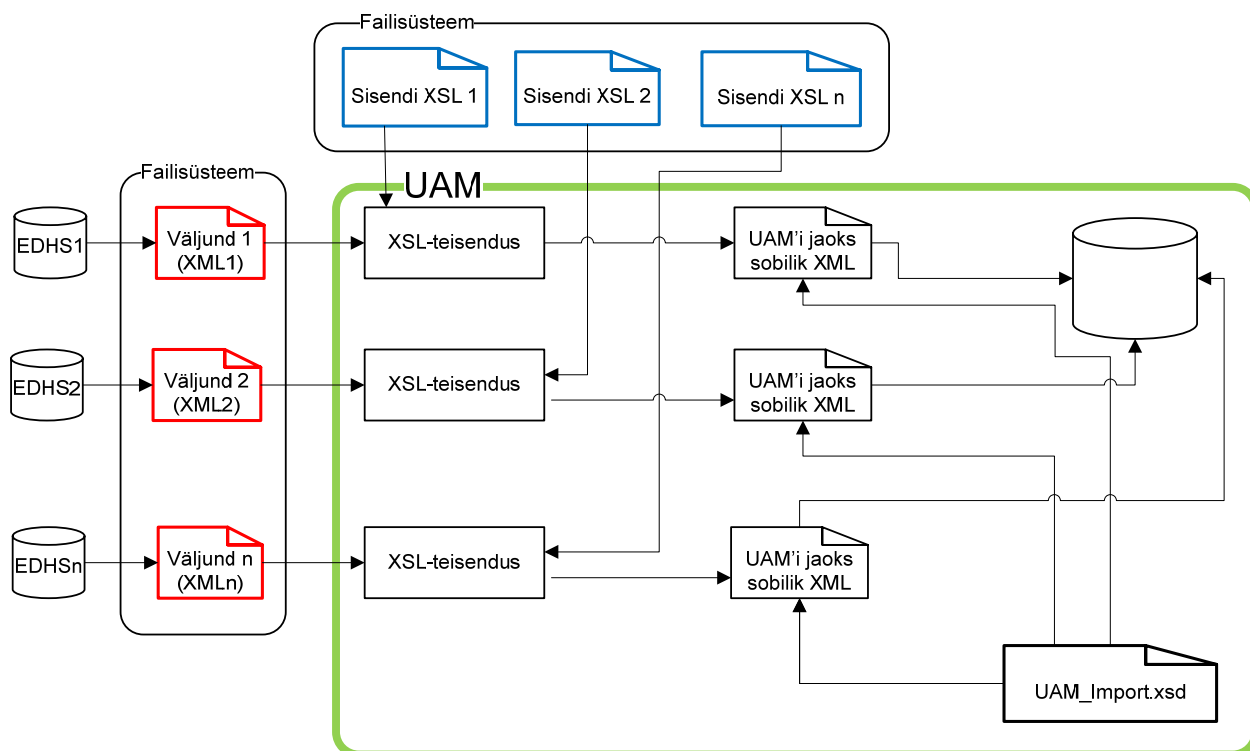


UAM'i sisendi seadistamine EDHS'i väljundi jaoks

Sisukord

Üldine kirjeldus.....	2
UAM'i sisendi ülesehitus	2
Teisenduse näide.....	4
Dokumendihaldussüsteemi väljundi XML	4
Teisenduseks kasutatava XSL 'i ülesehitus	4
XSL-i töötamise näide	6
XSL-i kohandamise näide	6
Olulised märkused	8
Hierarhia	8
Kuupäeva formaadid	9

Üldine kirjeldus



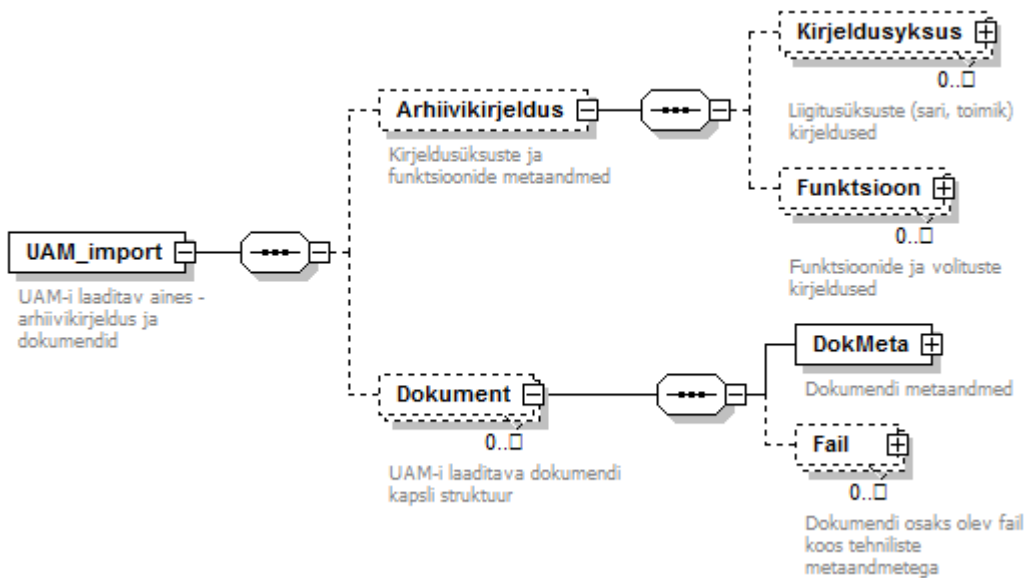
EDHS'i väljund tuleb teisendada UAM'i jaoks sobivale XML-kujule. UAM'i jaoks sobiv kuju on kirjeldatud XSD'ga UAM_Import.xsd

Teisendamiseks võib kasutada XSL'i, mille asukohta saab UAM'i seadistustes määrata. Niimoodi saab UAM'i laadida EDHS'i väljundit ilma muu vahepealse töötlusteta. Kasutada saab vaid XSL versiooni **1.0**.

UAM'i sisendi ülesehitus

XML'is on juurelemendi all 1 element „Arhiivikirjeldus“ ja loetelu elementidest „Dokument“, mis sisaldavad dokumentide metaandmeid ning faile. Elemendis „Arhiivikirjeldus“ on loetelu elementidest „Kirjeldusyksus“, mis sisaldavad sarjade või toimikute metaandmeid, ning loetelu elementidest „Funktsioon“, mis sisaldavad funktsioonide metaandmeid.

XSD üldine ülesehitus:



XML'i üldine ülesehitus:

```

<?xml version="1.0" encoding="utf-8"?>
<o:UAM_import xmlns:o="http://www.ra.ee/schemas/EDHS/import_v0.1">

  <o:Arhiivikirjeldus>
    <o:Kirjeldusyksus>
      .... Liigitusüksuse andmed
    </o:Kirjeldusyksus>
    ...
    <o:Kirjeldusyksus>
      .... Liigitusüksuse andmed
    </o:Kirjeldusyksus>

    <o:Funktsioon>
      .... Funktsiooni andmed
    </o:Funktsioon>
    ...
    <o:Funktsioon>
      .... Funktsiooni andmed
    </o:Funktsioon>
  </o:Arhiivikirjeldus>

  <o:Dokument>
    <o:DokMeta>
      .... Dokumendi andmed
    </o:DokMeta>
    <o:Fail>
      .... Dokumendi fail
    </o:Fail>
    ...
    <o:Fail>
      .... Dokumendi fail
    </o:Fail>
  </o:Dokument>
  ...
  <o:Dokument>
    <o:DokMeta>
      .... Dokumendi andmed
    </o:DokMeta>
  
```

```

    <o:Fail>
        .... Dokumendi fail
    </o:Fail>
    ...
    <o:Fail>
        .... Dokumendi fail
    </o:Fail>
</o:Dokument>

</o:UAM_import>

```

Teisenduse näide

Dokumendihaldussüsteemi väljundi XML

Näite aluseks on dokumendihaldustarkvara GoPro väljund, täpsemalt Riigikantseleis kasutatava GoPro paigalduse väljund.

Lähte XML on kujul, kus kõik üksused on kirjeldatud elemendis „document“ ning üksuse tüübi määrab ära atribuut „form“. „document“ elemendi all on loetelu „item“ elementidest, mis sisaldavad imporditava üksuse metaandmeid. Siinne näide on oluliselt lihtsustatud XML, millest on enamus „item“ elemente välja jäetud.

Lähte XML:

```

<?xml version="1.0" encoding="utf-8" ?>
<database>
<!-- 1. üksus -->
<document xmlns="http://www.lotus.com/dxl" version="7.0" maintenanceversion="3.0"
replicaid="C2256CC30041D1F8" form="frmCategory"
parent="4509A980C365984CC22575690045B85D" response="true">
<item name="Number"><text>1</text></item>
<item name="DateBegin"><datetime>20090201</datetime></item>
    <!-- Funktsiooni ülejäänud metaandmed -->
</document>
<!-- 2. üksus -->
<document xmlns="http://www.lotus.com/dxl" version="7.0" maintenanceversion="3.0"
replicaid="C2256CC30041D1F8" form="frmCategory"
parent="A8570E61EC7D249AC22575690045F18B" response="true">
<item name="Number"><text>1-2</text></item>
    <!-- Allfunktsiooni ülejäänud metaandmed -->
</document>
<!-- 3. üksus -->
<document form="InDoc" parent="D8041A7878C02EE2C225756D002A00B9" response="true">
    <!-- Dokumendi ülejäänud metaandmed -->
</document>
</database>

```

Teisenduseks kasutatava XSL 'i ülesehitus

Näite XSL on üles ehitatud nii, et alguses jagatakse töötlus hierarhia tasemete järgi. See tähendab, et

tuvastatakse funktsioonid, sarjad, toimikud ja dokumendid ning antakse need liigile vastavale template'le töötlemiseks. Toimub nn „viilutamine“.

```
<!-- "Viilutamine" ehk objektide tuvastamine liikide kaupa ning edasisse
töötlusesse suunamine. -->
<xsl:template match="i:database ">
  <o:UAM_import>
    <!-- Arhiivikirjeldus ehk kõik muu, mis ei ole dokumendid ja failid. -->
    <o:Arhiivikirjeldus>
      <!-- Funktsioon: tegemist on funktsiooniga, kui form="frmCategory" ja kui
alternatiivne viitekood ei sisalda. '-' -->
      <xsl:apply-templates select="i:document[@form='frmCategory' and
not(contains(i:item[@name='Number'],'-'))]" mode="funktsioon" />

      <!-- Sari: tegemist on sarjaga, kui form="frmCategory" ja kui
alternatiivses viitekoodis esineb '-'. -->
      <xsl:apply-templates select="i:document[@form='frmCategory' and
contains(i:item[@name='Number'],'-')]" mode="sari" />

      <!-- Toimik: tegemist on toimikuga, kui form='MainDoc'. -->
      <xsl:apply-templates select="i:document[@form='MainDoc']" mode="toimik"
/>
    </o:Arhiivikirjeldus>

    <!-- Dokumendid ja failid:
* Dokumendid on „document“ elemendid, millel atribuut „form“ on loetelust:
InDoc, Letter, frmOigusakt, Memo, frmDecree, frmLegalAct, frmResolution,
frmGovMeetingMinutes, frmGovernmentSession, frmDecision.
* Failid on vastavate dokumentide alamelemendid. -->
    <xsl:apply-templates
      select="i:document[@form='InDoc' or @form='Letter' or
@form='frmOigusakt' or @form='Memo' or @form='frmDecree' or
@form='frmAct' or @form='frmLegalAct' or @form='frmResolution' or
@form='frmGovMeetingMinutes' or @form='frmGovernmentSession' or
@form='frmDecision']"
      mode="dokument" />
    </o:UAM_import>
  </xsl:template>
```

Toodud XSL'ilõigu esimene <xsl:apply-templates> element kirjeldab seda, et

- iga „document“ elemendi puhul,
 - o millel on olemas atribuut „form“ väärtusega „frmCategory“
 - o ja millel pole sellist alamelementi „item“, mille atribuudi „Number“ väärtus sisaldab sidekriipsu,
- tuleb rakendada funktsiooni andmeid töötlev template.

Seda, et andmete töötlemiseks valitaks õige „document“ elemendile vastav template, antakse edasi mode'i abil: mode="funktsioon".

Funktsiooni andmeid töötlev template näeb välja selline:

```
<!-- Funktsioon: andmete teisendamine -->
<xsl:template match="i:document" mode="funktsioon">
  <xsl:variable name="viit" select="i:item[@name='Number']/i:text" />
  <xsl:variable name="id" select="i:noteinfo[@unid]/@unid" />
  <xsl:variable name="vanem" select="@parent" />
```

```

<xsl:variable name="datebegin" select="i:item[@name='DateBegin']/i:datetime" />
<xsl:variable name="marksona" select="i:item[@name='CaseKeywords']/i:text" />
<o:Funktsioon>
  <o:FunktsioonNimi>
    <xsl:value-of select="i:item[@name='Name']//i:text"/>
  </o:FunktsioonNimi>
  <o:KyViit>
    <xsl:value-of select="$viit"/>
  </o:KyViit>
  ....
</o:Funktsioon>
</xsl:template>

```

Antud template'is leitakse üksuse metaandmed „document“ elemendi alamelementide „item“ hulgast.

Näiteks leitakse lähte XML'ist „item“ element, mille name="Number", ning see salvestatakse „viit“ muutujasse (kuna seda väärtust läheb template'is vaja mitmel teisendusel). Väljundisse luuakse element „o:KyViit“, mille väärtuseks saab muutuja „viit“.

Sisendis: `<item name="Number"><text>1</text></item>`

Väljundis: `<o:KyViit>1</o:KyViit>`

XSL-i töötamise näide

Fragment lähte XML'ist:

```

<document xmlns="http://www.lotus.com/dxl" version="7.0" maintenanceversion="3.0"
replicaid="C2256CC30041D1F8" form="frmCategory"
parent="4509A980C365984CC22575690045B85D" response="true">
<item name="CategoryName"><text>Sarjad</text></item>
<item name="DateBegin"><datetime>20090201</datetime></item>
<item name="Name"><text>Vabariigi Valitsuse töö planeerimine</text></item>
<item name="Number"><text>01</text></item>
</document>

```

Transformeeritud fragment:

```

<o:Funktsioon>
  <o:FunktsioonNimi>Vabariigi Valitsuse töö planeerimine</o:FunktsioonNimi>
  <o:KyViit>01</o:KyViit>
  <o:KyVanemIdEDHS>4509A980C365984CC22575690045B85D</o:KyVanemIdEDHS>
  <o:FunktsioonTegevus>
    <o:TegevusNimetus>Funktsiooni loomine</o:TegevusNimetus>
    <o:TegevusAeg Formaati="yyyyMMdd">20090201</o:TegevusAeg>
  </o:FunktsioonTegevus>
</o:Funktsioon>

```

XSL-i kohandamise näide

Vaatame lihtsa näite abil, kuidas saab muuta ühe sisendi jaoks loodud XSL'i selliselt, et veidi teistsuguse struktuuriga sisend annaks sama lõpptulemuse.

Riigikantselei GoPro väljundis on dokumendi ID kirjeldatud elemendi „document“ alamelemendi „noteinfo“ atribuudis „unid“. XML fragment Riigikantselei GoPro väljundist:

```
<document form="frmApproval" parent="B85504B319248ECCC225764200282581"
response="true">
<noteinfo noteid="21d33a" unid="A458DF3D4AC6B837C2257642002B7F8F" sequence="4"/>
```

Fragmenti töötlev XSL'i osa, mis eraldab dokumendi ID muutujasse „id“:

```
<xsl:template match="i:document" mode="dokument">
  <xsl:variable name="id" select="i:noteinfo[@unid]/@unid" />
```

Analoogne XML fragment Tartu Linnavalitsuse GoPro väljundist väikse erinevusega „unid“ asukohas (elemendi „document“ alamelemendis „item“, mille atribuut „name“ on väärtusega „unid“):

```
<document form="InDoc" parent="9671BFC74A844184C225743A004A1DAC" response="true">
<item name="unid"><text>E720CCF4A9C3A0ECC225743A004A1D2C</text></item>
```

Fragmenti töötlev XSL'i osa:

```
<xsl:template match="i:document" mode="dokument">
  <xsl:variable name="id" select="i:item[@name=unid]/i:text" />
```

Riigikantselei GoPro väljundis on failid dokumendi alamelemendid. Järgnev näide kirjeldab kuidas peab kohandama XSL'i kui failid on väljaspool dokumendi elementi.

Failid on dokumendi alamelemendid (Riigikantselei GoPro):

```
<xsl:apply-templates select="i:item[@name='$FILE']"/>
...
<xsl:template match="i:item[@name='$FILE']">
  <xsl:variable name="file" select="i:object/i:file" />
  <o:Fail>
    <o:FailNimi>
      <xsl:value-of select="$file/@name"/>
    </o:FailNimi>
    <o:FailBase64>
      <xsl:value-of select="$file/i:filedata/text()"/>
    </o:FailBase64>
  </o:Fail>
</xsl:template>
```

Failid on väljaspool dokumendi elemente:

```
<xsl:variable name="parentID" select="i:noteinfo[@unid]/@unid" />
<xsl:apply-templates select="/i:database/i:file[i:parentID = $parentID]" />
...
<xsl:template match="/i:database/i:file">
  <o:Fail>
    <o:FailNimi>
      <xsl:value-of select="@name"/>
    </o:FailNimi>
    <o:FailBase64>
      <xsl:value-of select="i:filedata"/>
    </o:FailBase64>
  </o:Fail>
</xsl:template>
```


Olulised märkused

normalize-space

XSL'i koostamisel peab arvestama, et kui XML tag'ide sees on reavahetusi ja tab'e, siis need kanduvad ka edasi UAM'i, kus võivad põhjustada vigu (nt. ei õnnestu kuupäeva formaatimine kui kuupäeva kirjeldavas stringis on sees tab'id, base64 kujul faili sisu ei tohi sisaldada lisa tab'e, jne.). Seetõttu peab tab'e sisaldavate tag'ide kirjeldamisel kasutama XSL'i funktsiooni „normalize-space“.

XML:

```
<item name="Date">
  <datetime>
    19940106
  </datetime>
</item>
```

Vastav XSL:

```
<xsl:value-of select="normalize-space(i:item[@name='Date']/i:datetime)"/>
```

Ka järgnev XSL lisab alloleva XML'i teisendamisel reavahetused ja tab'id :

```
<item name="Date">
  <datetime>19940106</datetime>
</item>

<xsl:value-of select=" i:item[@name='Date']"/>
```

Seetõttu peab lisama normalize-space funktsiooni:

```
<xsl:value-of select="normalize-space(i:item[@name='Date'])"/>
```

Hierarhia

Peale sisend XML'i teisendust UAM'i jaoks sobivale XML-kujule pannakse paika imporditud üksuste hierarhia, st. otsitakse igale üksusele vastav ülemüksus. Hierarhia luuakse elementide „KyVanemIdEDHS“ või „KyVanemViit“ järgi. Ülemüksust otsitakse kõigepealt elemendi „KyVanemIdEDHS“ järgi. Kui see on täietud ja sellele ID'le vastav üksus leidub (laetavates XML'ides või andmebaasis leidub üksus, mille „KyIdEDHS“ võrdub otsitava „KyVanemIdEDHS“-ga), siis seatakse leitud üksus ülemüksuseks. Vastasel juhul otsitakse analoogselt „KyVanemViit“ elemendi järgi (otsitakse üksust, mille „KyViit“ võrdub otsitava „KyVanemViit“-ga). Seega peab hierarhia loomiseks kõigil ülemüksust omavatel üksustel olema antud element „KyVanemIdEDHS“ või „KyVanemViit“ ja ülemüksustele nendele vastav „KyIdEDHS“ või „KyViit“. Dokumentidel on vastavateks alluvust kirjeldavateks väljadeks „DokVanemIdEDHS“ ja „DokVanemViit“.

Allolevas näites otsitakse esmalt „KyVanemIdEDHS“ järgi, kuid vastet ei leita, sest vastava „KyIdEDHS“-ga üksus puudub. Seejärel otsitakse „KyVanemViit“ järgi ning sari „Kirjavahetus“ paigutatakse funktsiooni „Vabariigi Valitsuse töö planeerimine“ alla, sest sarja KyVanemViit=1 ning funktsiooni KyViit=1.

Näide:

```
<o:Funktsioon>
  <o:FunktsioonNimi>Vabariigi Valitsuse töö planeerimine</o:FunktsioonNimi>
  <o:KyViit>1</o:KyViit>
  <o:FunktsioonTegevus>
    <o:TegevusNimetus>Funktsiooni loomine</o:TegevusNimetus>
    <o:TegevusAeg Formaati="yyyyMMdd">20090201</o:TegevusAeg>
  </o:FunktsioonTegevus>
</o:Funktsioon>

<o:Kirjeldusyksus>
  <o:KyTasand>sari</o:KyTasand>
  <o:KyIdentiteediala>
    <o:KyViit>1-3</o:KyViit>
    <o:KyVanemViit>1</o:KyVanemViit>
    <o:KyVanemIdEDHS>A8570E61EC7D249AC22575690045F18B</o:KyVanemIdEDHS>
    <o:KyPealkiri>Kirjavahetus</o:KyPealkiri>
  </o:KyIdentiteediala>
  ....
</o:Kirjeldusyksus>
```

Kuupäeva formaadid

UAM'ile sobivasse kujusse teisendatud XML'is peavad kuupäevad olema antud koos formaadiga:

```
<o:TegevusAeg Formaati="yyyyMMdd">20090201</o:TegevusAeg>
```

Näited XSL'ist:

```
<o:DokRegAeg Formaati="yyyyMMdd">
  <xsl:value-of select="i:item[@name='Date']"/>
</o:DokRegAeg>

<o:DokRegAeg Formaati="yyyyMMddTHHmss,ffz">
  <xsl:value-of select="i:item[@name='DateSent']/i:datetime"/>
</o:DokRegAeg>
```

```

</o:DokRegAeg>

<o:PiirangAlgus>
  <xsl:attribute name="Formaat">
    <xsl:value-of select="$formaad" />
  </xsl:attribute>
  <xsl:value-of select="$piirangAlgus" />
</o:PiirangAlgus>

```

Formaat peab vastama .Net DateTime formaadile.
 DateTime formaadi kirjeldus:

<http://msdn.microsoft.com/en-us/library/system.globalization.datetimeformatinfo.aspx>

Eestikeelse UAMi korral võib formaadi osa ära jätta järgmisel kujul olevate kuupäevade korral:

2009-10-20
 2009/1/22
 2009.1.22

22.1.2009
 20-10-2009
 20/10/2009
 20 10 2009

Kõik ülal mainitud kuupäevad võivad sisaldada ka kellaega kujul:

„kuupäev“ 10:40 (nt. 22.1.2009 10:40)
 „kuupäev“ 10:40:50
 „kuupäev“ 10:40:50.1234

2009-01-20T02:30:50.1
 2009-01-20T02:30:50.1+03
 2009-01-20T02:30:50.1+03:00

Ingliseelses UAM'is on ülal mainitud formaatides, mis ei alga aastaga, kuu ja päeva järjekord vahetatud.
 Nt. 1.22.2009