Tallinna Ülikool
Informaatika Instituut

SCORM-sisupakettide esitlusvahend e-õppekeskkonnas IVA

Bakalaureusetöö

Autor: Vahur Rebas
Juhendaja: Mart Laanpere

Autor:.....2008.a.
Juhendaja: .....2008.a.
Instituudi direktor: .....2008.a.

Tallinn 2008

Sisukord

Sissejuhatus

Seoses veebi ja veebirakenduste kiire arenguga ja levikuga on üha rohkem levinud ka e-õpe. Koolid juurutavad e-õppekeskkondi, õpetajad loovad e-materjale ja õpilased osalevad e-kursustel. Võib jääda mulje, et e-õpe ei ole enam eriti uudne teema, kuid see mulje on ekslik. E-õppekeskkondade ja materjalide üha suurenev hulk toob kaasa selliseid probleeme, millele varem ei pööratud tähelepanu. Nende hulgast üks on e-õppematerjalide ja keskkondade koostalitusvõime – interoperaablus. Koostalitlusvõimelised veebirakendused suudavad vajadusel automaatselt üksteisega andmeid vahetada, muuhulgas ka õppematerjale. Tänapäeval on oluline e-õppematerjalide puhul tagada nende ristkasutatavus erinevates õppekeskkondades, vältimaks õppesisu puhul tootjalukustust (i.k. *vendor lock*) ja tagamaks võimalikult pikka kasutamisiga. Täiendavaid probleeme tekitab see, et üha enam luuakse õppematerjale nõndanimetatud Web 2.0 keskkonnas ja sotsiaalsete tarkvarade vahendusel. See suurendab veelgi vajadust erinevate keskkondade koostalitusvõime järele.

Enamus e-õppekeskkondi mainivad enesetutvustustes, et nad on koostalitusvalmid. Tegelikkuses see nii ei ole. Formaalselt on nad interoperaablust tagavatele standarditega kooskõlas, kuid tegelikult oskavad vähesed süsteemid kõiki funktsionaalsusi kaotamata õpiobjekte omavahel vahetada.

Üks selliseid koostalitlusvõime tagamiseks mõeldud spetsifikatsioon on SCORM - Shareable Content Object Reference Model. Selle standardi rakendamine veebiteenuste ülesehitamisel võimaldab erinevatel süsteemidel omavahel õpiobjekte vahendada. Käesoleva bakalaureusetöö eesmärk on uurida SCORM-spetsifikatsioonile vastavate õppematerjalide esitlusvõimalusi IVA keskkonna näitel.

Kuigi SCORM ei ole veel jõudnud ametliku ISO või IEEE standardi staatusesse, on see kujunenud *de facto* standardiks digitaalsete õppematerjalide tootmisel. Kõik suuremad õpikeskkonnad (WebCT, Moodle, jne) toetavad seda ammu, kuid IVA mitte. Kuna järgmise põlvkonna e-õppe lahendused on hajutatud ja teenusorienteeritud arhitektuuriga, siis SCORM-pakettide esitaja (*SCORM player*)

tuleks teha nii, et see poleks IVA-sisene moodul, vaid seda saaks veebiteenusena (*Web Service*) kasutada ka teised süsteemid.

Käesoleva bakalureusetöö raames uuritakse, kuidas realiseerida SCORM-pakettide esitlusvahend veebiteenuste abil. Eesmärgi saavutamiseks:
- tutvustatakse e-õppekeskkondi ja -vahendeid, mida kasutatakse SCORM pakettide vahendamise eksperimendis (IVA, LeMill, eXe),
- kirjeldatakse SCORM spetsifikatsiooni,
- vaadeldakse erinevaid veebiteenuste standardeid,
- luuakse IVA keskkonnaga koostalitlev SCORM-sisupakettide esitlusteenuse Caldoz prototüüp,
- testitakse Caldozi prototüübi jõudlust.

Käesolev bakalaureusetöö toetub autori poolt eelnevalt teostatud seminaritööl.

1. Veebirakendused e-õppe kontekstis

Järgnev peatükk käsitleb mõningaid e-õppega seotud programme ja veebirakendusi, mis võimaldavad kasutajal õppematerjale toota või teistega vahetada. Selles vaatlen digitaalsete õppematerjalidega seotud standardeid ja spetsifikatsioone.

1.1. Veebipõhised õpikeskkonnad ja muu seonduv tarkvara

1.1.1. IVA

IVA on veebipõhine, vabavaraline õpikeskkond, mis on loodud Tallinna Ülikooli haridustehnoloogia keskuse ja informaatika osakonna koostöös. IVA aluseks on võetud Helsingis loodud õpihaldussüsteem FLE3[1] (Future Learning Environment).
IVA jaguneb neljaks osaks:

- Veebilaud

  Õpilase personaalne ala, kus ta saab infot oma kodutööde kohta ning oma tehtud kodutööd üles laadida. Samuti saab õpilane veebilaualt tagasiside oma tegemiste kohta.
- Raamaturiiul

  Sektsioon, kuhu õppejõud saavad üles laadida kursuse materjale. Raamaturiiulil asub ka kursuse avaleht - blogi. Õppejõud saab siin avaldada infot selle kohta, kuidas kursus on üles ehitatud.
- Töötoad

  Rühmatööde jaoks mõeldud ala. Siit leiab neli vahendit:
  - Teadmuspaja – foorumilaadne tööriist.
  - Meediapaja – vahend erinevate meediafailidega töötamiseks.
  - Rühmaportfooliod – gruppide moodustamine ja grupitöö tulemusena valminud materjalide üles laadimine.
  - Mõistekaardid – Siin on võimalik, kas grupitööna või individuaalselt, koostada mõistekaarte.
- Haldus

  Siin saab õppejõud liita teisi IVA kasutajaid kursusele õppijaks või õppejõuks, anda kodutöid, koostada teste ja muuta kursuse infot.

[1] http://fle3.uiah.fi

IVA administraatoril on siin võimalik muuta IVA seadeid, luua uusi kasutajaid ja kursuseid.

Lisaks nendele neljale liitus IVA versioonis 0.7.4 veel ka sektsioon Esik, kus õppija saab vahetada kursust, muuta oma seadeid, pidada oma õpipäevikut jne.

IVA on kirjutatud Python[2] programmeerimiskeeles ja töötab Zope[3] rakendusserveris. [1]

1.1.2. LeMill

LeMill on veebipõhine tarkvara, mis võimaldab kasutajatel õppematerjale leida, koostada ja jagada. Kõik õppematerjalid on LeMillis *Creative Commons*[4] litsentsiga, mis tähendab, et inimesed võivad LeMillis asuvaid materjale vabalt kasutada ja muuta. [10]

LeMilli sihtgrupp on eelkõige õpetajad ja selle eesmärk on lihtsustada õpetajatel õppematerjalide loomist ning julgustada neid õppematerjalide jagamisel.

LeMill on jagatud neljaks sektsiooniks:

- Materjalid

  Siin sektsioonis asuvad kõik materjalid.
- Meetodid

  Siin saavad õpetajad kirjeldada meetodeid, mida nad tunnis õpetamisel rakendavad.
- Vahendid

  Siit võib leida erinevate tööriistade kirjeldusi, mida õpetajad kasutavad.
- Kogukond

  Siin sektsioonis saab vaadata kasutajaid, kes on liitunud LeMilliga. Samuti saab siin luua gruppe, et ühiselt materjale luua.

Veel võimaldab LeMill koondada endale sobivad materjalid, meetodid ja vahendid kokku üheks kogumikuks. Kogumikkude juurde on võimalik kirjutada pisikesi vihjeid selle kohta, kuidas seda kõige parem kasutada oleks. Viimane, ja autorile kõige rohkem huvi pakkuv lahendus, on kogumikkude allalaadimine. Kogumikke saab alla laadida HTML failidena või SCORM paketina. Viimase saab omakorda üles laadida mõnda õpikeskkonda, mis toetab SCORM pakette.

---

[2] http://www.python.org
[3] http://www.zope.org
[4] http://creativecommons.org/

LeMill on vabavaraline, avatud lähtekoodiga programm, mis on kirjutatud Python programmeerimiskeeles ja töötab Zope rakendusserveris. LeMill on ehitatud sisuhaldussüsteemi Plone[5] laiendusmoodulina.

1.1.3. eXe

eXe – eLearning XHTML editor – on vahend, mille abil saavad õppejõud luua ja avaldada materjale. eXe-ga töötamiseks on vajalik programm enne alla laadida ja oma arvutisse installeerida. eXe-ga töötamiseks ei ole interneti ühendus vajalik. eXe on üles ehitatud nii, et üks leht koosneb erinevatest vormielementidest, mida nimetatakse *instructional device* e. *iDevices*, mille kõige parem eestikeelne vaste on õpivahend. Õpivahendid sisaldavad tavapäraste piltide ja tekstilõikude kõrval ka pedagoogilise vahendeid, nagu näiteks eesmärgid ja õpitegevused. [11]

Mõned näited eXe õpivahenditest:

- Flash – video lisamise võimalus.
- Mõtisklus – reflekteerimisülesanne.
- SCORM Test – valikvastustega test, kus õppijale näidatakse tema tulemust protsentides. Tulemusi ei salvestata ja õpetajal pole võimalik neid tulemusi jälgida.
- Pildigalerii – paljude piltide esitamiseks ühel leheküljel.
- Väline veebileht – avab sisestatud internetiaadressil asuva veebilehe otse lehekülje sisse, etteantud suurusega aknas.
- Wikipedia artikkel – avab lehekülje sisse Wikipedia artikli, mida õpetaja eraldi toimetada ei saa, kuid õppija saab sealt hüperlinkide abil edasi liikuda.

eXe-s valmistatud materjale saab programmist eksportida kolmel viisil:

- SCORM paketina
- IMS paketina
- HTML failidena

SCORMi ja IMS spetsifikatsioonile vastavaid pakette saab omakorda importida näiteks mõnda õpikeskkonda, mis neid standardeid toetab. HTML faile saab aga näiteks oma kodulehele üles riputada.

eXe on vabavaraline avatud lähtekoodiga programm, mis on põhiliselt kirjutatud Python programmeerimiskeeles. eXe põhineb vabavaralisel veebilehitsejal Firefox[6].

---

[5] http://plone.org

1.2. Digitaalsete õppematerjalide standardid

Digitaalsete ja ka mitte digitaalsete õppematerjalidega on seotud mitmeid organisatsioone ja standardeid. Järgnevalt vaatlen mõningaid suuremaid ja laiemalt kasutuses olevaid standardeid.

AICC

AICC on rahvusvaheline lennundusega seotud ühendus, mis koondab ühe mütsi alla alla õppe professionaalid. [15]

AICC loodi 1988 aastal ning tema põhilised eesmärgid on:

- Arendada juhiseid, mis innustaksid arendama digitaalseid õppematerjale.
- Arendada juhiseid, et tagada interoperaablus.
- Pakkuda avatud foorumit, et arutada arvutipõhist ja teisi õpitehnoloogiaid.

Praeguseks on tehnilisi juhendeid kokku juba 11.

Dublin Core

Dublin Core (DC) on standard ressursside kirjeldamiseks, et hiljem hõlpsamalt ressursse leida. DC koosneb 15 meta-andmete elemendist, mis kõik on vabatahtlikud ja võivad ühes kirjes esineda ükskõik mitu korda. [17]

LOM

Learning Object Metadata (LOM) standard kirjeldab õpiobjekti meta-andmete süntaksi ja semantikat. LOM keskendub minimaalselt vajalikule kogusele meta-andmetele, et õpiobjekt oleks hallatav, leitav ja hinnatav. Mõned parameetrid, mida LOM defineerib on autor, formaat ja pealkiri. LOM võib kirjeldada ka pedagoogilisi parameetreid õpiobjekti kirjeldamiseks, nagu näiteks eeldusained, tase ja õpistiil.

Ühel objektil võib olla ka mitu komplekti meta-andmeid, mis võivad olla ka eri keeltes.

IMS

IMS *Global Learning Consortium* on rahvusvaheline mittetulundusorganisatsioon, mis tegeleb erinevate tehniliste õpistandardite spetsifikatsioonide väljatöötamisega.

---

[6] http://www.firefox.com

Laiemalt kasutatavad IMS spetsifikatsioonid:

- *IMS Content Packaging*

  IMS sisupakendus spetsifikatsioon räägib, kuidas digitaalseid õpimaterjale kirjeldada ja pakendada nii, et erinevad süsteemid saaksid neid kasutada.
- *IMS Question & Test Interoperability*

  IMS küsimuste ja testide spetsifikatsioon kirjeldab andmemudeli testi ja testiküsimuste esitamiseks nii, et need oleks kasutatavad erinevates testimissüsteemides.
- *IMS Learning Resource Meta-data Specification*

  Õpiressursi meta-andmete spetsifikatsioon kirjeldab, kuidas ja mis andmetega rikastada õpiobjekti. See põhineb LOM standardil.

Järgmises peatükis uurin veel ühte digitaalsete õppematerjalidega seotud standardit.

2. SCORM spetsifikatsioon

SCORM on saanud *de facto* standardiks, kus õpiobjekt suhtleb õpihaldussüsteemiga. Õpiobjekt SCORM mõistes on *Shareable Content Object* (SCO) ehk jagatav sisuobjekt. SCO koosneb tavaliselt paljudest pisikestest komponentidest – pildid, videod, audio, jne. Pisikesed komponendid on rikastatud meta-andmetega, mis vastavad IMS MD spetsifikatsioonile. SCORMi paketid on pakendatud IMS *Content Packaging* standardile vastavalt. SCORM on kollektsioon standarditest ja spetsifikatsioonidest veebipõhise e-õppe jaoks.

SCORM 2004 koosneb viiest raamatust:

- *SCORM Overview*
  Kirjeldab lühidalt SCORMi ja seda, kuidas ülejäänud raamatud on omavahel seotud.
- *Content Aggregation Model* (CAM)
  Kirjeldab, mis komponente on võimalik kasutada SCORM paketis, kuidas neid komponente pakendada, et neid transportida ühest süsteemist teise.
- *Run-Time Environment* (RTE)
  See raamat kirjeldab, kuidas mingi süsteem SCORM sisupakette serveerib. Räägitakse ka SCORM käitusaja API-st, mida SCO kasutab, et suhelda süsteemiga. Siin peetakse süsteemi all silmas süsteemi, mis serveerib SCORM paketi.
- *Sequencing and Navigation* (SN)
  Raamat kirjeldab, kuidas pakk defineerib liikumise ühelt peatükilt teisele ja mis RTE ühel või teisel juhul tegema peaks.
- *SCORM Conformance Requirements*
  Kirjeldab nõudeid, mis peavad olema täidetud, et programm saaks SCORM sertifitseerituks.

2.1. SCORMi ajalugu ja idee

SCORM on standard, millega tegeleb *Advanced Distributed Learning* (ADL) *Initiative*[7]. ADL moodustati USA Kaitseministeeriumi poolt aastal 1997 eesmärgiga moderniseerida USA armee koolitust.

Esimene versioon SCORM spetsifikatsioonist SCORM 1.0 sai valmis aastal 2000. Esimese spetsifikatsiooni näol oli tegemist SCORM spetsifikatsiooni tutvustusega ja seetõttu ei hakanud seda versiooni keegi kasutama.

Teine versioon, SCORM 1.1, valmis aastaks 2001. See oli esimene SCORM spetsifikatsioon, mille põhjal võis juba hakata rakendusi tegema. Peale mõningaste puuduste kõrvaldamist ja spetsifikatsiooni täiustamist lasti välja SCORM-i järgmine versioon.

2001 aastal valmis ka järgmine versioon - SCORM 1.2.

2004 aasta jaanuaris väljastati SCORM 2004, millele vahel viidatakse ka kui SCORM 1.3. Sellele järgnes sama aasta juulis teine väljalase.

Oktoobris 2006 väljastati SCORM 2004 kolmas trükk, mis on hetkel ka viimane. [2]

SCORMi põhiliseks ideeks on õppematerjalide taaskasutamine ja interoperaablus, mis tähendab, et üks kord valmis tehtud materjali peab olema võimalik kasutada kõigis süsteemides, mis toetavad SCORM spetsifikatsioonile vastavaid õppematerjale. Need on ka pikka aega kasutatavad ning õpisüsteemi vahetumine või uue versiooni tulek ei põhjusta seda, et kõiki õppematerjale tuleb muutma hakata või mis veel hullem – nad tuleb uuesti teha.

2.2. SCORM paketi ülesehitus ja toimimine

SCORM pakett on üks zip-fail, mis sisaldab kõiki vajalikke faile, et mingi süsteem saaks paketti esitada.

Üks sisupakk peab kindlasti sisaldama imsmanifest.xml faili ja kõiki ressursse, mida sisupakk/ta kasutab – pildid, audio- ja video failid jms. Ressurssi, mis suhtleb süsteemiga, nimetatakse SCO-ks (Shareable Content Object).

[7] http://www.adlnet.org

2.2.1. Manifest fail

Imsmanifest.xml on XML fail, mis kirjeldab sisupakki ja selle sisu. See fail on alati olemas igas IMS *Content Packaging* (CP) spetsifikatsioonile vastavas sisupakis. Kuna SCORM 2004 kasutab IMS CP spetsifikatsiooni, siis on ka igas SCORM spetsifikatsioonile vastavas sisupakis imsmanifest.xml fail olemas.

Imsmanifest.xml fail koosneb alati identifikaatorist, mis eristab ühte sisupaketti teisest, minimaalsest kogusest meta-andmetest, mis kirjeldavad sisupaketti ja ütlevad, mis SCORM versiooniga tegemist on ning vähemalt ühest ressursside nimekirjast ja peatükkidest.

```
<?xml version="1.0" standalone="no" ?>
<manifest identifier="MANIFEST_IDENTIFIER" version="1.0"
   xmlns="http://www.imsglobal.org/xsd/imscp_v1p1"
   xmlns:adlcp="http://www.adlnet.org/xsd/adlcp_v1p3"
   xmlns:adlnav="http://www.adlnet.org/xsd/adlnav_v1p3"
   xmlns:adlseq="http://www.adlnet.org/xsd/adlseq_v1p3"
   xmlns:imsss="http://www.imsglobal.org/xsd/imsss"
   xmlns:lom="http://ltsc.ieee.org/xsd/LOM"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.imsglobal.org/xsd/imscp_v1p1 imscp_v1p1.xsd
                       http://www.adlnet.org/xsd/adlcp_v1p3 adlcp_v1p3.xsd
                       http://www.adlnet.org/xsd/adlnav_v1p3 adlnav_v1p3.xsd
                       http://www.adlnet.org/xsd/adlseq_v1p3 adlseq_v1p3.xsd
                       http://www.imsglobal.org/xsd/imsss imsss_v1p0.xsd
                       http://ltsc.ieee.org/xsd/LOM lom.xsd">
   <metadata>
            <schema>ADL SCORM</schema>
            <schemaversion>CAM 1.3</schemaversion>
   </metadata>
   <organizations/>
   <resources/>
</manifest>
```

Näide 1: Tühi imsmanifest.xml

<metadata> element kirjeldab SCORM versiooni.

<organizations> element jagab sisu peatükkidesse.

```
   <organizations default="ORG-001">
     <!-- The default activity tree. -->
     <organization identifier="ORG-001" structure="hierarchical">
       <title>SCORM spetsifikatsioon</title>
       <item identifier="ACT-001" identifierref="RES-001">
         <title>SCORMi ajalugu ja idee</title>
       </item>
       <item identifier="ACT-002" identifierref="RES-002">
         <title>SCORM paketi ülesehitus ja toimimine</title>
       </item>
```

```
  </organization>
</organizations>
```

Näide 2: <organizations> element.

Näide 2 defineerib käesoleva töö teise peatüki ja kaks esimest alapeatükki. Mõlemad alapeatükid viitavad ressurssidele.

<resources> element on nimekiri ressurssidest, mida üks või teine peatükk kasutab.

```
<resources>
  <resource id="RES-001" type="webcontent" adlcp:scormType="asset">
    <file href="example2.xml"/>
  </resource>
  <resource id="RES-002" type="webcontent" adlcp:scormType="asset">
    <file href="example3.xml"/>
  </resource>
</resources>
```

Näide 3: <resources> element

Iga ressursside kogum viitab ühele või mitemele failile ning lisaks on võimalik sellele lisada veel sõltuvusi. Näiteks näites 3 toodud RES-001 võib RES-002 sõltuvusse panna järgmise elemendiga: <dependency identifierref="RES-002"/>.
Kõik failid, millele <file> elemendid viitavad, peavad olema zip-failis olemas.

Manifest faili olemust ja sisupakendust kirjeldab põhjalikult *Content Aggregation Model* raamat.

2.2.2. Kommunikatsioon ja andmemudel

Üheks SCORM sisupaketi omaduseks on ka võime suhelda ennast võõrustava keskkonnaga (RTE[8]), olgu selleks siis mõni õpisüsteem või spetsiaalne SCORM pakettide esitaja. Suhtlus kahe osapoole vahel on defineeritud API[9] ja kindla andmemudeliga.
Uurime esmalt API-t.

RTE on kohustatud pakkuma sisupakile API ja sisupakk on kohustatud selle API ise üles otsima. Spetsifikatsioon näeb ette, et API tuleb panna muutujasse nimega API_1484_11. Muutja pannakse objektina DOM[10] puusse. SCORM sisupakk peab rekursiivselt läbi otsima kõik aknad, mis temaga seotud on, et API instans leida.

[8] Run-Time Environment
[9] Application Programming Interface
[10] Document Object Model vt. http://www.w3.org/DOM/

SCORM 2004 spetsifikatsioon defineerib suhtluskanali jaoks kaheksa funktsiooni, millest kaks on sessiooni funktsioonid: Initialize() ja Terminate(). Esimene neist algatab ja teine lõpetab sessiooni RTE-ga.
Kolm järgmist funktsiooni on diagnostikaga seotud: GetLastError(), GetErrorString(parameeter1), getDiagnostic(parameeter1). Peale igat kutsungit peab SCO kontrollima, kas tema viimati teostatud operatsioon õnnestus või mitte.
Viimased kolm funktsiooni on andmete kandmiseks RTE-st SCO-sse ja vastupidi: GetValue(parameeter1), SetValue(parameeter1, parameeter2), Commit(““). Nende funktsioonide kaudu saadab SCO RTE-le infot õppija tegemiste kohta.

SCORM 2004 defineerib andmemudeli, mida RTE peab salvestama.
Mõned näited:

- *cmi.completion_status*
  Hoiab väärtust selle kohta, kas õppija on SCO läbinud või mitte.
- *cmi.session_time*
  Ütleb palju aega on õppija ühe SCO peale aega kulutanud.
- *cmi.total_time*
  Summa cmi.session_time aegadest.

API ja andmevahetus on üldjuhul teostatud JavaScript skriptimiskeele abil.

2.3. SCORMile vastavad rakendused

SCORM 1.2 sertifitseeritud rakendusi on 137.
Viimase, SCORM 2004 3rd Edition, spetsifikatsioonile vastavaid ja sertifitseeritud rakendusi on ADL andmetel kõigest 25, mille hulgas vabavaralisi ja avatud lähtekoodiga süsteeme pole.

Kuigi enamik populaarseid õpikeskkondi pole SCORM sertifitseeritud, suudavad nad siiski maha mängida SCORM spetsifikatsioonile vastava sisupaketi. Enamik SCORM mängijaid õpikeskkondades vastavad SCORM 1.2 spetsifikatsioonile ning toetavad mõningaid SCORM 2004 elemente. [18]

3. Veebiteenused

Mõistet veebiteenus on raske defineerida. Kõigil, kes veebiteenustega kokku on puutunud, on nende olemusest mingisugunegi ettekujutus olemas, kuid sellest hoolimata on veebiteenuse mõistet raske kõigile meelepäraselt defineerida. Üheks võimalikuks definitsiooniks oleks:

Veebiteenus on tarkvara või süsteem, mille eesmärk on toetada interoperaablust, masin-masin suhtluses arvutivõrku kasutades. Selle liides on kirjeldatud masin-töödeldavas formaadis. Teised süsteemid suhtlevad veebiteenusega viisil, mis on ette kirjutatud, kasutades SOAP[11]-sõnumeid, mis omakorda transporditakse kasutades HTTP protokolli ja XML-i.[12]

Veebiteenuse ehitamise viise on laias laastus kaks: SOAP ja REST[13].

3.1. SOAP veebiteenused

Mõiste veebiteenus võimaldab standardiseeritud moodusel integreerida omavahel erinevaid veebipõhiseid programme. Tavaliselt on veebiteenuse eesmärgiks kirjeldada mingi värav, mille kaudu andmeid vahetada. Võrreldes tavalise veebilehitsemisega ei paku veebiteenused tavakasutajale mingit graafilist liidest, vaid need on mõeldud teistele programmidele tarbimiseks.

Komponendid ja tehnoloogiad, mida pea kõik SOAP veebiteenused kasutavad on järgmised:

- XML – *Extensible Markup Language*
- SOAP – *Simple Object Access Protocol*
- WSDL – *Web Services Description Language*
- UDDI. – *Universal Description, Discovery and Integration*

Järgnevalt igast ühest täpsemalt.

3.1.1. XML

XML, Extensible Markup Language, on laiendatav märgistuskeel, mis kirjeldab, kuidas andmed XML dokumendis välja näevad ning annavad osaliselt programmidele juhiseid, mida andmetega peale hakata.

---

[11] SOAP - Simple Object Access Protocol
[12] http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#defs
[13] REST - Representational State Transfer

XML on välja kasvanud SGML ISO standardist. [9]

3.1.2. SOAP

SOAP on XML-il põhinev lihtne sõnumite vahetamise protokoll, mis lubab programmidel infot vahetada.

```
POST /service HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPrice>
      <m:StockName>IBM</m:StockName>
    </m:GetStockPrice>
  </soap:Body>
</soap:Envelope>
```

Näide 4: SOAP päring.

3.1.3. WSDL

WSDL on veebiteenuste XML-põhine kirjeldamise standard. WSDL on XML formaat, mis kirjeldab võrguteenuseid - andmeid, mida mingi veebiteenus vastu võtab, milliseid sõnumeid aktsepteeritakse ning samuti seda, kuhu mingi veebiteenus installeeritud on.[16]

3.1.4. UDDI

UDDI on platvorm kirjeldamaks veebiteenuseid. UDDI kataloogi salvestatakse infot veebiteenuste kohta. UDDI kataloogi salvestatakse ka WSDL failidest saadud info ning kogu liiklus UDDI kataloogis käib SOAP vahendusel. [13]

3.2. REST veebiteenused

Mõistet REST - Representational State Transfer - kasutas esimesena Roy Fielding oma dissertatsioonis. Erinevalt SOAP-ist ei ole REST tegelikult standard, vaid süsteemi arhitektuuri stiil. REST-stiilis veebiteenuse saab ehitada kasutades HTTP, XHTML/XML, URI[14]. [14]

[14] URI - Uniform Resource Identifier – identifitseerivad ressursse Internetis.

Ressurss on REST stiilis kesksel kohal. Ressurss on midagi, millel on aadress (URI).
REST defineerib nelja HTTP meetodi kasutuse:

- GET – ressursi hankimine
- PUT – uue ressursi loomine
- POST – ressursi uuendamine
- DELETE – ressursi kustutamine

Seda nelikut võrreldakse tihti ka create, read, update ja delete (CRUD) operatsioonidega.
Üks REST-i tugevaid eeliseid on tema lihtsus ja see, et õppimisele kulub vähe aega.

3.3. Veebiteenused Internetis

Veebiteenuste hulgast Internetis valisin välja kaks. Esimene neist kasutab teenuste edasiandmiseks SOAP-i, teine on võtnud suuna REST arhitektuurile.

3.3.1. Amazon.com-i veebiteenused

Amazon.com on Interneti portaal, kust on võimalik osta raamatuid, tarkvara, dvd plaate, ehteid ja palju muud.
Amazon.com avaldab 10 erinevat veebiteenust:

1. Amazon E-Commerce Service
   Võimaldab hankida toote infot erinevate toodete kohta, mis on amazon andmebaasis.
2. Amazon Elastic Compute Cloud (Amazon EC2) - Limited Beta
   Amazoni virtuaalserveri teenus.
3. Amazon Flexible Payments Service (Amazon FPS) - Limited Beta
   Paindlike maksete süsteem, mis võimaldab liigutada raha kahe süsteemi või inimese vahel.
4. Amazon Mechanical Turk (Beta)
   Inimressursi turg.
5. Amazon Simple Storage Service (Amazon S3)
   Andmete hoidla, andmebaas internetis.
6. Amazon Simple Queue Service (Amazon SQS)
   Sõnumite vahetamise teenus, mis vahendab sõnumeid ilma kadudeta.

7. Alexa Site Thumbnail

   Teeb interneti lehekülje pisikeseks pildiks.
8. Alexa Top Sites

   Teenus koostab edetabeli populaarsematest veebilehtedest internetis.
9. Alexa Web Information Service

   Genereerib põhjaliku info veebilehe külastatavuse kohta.
10. Alexa Web Search

    Internetiotsingu teenus.

Siinkohal ei hakka neid erinevaid teenuseid väga põhjalikult lahkama. Olgu vaid öeldud, et need veebiteenused võimaldavad erinevatel e-poe omanikel oma poe amazoniga integreerida - näidata oma veebilehel sobivaid tooteid amazoni kaubamajast, kasutada paindlikku amazoni maksesüsteemi või vaadata oma e-lehekülje külastatavust. Amazon kasutab kõikide veebiteenuste puhul SOAP-i. Mõnda on võimalik kasutada ka REST stiilis. [7]

3.3.2. Yahoo! veebiteenused

Yahoo! pakub arendajatele veebiteenuseid, mis kasutavad REST arhitektuuri. Nagu enne REST peatükiski (vt peatükk 3.2) räägitud, spetsiaalselt konstrueeritud URL-id tagastavad tulemused XML formaadis.

Yahoo! pakub kokku kaheksat erinevat veebiteenust:

1. *Answers*

   Ühed küsivad, teised vastavad, kolmandad otsivad küsimuste ja vastuste seast.
2. *Local*

   Lokaliseeritud teenus, kus saab otsida teenuseid ja firmasid ühest linnast. Kahjuks piirdutakse seal ainult Ameerika osariikide ja linnadega.
3. *Mail*

   Võimaldab programmselt läheneda Yahoo! poolt pakutud e-postkastile.
4. *Maps*

   Yahoo! kaartide teenus, mis võimaldab otsingut aadressi järgi ning pakub juhiseid sõitmiseks punktist A punkti B.
5. *Search*

   Internetiotsingu teenus.
6. *Shopping*

Ostukeskus.

7. *Travel*

   Reisiteenus, kus inimesel on võimalik uurida reisi sihtkohti, planeerida reisi, osta lennupileteid, broneerida hotelle ja palju muud.

8. *Utilities*

   Kasulikud „vidinad“. Hetkel on selles kategoorias ainult õiget kellaaega ütlev teenus.

Kõiki neid teenuseid saab kasutada ka veebiteenuste vahenduseta. Näiteks http://answers.yahoo.com või http://search.yahoo.com.

Yahoo! pakub oma veebiteenustega kaasa ka põhjaliku dokumentatsiooni selle kohta, kuidas teenused töötavad ja kuidas neid kasutada. Lisaks on veel Yahoo! poolt programmeerimiskeelte jaoks valmis tehtud teeke ja näidisprogramme Perl, Python, PHP, Java, Javascript ja Flash. [8]

3.4. Veebiteenused Zope rakendustes

Zope on rakendusserver, mis on peamiselt programmeeritud Python programmeerimiskeeles. Sellega tuleb kaasa objekt-orienteeritud andmebaas, kuhu on võimalik salvestada andmeid, HTML malle, skripte ja palju muud.

Zope-s on igal objektil oma aadress. Näitena võib tuua aadresside haldamise programmi, (http://zope/AddressBook.) mis töötab Zope serveris. Kui nüüd kõik programmi sisestatud aadressid on objektid, siis on neil ka oma URL-id, näiteks http://zope/AddressBook/address1, http://zope/AddressBook/address2 jne.

Siit on näha, et see sobib ideaalselt REST arhitektuuri kontseptsiooga ja ressurss-orienteeritud arhitektuuriga. Kuna Zope-i on sisse ehitatud ka veebiserver, siis toetab Zope ka PUT, POST ja muid REST-ile vajalikke meetodeid.

Natuke keerulisem on lugu aga SOAP-i veebiteenustega. Zope-l puudub sisse ehitatud tugi SOAP päringute vastu võtmiseks või välja saatmiseks. Selleks, et SOAP-il põhinevaid veebiteenuseid Zope platvormil pakkuda, tuleb installeerida paar lisamoodulit nii Zope-le kui ka Pythonile. Viimasele tuleb installida juurde SOAPpy[15] teek mis võimaldab Python programmeerimiskeeles SOAP päringuid lugeda ja

[15] http://pywebsvcs.sourceforge.net/

koostada. Teiseks on vaja õpetada Zope-le SOAP päringuid - selleks on vaja installeerida Zope laiendusmoodus SoapServer[16]. Ning alles peale seda on Zope võimeline SOAP päringuid vastu võtma.

Veebiteenuste tarbimisega Zope-s probleeme pole. REST stiilis veebiteenuste kasutamiseks on Pythonis vajalikud võimalused olemas. SOAP päringute koostamiseks on vaja vaid installeerida ülal mainitud SOAPpy ja veebiteenuste tarbimine võib alata.

[16] http://trac.htk.tlu.ee/icamp/browser/SoapServer

4. SCORM-sisupakettide esitlusteenus Caldoz

Seni olid IVA kasutajatel suhteliselt piiratud võimalused õppematerjalide õpikeskkonda üles seadmiseks. Tavapärase faili üles laadimise kõrval on üheks variandiks veel koostada ka memosi ning luua wiki lehti. Kuigi wiki pakub mõningaid võimalusi materjali kujundamiseks, on need vahendid siiski napid. Loodud õppematerjalid on seotud ühe kursusega ja nende sealt välja toimetamine või teisel kursusel uuesti kasutamine on keeruline.

Siin tuleb appi SCORM spetsifikatsioon, mille kohaselt saab kasutaja vastavalt pakendatud õppematerjali mugavalt üles laadida ning mis õppurite jaoks avaneb kõigile ühte moodi. Seejuures ei pea õppur õppematerjalide nägemiseks muid programme kasutama.

Käesolev peatükk otsib vastuseid järgmistele küsimustele:

1. Kuidas oleks võimalik SCORM-pakette IVAs kasutada.
2. Millistele nõuetega peaks SCORM-pakettide esitaja IVAs vastama.
3. Kuidas on realiseeritud SCORM-pakettide esitaja Caldoz .
4. Kuidas Caldozt veel kasutada saaks.

4.1. Sisupakettide esitlusteenuse kasutamise stsenaariumid ja kasutajalood

Käesoleva peatüki esimene alapeatükk alustab SCORM-sisupakettide esitleja arendamist stsenaariumi kirjeldamisega, mis loob pildi ideaalsest SCORM-sisupakettide esitleja kasutamisest ning funktsioonidest, mis sellel olla võiksid. Järgmises alapeatükis kirjutan ma stsenaariumile tuginedes mõned kasutajalood, mis kirjeldavad süsteemi kasutamist ning seda, kuidas see sellele reageerib. Seejärel uurin, millistele nõutele peaks sisupakettide esitleja vastama ning viimases alapeatükis käsitlen programmeerimist ning testimist.

4.1.1. Stsenaarium

SCORM-sisupakettide esitlejasse Caldoz saab üles laadida SCORM-sisupakette. Seda saab teha otse Caldoze liidese abil või kasutades selleks muud tarkvara, mis suhtleb Caldozega veebiteenuse kaudu. Mõlemal juhul genereeritakse uue SCORM-sisupaketi üles laadimise puhul kaks võtit, millega sisupaketile hiljem juurde pääseb. Üks võti lubab sisupaketti ainult vaadata, teine võti on sisupaketi haldamiseks.

Sisupaketi haldamise all on selle kustutamise või uute võtmete tellimise funktsioon. Samuti on sisupaketi omanikul võimalik sisupaketi üksikuid lehti muuta, kuid seda piiratud ulatuses ja kasutajal võib ette tulla olukordi, kus on vaja käsitsi HTML koodi kirjutada. See aga sõltub juba konkreetsest sisupaketist.

SCORM-sisupaketti on võimalik sisestada (*embed)* suvalisele lehele. Selleks genereerib Caldoz paar rida HTML-i, mis tuleb asetada veebilehele, kuhu sisupaketti tahetakse saada. HTML-kood, mille Caldoz genereerib, sisaldab ka vaatamiseks mõeldud võtit. Et sisupakett maailma eest jälle ära peita, tuleb selle omanikul tellida uued võtmed.

IVAl on lisamoodul Caldoze kasutamiseks, mis suhtleb Caldozega veebiteenuse vahendusel. Lisamoodulit on võimalik aktiveerida ja deaktiveerida. Samuti on võimalik lisamoodulile anda klienti identifitseeriv nimi (*identification string*). Caldoz on võimeline selle nime abil kindlaks tegema, mis sisupakette mingi tarkvara üles laadinud on. Caldoze haldusliideses on võimalik veebiteenuse juurdepääs avada kõigile, olenemata kliendi ID-st. Samas on võimalik ka juurdepääsu piiramine ainult teatud klientidele. Kliente on võimalik siduda ka IP aadressiga.

IVA kasutaja saab IVAs üles laadida SCORM-sisupakette. IVA peab arvet selle üle, milliseid sisupakette kasutaja on üles laadinud. Samuti salvestab IVA ka juurdepääsuvõtmed kasutajaprofiili juurde, mis ei ole seotud ühegi kursusega. Selline süsteem teeb võimalikuks ühe sisupaketi ristkasutuse mitmelt kursuselt. Sisupakett, kui oskab, annab infot oma kasutuse kohta IVAle. Sisupaketi omanikul on võimalik vaadata sisupaketi kasutusstatistikat. Õppejõul on võimalik sisupaketti ka kodutööna esitada. Kui sisupakett peab arvet selle üle, kuidas õppijal üldkokkuvõttes läinud on, siis kantakse see tulemus õppejõu korraldusel üle IVA hinnetekoondtabelisse.

4.1.2. Kasutajalood

Õppejõu lugu 1: Õppejõud on kuulnud üht-teist SCORMist ning teab, et on olemas Caldoz ja otsustab seda katsetada. Õppejõud salvetab eXe-ga valmistatud õppematerjali SCORM formaati ja laeb selle Caldozesse. Vastuseks saab ta sealt kaks võtit ja paar rida HTML koodi. Õppejõud paigutab viimase oma kodulehele, kus seejärel kuvatakse sisupakett.

Õppejõu lugu 2: Õppejõud teeb eXe-ga valmis õppematerjali ning salvestab selle SCORM-formaadis. Edasi logib õppejõud sisse IVA-sse, valib sobiva kursuse ja

navigeerib raamaturiiulile. Raamaturiiulil pakutakse õppejõule muuhulgas ka valikut „Lae üles SCORM-sisupakett“. Õppejõud klõpsab sellel, talle avaneb vorm, kust saab sisupaketi üles laadida. Peale üles laadimist antakse õppejõule teada, et fail on edukalt üles laetud. Õppejõud leiab sisupaketi raamaturiiulilt.

Õppejõu lugu 3: Järgmisel semestril loeb õppejõud teist kursust, kuid tahab siiski üliõpilastele anda sama õppematerjali, mida ta eelmisel semestril oli kasutanud. Selleks logib õppejõud jällegi IVAsse, valib kursuse, navigeerib raamaturiiulile ning klõpsab „Lae üles SCORM-sisupakett“. Jällegi on seal vorm faili üles laadimiseks, kuid seekord on seal ka nimekiri varem üles laetud sisupakettidest. Õppejõud valib nimekirjast sobiva materjali ning see ilmub jällegi raamaturiiulil.

Õppejõu lugu 4: Õppejõud koostab õppematerjali ja lisab iga peatüki lõppu viis küsimust. Õppematerjali salvestab õppejõud SCORM-sisupaketiks ning laeb üles IVAsse. Kursuse lõpus saab õppejõud ülevaate õppijate käekäigust kursusel.

Tugiisiku lugu 1: E-õppe tugiisik aitab õppejõul IVAs e-kursuste ette valmistada. Koostöös õppejõuga töötatakse välja mõned SCORM spetsifikatsioonile vastavalt pakendatud õppematerjalid ning tugiisik laeb need IVAsse üles. Peale õppematerjalide üles laadimist delegeerib tugiisik õppematerjalide kontrolli õppejõule.

Halduri lugu 1: Süsteemi administraator otsustab installeerida IVAle SCORM-sisupakettide toe. Selleks laeb ta esmalt alla Caldoze ja installeerib selle. Samuti määrab administraator ära, et Caldoze veebiteenust tohib kasutada ainult teatud klientide kaudu ja teatud IP aadressidelt. Administraator installeerib ka IVA lisamooduli ja aktiveerib selle.

4.2. Nõuete analüüs

SCORM-sisupakettide kasutusest parema ülevaate saamiseks viisin läbi intervjuu kahe Tallinna Ülikoolis töötava e-õppe spetsialistiga (vt. Lisa 1). Intervjuust selgus, et kuna SCORM on suhteliselt uus teema, mis on viimaste aastate sotsiaalse tarkvara buumi tõttu natuke ka tahaplaanile jäänud, siis erilisi nõudeid SCORM-sisupakettide

esitlejale just nagu polekski. Oluliseks peeti seda, et SCORM-sisupaketi puhul oleks õppematerjal üks tervik. Siis ei pea õppija alla laadima kümneid ja kümneid faile, et tarkus kätte saada. Ühe positiivse küljena, mis SCORM-iga seoses välja toodi, oli see, et SCORM laseb õppematerjale erinevate tegevustega rikastada - säilitades seejuures oma terviklikkuse - et õppijal oleks huvitavam materjaliga töötada.
Konkreetselt IVAst ja SCORMist rääkides oli esialgu siiski oluline vaid see, et õppematerjal korralikult lahti tuleks. Veel sooviti, et pisemaid kirjavigu saaks otse IVAs parandada, ilma, et peaks uue SCORM sisupaketi üles laadima.
Läbivalt oli aga olulisim punkt juurdepääsu kontroll. Mõlemad e-õppe spetsialistid kinnistasid, et juurdepääsu piirangute seadmise võimalus peab olema ning mida paindlikum see on, seda parem.

Järgmiseks toon välja nimekirja nõuetest, millele SCORM-sisupakettide esitaja vastama peab. Nimekiri on koostatud stsenaariumi, kasutajalugude ning intervjuu põhjal. Nõuded SCORM-sisupakettide esitajale olid järgmised:

- Kasutaja(d) peavad saama üles laetud SCORM-sisupakette alla laadida.
- Sisupakettide esitaja peab olema kasutatav ka ilma IVAta ning teiste keskkondade poolt.
- SCORM spetsifikatsiooni tugi vähemalt sisu esitamise osas.
- Sisupakettide esitaja peab kasutaja soovil kiivalt kaitsma üles laetud materjale.
- SCORM-sisupakettide esitajas peab olema juurdepääsu piirangute seadmise võimalus vähema sisu üles laadimise osas.

Olles välja segitanud kasutajate vajadused, koostanud ühe võimaliku stsenaariumi ning kirjutanud kasutajalood, püüan järgnevalt realiseerida Caldozt vastavalt kirjutatud kasutajalugudele.

4.3. SCORM-sisupakettide esitlusteenuse Caldoz realiseerimine

Kuna IVA on programmeeritud Zope rakendusserveri tootena, siis Caldoze platvormiks sai samuti valitud Zope, ning programmeerimiskeeleks Python. Esimese sammuna sai loodud tarkvara, mis lasi üles laadida Zip-faile ning salvestas need andmebaasi.
Faili üles laadimisel tehakse esmalt kindlaks, kas tegu ikka on zip-failiga. Järgmiseks püütakse zip-failist leida manifest faili (vt. Peatükk 2.2.1). Kui see fail on olemas, siis

võib üsna kindel olla, et tegemist võib olla SCORM-sisupaketiga. Esialgu rohkem kontrolle ei rakendata, kuid võiks, või isegi peaks kontrollima, kas tegemist ei ole libafailiga ja kas tegelikult ei laeta üles rämpsu.
Kui on kindlaks tehtud, et üles laetud fail on SCORM-sisupakett, genereeritakse kaks võtit – avalik ja privaatne. Üks võti tagab täieliku kontrolli üles laetud ressursi üle, mis tähendab seda, et selle võtme omanik saab sisupaketi serverist ära kustutada või tellida uued võtmed. Teine võti annab ainult vaatamise õiguse ning seda võtit võib vaatamiseks jagada teiste kasutajatega. Need võtmed kuvatakse kasutajale salvestamiseks.
Järgmiseks sammuks on sisupaketi lahtivõtmine. SCORM-sisupakett on zip-fail, mille lahtivõtmine käib lihtsalt. Kõik failid salvestatakse eraldi objektidena andmebaasi. Kuna failinimed võivad sisaldada ka erisümboleid, siis ei saa kasutada faili enda nime objekti id andmiseks, vaid tuleb genereerida suvaline nimi. Faili õige nimi ja asukoht paketis salvestatakse samuti genereeritud objekti juurde, et pärast oleks võimalik sisse tulevatele päringutele õige sisu serveerida.
Edasi on vaja sisse lugeda imsmanifest.xml fail, mis kirjeldab sisupaketi erinevaid peatükke ja seda, millist faili tuleks kasutajale näidata ühe või teise peatüki juures. Imsmanifest.xml faili näidis on peatükis 2.2.1. Imsmanifest.xml failist genereeritakse iga peatüki (<organization> tag) kohta üks objekt, mis on Zope andmebaasis oma olemuselt kaust, mis võib sisaldada alampeatükke. Iga <organization> tag all võib olla mitu alampeatükki (<item> tag). Need salvestatakse andmebaasi puu kujuliselt nii, nagu nad manifest failis kirjas on. Nii säilitatakse sisupaketi struktuur ka andmebaasis, hoolimata sellest, kas sisupaketi failid on salvestatud ühele tasandile.

Iga peatüki juurde manifest failis kuulub ka terve hulk meta-andmeid, mis juhendavad sisupaketi esitluskeskkonda. Näiteks võib sisupakett öelda käivitaja-keskkonnale, et paketi sisukorda ei tohi kasutajale näidata ning sisupaketi mängija peab ära peitma enda edasi-tagasi nupud. Samuti võib sisupakett seada piiranguid, millal kasutaja järgmise peatüki juurde pääseb.
Neid meta-andmeid SCORM-sisupakettide esitaja aga esialgu ei toeta. Meta-andmetest võetakse esialgu ainult pealkiri.

Kui sisupakett on üles laetud, lahti võetud, vastavad objektid genereeritud, failid andmebaasi salvestatud, peatüki-objektid genereeritud. Nüüd on vaja programmeerida kasutajaliides, mis serveeriks kasutajale sisupaketi.
Selleks tuli/tuleb esmalt programmeerida meetod, mis genereeriks sisukorra. Kuna peatükid on juba eelnevalt andmebaasis objektidena struktureeritud, siis see ülesanne ei valmista probleeme.

Probleemid kerkivad esile siis, kui püüda sisukorda kasutada. Selles viitavad lingid peatüki objektidele, näiteks chapter342/chapter234. Peatüki objektid viitavad mingile failile, mida on vaja näidata. Peatüki objekt teeb sirvijale/lehitsejale ümbersuunamise õigele failile, näiteks sissejuhatus.html või lõppsõna.html. Selliste id-dega objekte aga andmebaasis ei eksisteeri! Eelpool mainitud andmebaasi piirangust tulenevalt on failidele sissejuhatus.html ja lõppsõna.html antud hoopis id-d file1232 ja file4325.
Lahenduse pakub Zope platvorm. Nimelt, Zope platvorm laseb programmeerijal kirjutada adapteri, mis sekkub sel hetkel, kui Zope server hakkab otsima mingi id-ga objekti. Käesolev lahendus kasutab seda adapteri programmeerimise võimalust nii, et esmalt lastakse Zope-l otsida oma vahenditega kasutaja poolt soovitud objekti. Kui Zope ei suuda soovitud objekti leida, asub tegutsema Caldoze objekti avalikustamise adapter. Valmistatud adapter käib läbi kõik objektid, mis esindavad sisupaketist tulnud faile ja võrdleb nende õigeid nimesid soovitud failinimega. Kui objekt on leitud, tagastatakse see kasutajale. Seejuures kontrollitakse enne igat objekti otsimise protseduuri, kas kasutajal on õigust antud sisupaketti üldse vaadata.

4.3.1. Veebiteenus

Käesoleva töö eesmärgiks on realiseerida ka lihtne ja lihtsalt kasutatav veebiteenus. Nagu peatükist 3 selgub, on Zope platvormil kõige lihtsamini realiseeritavad REST-stiilis veebiteenused (vt lk. 20). Kuna REST-stiilis veebiteenuseid on lihtne kasutada ka teistel platvormidel (Java, Javascript, PHP, jne) valisingi Caldoz veebiteenuse valmistamiseks REST-i. SOAP veebiteenustega käib komplektis WSDL, mis kirjeldab veebiteenuse olemuse, sisend- ja väljundparameetrid. REST-i puhul ei ole veel sellist kokkulepitud standardit olemas ja sellist faili ei saa teha. Seega, antud veebiteenuse puhul, peab selle tegemine toimuma kahes osas:

1. Veebiteenuse dokumentatsiooni kirjutamine

2. Veebiteenuse enda realiseerimine

Veebiteenuse võib kokku kirjutada ka ilma dokumentatsioonita, kuid siis ei oskaks keegi peale autori aimatagi, et selline veebiteenus eksisteerib ning ka selle kasutamine oleks äärmiselt keeruline.

Järgnevalt loetelu veebiteenuse meetoditest peamooduli tasemel:

1) addNewPackage

   süntaks: addNewPackage(file, clientid)

   - file – base64 kodeeritud fail
   - clientid – veebiteenust kasutava kliendi ID. See on lisatud siia tulevikuperspektiiviga, et kunagi, kui tekib vajadus, saaks teenuse kasutamist lubada ainult teatud klientidele.

   Meetod võimaldab laadida süsteemi üles uue sisupaketi. Mõlemad parameetrid on kohustuslikud.

   Klient saab vastuseks lühikese xml-I, kus on kirjas, kas paketi üles laadimine õnnestus ning kui õnnestus siis tagastatakse kliendile ka võtmed, paketi id ja sisupaketi pealkiri.

   Näidis vastus:

```
<response>
  <id>1</id>
  <status>0</status>
  <statusText>Success</statusText>
  <name>Title of the package</name>
  <public>f928n4f</public>
  <private>2948hn4</private>
</response>
```

2) startSession

   süntaks: startSession(package_id, key, username)

   - package_id – Sisupaketi id
   - key – Avalik või privaatne võti.
   - username – kasutajanimi, kes sessiooni alustab. Mittekohustuslik parameeter.

   Klient alustab uut sessiooni, mis on seotud sisupaketiga. Võtme mittesobimisel tagastatakse kliendile vastus staatusega 1.

   Uue sessiooni algatamisel seatakse võimalusel kasutaja brauserisse ka küpsis (*cookie*), mida kasutatakse edaspidi sisupaketi tasemel kasutaja õiguste kontrollimiseks.

   Näide:

```
<response>
  <status>0</status>
  <session_id>9248fn2394f</session_id>
</response>
```

3) getTOC

süntaks: getTOC(session_id)

- session_id – sessiooni id mis on saadud startSession meetodiga.

Paketi sisukord. Vastus HTML kujul. Vale sessiooni id korral kliendile vastust ei saadeta.

4) getNavigation

süntaks: getNavigation(session_id)

- session_id – sessiooni id mis on saadud startSession meetodiga.

Meetod tagastab paketi navigeerimisnupud (eelmine, järgmine, välju). Selle meetodi väljund sõltub sellest, kas pakett lubab või ei luba käivitaja-süsteemil oma navigeerimisnuppe kuvada. Vastus HTML kujul.

5) destroy

süntaks: destroy(session_id)

- session_id – sessiooni id mis on saadud startSession meetodiga.

Kustutab sisupaketi serverist kui sessiooni on alustatud privaatse võtmega. Vastuseks on staatuse kood 0 või ebaõnnestumise korral 1. Vastus XML kujul.

6) getNewTokens

süntaks: getNewTokens(session_id)

- session_id – sessiooni id mis on saadud startSession meetodiga.

Genereerib sisupaketile uued juurdepääsuvõtmed kui sessiooni on alustatud privaatse võtmega.

Vastuseks on staatuse kood 0 või ebaõnnestumise korral 1 ja uued võtmed. Vastus XML kujul.

Näidis:

```
<response>
  <status>0</status>
  <public>f9as2228sdnf4f</public>
  <private>2dsf948hnda4</private>
</response>
```

7) endSession

süntaks: endSession(session_id)

- session_id – sessiooni id mis on saadud startSession meetodiga.

Lõpetab sessiooni. Vastus puudub.

Loetelu veebiteenuse meetoditest sisupaketi tasemel:

1) getTOC

   Dubleeritud. Vt. Peamooduli meetodid

2) getPrevChapter

   Tagastab eelmise peatüki aadressi.

3) getNextChapter

   Tagastab järgmise peatüki aadressi.

4) getLMSValues

   Tagastab sisupaketi poolt salvestatud info.

5) setLMSValues

   Salvestab sisupaketi poolt saadetud info.

6) getStartingPage

   Tagastab sisupaketi esilehe aadressi.

7) getTitles

   Tagastab sisupaketi peatükkide pealkirjad ja aadressid. Tavaliselt on ühel sisupaketil üks peatükk.

   Näide:

```
<response>
  <chapter>
        <title>Example SCORM package</title>
        <url>http://localhost/Caldoz/package_23/chapter33</url>
  </chapter>
</response>
```

Sisupaketi taseme meetodite kasutamiseks on vajalik teada paketi id-d. Samuti on vajalik, et päringuga saadetakse ka küpsis nimega session_key, mis on session_id, mis on saadud startSession meetodi tulemusena.

Peamooduli meetodite aadress on Caldoze aadress pluss meetodi nimi.

Näide: http://localhost/Caldoz/startSession

Sisupaketi taseme meetodite kasutamiseks tuleb Caldoze aadressile liita veel sisupaketi id. Näide: http://localhost/Caldoz/package_23/getTitles

4.4. Caldoze klient IVA laiendusena

IVAl on olemas lisamoodulite arhitektuur, mis võimaldab ehitada IVAle Caldoze klient nii, et IVA lähtekoodi väga palju muutma ei pea.

Esmalt loon Zope-le uue rakenduse – *CaldozClient* (edaspidi CC). Zope rakendusserveri taaskäivitamisel annab CC endast automaatselt IVAle teada. CC lisab

ka adapteri, mida on IVA kontekstis võimalik adapteerida. IVA halduril on seejärel võimalik IVA halduse sektsioonis vastav lisamoodul aktiveerida. Selle peale otsib IVA üles vastava adapteri ning käivitab _install meetodi. Mainida tuleb veel ka, et CC vastav adapter vastab IIVAPlugin liidese (i.k. *Interface*). IIVAPlugin liides defineerib neli meetodit: *_install*, *_uninstall*, *isInstalled* ja *isVisible*.
IVA käivitab neid meetodeid vastavalt sellele, kas tahetakse lisamoodul aktiveerida või deaktiveerida. *isInstalled* meetod ütleb, kas lisamoodul on aktiveeritud või mitte ning *isVisible* meetod ütleb, kas lisamoodul on kasutajatele nähtav. Nimelt, IVAs on võimalik lisamoodul ka nii aktiveerida, et see ei ole kasutajatele nähtav.

Peale seda, kui IVA haldur on CC lisamooduli aktiveerinud, tuleb see seadistada. Selleks tuleb CC-le öelda Caldoze aadress ning märkida linnukesega, kas CC poolt pakutavad funktsionaalsused on kasutajatele nähtavad. CC-le tuleb anda ka clientid ehk kuidas CC ennast Caldozele tutvustab (vt. Lk 27 addNewPackage meetodi kommentaari).

Seejärel on IVA kasutajal võimalus üles laadida SCORM-pakette. Link selleks asub veebilaual „Lisa fail“ all. SCORM-paketi üles laadimisel võetakse ühendust Caldoz serveriga ning sisupakett saadetakse sinna. Kui kõik õnnestub, antakse kasutajale teada, et operatsioon oli edukas. IVA salvestab samal ajal vastuse kasutaja profiili juurde ning peab arvet kasutaja poolt üles laetud sisupakettide kohta. Samuti peab IVA meeles ka võtmeid, millega sisupaketi lahti saab. Seejärel leiab kasutaja viite üles laetud SCORM-sisupaketile veebilaualt. Sellel klõpsates võtab CC kõigepealt ühendust Caldoz serveriga ja alustab uut sessiooni. Saanud sealt sessiooni võtme, saadab CC kasutaja brauseri Caldoz serverisse küpsist saama. See on vajalik selleks, et kui sisupakett pöördub otse Caldoz serveri poole, siis kasutaja saaks sealt vajalikud failid kätte. Isegi kui CC sätiks kasutaja brauserisse küpsise, võib tekkida probleeme sellega, et Caldoz võib olla installeeritud hoopis teise aadressi peale ja siis ei ole brauseril enam õigust CC poolt sätitud küpsist lugeda.
Probleem, mis siin tekib, on sarnane probleemile, mis on kirjeldatud Caldozt käsitlevas peatükis (vt. Lk 26). Nimelt faile, mida sisupakett tahab, ei ole IVA andmebaasis olemas. CC-le tuleb samuti programmeerida adapter, mis sekkub siis, kui vastavat objekti IVA juurest ei leita. Adapter võtab ühendust Caldoz serveriga ja edastab tulnud päringu sinna, andes kaasa ka sessiooni id.

Selline skeem töötab hästi, kuni andmed, mida IVA CC ja Caldoz omavahel vahetavad on väikesed. Kui aga andmed lähevad suuremaks, muutub ka süsteem aeglasemaks, sest üle võrgu andmete liigutamine võtab rohkem aega. Määravaks saab see, kui kiire on võrk IVA ja Caldoz serveri vahel. Samuti muutub IVA seisukohalt oluliseks ka see, et Caldoz reageeriks päringutele kiiresti. Vastasel korral võib tekkida situatsioon, kus terve Zope server tegeleb sellega, et aktiivselt Caldoz serverist ressursse hankida ning IVA ei saa inimesed sel juhul üldse kasutada.

4.5. Caldoze kasutamine muudest rakendustest

Sarnaselt IVA Caldoze kliendile on võimalik Caldoze klient ehitada ka muudele rakendustele. Piiranguid Caldoze kasutamisele muudest rakendustest ei seata. Kuna aga sisupakett soovib suhelda serveriga JavaScript abil, siis ei saa sisu esitaja olla mujal, kui Caldoz serveris, muidu ei ole JavaScript-il lubatud serveriga ühendust võtta. Järgnevalt toon näite esimese õppejõu kasutajaloo varal (vt lk 23). Kui õppejõu koduleht on näiteks TLÜ serveris, siis JavaScriptil, mis sel lehel asub, ei ole õigust Caldoz serveri poole pöörduda. Appi tuleb siin *iframe,* mille sisu võib tulla hoopis teisest serverist, näiteks Caldoze serverist. Iframe puuduseks on aga see, et kasutaja saab määrata ainult selle suurust, kõige muu üle, mis teisest serverist tuleb, veebilehe omanikul kontroll puudub.

5. Testimine

Lisaks programmeerimistööle on vaja programmi ka testida. Nagu ka peatükist 4.4 selgub, on sellise rakenduse puhul oluline ka jõudlus ja vastupidavus, vastasel juhul võib üks nõrk koht ühes süsteemis halvata ka teised, seda süsteemi kasutavad süsteemid.

5.1. Jõudlustestid

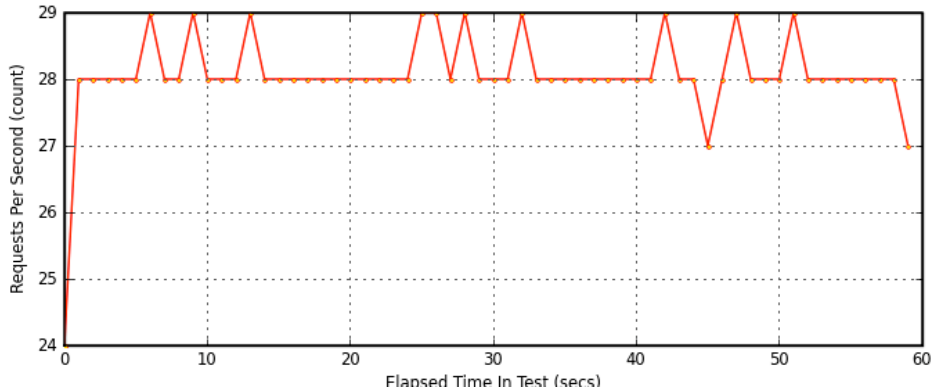
Jõudluse testimiseks tuleb esmalt valida tarkvara. Testimistarkvarad töötavad kõik suhteliselt ühte moodi. Tarkvarale tuleb ette anda aadressid, mida see siis korduvalt külastab ning sealjuures võtab see aega, kui kiiresti sisu kätte anti. Oluline on, et tarkvara loeks aadressid sisse mingist failist, lubaks simuleerida mitme kasutaja süsteemi kasutamist ning genereeriks lõpuks mingi mõistliku raporti. Peale mitme tarkvara katsetamist valisime testide läbiviimiseks Pylot[17]-i. Pylot on vabavaraline tööriist veebirakenduste ja -teenuste jõudluse testimiseks. Pylot on valmistatud Python programmeerimiskeeles ja töötab igas süsteemis, kus Pythongi. Pyloti sisendfailiks on XML fail, kus on kirjas aadressid, mida tarkvara külastama peab.

Arvuti, mille peal testid läbi viidi, oli tagasihoidlik - 64-bitise, 1.3Ghz taktsagedusel töötava protsessoriga, mälu maht 1.5Gb.
Testimiseks tegin mitu erinevat testi, mis peaksid kätte näitama süsteemi kitsaskohad. Olgu veel öeldud, et kõik testid kestsid 60 sekundit.

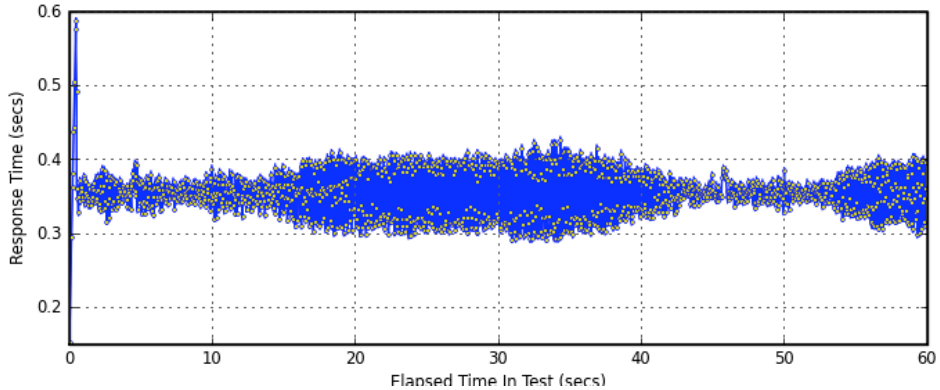
Test 1

Esmalt tegin lihtsa testi, kus 10 üheaegset kasutajat soovisid kõik saada ühe sisupaketi esilehte. Selle testi juures on oluline märkida, et Zope veebiserveri tegevus siin on minimaalne. Serveril tuli serveerida lehekülg ning genereerida sellele sisupaketi sisukord ning navigatsiooninupud.

[17] http://www.pylot.org/



Joonis 1. Töödeldud päringute arv sekundis.



Joonis 2. Reageerimisaeg sekundites.

Keskmine aeg, mille jooksul server vastuse andis on 0.36 sekundit (vt. joonis 2). Päringuid oli sekundis keskmiselt 28 (vt. joonis 2). Kokku jõuti teha 1684 päringut. Andmemaht 8,27Mb.

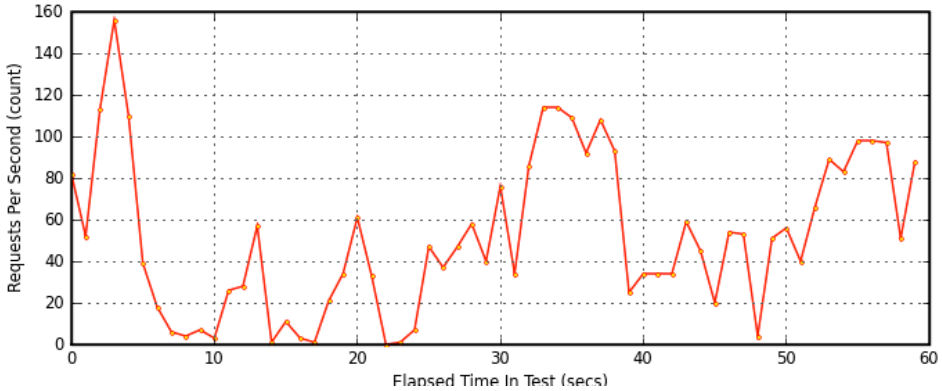
Kuigi reageerimisaeg oli suhteliselt hea, ei saa selle testi tulemusega päris rahule jääda. Lihtsa lehe genereerimine võiks võtta oluliselt vähem aega, kuid tulemus ise viitab sellele, et sisukorra genereerimine on programmeeritud ebaefektiivselt.

Test 2

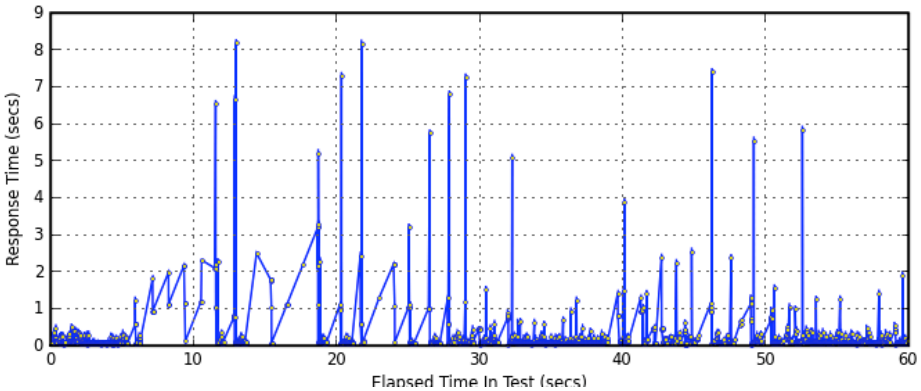
Järgmise testi koostasin nii, et laadisin testsüsteemi 6 sisupaketti ning käisin need kõik pisteliselt läbi. Tulemuseks sain veebiserveri logisse kõik päringud, mis brauser serverile esitas. Selleks, et päringud Pylot-ile sobivasse XML formaati saada, kirjutasin Pythoni skripti, mis selle töö ära tegi.

Selles testis on märkimist väärt see, et siin on sees ka need päringud, mis sisupakett saatis serverile, et see andmeid salvestaks. Hoolimata sellest, et Caldoz ei oska veel sisupaketist tulevate andmetega midagi teha, on salvestatud meetod Caldozes siiski

olemas. Andmeid ei salvestata küll andmebaasi, kuid andmebaasile antakse märku, et andmetes on toimunud muudatus, mida on vaja salvestada. Simulatsiooni mõttes on hea saada ülevaade sellest, kuidas andmete salvestamine mõjutab jõudlustesti tulemust.



Joonis 3. Töödeldud päringute arv sekundis.



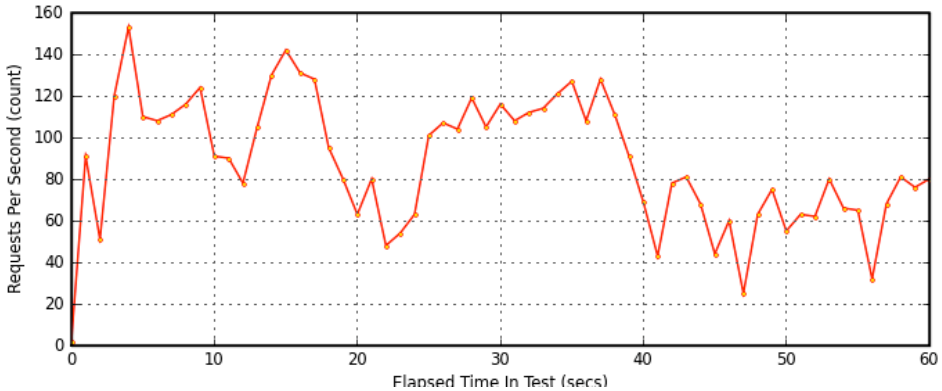
Joonis 4. Reageerimisaeg sekundites.

Kokku sooritati 3108 päringut. Keskmine reageerimisaeg oli 0.19 sekundit (vt. joonis 4) ja päringuid töödeldi keskmise kiirusega 51.80 sekundis (vt. joonis 4). Andmete maht on siin 27.4Mb.

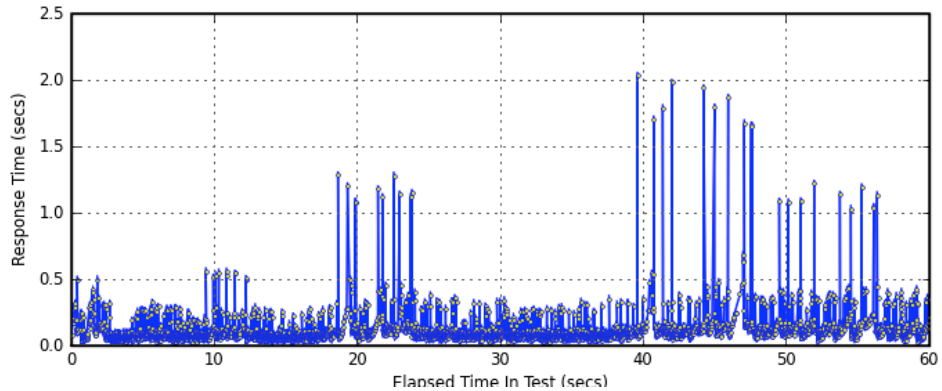
Selle testi tulemused on oluliselt paremad, kui esimesel testil. Esimese testiga võrreldes suudeti serverile esitada ligi 100% rohkem päringuid, sama kehtib ka päringute arvu kohta. Siiski tõuseb reageerimisaeg, ilmselt siis salvestamise hetkel, üle 8 sekundi. See on aga ilmselgelt liiga pikk ja tõenäoliselt ei taha ükski kasutaja nii kaua oodata. Reageerimisaja graafikult on ka näha, kuidas serveri suutlikkus päringuid töödelda muutub pea olematuks.

Test 3

See test on koostatud sarnaselt test 2-le, kuid välja on jäetud andmeid salvestavad päringud. Seega oli serveri põhiline funktsioon serveerida sisupaketist tulevaid ressursse – HTML, Flash ja muud meediafailid.



Joonis 5. Töödeldud päringute arv sekundis.



Joonis 6. Reageerimisaeg sekundites.

Kokku sooritati 5370 päringut. Keskmine reageerimiseaeg oli 0.11 sekundit (vt. joonis 6) ja päringuid töödeldi keskmise kiirusega 88.03 sekundis (vt. joonis 5). Andmete maht on 47.2Mb. Selle testi tulemustega võib igati rahule jääda. Päringute arv, mis serverile suudeti esitada, on kasvanud. Keskmine reageerimisaeg, test 2-ga võrreldes, on vähenenud 0.08 sekundit. Kuigi testi lõpu poole reageerimisaeg kohati kasvab kuni kahe sekundini, on see siiski kuni neli korda väikesem, kui test 2-s.

Siit võib järeldada, et sisu serveerimiseks sobib platvorm igati. Võttes arvesse ka testimises kasutatud riistvara, võib arvata, et korraliku serveri peal jooksutades, on tegemist üsna töökindla ja kiire süsteemiga. Kitsaskohad olid andmete salvestamine ja sisukorra genereerimine. Andmete salvestamise osas kahjuks midagi kiiremaks muuta ei saa, pigem läheb süsteem reaalseid andmeid töödeldes aeglasemaks. Siit võib

järeldada, et Zope andmebaas ei ole just kõige sobivam koht, andmete salvestamiseks, kuna see tirib alla terve süsteemi kiiruse. Alternatiivina saab siin andmete salvestamiseks kasutada mõnda relatsioonilist andmebaasi, mille andmete salvestamiskiirus on suurem. Selle realiseerimine ja testimine aga jäävad selle töö skoobist välja.

Teiseks testides ilmnenud probleemiks, oli see, et sisukorra genereerimine ei ole eriti nobe. Lahendus on siin lihtne. Algoritm, mida sisukorra genereerimiseks kasutatakse, tuleb välja vahetada efektiivsema vastu. Võimalusel tuleks kasutada ka puhverdamist – sisukord ju ei muutu, kui selle ühel korral välja arvutab, võib tulemuse salvestada. Kõikvõimalike andmete puhverdamine ja valmis genereerimine annab kindlasti olulise võidu.

Kokkuvõte

Käesoleva bakalaureusetöö eesmärgiks oli uurida ja testida praktikas võimalusi IVA õpikeskkonnale SCORM-sisupakettide esitlusteenuse loomiseks. Tulenevalt püstitatud eesmärgist viidi läbi analüüs digitaalsete õppematerjalidega seotud standarditest, uuriti lähemalt SCORM spetsifikatsiooni olemust ning e-õppe keskkondi SCORM kasutamise vaatenurgast. Läbi viidud uurimuse tulemustest järeldati, et SCORM spetsifikatsioon on perspektiivikas, kuigi veebiteenusena toimivaid SCORM sisupakettide esitajaid hetkel maailmas autori teadmisel ei eksisteeri. Arvestades järgmise põlvkonna e-õppe hajutatud arhitektuuriga ja veebiteenustel põhinevaid lahendusi ning tulenevalt läbi viidud analüüsi tulemustest jõudis töö autor järeldusele, et koostalitluse tagamiseks erinevate e-õppekeskkondadega oleks kõige targem luua SCORM-mängija eraldiseisva REST-tüüpi veebiteenusena. Bakalaureusetöö raames valmis SCORM-sisupakettide esitlusvahendi Caldoz prototüüp, millega viidi läbi jõudlustestid ja koguti tagasisidet kasutatavuse kohta ka IVA kasutajatelt. Testimine andis positiivse tulemuse ja julgustas autorit Caldoz veebiteenust edasi arendama. Eelkõige tuleks arendustöö järgmises faasis saavutada täielik andmevahetus Caldozi ja IVA (ka teiste e-õppekeskkondade) vahel – SCORM-sisupakett peaks oskama IVA-le edasi anda enda kasutamisstatistikat õpilaste poolt. Samuti ootavad IVA kasutajad käepärast SCORM-sisupakettide redaktorit, mis võimaldaks vajadusel kiiresti SCORM-sisupaketis teksti- ja kujundusvigu parandada.

Summary: SCORM-player for e-learning environment IVA

The aim of this B.A. thesis was to investigate problems and solutions related to presentation of SCORM-compliant e-learning content packages with a dedicated Web Service that is interoperable with an open-source e-learning environment IVA. SCORM (Shareable Content Object Reference Model) has become the main de facto standard for distributing re-usable e-learning materials. This is why the lack of ability to handle SCORM-packages was seen as a strategic deficiency of IVA system.

The first part of this paper provides a comparative overview of existing SCORM-compliant products: both systems that have ability to export content as SCORM-packages and authoring tools that are used for creating or editing SCORM-packages. The author also provides an overview of Web Services technology and its uses in developing new Web applications that are based on Service-Oriented Architecture.

The second part of this paper focuses on the development of a prototype of a new Web Service called Caldoz for presenting SCORM-packages in IVA learning environment. The results of stress testing of Caldoz tool are presented and analysed, suggestions for improving Caldoz in the future are given.

Kasutatud kirjandus

1. Laanpere, M., Kippar, J., Põldoja, H. (2003) Kodumaine õpihaldussüsteem IVA: pedagoogiline ja tehniline kontseptsioon artikkel ajakirjas A&A nr 1, 2003
2. ADLNET (2007) Advanced Distributed Learning. Viimati vaadatud 04.11.2007: http://www.adlnet.org
3. Ostyn Consulting (2006) Brief History of SCORM. Viimati vaadatud 01.11.2007: http://www.ostyn.com/standards/docs/HistoryOfSCORM.htm
4. SCORM® 2004 3rd Edition Content Aggregation Model (CAM) Version 1.0
5. ILIAS (2007) User Manual for ILIAS. Viimati vaadatud: 04.11.2007: http://www.ilias.de/ios/docs-e.html
6. Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F. (2006) Extensible Markup Language (XML) 1.0. Viimati vaadatud 30.10.2007: http://www.w3.org/TR/2006/REC-xml-20060816/
7. Amazon (2007) Amazon Web Services. Viimati vaadatud 04.11.2007: http://aws.amazon.com/
8. Yahoo! (2007) About Applications that Use Yahoo! Web Services. Viimati vaadatud 04.11.2007: http://developer.yahoo.com/about/
9. Haas, H., Brown, A. (2004) Web Services Glossary. Viimati vaadatud 04.11.2007: http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/#defs
10. LeMill Project (2007) Lemill. Viimati vaadatud 04.11.2007: http://lemill.org/trac/wiki/LeMill
11. eXe Project (2007) eLearning XHTML editor. Viimati vaadatud 30.10.2007: http://exelearning.org/
12. Ogbuji, U. (2000) Using WSDL in SOAP applications. Viimati vaadatud 04.11.2007: http://www.ibm.com/developerworks/library/ws-soap/?dwzone=ws
13. OASIS (2004) Introduction to UDDI: Important Features and Functional Concepts. Viimati vaadatud 04.11.2007: http://uddi.xml.org/files/uddi-tech-wp.pdf

14. Fielding, R. T. (2000) Architectural Styles and the Design of Network-based Software Architectures. Viimati vaadatud 04.11.2007: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
15. AICC (2006) AICC FAQ. Viimati vaadatud 04.11.2007: http://aicc.org/pages/aicc_faq.htm
16. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S. (2001) Web Services Description Language (WSDL) 1.1. Viimati vaadatud 04.11.2007: http://www.w3.org/TR/wsdl
17. DCMI Usage Board (2007) DCMI Metadata Terms. Viimati vaadatud 04.11.2007: http://dublincore.org/documents/dcmi-terms/
18. RELOAD (2005) Practical Interoperability. Viimati vaadatud 04.11.2007: http://www.reload.ac.uk/interop.html
19. Goldberg, C. (2008) Pylot – Web Performance Tool. Viimati vaadatud 16.04.2008: http://www.pylot.org/

Lisa 1. Intervjuu

Caldoz arendamise vajadused ja ootused kasutaja vaatenurgast

1. Millised on Sinu varasemad kogemused SCORM-sisupakettide arendamisel ja kasutamisel?
2. Kuivõrd (ja miks) on üldse oluline kasutada SCORM-pakette e-õppes? Nüüd ja tulevikus?
3. Mil määral oled SCORM-sisupakette kasutanud IVAs? Aga muudes keskkondades?
4. Milliseks võiks kujuneda IVA SCORM-playeri kasutamisintensiivsus TLÜ õppejõudude seas?
5. Kuidas näeks Sinu arvates välja SCORM-sisupaketi kasutamise stsenaarium IVAs (õppejõu pilguga)?
6. Millised peaksid olema juurdepääsupiirangud SCORM-sisupakettidele?
7. Kui oluline on paketi ja IVA vaheline suhtlus õppejõu jaoks?
8. Kui oluline on õppejõu jaoks SCORM-playeri kasutamisvõimalus väljaspool IVA (eraldiseisva teenusena)