla tai ne piilotetaan pois jolloin niitä ei näe ollenkaan. (De Graeve 2011.)

Kuvan pienentyessä ympäristönsä mukaan voi kuvasta pahimmassa tapauksessa tulla katselukelvoton, koska siitä ei saa mitään selvää. Sivuston kehittäjien täytyy miettiä tarkkaan sivuston elementtejä ja sitä ovatko kaikki niistä tarpeellisia.

3.3 Mediahaut ja Viewport meta tag

Mediahaut nostettiin vuonna 2012 World Wide Web Consortium -suositusten, W3C-suositusten, joukkoon. Mediahaut ovat ehtoja CSS-tiedostoissa. Niiden avulla voidaan määrittää, että tietyn ehdon täyttyessä, käytetään ehdon sisään asetettuja tyylityksiä CSS-tiedostosta. Mediahaut ovat suuressa asemassa responsiivisuudessa, sillä niiden avulla voidaan kohdistaa CSS-tiedostojen komentoja jopa laitteen fyysisten ominaisuuksien, kuten näytön koon, mukaan. (World Wide Web Consortium 2012.)

Viewport meta tag ei sinänsä kuulu responsiivisen web-suunnittelun piiriin, mutta sitä käytetään melko yleisesti parantamaan mobiilia käyttökokemusta. Meta tag sijoitetaan HTML-tiedoston header-osioon ja se voi kertoa tietoa sivustosta. Viewport meta tag on tärkeä responsiivisuuden osa useissa web-sivuissa. Matkapuhelimen selaimissa on oletuksena toiminto, joka skaalaa sivustoa niin, että se näkyisi kokonaan puhelimen näytöllä. Useimmissa tapauksissa selain joutuu skaalaamaan sivuston niin pieneksi, ettei käyttäjä voi saada selvää sivustosta ilman omaa zoomausta. Viewportin määrittämisen jälkeen selain ymmärtää, ettei sen tarvitse näyttää kaikkea kerralla heti aluksi. (Frost 2015.) W3C pyrkimyksenä on standardisoida @viewport-mediaehto. @viewportia käytettäisiin CSS-tiedostosta käsin (World Wide Web Consortium 2015a). Käytetyimmistä selaimista Internet Explorer ja Google Chrome tukevat kyseistä komentoa osittain (Mozilla foundation 2014b).

3.4 Cascading Style Sheet - CSS

CSS lisää tyylejä sivuston HTML-dokumenttiin. (World Wide Web Consortium 2015b). Se on useimmiten ulkoinen tiedosto, johon listataan sivuston tyylisäännöt. Näitä sääntöjä voidaan kohdistaa sivuston elementteihin yleisesti tai suoraan sivuston elementtien luokkiin ja tunnuksiin. (World Wide Web Consortium 2011.) Ensimmäinen versio CSS:stä julkaistiin vuonna 1996 nimellä CSS1 (World Wide Web Consortium 2008). Yleisimpinä responsiiviseen verkkosuunnitteluun liittyviä tyylityksinä voidaan pitää width ja max-width. Kuvassa 2 esitetään tyylitysten esimerkkiä.

```
#wrapper {
    width: 95%;
}
fieldset, img {
    max-width: 100%;

}
```

Kuva 2. Esimerkki widthin ja max widthin käytöstä.

Width eli leveys on ominaisuus, joka vaikuttaa elementin leveyteen. Leveys voidaan määrittää käyttämään arvoja auto, pikseli, prosentti, initial tai inherit (World Wide Web Consortium 2011). Responsiivisuuteen tähtäävissä tyylityksissä useimmiten käytetään prosenttiarvoja. Prosenttiarvoa käytettäessä elementti, johon ominaisuus on liitetty, käyttää tietyn prosenttiosuuden ympäröivään osioon verrattuna (World Wide Web Consortium 2011).

Max-width eli maksimi-leveysominaisuus määrittää elementin suurimman mahdollisen leveyden (World Wide Web Consortium 2011). Responsiivisessa ympäristössä max-width-ominaisuutta monesti kohdistetaan kuviin, jolloin kuva voi skaalautua kooltaan pienemmäksi.

Mediahaut tarkastavat täyttyvätkö niihin kirjoitetut ehdot. Jos ehto on tosi, CSS-tiedostossa tulevat voimaan mediahaun sisään kirjoitetut tyylitykset. Ne ovat siis loogisia ilmauksia, jotka voivat olla tosia tai epätosia. Mediaehto alkaa merkeillä

@media. (World Wide Web Consortium 2012.) Yleisiä responsiivisuuteen liittyviä mediahakuja ovat width, height, device-width, device-height sekä orientation. Kuvassa 3 esitellään mediahakua, jossa käytetään max-device-widthiä sekä max-widthiä.

```
@media(max-device-width:1055px),(max-width:1055px){

}
```

Kuva 3. Esimerkki mediahausta.

Mediahaut width ja height tarkastavat näyttöalueen korkeuden ja leveyden. Ehtoja käyttäessä voidaan hyödyntää etuliitteitä min ja max, jotka antavat ehdolle käyttöön pienempi kuin- sekä suurempi kuin -merkit. (World Wide Web Consortium 2012.)

Mediahaut device-width sekä device-height tarkastavat laitteen näytön korkeuden ja leveyden. Toiminto on lähes identtinen mediakyselyjen width ja height kanssa, mutta eroaa sillä tavalla, että ehto tarkastelee laitteen kokoa. Ehtojen kanssa voidaan käyttää myös etuliitteitä min ja max. (World Wide Web Consortium 2012.)

Älypuhelimilla on kaksi eri näytön asentoa, riippuen millä lailla käyttäjä pitää puhelintaan. Jos puhelimen leveys on suurempi kuin pituus, puhelimen orientaatio on landscape. Puhelimen pituuden ollessa leveyttä isompi, useimmiten puhelimen pystyasento, orientaatio on picture. Orientation-mediakysely selvittää, onko median ominaisuus portrait vaiko landscape. Kysely tarkastelee height- sekä width-mediahakujen arvon joita vertaamalle se päättelee, käytetaankö portrait vai landscape arvoa. Orientation-mediakyselyllä voidaan siis kohdistaa tyylityksiä riippuen näytön asennosta. (World Wide Web Consortium 2012.)

Mediahakujen selaintuki on erittäin laaja ja niiden pitäisi toimia useimmilla päivitetyillä matkapuhelinselaimilla. Ongelmia tuen kanssa voivat aiheuttaa selainten vanhentuneet työpöytäversiot. (Marcotte 2010.)

3.5 Mobiili ensin -periaate

Mobiili ensin -periaatteella tarkoitetaan, että sivusto rakennetaan ensin mobiilille alustalle ja vasta sen jälkeen siitä tehdään muut suuremman näyttökoon versiot (Pettit 2014). Sen voidaan katsoa hyödyntävän Progressive Enhancementin-periaatteita. Aluksi kartoitetaan halutut näyttöjen koot sekä ominaisuudet, jotka halutaan liittää näihin kokoihin. Sivustoja rakennetaan pienimmästä tuetusta näyttökoosta isoimpaan ja lisätään toiminnallisuutta ja toimintoja näyttökoon kasvaessa. (Johnson 2013.)

Progressive Enhancement on toinen vallitsevista periaatteista responsiivisia web-sivuja rakennettaessa. Graceful Degradationissa suunta on päinvastainen. Sivusto rakennetaan ensin tietokoneella käytettäväksi työpöytäversioksi, josta lähdetään alustan pienentyessä skaalaamaan sivua pienemmäksi ja ottamaan toimintoja pois. Web-kehittäjät joutuvat käyttämään Graceful Degradationia useasti tapauksissa, joissa jo olemassa olevaa sivustoa lähdetään muuttamaan mobiiliksi, eikä uutta sivua voida tehdä täysin alusta asti. (Johnson 2013.)

Mobiili ensin -toimintoja lisäävällä lähestymistavalla on monia hyötyjä verrattuna Graceful Degragationiin. Mobiilisivustoissa on hyvä käytäntö pitää ne kooltaan pienenä. Jos sivustot on valmiiksi suunniteltu mobiilille alustalle, sivustoon voidaan lisätä toimintoja. Kehittäjän on usein helpompi lisätä materiaalia sivustoon kuin leikata osia pois. (Johnson 2013.) Toinen tärkeä seikka mobiili ensin -periaatteen puolesta on se, että työpöytäversiosta pienennetyt sivustot voivat olla raskaita kooltaan. Kehittäjien usein käyttämä tekniikka on se, että sivustolle annetaan tyylitiedosto, joka piilottaa materiaalia käyttäjältä tämän selatessa niitä pienempinäyttöiseltä ruudulta. Selain kuitenkin lataa piilotetun tiedoston, vaikka ei näytä sitä. (De Graeve 2011.) Mobiili ensin-periaate on kuitenkin siitä huono, että se pahimmissa tapauksissa rajoittaa kehittäjää. Suunnittelussa on otettava huomioon pieni näyttö sekä pienempi teho. (Johnson 2013.)

4 Toimeksiantoprojekti

Projekti piti sisällään tiedonhankintaa, jossa selvitettiin responsiivisuuteen liittyviä periaatteita ja parhaita käytäntöjä. Näitä teoreettisia taitoja käytettiin testattaessa erilaisia lähestymistapoja projektin ongelmaan. PTTK Oy tarjosi tunnukset demosivustolle, jota voitiin käyttää testattaessa löydettyjä ratkaisuja, joten toimeksiannon työosiota voitiin tehdä pääsääntöisesti kotoa käsin. Kotikoneelleni asennettiin Liferay Community Edition 6.1, jota käytettiin syvällisempään Liferay-portaalitarkasteluun. Portleteille ja teemalle kehitysympäristönä käytettiin Eclipseä ja Liferay IDE:tä.

4.1 Projektin tavoite

Opinnäytetyön tehtävänä oli selvittää, kuinka responsiivisuuden periaatteita voitaisiin toteuttaa mahdollisimman mielekkäästi ja kustannustehokkaasti PTTK Oy:n asiakkaiden Liferay-sivustoilla. Opinnäytetyö myös pohtii, voidaanko Liferayn layoutiin tai teemaan liittyviä tiedostoja muuttaa niin, että responsiivisuuden periaatetta voidaan hyödyntää ilman, että kaikkien asiakkaiden eri sivujen CSS-tiedostoja jouduttaisiin muuttamaan yksitellen. Lisäksi otettiin kantaa kokonaan uuden responsiivisen teeman rakentamiseen.

Responsiivisen web-suunnittelun kannalta olisi otollisinta, jos sivut voitaisiin luoda mobiili ensin -periaatteella. Näin ei kuitenkaan voida enää tehdä, koska PTTK Oy:n asiakkaiden sivut ovat jo tuotannossa, ja kaikkien selailtavissa. Ne käyttävät pohjanaan samaa Liferay-teemaa ja rakenteita, mutta asiakkaat ovat voineet antaa ulkonäkötoivomuksia omille sivustoilleen, jotka PTTK Oy on toteuttanut mahdollisuuksien mukaan.

Suurimpia selvitettäviä ongelmia opinnäytetyössä olivat Liferayn portlettien käyttäytyminen responsiivisessa ympäristössä sekä asiakkaiden sivustojen omat tyylitykset. Suurin osa ongelmista voitaisiin ratkaista tekemällä PTTK Oy:n asiakkaille uusi oma yhteinen teema, joka noudattaisi responsiivisuutta alusta asti. Asiakkaiden omat yksilölliset muutokset voitaisiin tämän jälkeen tehdä

myös responsiivisuuden periaatteita noudattaen. Vaikka opinnäyetyössä tarkastellaan teeman tekemistä yhtenä mahdollisuutena, se keskittyi ensisijaisesti jo olemassa olevan teeman muokkaamiseen sekä parantamiseen.

Projektissa haasteena voitiin pitää myös Arcusys Oy:n osuutta PTTK Oy:n asiakkaiden sivustoissa. Sekä sivustot, että demosivu, sijaitsevat Arcusys Oy:n palvelimella, mikä hankaloittaa portaalin tietojen muokkaamista. Pääasiallisena ratkaisuna käytetään testisivuston hallintapaneelia, jossa voidaan muuttaa sivustoa koskevia CSS- ja JavaScript-koodia. Hallintapaneelista pääsee myös tekemään rakenteisia portletteja, mutta syvällisemmät muutostoiveet täytyi ilmoittaa sähköpostitse Arcusys Oy:lle. Oli myös mahdollista, että Arcusys Oy:n päivitykset sivustoilla voisivat myös muuttaa kehitystä tavoilla, joita ei voi ennaltaehkäistä tai ennakoida. Päivityksistä ei kuitenkaan ollut haittaa toimeksiannon työosuuden aikana.

4.2 Tutkimusmenetelmä ja ympäristö

Toiminnallisen opinnäytetyössä yhdistetään kaksi osiota: toiminnallinen osio sekä raportointi. Toiminnallinen osio voi vaihdella jonkin konkreettisen toimen suunnittelemisesta tapahtuman toteuttamiseen. Käytännön toteutuksen tuotos vaihtelee sen mukaan, keille työ kohdistetaan. Se voi olla muun muassa verkkosivu, kirja tai ohjelehtinen. (Airaksinen & Vilkka 2003, 9.) Toiminnallisen opinnäytetyön raportointiosiossa selvitetään, mitä opinnäytetyössä on tehty, miksi ja kuinka valittuihin ratkaisuihin on päädytty. Se sisältää myös opinnäytetyön tekijän oman arvioinnin prosessista sekä raportoinnista. (Airaksinen & Vilkka 2003, 65.)

Raportoinnin sekä toiminnallisen osuuden tavoitteena tavoitteena on, että lukija ymmärtäisi opinnäytetyön tekijän ammatillisen osaamisen sekä työn aikana koetun henkilökohtaisen kehityksen (Airaksinen & Vilkka 2003, 65). Toiminnallisella opinnäytetyöllä on hyvä olla toimeksiantaja, sillä usein toimeksiantajan projekteissa voidaan haastaa omaa osaamista oikeissa työelämän tehtävissä. Airaksinen ja Vilkka ottavat esille myös toiminnallisen

opinnäyttyön edun, jonka mukaan toimeksiantaja voi kiinnostua opinnäytetyöntekijästä työntekijänä avaten mahdollisia valmistumisen jälkeisiä työpaikkoja. (Airaksinen & Vilkka 2003, 16 – 17.)

Projektityössä käytettiin PTTK Oy:n tarjoamaa testiympäristöä, joka hyödyntää Liferay Community Edition 6.1:tä. Ympäristössä PTTK Oy:n asiakkaat ovat voineet harjoitella ja testata Liferay-portaalia ennen ja jälkeen omien sivujensa käyttöönoton. Testiympäristö tarjosi hyvän kehittämisalustan, sillä se käyttää samaa layoutia asiakkaiden sivujen kanssa, sekä useimmat sen portleteista ovat samoja kuin asiakkaiden sivuilla. Testisivustolla voidaan muokata helposti sivuja koskevia tyylityksiä ja näin testata CSS:llä luotavaa reponsiivisuutta hallintapaneelin kautta.

Syvällisempään teeman tarkasteluun ja testaamiseen käytettiin Liferay Community Edition 6.1:tä, joka asennettiin tietokoneelleni. Erillinen testiympäristö tarvittiin, koska asiakkaiden sivut ovat Arcusys Oy palvelimilla, eikä teeman ja Liferayn osien muutoksia voitu tehdä omatoimisesti. Muutoksista olisi pitänyt ottaa yhteyttä Arcusys Oy:hyn. Prosessissa olisi kulunut liian kauan aikaa. Varsinkin pienissä hienosäädöissä välittömät muutokset olivat toivottavia. Kehitysympäristönä käytettiin Eclipseä ja Liferay IDE:tä.

4.3 Projektin toteutus

4.3.1 Aloitus

Opinnäytetyön aloitus ajoittui joulukuun alkuun sopimusten kirjoittamiseen. Toimeksiantajan edustajan kanssa sovittiin, että aluksi keräisin tietoperustaa työ-osiota varten. Teoriaa kerätessä huomattiin, ettei painettua kirjallisuutta Liferay-portaalista ollut saatavilla ilmaiseksi. Tästä johtuen päätettiin turvautua Liferayn omiin sivustoihin sekä Liferay-kehittäjille tarkoitettuun dokumentaatioon. Sivuston tarjoamat ohjeet ovat laajat, mutta melko vaikeita käyttää. Haetun asian löytäminen voi olla hankalaa, sillä useasti ongelma on monitasoinen.

Toinen selvitettävä asia oli responsiivisuuden teoreettinen pohja. Responsiivisuudesta löytyy myös monia sivustoja internetistä, sillä periaate näyttää olevan nousussa vuodesta 2011 lähtien.

Toimeksiantajan edustajan kanssa ensimmäisissä kokoontumisissa tarkasteltiin myös erilaisia responsiivisia verkkosivuja ja keskusteltiin niiden ominaisuuksista. Tärkeässä asemassa oli myös keskustelu opinnäytetyön lopputuloksesta ja siitä, mitä tulevalta mahdolliselta ratkaisulta toivottaisiin. Vaikka vaatimuksia ei ollut paljon, vertailukohdat olivat selkeät. Sivustojen tulisi skaalautua selkeästi mahdollisimman monella laitteella. Muut suunnitelmat päätettiin jättää myöhempiin tapaamisiin.

Joulukuun aikana suoritettiin myös ensimmäisiä kokeiluja responsiivisuuden keskeisimmistä aiheista, kuten sivuston sekä kuvan skaalautumista sivun mukana. Joulukuussa PTTK Oy:n edustaja lähetti myös minulle omakehittämänsä teeman, jossa hän oli testannut joitakin yleisiä periaatteita responsiivisuudesta, kuten sivuston skaalautumista selaimessa. Teeman ratkaisuja otettiin huomioon omassa ratkaisussa, ja niitä pyrittiin parantelemaan mahdollisuuksien mukaan.

4.3.2 Työkalut

Seuraavaksi tarkasteltiin työkaluja. Liferay tarjoaa monia tapoja kehitykselle. Kahtena esimerkkinä voidaan mainita Apache Ant sekä Plugins SDK:n ja Eclipse yhdistettynä Liferay IDE:hen Monia muitakin ratkaisuja voidaan käyttää. Yksi suosittu command-line-vaihtoehto on Maven. Liferay tukee myös erilaisia IDE;tä kuten NetBeans ja IntelliJ IDEAa. Kehitystyövälineen valintaa miettiessä otettiin huomioon seuraavat kriteerit: kehitysympäristön helppokäyttöisyys, suosio ja tuttuus. Suurin painoarvo annettiin ympäristön tuttuudelle.

Eclipse on kenties yksi suosituimmista Java IDE:stä ja siihen löytää ongelmatilanteissa internetistä ohjeita melko helposti. Olin myös tutustunut aikaisemmin Eclipsen käyttöön omissa Liferay-projekteissa. Liferay IDE:tä oli

näissä projekteissä myös käytetty. Liferay-portaali asennettiin Tomcat-yhdistelmän kanssa. Tomcat on avoimen lähdekoodin ohjelmistotuote Java Servlet ja JavaServer pagesista (Apache Software Foundation 2015). Latauksen jälkeen käyttö on yksinkertaista. Ladatussa kansiosta täytyy löytää bin-kansio, jossa sijaitsee startup.bat-tiedosto. Tiedostoa klikkaamalla Liferay käynnistyy pienen viiveen jälkeen ja aukaisee selainnäkymän.

Eclipse ja Liferay IDE asennettiin, koska tarvittiin kehitystyökaluja joilla voitaisiin tehdä portletteja sekä teemoja. Liferay IDE:n asentaminen onnistuu helposti Eclipsen help-valikon kautta. Valitsemalla osiosta Install new software esiin tulee palkki, johon kirjataan Liferayn sivuilta saatava osoite. Kirjauksen jälkeen seurataan yksinkertaisia ohjeita, jonka jälkeen Eclipse asentaa Liferay IDE -lisäosan. Eclipsen ja Liferay IDE -yhdistelmän voi myös ladata Liferayn sivuilta. (Liferay Inc. 2015u.)

4.3.3 Tyylityksiä ja Meta tag viewport

Tammikuussa aloitettiin demo-sivuston tyylitysten muokkaaminen. Testaus kotikoneella oli luontevaa, mutta muutosten vienti demo-sivustolle oli aluksi hankalaa. Kuten alussa asti oli jo selvillä, demosivuston rakennetiedostoihin ei päässyt käsiksi helposti. Näin ollen teeman tyylikansioon, johon muutokset lisättäisiin, ei päästy käsiksi nopeasti. Ratkaisuksi osoittautui hallintapaneeli, jonka kautta voitiin sivustoihin kohdistaa CSS-tyylityksiä. Hallintapaneelissa on myös osio JavaScript-koodille, jota päätettiin käyttää aina kun haluttiin lisätä sivuston portletteihin toiminnallisuutta. Molemmat edellä mainituista osioista antavat admin-käyttäjälle mahdollisuuden kirjoittaa koodia, joka aina ladataan sivustolle viimeiseksi.

Demosivuston hallintapaneelia käytettiin paljon responsiivisuutta testattaessa sivustolla sekä myöhemmässä vaiheessa kun ratkaisuja testattiin tuotannossa. Vaikka omalla kotikoneella kaikki luodut tyylitykset saattoivat toimia moitteettomasti, oli ne syytä testata demoympäristössä, koska kyseinen ympäristö käyttää PTTK Oy:n asiakkaiden sivustojen pohjaa. Modernit

matkapuhelinten selaimet olivat myös haasteellisia, sillä ne pyrkivät useimmiten näyttämään koko sivuston puhelimen näytöllä. Selain zoomaa sivustoa ulospäin niin kauan, että niistä ei välttämättä erota mitään. Käyttäjä joutuu manuaalisesti zoomaamaan nähdäkseen haluamansa elementit.

Ongelmaa yritettiin aluksi ratkaista CSS-tyylityksillä, jotka kohdistettiin ongelmaosioihin. Tyylitykset sijoitettiin mediahakuun niin, että ne tulisivat voimaan ainoastaan tietyissä näyttöleveyksissä. Esiin nousi kuitenkin uusia ongelmia: selain- ja laitekohtaiset erot mobiilissa ympäristössä voisivat aiheuttaa ongelmia, vaikka CSS-ratkaisu saataisiin toimimaan. Pikseleiden koko saattaisi vaihdella niin, että ulkoasua olisi vaikea ennustaa eri puhelin- ja selain-yhdistelmissä. Vaikka tällä ratkaisulla sivut saataisiin näyttämään hyviltä mobiilissa ympäristössä, ne näyttivät erittäin oudoilta tietokoneen selaimessa niitä skaalattaessa pienemmäksi. Tähän pohdittiin ratkaisuna mediaehdon tekemistä niin, että ongelmakohtien muokkaukset eivät vaikuttaisi selainversiossa.

Ratkaisuksi löydettiin viewport meta tag, joka liitetään sivuston header-osioon. Meta tag kertoo selaimelle sivun olevan sellainen, jota ei tarvitse lähteä skaalaamaan (Mozilla Foundation 2015c). Meta tag toimi käytännössä niin kuin teoriassa oletettiin ja tyylityksiä voitiin tehdä tästä eteenpäin olettamuksella, että sivustoa ei tarvitsisi heti sinne mentäessä ruveta zoomaamaan. Seuraavaksi oli edessä meta tagin saaminen demosivuston header-osioon, mielellään myös niin ettei Arcusys Oy:hyn tarvitsisi ottaa yhteyttä, olihan kyseessä vain testi. Ainoaksi löydetyksi ratkaisuksi osoittautui hallintapaneelin javaScript-osio. JavaScriptillä oli mahdollista etsiä sivuston header ja lisätä siihen haluttu meta tagi. Vaikka testiolosuhteissa tämä toimiikin, suositellaan lopussa välttämään ylimääräisiä skriptejä ja tekemään muutoksen sivustojen headereihin muokkaamalla sivuston tiedostoja, jos vain mahdollista.

4.3.4 Navigaatio

Seuraavaksi ongelmaksi muodostui navigaatio. Nyt puolittain responsiivisessa sivustossa navigaatio siirtyili sivustolla paikasta toiseen sivuja skaalatessa ei-toivotulla tavalla. Navigaatio-ongelman ratkaisua pohdittiin ja testattiin pitkään. Pieneen näyttöön ei mahdu paljoakaan alivalikoita ja PTTK Oy:n asiakkaiden navigaatioissa oli kymmeniä alisivuja. Keskusteltaessa toimeksiantajan yhteyshenkilön kanssa päädyttiin lopputulokseen, jossa normaalissa näkymässä nähtävä navigaatio piilotetaan näkyvistä näytön pienentyessä tarpeeksi. Tällöin uuden navigaation nappi tulee näkyviin sivuston yläosaan, joka seuraa käyttäjää tämän selaillessa sivua vasemmassa yläkulmassa. Mediahaku tyylitiedostossa tarkastaa, onko näyttö tarpeeksi pieni, jotta navigaationappi tulee näkyviin.

Mediahaun sisään sijoitetaan myös muita tyylityksiä, joiden halutaan astuvan voimaan näytön ollessa halutun mobiiliympäristön levyinen. Käyttäjän painaessa nappia ensimmäisen tason valikko tulee näkyviin. Sivuston alavalikoita ei tarjota käyttäjälle ennen kuin hän on siirtynyt päänavigaation aiheen painamiseen jälkeen uudelle sivulle. Tällöin työpöytäversiossa vasempaan palkkiin sijoitettu aiheen navigaatio on sijoitettu ensimmäiseksi asiaksi, minkä käyttäjä näkee. Tässä alinavigaatiossa on useasti myös alisivun alisivuja, jotka klikataan aiheen vasemmalla sijaitsevasta nuolesta esiin. Alinavigaation rakenne mahdollisti, että sen alisivuihin päästään käsiksi suurentamalla klikkausnuolen osaa. Toinen käyttäjää helpottava ratkaisu oli, että paljastuvat alisivustot sisennettiin. Näin käyttäjän voi olla helpompi seurata, mitkä sivut ovat alisivuja. Ratkaisuna asiakkaiden suurelle määrälle navigaatio elementtejä päätettiin luoda nappi, jolla se saadaan piiloon. Nappi sijoitettiin alinavigaatio-osion yläosaan, josta käyttäjä löytäisi sen helposti.

4.3.5 Rakenteinen portletti

Yläpalkin testiversioksi laadittiin yksinkertainen rakenteinen portletti, jonka avulla voitiin automaattisesti lisätä halutut luokat ja tunnisteet sivustoon

portlettia käyttämällä. Portletti ottaa vastaan käyttäjän antaman kuvan, jota käytetään navigaation painikkeena. Portlettiin lisätään myös halutut viittaukset sisäisiin sivuihin, minkä pohjalta yläpalkin navigaatio muodostuu. Sen pohjana käytetty demosivuston oma header portlet sisältää myös linkit ja toiminnallisuuden, joka aikaisemmin sivustossa myös oli. Yläpalkki voi kuitenkin vaihdella niin toiminnallisuudeltaan, kuin ulkoasultaan eri asiakkaiden sivustoilla. Tehtäväksi jääkin käyttöönottovaiheessa arvioida, mitä kaikkea sivuston yläpalkki tarvitsee ja mahdollisuuksien mukaan ottaa turhia tietoja pois.

Tarjotussa ratkaisussa yläpalkin elementteihin kohdistetaan CSS-tyylitykset display:none, joka piilottaa elementin loppukäyttäjältä, sekä display: block, joka tuo elementin esiin, riippuen näytön koosta. Portlettiin lisättiin myös toiminnallisuutta JavaScript-koodilla, joka tunnistaa yläpalkissa sijaitsevan kuvan klikkaamisen. Skripti lisää mobiili-navigaatioon halutun luokan sitä painettaessa (kuva 4).

```
/*--Change class name toggle navi test -- */
document.getElementById('nav-pic').onclick = function(){
    document.getElementById('toggle-nav');
    if ( document.getElementById('toggle-nav').className.match(/(?:^|\s)toggleAuki(?!\S)/) ) {
        document.getElementById('toggle-nav').className = document.getElementById('toggle-nav').className.replace( /(?:^|\s)toggleAuki(?!\S)/g , '' );
        document.getElementById('toggle-nav').className += " toggleKiinni";
    } else {
        document.getElementById('toggle-nav').className = document.getElementById('toggle-nav').className.replace( /(?:^|\s)toggleKiinni(?!\S)/g , '' );
        document.getElementById('toggle-nav').className += " toggleAuki";
    }
}
```

Kuva 4. JavaScript-toiminnallisuuden lisääminen yläpalkkiin.

4.3.6 Monisarakkeisuus

Asiakkaiden sivustojen layout on monisarakkeinen ja näin ollen ei tule kelpaamaan ainakaan matkapuhelimella käytettävään sivustoversioon, koska pienellä näytöllä ei yksinkertaisesti ole tilaa. PTTK Oy:n asiakkaiden sivustolla body-elementissä sijaitsevat layout-ontop, jossa sijaitsee yläpalkki, wrapper, jossa esitetään sivuston yleiset elementit ja tekstit sekä footer-osio, jossa sijaitsee sivustoon loppuun mahdollisesti lisättävä osio. Wrapperin sisään

sijoitetaan myös content-osio, jossa on layout-templatesta riippuva määrä sarakkeita. Sarakkeiden sisään sijoitetaan eri elementtejä, jotka näkyvät asiakkaille halutussa järjestyksessä.

Näitä sarakemääriä voitaisiin muutella Liferayn Mobile Rules -osiossa, mutta sitä ei projektissa voida käyttää. Yksinkertaiseksi ja ympäristössä toimivaksi ratkaisuksi löytyi width: 100 % -tyylikomennon kohdistaminen sarakkeisiin. Tyylikomento pakottaa sarakkeet allekkain niin, että jokainen niistä on 100 %:n levyinen verrattuna ympäröivään sivustoelementtiin eli content-div-osioon.

4.3.7 Tuotantotestaus

Ratkaisuja testattiin demosivustolla, ja niiden pääosin toimiessa PTTK Oy:n yhteyshenkilö esitti, että niitä voitaisiin testata myös tuotannossa. Opinnäytetyön toiminnallinen osuus oli tehty niin, että sivuston muutokset voitiin tehdä hallintapaneelin kautta, joten tuotantotestaus oli myös mahdollinen melko helposti. Tyylitykset sekä JavaScript-osio lisättiin sivuston ulkoasuun ja uudet rakenteiset portletit pystyttiin luomaan hallintapaneelista käsin. Tuotantotestauksessa oli tarve luoda yksinkertaisia graafisia elementtejä, sillä demosivuston napit oli tehty sivustosta löydetyillä paikantäytteillä.

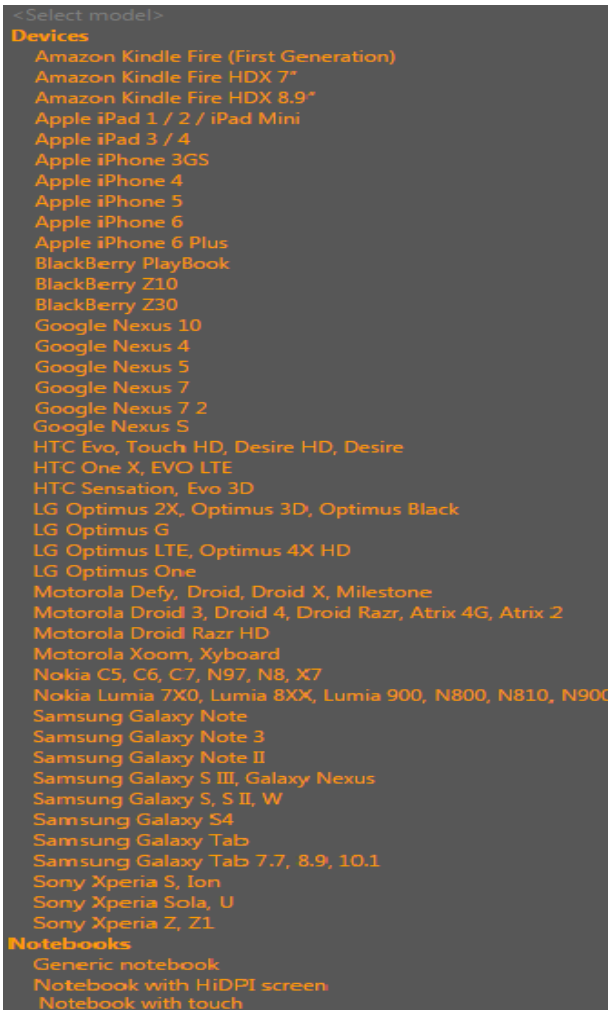
Esivalmistelujen jälkeen ratkaisut lisättiin sivustoon ohjeiden mukaan. Tuotannossa ratkaisut ja pääperiaatteet toimivat odotetusti. Muutamia kauneusvirheitä kuitenkin havaittiin. Pienennettäessä selainikkunaa sivuston käyttäessä tablet CSS -tyylityksiä, oli n. 10 pikselin ajan headerissa ongelma. Skaalautuminen pienensi headerissa tilaa niin paljon, etteivät siellä olleet elementit enää pysyneet paikoillaan, vaan ne työntyivät pois palkista. Ratkaisuksi osoittautui puhelintyylitysten alkamisleveyden suurentaminen.

Demosivusto pakottaa joillekin sivustoille omia tyylityksiään. Palautteen antamissivuun täytyi tehdä tyylitykset erikseen, jotta ne toimivat paremmin mobiilissa ympäristössä. Myös kysymyssivu vaati korjaamista. Näiden sivustojen portletit saivat tyylityksiä, joko custom.css-osiosta tai jostain muusta

tyylitiedostosta. Tästä voidaan päätellä, että myös PTTK Oy:n asiakkaiden sivustot voivat pakottaa tyylityksiä joihinkin portletteihin sekä sivustoihin. Nämä täytyy siis myöhemmin responsiivisuutta otettaessa käyttöön tyylittää parempaan responsiivisuuteen yksitellen. Yksi mahdollisuus olisi tarkastella jokaista sivuston käyttämää CSS-tyylitiedostoa sekä korjata löydetyt tyylitykset suoraan niihin.

4.3.8 Lopputestaus

Yhtenä tärkeimmistä tuotekehityksen vaiheista voidaan pitää testaamista. Tuotantotestauksen jälkeen oli demosivu lähes valmis. Lopputestauksen tarkoituksena oli varmistaa kehitettyjen ratkaisujen toimiminen mahdollisimman laajalla laite- ja selainkokonaisuudella. Pääasialliseksi työn testivälineeksi osoittautui Google Chromen kehittäjän työkalut, joiden avulla voidaan tarkastella sivun lähdekoodia sekä CSS-komentoja. Kehittäjän työkaluissa sijaitsee myös mobiiliin testaukseen hyvin soveltuvat laitetilat, jolloin Chrome emuloi valitun laitteen näytön leveyttä ja sivuston käyttäytymistä kyseisellä laitteella. Kuvassa 5 on esitelty laitteet, joita voidaan emuloida Chromessa.



Kuva 5. Google Chromen kehitystyökalujen laitelista.

Emuloidessa laitteita kaikki toimi odotetusti ja sivut skaalautuivat halutulla tavalla. Emulointityökalut ovat kuitenkin vain suuntaa antavia, eivätkä ne korvaa fyysistä laitetestausta. Fyysisessä laitetestauksessa käytettiin Samsung Galaxy S II -älypuhelinta, Apple iPhone 5:sta neljän tuuman näytöllä, iPad miniä sekä Nokia Lumia 620:aa. Samsung Galaxy S II -puhelimella testattiin Google Chrome- ja Firefox-selaimia sekä puhelimen omalla internetselainta. Apple iPhonea testattiin sen Safari-selaimella sekä Google Chromella. Nokia Lumia -puhelinta käytettiin Internet Explorer-selaimella.

Testattaessa puhelimilla käytettiin niiden omia datayhteyksia, enintään 3 G:n nopeus, sekä kiinteää verkkoa wi-fi-yhteydellä. Selainkohtaiset erot testauksessa olivat huomattavia. Google Chromea käytettäessä kaikki toimi yhtä hyvin, kuin pöytäkoneelle tarkoitetussa versiossa, eikä ongelmia testauksessa löytynyt. Firefox-selaimella sivusto skaalautui halutulla tavalla,

mutta toimi usein testauksessa hitaasti. Nokia Lumia ja Internet Explorer aiheuttivat eniten ongelmia. Niillä oli joskus tapana tulkita tyylityksiä ja viewport-komentoa väärin, jolloin selain ei käyttänyt hyväkseen koko näytön leveyttä. Asian pystyi korjaamaan zoomaamalla sivustoa isommaksi ensin ja sitten palauttamalla sivut normaalin kokoiseksi, jolloin sivusto asettui oikean kokoiseksi puhelimen näytölle.

Käyttäjälle ratkaisu ei ollut mielekäs, joten korjausta alettiin suunnitella. Tutkimuksen jälkeen löytyi Internet Explorerille kohdistuva ms-viewport-komento, jonka sisään laitettiin height- ja width-arvolle auto. Ongelma näytti ratkeavan, mutta komento koskee kaikkia Internet Explorer-selaimia, ja niitä ei fyysisenä testissä ollut kuin yksi. Ei siis voida olla täysin varmoja Internet Explorer-tuesta, ja selainta olisi hyvä testata ennen mahdollista tuotantoon ottoa mahdollisimman paljon.

4.4 Keskeiset painopisteet

4.4.1 Portlet ja responsiivinen ympäristö

Portletit ovat osasia sivustosta ja niiden paikka määräytyy Liferayn käyttämän layoutin mukaan. Responsiivisessa sivustossa layout täytyy saada yksisarakkeiseksi, jotta sisältö näyttää luontevalta matkapuhelimen ruudulta tarkasteltuna. Tämä muutos voidaan tehdä joko Liferayn Mobile Rules -osiolla tai CSS-tyylityksillä. Opinnäytetyössä muutostyyliksi valikoitui CSS. CSS-ratkaisussa sarakkeet pakotetaan allekkain määrittämällä sarakkeiden leveydeksi 100 %. Portletit pitävät paikkansa sarakkeissa, joten ne siirtyvät näytöllä sarakkeiden mukaan, jos niihin ei haluta kohdistaa vielä erillisiä tyylikomentoja.

CSS-ratkaisussa löytyy myös ongelmia. Joissakin portleteissa on kirjoitettu tyylitykset suoraan rakenteen sisään. Näin ollen vaikka portletin asemointi sivustossa vaihtuu halutusti, sen omat tyylitykset eivät aina vastaa

sivustokohtaisia ratkaisuja. Responsiivisuus ei näissä portleteissa tällöin välttämättä toteudu. Erittäin ongelmallisiksi tällaisen portletin tekee se, että useasti näihin versioihin on määritelty leveydet pikseleittäin ja portlet saattaa levitä mobiilin näyttöruudun ulkopuolelle. Ratkaisuksi portletteihin havaittiin joko portletin omien staattisten tietojen muutos tai portletin osiin kohdistettu CSS-tyylitys. Opinnäytetyön aikaisempia periaatteita noudattaen ratkaisuksi valittiin kohdistettu CSS-tyylitys, kun löydettiin portletti joka sisälsi sisäisiä CSS-komentoja. Vaikka löydetyistä portleteista tyylitykset muutettiin, voivat PTTK Oy:n asiakkaiden sivustoissa olla portletteja, joissa vielä näitä tyylityksiä on. Näissä tapauksissa täytyy muistaa muokata portetit niiden löytyessä.

4.4.2 Tulosten ja Liferayn arviointia responsiivisuuden kannalta

Yksinkertaisten ratkaisujen avulla haettu responsiivisuus onnistui hyvin. Asiakkaille käytössä ollut yhteinen rakenne sivustoissa johti siihen, että yleiset tyylitykset voitiin tehdä niin, että perusperiaatteiden voidaan uskoa toimivan lähes kaikilla sivustoilla. Jo olemassa olevat tyylitykset voivat estää ratkaisun tyylitysten toimivuutta, mutta otettaessa ratkaisuja käyttöön ne on huomioitava sivustokohtaisesti.

Liferay-portaalin rakenne mahdollistaa tehokkaan responsiivisten tyylitysten kehittämisen myös jälkikäteen. Tämä lähestymistapa johtaa ulkoasulliseen responsiivisuuteen, mutta johtaa ongelmiin toisen hyvän responsiivisuuden ominaisuuden kanssa. Sivustojen koko on merkittävässä asemassa mobiilialustoissa. Jos käyttäjä ei voi yhdistää internetiin wi-fi-yhteyden kautta, vaan joutuu käyttämään puhelimen data-yhteyttä, voivat suuret sivut latautua hitaasti. Opinnäytetyön oloissa ja käytetyissä ratkaisuissa keskityttiin sivuston ulkoasulliseen responsiivisuuteen. Jo olemassa olevien sivustojen koko-ongelmia voidaan ratkaista vain suunnittelemalla ja rakentamalla sivustot kokonaan uudestaan tai suunnittelemalla sivut erittäin tarkasti ja muokkaamalla niitä.

Huomionarvoinen seikka on myös se, että opinnäytetyön ratkaisuissa käytetään paljon JavaScript-pohjaisia ratkaisuja hyväksi. Moni JavaScriptillä lisätyistä toiminnoista olisi voitu lisätä sivustoon myös portaalin tiedostoihin suoraan ja käytetty JavaScript-koodi onkin useasti vain kiertotie, jonka avulla ei tarvinnut ottaa yhteyttä Arcusys Oy:hyn. JavaScript voi hidastaa sivustojen toimintoja mobiilissa ympäristössä. Jos PTTK Oy käyttää ratkaisua myöhemmässä vaiheessa, suositellaan ainakin viewport meta tagin lisäämistä header-osioon suoraan sivuston tiedostoon.

Vaikka opinnäytetyön loppuvaiheessa keskityttiin ratkaisujen testaamiseen, voidaan sanoa että sitä olisi voinut tehdä enemmänkin. Valitettava seikka testauksessa oli myös se, ettei sivustoa voinut testata kovin monella fyysisillä laitteilla, koska saatavuutta näihin ei ollut. Emuloinnin ja laitetestien mukaan ratkaisu kuitenkin toimii, joten sivuston pääperiaatteet vaikuttavat toimivilta. Opinnäytetyö tehtiin myös ilman PTTK Oy:n asiakkaiden design-suunnitelmia, joten ei voida olla täysin varmoja tyydyttääkö ratkaisun tämänhetkinen ulkoasu asiakkaita vai joudutaanko sivustoja räätälöimään ulkoasullisesti erittäin paljon. Toisaalta opinnäytetyön tarkoitus oli löytää mahdollisimman helpot keinot responsiivisuuten, joiden päälle PTTK Oy voi rakentaa mahdolliset tulevat asiakassivustoratkaisunsa.

4.4.3 Asiakkaiden sivustot

PTTK Oy:n asiakkaiden yhteisesti käyttämä sivustopohja rakentuu loogisesti ja siihen voidaan vaikuttaa erittäin paljon CSS-tyylityksillä ja JavaScriptillä. Responsiivisuus on mahdollista saavuttaa sivustolla kohdistamalla tyylitykset tiettyihin kohtiin rakennetta: yläpalkkiin, footer-osioon sekä kaikkiin palstoihin. Täysin uutta teemaa ei varmastikaan tarvitse tehdä välttämättä, mutta vanhaa teemaa voisi päivittää tai ainakin nykyisten sivujen suunnittelua miettiä responsiivisuuden kannalta. Suurimpia huomiotavia seikkoja ovat navigaatio sekä kuvat. Sivustoilla on käytetty paljon kuvallisia elementtejä ja jopa kuvakarusellejä. Ratkaisussa kuvat saadaan skaalautumaan näyttöä silmälläpitäen, mutta ne ovat raskaita elementtejä ladattavaksi. Kuvien määrää

olisi hyvä miettiä sivustoilla, ja niiden kokoa mahdollisuuksien mukaan pienentää.

PTTK Oy:n asiakkaiden navigaatio on erittäin haasteellinen responsiivisuuden osalta. Pääkategorioita sivustoilla on vähintään 6 kappaletta poikkeuksetta, ellei jopa enempää. Opinnäytetyössä esitetty ratkaisu toimii parhaiten, jos päänavigaatiossa on 6 kappaletta tai vähemmän painettavia elementtejä. Jos tästä määrästä mennään yli, ei navigaatio välttämättä mahdu enää mobiilille näytölle ja sitä täytyy tällöin muokata CSS-tyylityksillä pienemmäksi. Pienentämisellä on aina riskinsä, sillä on muistettava, että mobiilissa ympäristössä elementtien painaminen tapahtuu sormella. Alisivustoja voi periaatteessa ratkaisussa olla niin paljon kuin vain asiakkaat haluavat. Mitä enemmän niitä on, sitä enemmän käyttäjät joutuvat vierittämään sivua pelkästään selatakseen alisivu mahdollisuuksia. Alisivuissakin täytyy siis käyttää myös kohtuutta, jos halutaan taata käyttäjätyytyväisyys.

Opinnäytetyössä esitetty ratkaisu on pyritty pitämään ulkoasullisesti mahdollisimman neutraalina. Kaikissa sivustoissa, joissa ratkaisua tullaan hyödyntämään, täytyy tarkasti varmistaa sivuston tyylitysten yhdenmukaisuus. Yläpalkin päänavigaatio täytyy ainakin tyylittää, sekä tehdä nappi-elementit, joita sivuilla tullaan käyttämään. Sivustoilla käytettävien portlettien toimivuus tulisi myös tarkistaa.

4.5 Projektin arviointi

Projektin työ-osuus onnistui hyvin ja PTTK Oy:n yhteyshenkilö oli siihen tyytyväinen. Projektissa onnistuttiin sen päätavoitteissa: löydettiin ratkaisu, jota voitiin käyttää saavuttamaan responsiivisuus PTTK Oy:n asiakassivustoissa. Projektissa nähtiin onnistumisia sekä epäonnistumisia. Onnistumisiin voidaan luetella työ-osuus, toimeksiantajan tyytyväisyys sekä opinnäytetyöntekijän kokemuksen karttuminen. Epäonnistumisiin voidaan katsoa raportointi sekä ajankäyttö.

4.5.1 Suunnittelu

Opinnäytetyössä suunniteltiin ensimmäisenä aikataulua ja alkuperäiseksi suunnitelmaksi muodostui: joulukuussa tietojenkeruuta, tammi-maaliskuussa projektityön tekemistä samalla täydentäen aikaisempaa tietoperustaa. Raportin kirjoittaminen olisi samanaikaista toimeksiannon työn kanssa. Maaliskuussa oli tarkoitus viimeistellä projekti ja raportti. Tietojenkeruu alkoi suunnitellusti, mutta pääsy demosivustolle ja omasta kiinnostuksesta johtuen työosion aloitus siirtyi myös joulukuuhun. Raportin kirjoittaminen ei kuitenkaan siirtynyt työn aloittamisen ohessa ja vuodenvaihteen lomien jälkeen minun oli hankala aloittaa kirjoittamaan raportin osuutta jo tehdystä työstä.

Ulkoasun suunnittelua ei projektissa ollut paljoa. Jokaisella PTTK Oy:n asiakkaalla on oma tyylinsä sivustollansa, joten jokaista sivua täytyy tyylittää myös ratkaisun hyödyntämisen jälkeen erikseen. Ulkoasussa pyrittiin kuitenkin noudattamaan periaatetta, että muokattujen sivustojen elementtien ulkoasu noudattaisi mahdollisimman tarkasti työpöytäversioita. Näin elementtien tyylitys helpottuisi myöhemmissä vaiheissa.

Helmikuussa keskustellessa tarkemmin PTTK Oy:n yhteyshenkilön kanssa päädyttiin muutaman ehdotuksen jälkeen ratkaisuun, jossa päätettiin keskittyä enemmän puhelinversioon. Tablet-versio tyydyttiin jättämään pienennettyyn versioon työpöytäversiosta, jossa olisi muutettu ainoastaan yleisiä toimintoja hiirettömään ympäristöön. Yhteyshenkilö ja projektityöntekijä myös tällöin suunnittelivat yhdessä navigaation ulkoasua.

4.5.2 Ajankäyttö

Projektityön osalta ajankäyttö oli tehokasta. Alun tietojenkeräyksen sekä työvälineistön asennuksen jälkeen projekti lähti hyvin käyntiin. Projektin tekoa oli pääsääntöisesti joulukuusta 2014 helmikuuhun 2015. Maaliskuussa työhön tehtiin viimeiset testaukset sekä hiomiset. Projektin työtahti oli pääsääntöisesti kaksiviikkoinen, eli PTTK Oy:n yhteyshenkilöön pidettiin yhteyttä kahden viikon

välein. Yhteyshenkilön kanssa katsottiin yhdessä läpi miltä ratkaisu näytti sillä hetkellä tai hän kommentoi asiaa sähköpostiviestin kautta. Kommenttien jälkeen tiedettiin mitä osioita pitäisi työstää.

Kirjoitusprosessi alkoi joulukuussa 2014 tietojen keräämisellä. Prosessi eteni aluksi melko hyvin ja yleistä tietoa aiheesta löytyi. Kirjoituksen ajankäytössä ilmeni kuitenkin ongelmia opinnäytetyön työn raportointivaiheessa alussa sekä loppupuolella ja opinnäytetyö viivästyi tavoiteaikataulusta. Olin aliarvioinut kirjoitusprosessiin tarvittavan ajan.

4.5.3 Yhteydenpito

Toimeksiantajan kanssa sovittiin pääsääntöiseksi yhteydenpitovälineeksi sähköposti ja yhteiset kokoontumiset tarvittaessa. Mahdollisuus oli myös videokokoukseen, mutta sitä ei käytetty kertaakaan opinnäytetyön aikana. Alun tietoperustan keräämisen ajaksi sovittiin, että toimeksiantajan edustajaan otetaan yhteyttä kun tietoperusta on kerätty. Tietojen keräyksen jälkeen yhteydenpito alkoi tiiviimmin. Projektin tiimoilta otettiin tavoitteeksi, että yhteyttä yhteyshenkilöön pyrittäisiin ottamaan vähintään kahden viikon välein. Toiseksi tavoitteeksi otettiin, ettei yhteydenpitokatkos kestäisi pahimmassakaan tapauksessa yli kuukautta. Toimeksiantajan edustaja toivoi, että yhteyttä otettaisiin aina kun jotain kysyttävää ilmenisi tai ratkaisu olisi edennyt merkittävästi. Pääsääntöisesti yhteydenpitoaikataulutuksessa pysyttiin. Kuitenkin muutaman kerran yhteydenpitokatkos kesti 3–4 viikkoa, koska projektissa ei tuntunut, että siinä olisi edetty tarpeeksi paljon tulosten etenemiseen tai muutoksilla saataisiin vielä nopeasti oma ratkaisu aikaan. Näissä tapauksissa useasti ratkaisu kuitenkin viivästyi.

Yleisten esittelyjen sekä ratkaisujen kommentoimisten lisäksi tärkeässä asemassa oli edustajan kommentit käyttöliittymän ulkoasuun liittyen. Yhteyshenkilö oli tehnyt PTTK Oy:n asiakkaiden kanssa läheistä yhteistyötä heidän sivustojensa työpöytäversioissa. Näin ollen hänellä oli kokemusta sanoa, minkälaiseen ulkoasuun kannattaisi pyrkiä.

Sähköposti pääsääntöisenä välineenä yleisissä raportoinneissa toimi hyvin, koska sen avulla voitiin ilmoittaa onnistumisista sekä epäonnistumisista PTTK Oy:n edustajalle. Suuria tiedostoja ei tarvinnut lähettää edestakaisin, koska molemmilla oli pääsy demosivustolle josta ratkaisut näki hyvin. Tapaamiset olivat myös tärkeitä ja aina rakentavia. Niissä suunniteltiin aina pitemmän aikavälin tavoitteita sen hetkisten tulosten kommentoinnin lisäksi.

5 Yhteenveto

PTTK Oy:n asiakkaiden yhteinen maakunnallinen teema rakentuu loogisesti ja siihen on helppo kohdistaa tyylityksiä. Rakenteisella portletilla saadaan myös luotua portletille ja sen toiminnoille luokat ja id:t, joiden avulla voidaan kohdistaa tyylityksiä sen elementteihin. Asiakkaiden sivustojen ulkoasua on siis tehokasta muokata CSS-komennoilla ja koska monilla asiakkailla on samaan pohjaan, layoutiin, tehty sivut, voidaan niitä tehdä tehokkaasti.

Opinnäytetyön tuloksena demo-sivulle tehtiin ulkoasulliseen responsiivisuuteen mahdollistava CSS-tiedosto. Uusia rakenteisia portletteja jouduttiin myös luomaan, ja JavaScriptillä luotiin uutta toiminnallisuutta. Responsiivisuus toteutettiin mediahauilla. Toimeksiantajalle tehtiin myös ratkaisuun pohjaavat ohjeet, joita hyväksikäyttämällä PTTK Oy:n asiakkaiden sivut voidaan muuttaa responsiivisiksi. Ohjeissa käytettiin paljon jo olemassa olevia materiaaleja, joten yhteisymmärryksessä toimeksiantajan kanssa päätettiin käyttöohjeet jättää julkaistavan raportin ulkopuolelle.

Projektin aikana opittiin monia uusia asioita Liferay-portaalista, vaikka se oli tuttu jo aikaisemmasta harjoittelusta PTTK Oy:ssä sekä omatoimisissa kokeiluissa. Sen syvällisempi tarkastelu oli aiemmin jäänyt hieman vähemmälle huomiolle. Portaalin yleisestä toiminnasta oli hyvä opiskella syventävää teoreettista tietoa ja rakenteinen portletti on tärkeä kehitystyökalu Liferayn-toiminnassa, joten oli erittäin mielenkiintoista tutustua siihen syvemmin. Responsiivisuus oli käsitteenä tuttu, mutta sitä ei aikaisemmin ollut käytännössä

tehty tai testattu. Opinnäytetyön ansiosta tutustuttiin yleiseen web-kehitykseen ja sen historiaan hieman tarkemmin. Samalla saatiin parempaa kuvaa web-kehityksen tulevaisuudesta ja mihin suuntaan se on kehittymässä. Tiedoista on paljon hyötyä, jos päätetään lähteä töihin web-kehityksen saralle.

Opinnäytetyön tieto-osuus pohjautuu etupäässä englanninkielisiin internet-lähteisiin, mutta myös kirjallisuuslähteitä on käytetty. Liferay-alustan kirjallisuutta ei löytynyt ja tietoperusta siitä täytyi etsiä pelkästään internetistä. Liferayn omilla sivuilla on kehittäjille suunnattua materiaalia. Alustan yhteisöpohjaisuus takasi sen, että ongelmakohtiin löytyi myös suuntaa antavia keskusteluja samoista ongelmista. Responsiivisuutta käsittelevät lähteet ovat myös internetistä. Tänä päivänä web-suunnittelussa on relevanttia, että aiheita pohditaan artikkeleissa ja blogeissa. Näitä on myös hyödynnetty tässä opinnäytetyössä.

Liferay-portaali vaikuttaa tehokkaalta tavalta tehdä suurempia sivustokokonaisuuksia. Sen peruskäyttö on helposti omaksuttavissa myös ihmisille, jotka eivät tietokoneita ole paljoa käyttäneet ja he voivat toimia sisällöntuottajina sivustoilla pienellä perehdytyksellä. Sivuston osien omakohtainen rakentaminen vaatii jo suurempaa tietotaitoa ja paneutumista portaaliin, mutta sekin on melko helposti omaksuttavissa.

Opinnäytetyön toiminnallinen osuus onnistui melko hyvin. Toimeksiantaja oli tyytyväinen ratkaisuun, ja työn tekemiseen. Ylitsepääsemättömiä ongelmia projektissa ei kohdattu. Haasteellisimmat osiot liittyivät yläpalkin rakenteiseen portlettiin, vaikka aikaisempaa kokemusta aiheesta myös oli. Portletin pohjan Velocityn syvällisempi käyttäminen oli haastavaa. Työssä yleiset responsiivisuuden periaatteet saatiin toimimaan melko helposti. Kuitenkin, hienosäätö, testaus ja sivuston toimiminen halutulla tavalla vaati odotettua suuremman osan projektityön suunnitellusta ajasta. Lopputestauksen voidaan todeta jääneen hieman puutteelliseksi sekä fyysisten testilaitteiden määrä olisi voinut olla myös suurempi.

Raportointiosuus kohtasi haasteita, vaikka se aloitettiin hyvissä ajoin. Alussa raportointi ja muistiinpanot työstä olivat osaksi tajunnanvirtaa, eikä täysin

mietittyä materiaalia. Tästä johtui, että raporttia kirjoittaessa täytyi lähes kaikki kirjoittaa uudestaan. Kerran kirjoitettua tekstiä oli vaikea myöhemmin palata korjaamaan. Opinnäytetyöprosessin aikana oman tekstin kieliasun kriittinen arviointi kuitenkin kehittyi.

Lähteet

Airaksinen, T & Vilkka, H.2003. Toiminnallinen opinnäytetyö. Helsinki: Tammi.

Apache Software Foundation. 2015. Apache Tomcat. http://tomcat.apache.org/. 17.4.2015.

Carlson, R. 2011. The Evolution Of Web Design: 1990-Present. DesignJuices. 1.09.2011. http://www.designjuices.co.uk/2011/09/web-design-evolution/. 17.4.2015.

Cavanaugh, N. 2014. The future of UI Development and AlloyUI in Liferay 7. Company Blogs. 17.9.2014. https://www.liferay.com/web/nathan.cavanaugh/blog/-/blogs/the-future-of-ui-development-and-alloyui-in-liferay-7. 17.4.2015.

Chapman, C. 2009. The Evolution of Web Design. Six Revisions. http://web.archive.org/web/20131030030802/http://sixrevisions.com/web_design/the-evolution-of-web-design/. 17.4.2015.

De Graeve, K. 2011. Responsive Web Design. Microsoft. https://msdn.microsoft.com/en-us/magazine/hh653584.aspx. 17.4.2015.

Frost, B. 2012. Creating a Mobile-First Responsive Web Design. HTML5Rocks. http://www.html5rocks.com/en/mobile/responsivedesign/. 17.4.2015.

Johnson, J, 2013. Mobile First Design: Why It's Great and Why It Sucks. DesignShack http://designshack.net/articles/css/mobilefirst/. 17.4.2015.

Liferay Inc. 2015a. Portal Features. http://www.liferay.com/products/liferay-portal/features/portal. 17.4.2015.

Liferay Inc. 2015b. Web Platform. http://www.liferay.com/products/what-is-a-portal/web-platform. 17.4.2015.

Liferay Inc. 2015c. Web Content Management System. http://www.liferay.com/products/what-is-a-portal/wcm. 17.4.2015.

Liferay Inc. 2015d. Integration Platform. http://www.liferay.com/products/what-is-a-portal/integration-platform.17.4.2015.

Liferay Inc. 2015e. Collaboration Platform. http://www.liferay.com/products/what-is-a-portal/collaboration-platform. 17.4.2015.

Liferay Inc. 2015f. Social Applications Platform. http://www.liferay.com/products/what-is-a-portal/social-apps-platform. 17.4.2015.

Liferay Inc. 2015g. Get Liferay Portal. https://www.liferay.com/downloads/liferay-portal/overview. 17.4.2015.

Liferay Inc. 2015h. Portlets. https://dev.liferay.com/participate/liferaypedia/-/wiki/Main/Portlets. 17.4.2015.

Liferay Inc. 2015i. Robust Functionality. https://www.liferay.com/documentation/liferay-portal/6.0/administration/-/ai/robust-functionali-2. 17.4.2015.

Liferay Inc. 2015j. Developing applications for Liferay. https://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/developing-applications-for-lifer-3. 17.4.2015.

Liferay Inc. 2015k. Understanding the Two phases of Portlet execution. http://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/understanding-the-two-phases-of-portlet-executi-4. 17.4.2015.

Liferay Inc. 2015l. Advanced content with structures and templates. https://www.liferay.com/documentation/liferay-portal/6.1/user-guide/-/ai/lp-6-1-ugen03-advanced-content-creatio-10. 17.4.2015.

Liferay Inc. 2015m. Creating Liferay Themes. https://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/creating-liferay-them-7. 17.4.2015.

Liferay Inc. 2015n. Creating a Theme. https://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/creating-a-the-4. 17.4.2015.

Liferay Inc. 2015o. Anatomy of a theme. https://dev.liferay.com/develop/tutorials/-/knowledge_base/6-1/anatomy-of-a-theme. 17.4.2015.

Liferay Inc. 2015p. Extending and customizing Liferay. https://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/extending-and-customizing-lifer-3. 17.4.2015.

Liferay Inc. 2015q. Creating Liferay Layout Templates. https://dev.liferay.com/develop/tutorials/-/knowledge_base/6-1/creating-liferay-layout-templates. 17.4.2015.

Liferay Inc. 2015r. JavaScript. https://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/javascri-5. 17.4.2015.

Liferay Inc. 2015s. Displaying site pages to mobile devices. https://dev.liferay.com/discover/portal/-/knowledge_base/6-1/displaying-site-pages-to-mobile-devices. 17.4.2015.

Liferay Inc. 2015t. Device Recognition Provider EE. http://www.liferay.com/marketplace/-/mp/application/35419014. 17.4.2015.

Liferay Inc. 2015u. Installation. https://www.liferay.com/documentation/liferay-portal/6.1/development/-/ai/installati-6. 17.4.2015.

Lecomte, J. 2014. Important Announcement Regarding YUI. Yahoo Engineering http://yahooeng.tumblr.com/post/96098168666/important-announcement-regarding-yui. 17.4.2015.

Marcotte, E. 2010. Responsive Web Design. A List Apart. http://www.alistapart.com/articles/responsive-web-design/. 17.4.2015.

Mozilla Foundation. 2014a. Specificity. https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity.17.4.2015.

Mozilla Foundation. 2014b. @viewport. https://developer.mozilla.org/en-US/docs/Web/CSS/@viewport. 17.4.2015.

Mozilla Foundation. 2015a. About JavaScript. https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript. 17.4.2015.

Mozilla Foundation. 2015b. A re-introduction to JavaScript (JS-tutorial). https://developer.mozilla.org/en-US/docs/Web/JavaScript/A_re-introduction_to_JavaScript. 17.4.2015.

Mozilla Foundation. 2015c. https://developer.mozilla.org/en-US/docs/Mozilla/Mobile/Viewport_meta_tag. 17.4.2015.

Pettit, N. 2014 The 2014 Guide to Responsive Web Design. Treehouse blog. 2.6.2014 http://blog.teamtreehouse.com/modern-field-guide-responsive-web-design. 17.4.2015.

Ruluks, S. 2014. A brief history of web design for designers. FROONT Open design blog. 4.12.2014. http://blog.froont.com/brief-history-of-web-design-for-designers/. 17.4.2015.

Suomen virallinen tilasto(SVT). 2014. Internetin käytön yleiset muutokset (korjattu 25.11.2014). http://www.stat.fi/til/sutivi/2014/sutivi_2014_2014-11-06_kat_001_fi.html. 17.4.2015.

World Wide Web Consortium. 2008. Cascading Style Sheets, level 1. http://www.w3.org/TR/REC-CSS1/. 17.4.2015.

World Wide Web Consortium. 2011. Cascading Style Sheets Level 2 Revision 1 (CSS 2.1). http://www.w3.org/TR/CSS2/. 30.3.2015.

World Wide Web Consortium. 2012. Media Queries. http://www.w3.org/TR/css3-mediaqueries/. 17.4.2015.

World Wide Web Consortium. 2015a. CSS Device Adaption Module Level 1. http://dev.w3.org/csswg/css-device-adapt/. 30.3.2015.

World Wide Web Consortium. 2015b. What is CSS? http://www.w3.org/Style/CSS/ 17.4.2015.