SURIMA-hankkeessa toteutetut tekniset asiat (UEF)

Ilja Sidoroff

Yleistä

Seuraavassa kuvataan hankkeen teknisiä osia. Erityisesti keskitytään DSpace-julkaisuarkiston pystyttämiseeen, konfigurointiin ja sen tutkimustietojärjestelmäintegrointiin.

Teknistä dokumentaatiota löytyy myös yliopiston wikistä:
https://wiki.uef.fi/display/libdspace/Kirjaston+DSpace-instanssi+Home

Yleisiä suunnitteluperiaatteita

Rinnakkaistallennettujen artikkelien ja muun aineiston tallennuspaikaksi oli hankkeen alussa valittu DSpace-niminen avoimen lähdekoodin ohjelmisto [1]. DSpace on tarkoitettu erilaisten instituutioiden julkaisuarkistoksi – se muistuttaa paljon verkkopohjaisia sisällönhallintajärjestelmiä, kuten Drupal, Wordpress tai Joomla, mutta poikkeaa niistä siinä, että ensisijaisia hallittavia kohteita ovat objektit (item) ja niihin liittyvä metadata, eikä esimerkiksi artikkeli- tai muun muotoinen sisältö.

DSpacea mainostetaan toisaalta helposti käyttöönotettavana, mutta toisaalta myös helposti muokattavana, mikä osoittautui ainakin osittain todeksi myös tämän hankkeen aikana (ks. alla). DSpace on Suomessa käytössä useassa yliopistokirjastossa, joten sen käytöstä on kertynyt kokemuksia. Ohjelmistoteknisesti DSpace on kypsä ja luotettava. Lisäksi kun DSpace-arkisto on toiminnassa, voi sen käyttöä laajentaa helposti myös muihin käyttötapauksiin rinnakkaistallentamisen lisäksi, esimerkiksi opinnäytteiden tai tutkimus(meta)datan säilyttämiseen.

DSpace-käyttökokemusten perusteella eräs DSpacen ongelmallinen ominaisuus liittyy sen muokkaamiseen ja laajentamiseen. DSpacen suhteellisen helppo muokattavuus voi houkutella lisäämään tai muokkaamaan ohjelmiston koodia käyttöliittymätason alapuolella. Omien muutosten tekeminen ja toiminallisuuksien lisääminen kuitenkin vaikeuttavat tai ainakin hidastavat uusien DSpace-versioiden käyttöönottoa, kun kaikkien muutosten toimivuus täytyy varmistaa päivityksen yhteydessä. Tämä kävi selväksi *dspace-community*-keskustelulistan perusteella [2] ja keskusteluista muiden DSpace-ylläpitäjien kanssa.

Niinpä käyttöön otettiin kolme suunnitteluperiaatetta. Ensiksi, toiminallisuuksien luomisessa pyrittiin hyödyntämään mahdollisimman paljon DSpacen olemassa olevaa tai tulevaa toiminnallisuutta. Tulevien toiminallisuuksien hyödyntäminen tarkoittaa sitä, että käyttöön otetussa versiossa 5.x saatettiin käyttää koodia, joka oli tehty versiolle 6.x.

Toiseksi, oma, lisätty ja muutettu toiminallisuus pyrittiin pitämään pieninä, loogisina kokonaisuuksina, jotta kunkin yksittäisen toiminnallisuuden toteuttavat muutokset ovat helposti eroteltavissa DSpacen lähdekoodista. Tämä tehtiin käytännössä versionhallintaohjelmiston (git) avulla, ja jää nähtäväksi onko

tavoitteessa onnistuttu myös muiden kuin aktiivisesti järjestelmän kehittämisessä mukana olleiden osalta.

Kolmas periaate oli se, että kaikki muutokset ja bugikorjaukset pyrittiin saamaan lisättyä DSpacen kanoniseen lähdekoodiin (*upstream).* Tämä tarkoittaa käytännössä sitä, että koodimuutokset tarjottiin *github*-alustan [3] kautta DSpacen kehittäjille, jotka pystyivät hyväksymään tai hylkäämään tarjotun muutoksen. Mikäli muutos hyväksyttiin, tuli siitä osa DSpacen virallista lähdekoodia ja se on virallisen ylläpidon piirissä. Mikäli muutosta ei hyväksytty, jäi muutos kuitenkin julkisesti saataville.

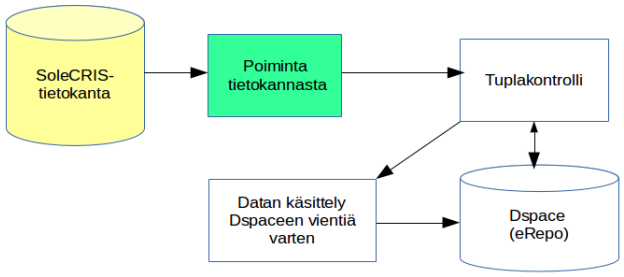
Kolmas periaate aiheutti toisaalta sen, että koodimuutosten täytyi olla tarpeeksi huolellisesti ja yleispäteviksi tehtyjä, että ne täyttivät DSpacen koodin laatukriteerit ja yleisen hyödyllisyyden. Tällä tavalla koodaaminen on hitaampaa, kuin pelkästään omaan käyttöön tehty, tietyssä ympäristössä tietyn ongelman ratkaiseva koodi, mutta toisaalta jaettu koodi käy läpi laajemman testauksen ja jos se päätyy viralliseen jakeluun, ylläpitovelka jää mitättömäksi.

Käytännössä koodin saaminen virallisen DSpacen lähdekoodiin oli melko hidasta. Tämä johtuu ennen kaikkea siitä, että projektilla ei ole vakituisesti ja jatkuvasti käytössään kokoaikaista kehittäjähenkilöstöä, vaan kehittäjät toimivat joko osa-aikaisesti tai vapaaehtoiselta pohjalta. Projektin aikana kehittäjäyhteisön tavoitteena oli myös saada DSpacen versio 6 julkaistua ja kehitystyön painopiste oli ohjelman vakauttamisessa, testaamisessa ja löydettyjen bugien korjaamisessa. Tällöin tarjottujen muutosten läpikäyminen voi kestää jopa kuukausia. Toisaalta DSpace-kehittäjäyhteisön kanssa toimimisesta oli myös suoraa hyötyä. Esimerkiksi DSpacen-kansilehti –toiminnallisuutta kehitettäessä havaittiin, että DSpacessa oleva kansilehden perustoiminnallisuus oli ollut rikki kehitysversiossa ainakin yhdeksän kuukautta. Korjaava koodimuutos saatiin nopeasti läpi viralliseen koodiin ja se hyödyttää kaikkia DSpace 6-käyttäjiä.

DSpacen lisäksi projektissa kehitettiin yksisuuntainen integraatio tutkimustietojärjestelmä SoleCRISistä DSpaceen. Tätä tarkoitusta varten ei ollut saatavilla valmista koodia, joten se kirjoitettiin projektin aikana itse. Tämäkin koodi kirjoitettiin mahdollisimman yleispäteväksi ja laitettiin avoimesti saataville. [4]

Tekniset toteutukset

SoleCRIS-DSpace –tiedonsiirto



Kuva 1

Tiedonsiirto SoleCRISin ja DSpacen välille tehtiin yhteistyössä tietotekniikkapalveluiden kanssa. Kuvassa 1 on esitetty yleiskuva siirrosta. SoleCRIS-tietokannan muoto (keltaisella) on toimittajan vastuulla. Tietotekniikkapalvelut (vihreällä) suorittaa poiminnan tietokannasta haluttujen kriteerien mukaan. Poiminta tehdään kerran päivässä ja tieto siirretään automaattisesti DSpace-palvelimelle. Poiminta SoleCRIS-tietokannasta on tiedonsiirron kannalta kriittisin vaihe, koska toimittaja voi muuttaa tietokannan rakennetta kehittäessään ohjelmistoa ja tällöin kyselyn rakennetta voi joutua muuttamaan. Suora tietokantakysely oli kuitenkin ainoa tapa, jolla pystyimme saamaan tarvittavan tiedon SoleCRIS-järjestelmästä, koska järjestelmässä ei ollut muuta, valmista ja soveltuvaa rajapintaa tiedonsiirtoon. Toisaalta kysely tietokannasta on melko yksinkertainen ja sen pitäminen ajan tasalla ei pitäisi olla vaikeaa. Lisäksi tietotekniikkapalveluiden kehitysohjelmassa on jaetun tietovaraston käyttöönotto, jolloin mahdollisesti tulevaisuudessa SoleCRIS-tiedot voidaan poimia tietovarastosta (ja tietovarasto - tutkimustietojärjestelmä integraatio on toteutettu stabiilisti).

Kuvassa valkoisella olevat osat on toteutettu projektin omana työnä ja ne liittyvät siirretyn datan jatkokäsittelyyn.

Poimitusta datasta karsitaan ensin sellaiset rivit, jotka jo ovat DSpacessa, joko valmiina tietueina tai käsittelyjonossa. Karsinta tehdään vertaamalla DSpacen tietokannan sisältöä tuleviin riveihin, käyttämällä tunnisteena SoleCRISissä olevaa id-tietoa. DSpacessa ei valitettavasti ole mahdollista saada helposti tietoa kaikista käsittelyssä olevista tietueista, koska olemassa olevat, tarkoitukseen laaditut toiminnallisuudet (REST-rajapinta, SOLR-kysely) ovat joko puutteellisia tai hieman virheellisiä. Sen vuoksi poiminta tehtiin suoraan DSpacen tietokannasta, mikä tarkoittaa sitä, jos DSpacen tietokantarakenne muuttuu merkittävästi, pitää poimintakysely tehdä uudestaan. Projektissa toteutettiin poiminta sekä DSpacen versioille 5.x ja 6.x. Lisäksi DSpacen tuleviin versioihin on odotettavissa toiminnallisuutta (REST-rajapinnan laajentuminen), jolla tietokantakyselyn voi tehdä käyttämällä valmiita rajapintoja.

Tuplakontrollin jälkeen SoleCRISista saatu data muokataan DSpacen hyväksymään muotoon. Muokkaaminen tehdään kolmessa vaiheessa. Ensimmäisessä vaiheessa muokataan SoleCRISin lähtödata DSpacessa käytettävään muotoon (ohjelmisto prepare-csv). Tämä tarkoittaa metadatan muuttamista Dublin Core –formaattiin. Käytännössä esimerkiksi useat tekijät erotetaan omiin kenttiinsä ja metadatan muotoa vakioidaan (esim. Doi-tunnusten muoto pyritään yhdenmukaistamaan, aineiston tyyppi-merkinnät muunnetaan OpenAIREn hyväksymään muotoon). Tässä vaiheessa SoleCRIS dataa voi myös

rikastaa, esimerkiksi rakentamalla tietueisiin uusia kenttiä olemassaolevan datan pohjalta (esim. Viittauskenttä; dc.identifier.citation).

Toisessa vaiheessa käsitellystä datasta rakennetaan DSpacen ymmärtämä datapaketti, ns. Simple Archive Format (SAF) -paketti (ohjelmisto saf-archiver). [5] Kolmannessa vaiheessa pakettiin lisätään vielä oletuslisensiointitieto (ohjelmisto add-file) ja tehty SAF-paketti viedään DSpace-järjestelmään, jossa saapuneet tiedostot jäävät käsittelyjonoon.

Näitä vaiheita varten ei ollut olemassa valmiita, soveltuvia ohjelmistoja, joten ne kirjoitettiin itse projektin aikana. Ohjelmistot tehtiin Googlen kehittämällä go-ohjelmointikielellä (prepare-csv, saf-archiver, add-file) tai Bourne Shell -skriptauskielellä (tuplakontrolli, prosessin kontrollointi). Ohjelmistot ovat vapaasti saatavilla osoitteesta [4], osoite https://github.com/Dspace-Fi/saf-archiver sisältää SAF-pakettien laadinnan kannalta olennaiset tiedostot, osoite https://github.com/Dspace-Fi/SoleCRIS-DSpace puolestaan sisältää tuplakontrollointi- ja koko prosessin hallintaan liittyvät ohjelmat.

DSpace-ympäristön rakentaminen (eRepo)

Tiedon siirron ohella toinen laaja kokonaisuus oli DSpace-ympäristön rakentaminen. Valmis julkaisuarkisto sijaitsee osoitteessa https://erepo.uef.fi ja se pohjautuu tämän kirjoitushetkellä DSpace-versioon 5.6. Julkaisuarkiston nimi on UEF eRepository tai lyhyesti eRepo.

Julkaisuarkisto pyörii CentOS 7-virtuaalipalvelimella, tietotekniikkapalveluiden alustalla. Tietotekniikkapalvelut vastaavat palvelimen ylläpidosta ja kirjasto vastaa DSpace-ylläpidosta ja kehityksestä.

Projektissa oli alkuaan tarkoitus ottaa käyttöön DSpacen uusin versio 6.x, jonka julkaisun oli luvattu tapahtuvan alkuvuodesta 2016. Version 6 julkaiseminen kuitenkin viivästyi eri syistä - osasyynä olivat suuret sisäiset muutokset, joita lähdekoodiin oli tehty, ja niistä aiheutuneet ohjelmistoviat – ja koska versiota 6 ei oltu lokakuun 2016 alkuun mennessä julkaistu, päätettiin pysyä versiossa 5.x, joka oli otettu käyttöön jo heti projektin alussa, lokakuussa 2015. Päätös perustui kokemuksiin siitä, ettei uusia ohjelmistoversioita kannata ottaa käyttöön ensimmäisten joukossa ja myös siihen, että projektin loppu alkoi häämöttää, eikä uusien vakavien vikojen ilmaantuessa olisi välttämättä tarpeeksi aikaa korjata niitä.

Projektin aikana kuitenkin testattiin laajasti myös julkaisematonta versiota 6, ja kaikki DSpacen koodiin tehdyt muutokset ovat olemassa myös versiota 6 varten, mikä tarkoittaa sitä, että versiopäivitys 5 -> 6 on mahdollista tehdä suhteellisen turvallisin mielin – ainakin omien muutosten osalta, kun DSpace 6 on todettu riittävän stabiiliksi.

Muutokset DSpaceen pyrittiin tekemään ensisijaisesti konfiguraation kautta. Toissijaisesti muutettiin DSpacen lähdekoodia. Kaikki muutokset tallennettiin versionhallintaan, jonne lisättiin myös skripti, joka kopioi tehdyt muutokset alkuperäisen DSpacen lähdekoodin päälle. Tällä tavalla muutoksia on mahdollista hallita suhteellisen helposti, ainakin toistaiseksi, kun muutosten määrä ei suuri. Muutosten testaaminen on myös helppoa, koska tuotantoympäristöä vastaavan ympäristön voi rakentaa esimerkiksi omalle tietokoneelle ja näin testata tehtyjä muutoksia turvallisesti. Muutokset on

tallennettu yliopiston (UEF Bioinformatics Center) hallinnoimalle palvelimelle, jonne pääsyyn vaaditaan yliopiston käyttäjätunnus ja pääsyoikeuksien lisääminen versionhallintaohjelmistoon [6].

Sellaiset muutokset, jotka eivät erityisesti rajoitu Itä-Suomen yliopiston tarpeisiin, tarjottiin lisättäväksi DSpacen lähdekoodiin; merkittävin tällainen muutos oli lähinnä muutokset kansilehti-toiminnallisuuteen. Lisäksi DSpacen lähdekoodiin lähetettiin useita bugikorjauksia.

DSpacen ulkoasu muokattiin yliopiston ilmeen mukaiseksi ja sen pohjana käytettiin responsiivista Mirage 2 –teemaa, joka toimii myös mobiililaitteilla.

Vaikka DSpace on käytössä Suomessa useissa eri instituutioissa, ei DSpacen virallinen levityspaketti sisältänyt suomenkielistä käännöstä. Projektin yhteydessä toteutettiin myös suomenkielinen raakakäännös DSpacesta, joka liitettiin osaksi DSpacen virallista lähdekoodia. Käännöksen tekemiseen saatiin apua Jyväskylän yliopiston ja Kansalliskirjaston aikaisemmista käännöksistä. Koska käännös toteutettiin kehittäjien voimin, ei sen taso vastaa ammattilaisten tekemää käännöstä, mutta se voinee joka tapauksessa toimia pohjana jatkokäännöksille. [7] DSpacen kehitystyössä ensisijaisena kielenä käytettiin englantia, ruotsinkielisen käännöksen tekemiseen tai hankkimiseen ei valitettavasti ollut aikaa eikä osaamista.

Kokemuksia ja jatkokehitysmahdollisuuksia

Projektissa hankitut kokemukset ovat vielä tässä vaiheessa vaillinaisia, koska eRepo-järjestelmää ei ole otettu vielä täyteen tuotantokäyttöön, vaikka sitä onkin jo melko laajasti testattu. Tärkein palaute järjestelmästä tulee sen loppukäyttäjiltä, eivätkä loppukäyttäjät ole vielä pystyneet käyttämään järjestelmää tarpeeksi, jotta nähdään mitkä osat ovat toimivia ja mitkä eivät. Tämän vuoksi parhaan tuloksen saamiseksi voi olla järkevää kerätä kokemuksia eRepon ja siihen liittyvien järjestelmien toiminnasta ja varata resursseja järjestelmän jatkokehittämiseen myöhemmin.

Teknisesti järjestelmän kehittäminen ei osoittautunut vaikeaksi, vaikka tiedonsiirto SoleCRIS-järjestelmän ja DSpacen välillä ei olekaan laadittu teknisesti hyvien käytäntöjen (muuttumattomien rajapintojen) kautta. Tehty ratkaisu oli kuitenkin helppo, eikä tuottanut lisäkustannuksia esimerkiksi ohjelmistokehitysmaksujen muodossa, ja tulevat vuodet osoittavat miten se jatkossa toimii. Projektin aikana havaittiin, että yhteistyö julkaisutietojärjestelmien ja julkaisuarkistojen välillä on haastavaa myös muiden ympäristöjen välillä, eikä mitään yhtä, valmista ratkaisua ole vielä olemassa.

Projektissa toteutettu malli perustuu DSpacen perustoiminnallisuuteen ja sen tukena käytetään muita tietojärjestelmiä (julkaisutietojen syöttölomake). DSpace kuitenkin mahdollistaisi myös monimutkaisempien työnkulkujen toteuttamisen järjestelmän sisällä (xmlworkflow). Valitettavasti tällaisen järjestelmän suunnitteluun ja toteutukseen ei ollut projektin sisällä aikaa. Yhtenä jatkokehitysmahdollisuutena voisi olla koko julkaisutietojen syöttö- ja rinnakkaistallennustoiminnallisuuden tekeminen DSpacen omilla välineillä.

Yhteistyö DSpace-kehittäjäyhteisön kanssa vaikuttaa osoittautuneen hyväksi ratkaisuksi. Yhteistyö hidastaan jonkin verran teknisten toteutusten tekemistä, mutta tuo mahdollisesti etuja helpottuneen ylläpidon muodossa. Toinen tärkeä seikka on verkottuminen ja vuorovaikutus kansainvälisen

kehittäjäyhteisön kanssa, mikä puolestaan tuo erilaisia aineettomia hyötyjä (osaamisen jakaminen, mainehyödyt, epävirallisen tuen saaminen).

Projektin aikana pyrittiin myös luomaan epävirallinen verkosto suomalaisten DSpace-kehittäjien välille (DSpace-FI kanava Slack-palvelussa, DSpace-FI –koodivarasto Github-palvelussa). Jää vielä nähtäväksi, ottaako tällainen epävirallinen yhteistyö tulta alleen.

Viitteet

[1] http://www.dspace.org/

[2] https://groups.google.com/forum/#!forum/dspace-community

[3] https://github.com/DSpace/DSpace

[4] https://github.com/dspace-fi

[5] https://wiki.duraspace.org/display/DSDOC5x/Importing+and+Exporting+Items+via+Simple+Archive+Format

[6] https://bioinformatics.uef.fi/gitlab/uef-lib/UEF-DSpace

[7] https://wiki.uef.fi/pages/viewpage.action?pageId=40417094