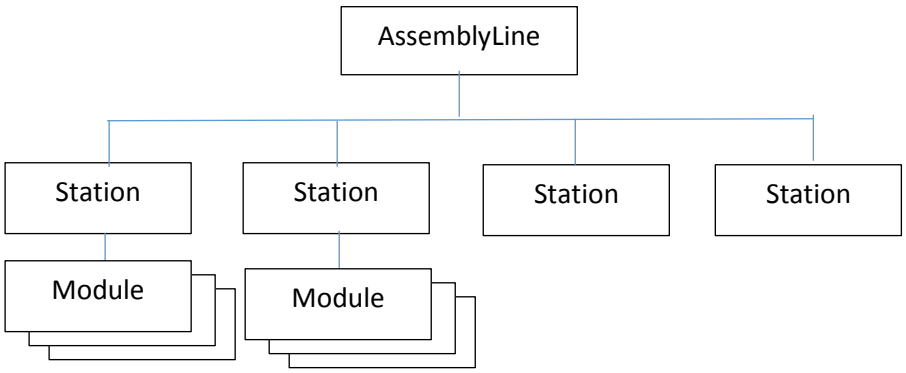
Tehtävä_4_XML_Schema.pdf
ELEC-C8203 Automaatiojärjestelmät 2
XML Tietointegraatio

Tehtävä 4: XML Schema

1. XML Scheman luominen AL-dokumenteille

Tässä tehtävässä tehdään XML Schema, joka määrittelee uuden XML-kielen ALML (AssemblyLineMarkupLanguage) *assembly_line*-tyyppisten xml-dokumenttien tarkistamista/ validointia varten. Lähtökohtana käytetään tehtävässä 1 tehtyä *assembly_line_base.xml* (yksinkertaistettu AL.xml) dokumenttia, jonka tulee olla validi määriteltävän XML Scheman suhteen. Huomaa, että annettu AL.xml on vain yksi mahdollinen määriteltävän scheman mukainen dokumentti. Lopullisen scheman tulee määritellä kaikki AL.xml:ssä esiintyvät elementit sekä attribuutit ja määrittelyjen tulee vastata siinä esiintyviä tyyppejä ja rakenteita.



Kuva 1. assembly_line.xml dokumentin ylimpien tasonjen elementtirakenne fyysisten laitteiden osalta. Muista, että xml-dokumentti sisältää myös ei rakenteellisia elementtejä kuten <name> ja <description>

2. Scheman vaiheittainen kehitys

Osatehtävä 1

Ensimmäisessä osatehtävässä tehdään xml-schema *ALML_schema.xsd* (pohjana käytetään annettua tiedostoa *ALML_schema_pohja.xsd*), joka validoi yksinkertaistetun *assembly_line_base.xml* dokumentin elementtien rakenne-hierarkian ilman attribuutteja tai tekstisisältöä.
Scheman kehittäminen kannattaa tehdä vaiheittain aloittaen yksinkertaisesta schemasta, joka sallii/hyväksyy validiksi lähes kaikki xml-dokumentit, joiden juurielementti on *assembly_line*. Vaiheittain inkrementaalisesti lisätään schemaan määrityksiä, jotka rajoittavat validin dokumentin rakennetta (*XML_Schema_extra.pdf* kalvoissa 50-56 on esitetty esimerkki vaiheittaisesta scheman kehityksestä. Siihen kannattaa tutustua ennen tehtävän aloittamista).
Huom. Koska tässä vaiheessa ei vielä haluta, että schema tarkistaa attribuutti-määritykset, jokaisen *xs:complexType* elementin lapsena *xs:sequence*-elementin jälkeen tulee olla seuraava rivi (’skippaus’)
<xs:anyAttribute processContents="skip"/>

Scheman vaatimuksia:

Scheman tulee osoittaa validiksi kaikki xml-dokumentit, joiden rakenne vastaa *assembly _line_base.xml* dokumentin rakennetta. Tarkoitus on vaiheittain lisätään elementtien tyyppimäärittelyjä (mutta skipataan vielä attribuutit):

1. Juurielementin *<assembly_line>* lapsina saa olla vain *<station>* elementtejä
2. jokaisella *<station>* elementillä saa olla rajoittamaton määrä <module> elementtejä lapsielementteinään
3. moduuli-elementtien lapsina voi olla vain elementtejä, jotka on esitetty *assembly_line_base.xml* instanssidokumentissa ja niiden on oltava samassa esitetyssä järjestyksessä
4. Kaikkien elementtien, joilla on lapsia, tyyppi tulee määritellä *xs:complexType* määrittelyllä, joka sisältää *<xs:sequence>* malliryhmän
5. Yksinkertaisuuden vuoksi myös lapsettomien laitteisto-elementtien (puun lehtielementit, kuten *sensor*) tyyppi määritellään kuten edellä. Tämä tarkoittaa sitä, että sallitaan myös niille lapsielementit *'name'* ja *'description'* (Huom. vaihtoehtona olisi käyttää *complexType-simpleContent* tyyppimäärittelyä)
6. Tekstimuotoista metatietoa sisältävät elementit *<name>* ja *<description>* ovat yksinkertaista tyyppiä (SimpleType) ja niiden tyypin tulee olla xs:string. esim: *<xs:element name="name" type="xs:string"/>*
7. Scheman tulee sallia kaikkien *xs:complexType* määrittelyn *xs:sequence*:ssa esiintyvien elementtien lukumäärän (kardinaliteetti = #n) olevan välillä (#0 - unbounded) esim: *<xs:element name="sensor" minOccurs="0" maxOccurs="unbounded"/>*
8. Tarkista, että assembly_line_base.xml dokumentti on validi ALML-dokumentti kirjoitamasi ALML scheman suhteen (ks. ohjeet luku 4)

Osatehtävä 2

Tässä osatehtävässä määritellään kaikkien elementtien tyypit myös niiden attribuuttien osalta.

1. Lisää elementille tarvittavat attribuutti-määritykset, jokaisen *xs:complexType* elementin lapsena olevan *xs:sequence*-elementin jälkeen (poista samalla vaiheessa 1 lisätty 'skippaus').

   esim: *<xs:attribute name="type" type="xs:string"/>*

2. Muuta schemassa elementeille määrättyjä kardinaliteetti-raja-arvoja (minOccurs maxOccurs) niin, että ne sallivat vain *assembly_line_base.xml*:ssä esiintyvien elementtien minimi ja maksimimäärien välillä olevat lukumäärät.
3. Tarkista, että assembly_line_base.xml dokumentti on validi ALML-dokumentti kirjoitamasi ALML scheman suhteen (ks. ohjeet luku 4).

3. Vinkit

1. Tässä osatehtävässä kannattaa käyttää vaiheittaista inkrementtaalista scheman kirjoittamistapaa (ks. esimerkki *XML_Schema_extra.pdf* kalvoissa 50-56)
2. Jokaisen täydennyksen jälkeen kannattaa itse tarkistaa, että schema edelleen validoi AL.xml:n. (Notepad++ tai http://www.xpathtester.com/validate)
3. Tehtävän ratkaisemiseksi kannattaa tutustua mm. scheman *xs:complexType* määrittelyyn

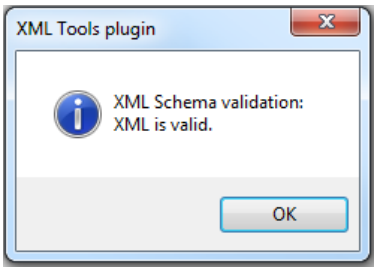
4. Monitahoisten tyyppien <*xs:complexType*> sisältömallien määrittelyssä on tärkeää käyttää oikeaa malliryhmää. Tämän tehtävän rakaisussa tarvitsee kuitenkin käyttää ainoastaan <*xs:sequence*> malliryhmä määrittelyjä.

4. XML-dokumentin validointi kahdella eri tavalla

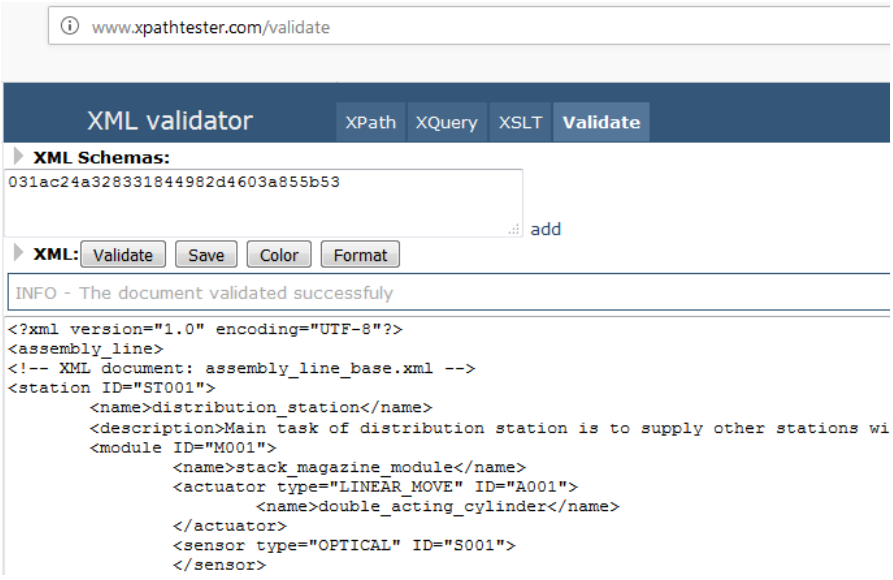
Validointi Notepad++ ohjelmalla

Avaa *assembly_line_base.xml* tiedosto Notepad++ editorilla (Jos se on asennettuna). Valitse menusta *Plugins/XML tools/validate now*, jolloin *'Select XML Schema file'* dialogi aukeaa. Navigoi (…) ja avaa tiedosto *ALML_schema.xsd* ja paina *'OK'* painiketta. Validoinnin tuloksena tulisi aueta 'XML Tools plugin' dialogi-ikkuna, joka ilmoittaa validoinnin tuloksen: XML is Valid/Invalid.
Tekstieditori: Notepad++ with XML-plugin
Lähtötiedostot:

- Kohde XML: *Assembly_line_base.xml* (= tehtävässä 1 kirjoitettu perusmalli)
- XSL: *ALML_schema.xsd* (tässä tehtävässä kirjoitettu tiedosto)



Kuva 2Validoinnin tulosilmoitus Notepad++:ssa.



Kuva 3 XML-dokumentin validointi online työkalulla.

Validointi online XSD ohjelmalla

http://www.xpathtester.com/validate

Paina XML Schemas-kentän perässä olevaa *'add'*-painiketta ja kopioi *ALML_schema.xsd* tiedoston sisältö *'Content'*-kenttään ja paina *'Save'*. Tämän jälkeen XML lähtödokumentti kopioidaan XML-kenttään, jonka jälkeen voidaan ajaa validointi painamalla *'Validate'* painiketta. Validoinnin tulos näkyy INFO-kentässä: *INFO - The document validated successfuly*

On-line XSLT tester: http://www.xpathtester.com/validate

Lähtötiedostot:

- Kohde XML: *Assembly_line_base.xml* (= tehtävässä 1 kirjoitettu perusmalli)
- XSL: *ALML_schema.xsd* (tässä tehtävässä kirjoitettu tiedosto)

5. Jatkotehtäviä

JT1: XSD-prosessorin ilmoitukset

Kielen formaali määrittely Scheman avulla mahdollistaa sisäänluetavien instanssidokumenttien tarkistuksen ennen niiden sisällön varsinaista käsittelyä. Tässä tehtävässä tutkitaan minkälaisia ilmoituksia XSD-prosessori tuottaa epävalidia tiedostoa tarkistettaessa.

1. Kopioi XML tiedosto ja nimeä se *assembly_line_invalid.xml* :ksi ja lisää siihen jonkin moduulin lapsiksi *sensor*-elementtejä niin että niiden lukumäärä ylittää *maxOccurs*-raja-arvon.
2. Testaa sen validiteetti ja tutki *invalid*-ilmoitusta; selviääkö siitä virheen lähde?
3. Kommentoi pois edellä lisätyt ylimääräiset sensor-elementit
4. Tee samanlainen testi lisäämällä ylimääräisiä attribuutteja joihinkin elementteihin ja poistamalla joistakin elementeistä Scheman vaativia attribuutteja. Tutki jälleen invalid-ilmoituksen kuvausta.
5. Tarkista laajennetun AL-mallin *assembly_line_ext.xml* ALML-Scheman suhteen. Onko se validi ALML-kielinen dokumentti?
6. Täydennä *ALML_schema.xsd* schemaa niin, että se hyväksyy myös laajennetun AL-mallin *assembly_line_ext.xml* kielen mukaiseksi.

JT2: ALML-kielen määrittelyn viimeistely - Nimiavaruuden määrittely

Viimeistellään kehittämämme ALML-kielen formaali määrittely ilmoittamalla schemassa kohde-nimiavaruuden URI (*Uniform Resource Identifier*). Aloittaessamme kielemme suunnittelun tehtävässä 1 listasimme jo kaikkien elementtien nimet, joita tulemme käyttämään. Nyt määrittelemme kohdenimiavaruuden (URI), johon kaikki käyttämiemme elementtien nimet kuuluvat (tehtävässä käytettävä URI on täysin keksitty; URI:n ei tarvitse olla toimiva web-osoite eli URL):

targetNamespace="http://alml.org/2019/AlmlSchema"

Lisäykset XML-Schemaan

1. Tee kopio schemasta ALML_Schema.xsd ja nimeä se *ALML_Schema_with_NS.xsd:*ksi
2. Lisää *ALML_Schema_with_NS.xsd:n* juurielementtiin *xs:schema* seuraavat attribuutit:
   - *<xs:schema xmlns:xs=*"http://www.w3.org/2001/XMLSchema" *xmlns=*"http://alml.org/2019/AlmlSchema" *targetNamespace=*"http://alml.org/2019/AlmlSchema" *elementFormDefault=*"qualified">
3. Attribuutilla *targetNamespace* määritellään kohdenimiavaruuden nimi ja attribuutin *elementFormDefault* arvo "*qualified*" kertoo, että kaikki elementit, myös lokaalisti schemassa määritellyt, kuuluvat ko. nimiavaruuteen ja ne täytyy siis *'qualifioida'* XML-instanssidokumentissa

Lisäykset XML-instanssidokumenttiin

1. Tee kopio dokumentista assembly_line_base.xml ja nimeä se tiedostoksi *assembly_line_base_with_NS.xml*
2. Lisää *sen* juurielementtiin *xs:assembly_line* seuraavat attribuutit:

   <assembly_line xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" *xmlns=*"http://alml.org/2019/AlmlSchema" *xsi:schemaLocation=*"http://alml.org/2019/AlmlSchema *ALML_Schema_with_NS.xsd*">

3. Attribuutti xmlns nimeää käytetyn kohdenimiavaruuden ja attribuutti *xsi:schemaLocation* nimeää paitsi scheman nimiavaruuden URI:n myös osoittaa schema-tiedoston nimen (kun schemaa ei löydy URI:n osoittamasta web-osoitteesta.

Dokumentin validointi schemaa vasten

1. Aja *assembly_line_base_with_NS.xml* dokumentin validointi *ALML_Schema_with_NS.xsd* schemaa vasten
2. Jos validointi epäonnistuu, korjaa mahdolliset kirjoitusvirheet ja yritä uudelleen
3. Kun XSD-prosessori ilmoittaa dokumentin validiksi, ALML-kielen täydellinen formaali määrittely on valmis.

6. Reflektiokysymykset

Saadaksesi kokonaiskäsityksen siitä mitä olet tämän XML moduulin harjoitustehtävissä oppinut, palauta ne mieleesi ja pohdi missä tehtävissä mm. seuraavat asiat on opittu:

A. Olet mallintanut tietyn KP-linjan rakenteen XML-muodossa lähtien liikkeelle sen vain ihmisen luettavaksi tarkoitetusta tekstuaalisesta kuvauksesta. Kirjoitettu ALML-kielinen dokumentti on myös koneen luettavissa ja siten sen tietosisältö ohjelmallisesti edelleen käsiteltävissä.
B. Olet kirjoittanut XSLT-muunnosdokumentin, jonka sääntöjen ohjaamana sovellus voi generoida mistä tahansa samalla ALML-kielellä kirjoitetusta KP-linjan kuvauksesta standardimuotoisen AutomationML kuvauksen. Täten kirjoittamasi KP-linjan malli on useiden standardia tukevien suunnittelutyökalujen luettavissa.
C. Olet myös tutustunut ja täydentänyt XPath-hakulausekkeilla XSLT-tyylisivua, jonka sääntöjen ohjaamana voidaan KP-linjan mallista generoida yhteenvetoraportti ja esittää se web-sivuna pelkästään web-selainta käyttäen.
D. Olet määritellyt KP-linjan mallissa käyttämäsi ALML-kielen myös formaalisti XMLSchema kielen avulla. Tämän formaalin määrittelyn avulla voidaan ohjelmallisesti etukäteen varmistaa, että käsiteltävät XML-dokumentit ovat todella valideja ALML-kielisiä dokumentteja.

Tehtävän 4 demoaminen tai palautus sähköpostilla

Tehtävät on tarkoitus suorittaa demoamalla harjoitusryhmissä eli siis viimeistään ti 12.2. klo 10-12. Tämän itsenäisesti suoritettavan tehtävän 4 'XML Schema ja ALML-kieli' osaratkaisu on kuitenkin mahdollista palauttaa sähköpostilla vielä saman viikon aikana (osaratkaisu: max 3p).

Ohjeet sähköpostipalautukseen

Viimeinen email-palautuksen ajankohta su 17.2 klo 24:00.
Lähetä email-viesti osoitteeseen: pekka.aarnio@aalto.fi seuraavin tiedoin
Otsikko:XML tehtävän 4 ratkaisu

Työparin op-numerot ja nimet
Ratkaisun tulee sisältää seuraava online validointiajo:
Validointi:
assembly_line_base.xml
ALML_Schema.xsd

Aja xml-tiedoston validointi online ohjelmalla: http://www.xpathtester.com/validate
ja talleta se *save*-painikkeella. Kopioi INFO kentästä ajon permalinkki viestiisi
esim: INFO - Permalink (xpath): http://www.xpathtester.com/validate/ee7b2ebbd549ce81064

Liitetiedostot: Lisää viestin liitteiksi validointiajossa käyttämäsi tiedostot 2kpl