hyväksymispäivä arvosana

arvostelija

Viitemallin profilointi terveydenhuollon sanomien määrittelyssä

Markus Montonen

Helsinki 7.10.2013

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta – Fakultet –Faculty	Laitos – Institution – Department
Matemaattis-luonnontieteellinen tiedekunta	Tietojenkäsittelytieteen laitos

Tekijä – Författare – Author

Markus Montonen

Työn nimi – Arbetets titel – Title

Viitemallin profilointi terveydenhuollon sanomien määrittelyssä.

Oppiaine – Läroämne – Subject

Tietojenkäsittelytiede

Työn laji – Arbetets art – Level	Aika – Datum – Month and year	Sivumäärä – Sidoantal – Number of pages
Pro gradu	7.10.2013	70 sivua

Tiivistelmä – Referat – Abstract

Terveydenhuollon tietojärjestelmät ovat luotu usean vuosikymmenen aikana ja ne perustuvat usein toisistaan poikkeaviin tekniikoihin, menetelmiin ja standardeihin. Tällaisten keskenään heterogeenisten järjestelmien välinen yhteistoiminta on ollut hankalasti toteutettavissa. Erityisesti kliinisten dokumenttien välitys järjestelmien välillä on vaatinut huomattavasti ihmistyötä. Järjestelmien yhteistoiminnan automatisoimiseksi on esitetty useita standardeja, joista keskeisimmässä asemassa on ollut HL7-organisaation ylläpitämä sanomastandardi. Sen keskiössä on UML-luokkakaaviona esitetty RIM-viitemalli, joka mallintaa kliinisissä dokumenteissa käytetyn käsitteistön. Edelleen viitemallia rajoittamalla voidaan siitä muodostaa profiileja, jotka kuvaavat ainoastaan tiettyä käyttötarkoitusta varten luodun kliinisen dokumentin käsitemallin, jota kutsutaan RMIM-kaavioksi. Siitä voidaan edelleen johtaa XML-skeema, jota käyttämällä viestin toteuttaja voi tarkistaa toteuttamansa viestin oikeellisuuden. XML-skeema ei kuitenkaan yksinään riitä tarkistamaan kaikkea RMIM-kaaviossa määriteltyjä viestin rajoitteita. Tätä varten on kehitetty mallinteet, jotka lisäävät tarkistuksia joko kerronnallisin toteutusoppain tai koneellisesti prosessoitavilla Schematron-tarkistuksilla. Tässä työssä tarkastellaan miten terveydenhuollon järjestelmien välittämien kliinisten dokumenttien kuvaukset määritellään. Lisäksi työssä tarkastellaan Suomen sähköisen lääkemääräyksen kuvausta RIM-viitemallin ja mallinteiden avulla.

ACM Computing Classification System (CCS):
Applied computing → Health informatics
Applied computing → Health care information systems
Social and professional topics → Health information exchanges
Software and its engineering → System description languages

Avainsanat – Nyckelord – Keywords

HL7, yhteentoimivuus, terveydenhuolto, sanoman välitys, CDA

Säilytyspaikka – Förvaringställe – Where deposited

Kumpulan tiedekirjasto, sarjanumero C-

Muita tietoja – Övriga uppgifter – Additional information

Sisältö

1 Johdanto

Tietojärjestelmät ovat yleisiä sairaaloissa ja terveyskeskuksissa [HDS04]. Näiden järjestelmien yhteentoimivuus on kuitenkin ongelma, koska järjestelmät on mallinnettu itsenäisesti [Sas12] ja eri toimijoiden toimesta [HDS04]. Toisin sanoen järjestelmät eivät ymmärrä toisiaan, ja tiedon lähettäminen järjestelmästä toiseen vaatii ihmisen ymmärrystä tiedon käsittelemiseen, niin lähettävän kuin vastaanottavankin järjestelmän päässä. Kolmannen osapuolen mukaan tulo tiedonvälitykseen on hankalaa ja vaatii määrittelyä sekä dokumentointia suunnittelijoiden välillä [Ben10].

Alun perin järjestelmien yhteentoimivuudella pyrittiin vähentämään paperien määrää sekä parantamaan taloutta ja hallinnon päätöksentekoa. Myöhemmin sillä on parannettu palvelun laatua ja nopeutta. Organisaation toiminta tehostuu yhteentoimivuuden lisääntyessä järjestelmässä, esimerkiksi turha ihmisen tekemä tiedon syöttö järjestelmästä toiseen vähenee. Terveydenhuollon alalla on lisäksi tarvetta koota potilaan missä tahansa kirjatut koko elämän aikaiset hoitotiedot hoitohenkilökunnan käyttöön [HDS04].

Tiedon automaattinen ja koneellinen käsittely vaatii tiedon olevan rakenteellista ja koodattua. Vaatimusten täyttämiseen on kehitetty terveydenhuollon standardeja, joista käytetyin on Health Level 7 eli HL7 [HL713] organisaation luomat standardit sanomien määrittelyyn ja välitykseen [HMN12, ETD07]. Organisaation luomat kaksi standardia, HL7 versio 2 ja 3, määräävät tiedon rakenteellisuuden ja koodauksen.

HL7 versiossa 2 ei ole semanttista yhteentoimivuutta. Siitä puuttuu yhteinen koko aihealueen kattava tietomalli, vakaasti koneellisesti käsiteltävät tietotyyppimääritykset, sanastojen yhtenäinen sitominen viestiin ja formaali ylhäältä alaspäin tehtävä viestin kehitysprosessi. Versiossa 2 on mahdollista lisätä vapaaehtoisia kenttiä. Niiden vuoksi versio 2 ei ole täysin syntaktisesti yhteentoimiva. Versiossa 3 sen sijaan on kaikki yllämainitut ominaisuudet. Ne eivät ole kuitenkaan riittäviä estämään saman asian sanomista eri tavoilla sanomassa. Siksi versio 3 tarvitsee lisämäärityksiä viestin profiloinnissa, kuten mallinteita [Mea06].

Tutkielma koostuu teoriasta sekä siihen liittyvästä esimerkkitapauksesta. Teoriaosassa tarkastellaan, miten semanttinen yhteentoimivuus saavutetaan luomalla CDA-arkkitehtuurin mukaisia määrittelyjä välitettäville sanomille, miten sanomien määrittelyä tarkennetaan mallinteilla (templates), ja mihin asti määrittelyä pelkkä viitemallin käyttäminen

riittää. Esimerkkitapauksessa tutkitaan Suomen terveydenhuollossa käytetyn sähköisen lääkemääräyksen, eReseptin [Ere], toteutusta, ja miten siinä on käytetty CDA-arkkitehtuurin määrittelemiä mallinteita ja missä kulkee raja, että niitä tarvitaan.

Tutkielman rakenne on seuraava: Luvussa kaksi käsitellään tiedon välitykseen tarvittavaa semanttista yhteentoimivuutta. Luvussa kolme esitellään HL7 versio 2, miten sitä käytetään ja miksi sen määrittelemät sanomat eivät ole semanttisesti yhteensopivia toistensa kanssa. Luvussa neljä esitellään HL7 versio 3 ja sen käyttö sekä miksi se tuo sanomiin semantiikkaa. Luvussa viisi käsitellään HL7 version 3 määrittelemää RIM-viitemallia. Luvussa kuusi esitellään RIM-viitemallin profilointia RMIM-kaavioiksi sekä XML-skeemaksi. Luvussa seitsemän käsitellään HL7 version 3 käytetyintä arkkitehtuurimallia CDA-arkkitehtuuria. Luvussa kahdeksan on CDA-arkkitehtuurin mukainen viestien profilointi ja sen määrittelemät mallinteet. Luvussa yhdeksän on Suomen sähköisen lääkemääräyksen analysointi ja mallinteiden käyttö. Luvussa kymmenen esitetään kritiikkiä HL7-standardia vastaan ja miten HL7-organisaatio on siihen vastannut. Viimeisessä luvussa on yhteenveto.

2 Semanttinen yhteentoimivuus

Kaksi järjestelmää ovat yhteentoimivat (interoberable), kun toinen järjestelmistä pystyy vastaanottamaan tietoa tai palvelupyynnön toisesta järjestelmästä ja suorittamaan sen mukaisen mielekkään tehtävän ilman muun toimijan väliintuloa [Dog12]. Semanttista yhteentoimivuutta (semantic interoperability) on kyky välittää palveluita ja tietoa järjestelmien välillä niin, että molemmat osapuolet ymmärtävät välitettävän tiedon sisällön samalla tavalla [Hei95]. Yhteentoimivuutta on kolmea tasoa. Syntaktinen, osittainen semanttinen ja täysin semanttinen yhteentoimivuus [BBG10].

Semanttisuudella kuvataan tarkoitusta, kun sen sijaan *syntaksilla* kuvataan rakennetta. Esimerkkinä semanttisuudesta voi pitää kahta lausetta; koira syö punaista lihaa ja koira laulaa sinisiä puita. Rakenteellisesti lauseet ovat samanlaisia, niissä on tekijä, verbi ja kohde. Semanttisesti ensimmäinen lause on järkevä ja jälkimmäinen ei. *Syntaktinen yhteentoimivuus* takaa rakenteellisesti samanlaisen tiedon siirtymisen järjestelmien välillä. Se ei ota kantaa, että molemmat siirron osapuolet ymmärtäisivät viestin sisällön samalla tavalla. Esimerkiksi HTML- tai XML-tiedostot ovat syntaktisesti siirrettäviä tiedostoja.

Eri verkkoselaimella tai ohjelmalla avattaessa tiedosto voi näyttää erilaiselta riippuen ohjelman tulkinnasta. Tämä ei ole ongelma, koska tiedostojen semanttisen tulkkaus on tarkoitettu ihmisen tehtäväksi. *Semanttinen yhteensopivuus* sen sijaan takaa molempien osapuolten yksiselitteisen tiedon ymmärtämisen. Kliiniset dokumentit kuten leikkauspöytäkirja, lähetteet tai lääkemääräykset noudattavat kliinistä sanastoa ja yleisiä käytäntöjä, jotka takaavat kliinikoiden välisen ymmärryksen. Tietokonejärjestelmien välinen semanttinen yhteensopivuus takaa, että järjestelmät ymmärtävät siirrettävän tiedon yksiselitteisesti. Tämä ei tarkoita, että kaikki järjestelmät käsittelisivät tietoa samalla tavalla vaan jokainen järjestelmä päättää käsittelytavasta tiedon merkityksen perusteella [Mea06].

Syntaktinen ja osittainen semanttinen yhteentoimivuus perustuu sopimuksiin lähettäjien ja vastaanottajien kesken. Sovittavia asioita ovat esimerkiksi viestien kuljetusprotokolla, metodien nimet, virhekoodit ja parametrien tietotyypit. Semanttinen yhteentoimivuus varmistaa, että sovitut asiat ovat kaikille osapuolille yhteisessä ymmärryksessä. Esimerkiksi vastaanottavan järjestelmän tulee ymmärtää mitä potilaalla tarkoitetaan lähettävässä järjestelmässä, jolloin se voi käsitellä saamaansa tietoa hyödyllisesti [Hei95]. Tämä on osittaista semanttista yhteentoimivuutta, koska ennen tiedon välittämistä on järjestelmien välillä sovittu mitä lähetetään [BBG10].

Osittainen semanttinen yhteentoimivuus perustuu sopimuksille välitettävän tiedon keräämisen algoritmeista ja niiden sivuvaikutuksista (side effect) sekä välitettävän tiedon oikeellisuudesta ja tietomallista. Semanttisen yhteensopivuuden sopimukset voivat puuttua, jos vastaanottaja ja lähettäjä ovat itsenäisiä projekteja ilman ennalta sovittuja sääntöjä. Ne voivat myös puuttua, jos eri asiaan tehtyä algoritmia tai tietoa uudelleen käytetään eri yhteydessä. Tällöin tiedon pyytäjä ei voi olla varma siitä, että hänen oletuksensa ovat samat kuin tiedon tarjoajan. Vastaanottajan saama tieto voi olla väärää vastaanottajan tietämättä, jolloin vastaanottajan jalostaman tiedon lopputulos voi olla väärä. Esimerkiksi Yhdysvaltojen puolustusministeriö pyrki jalostamaan tietokannastaan veteraanien osoitteiden perusteella tietoa hakeutuvatko veteraanit sairaanhoidon piiriin. Järjestelmä oli kuitenkin suunniteltu seuraamaan aktiivisia sotilaita, joten sinne tallennettu tieto osoitteista saattoi olla väliaikaisia työtehtäviä eikä veteraanien kotiosoitteita, mikä teki tiedosta uuteen tarkoitukseen hyödyttömän [Hei95].

Terveydenhuollon järjestelmien välistä semanttista yhteentoimivuutta tarvitaan kliinisten tietojen saatavuuden parantamiseen, potilaiden liikkuvuuteen ja tiedon saatavuuteen eri maiden välillä, vähentämään kliinisiä virheitä ja näin ollen parantamaan turvallisuutta,

parantamaan laatutiedon saatavuutta potilaille ja ammattilaisille ja parantamaan terveydenhuollon prosessien tehokkuutta [BLD05]. Virheet vähenevät ja tehokkuus paranee kun kliinisiä tietoja ei tarvitse tulostaa toisesta järjestelmästä ja käsin syöttää toiseen järjestelmään.

Semanttisen yhteentoimivuuden saavuttamiseen Landgrebe ja Smith ovat artikkelissaan [LaS11] keränneet joukon periaatteita: Kehikon tulee määritellä yksi yhtenäinen toimialue (domain), esimerkiksi terveydenhuolto. Toimialueelle tulee määrittää selvät rajat, joiden sisällä semanttinen yhteentoimivuus on voimassa. Lisäksi kehikon tulee toteuttaa parhaita käytäntöjä (best practices), jotta sen käyttöönottoon on matala kynnys. Lisäksi semanttisen yhteensopivuuden määrittelemä abstraktio tulee olla sillä tasolla, että se palvelee eri teknologioita käyttäviä järjestelmiä. Yhteentoimivuuden kannalta kehikon tulee kuvata miten tunnistetaan ja muotoillaan yrityksessä käytetyt prosessit kehikon määrittelemään muotoon. Kehikon tulee erottaa tietosisältö ja tiedonvälitystapa. Sen tulee määrittää miten tieto järjestellään onnistuneen tiedonvälityksen saavuttamiseksi niin tietomallin kuin reaalimaailman suhteiden tasolla. Kehikon tulee määritellä yksittäisten järjestelmien käyttäytymismallit ja interaktiot järjestelmien välillä siten, että ne sopivat kehikon määrittelyyn. Kehikon tulee kuvata järjestelmäarkkitehtuurin vaatimukset missä kehikon on tarkoitus toimia.

Yllä kuvatut vaatimukset kehikolle eivät vielä tuota semanttisesti yhteentoimivaa kehikkoa, mutta tarjoavat ohjenuorat sen määrittämiseen.

HL7 (Health Level 7) on tehnyt terveydenhuollon alalle standardeja vuodesta 1987. Standardien tavoitteena on yhteentoimivuus alan järjestelmien välillä. Versio 3 tuo mukanaan semanttista yhteentoimivuutta, joka on määritelty kahden tai useamman järjestelmän väliseksi kommunikoinniksi missä vastaanottavat järjestelmät ymmärtävät tiedon samalla tavalla kuin lähettävä järjestelmä on sen tarkoittanut. Tavoitteen saavuttamiseksi versiossa 3 on keskiössä RIM-viitemalli. HL7-organisaation ideana on, että kaikki tietosisältö johdetaan RIM-mallista, jolloin semanttiseen yhteentoimivuuteen vaadittavat sopimukset täyttyisivät.

RIM-viitemallin käyttöönotto on kohdannut useita vaikeuksia. Se on monimutkainen ja sen perusta on omaperäinen teknisesti ja ontologisesti. Siinä käytetyt tekniikat ovat sekoitus UML-kuvauskieltä ja XML-metakieltä, mikä on tehnyt siitä vaikeakäyttöisen ih-

misille ja tietokoneille. HL7-organisaatio on kehittänyt palvelutietoisen yhteentoimivuuskehikon (Service-Aware Interoperability Framework, SAIF). Sen tavoite on tarjota toimiva yhteentoimivuus terveydenhuollon alalla. SAIF käsittää kolme yhteentoimivuusparadigmaa; viestitys (messaging), dokumentit ja palvelut. Viestitys keskittyy tapaan vaihtaa tietoa, dokumentit ovat CDA-standardin määrittelemiä terveydenhuollon tietodokumentteja ja palvelut ovat järjestelmien tarjoamia kyselyrajapintoja. Yhteentoimivuuden saavuttamiseksi tulee huomioida välitettävän tiedon rakenne ja järjestelmien käyttäytyminen. HL7 keskittyy tiedon rakenteeseen, mutta käyttäytymiseen keskitytään vasta palveluparadigman yhteydessä. SAIF ei rajaa kehikkoa vain terveydenhuoltoon eikä RIM-malli erottele loogista mallinnusta ja teknistä toteutusta. Malli ei ole teknisesti itsenäinen ja se käyttää HL7-organisaation määrittelemiä XML-suunnittelu periaatteita. SAIF ei siis täytä kaikkia semanttisen yhteentoimivuuden saavuttamiseksi tarvittavia periaatteita [LaS11].

Täysin semanttinen yhteentoimivuus ei tarvitse sopimuksia järjestelmien välille. Tällöin kieli tai teknologiset erilaisuudet eivät estä järjestelmää vastaanottamasta tai lähettämästä tietoa. Tämä saavutetaan käyttämällä semanttisen verkon teknologioita, kuten OWL-ontologiaa ja siihen liittyviä päättelykoneita. Toisin kuin XML-metakielisissä, OWL-ontologian mukaisissa viesteissä on semantiikkaa mukana [BBG10].

3 HL7 versio 2

3.1 Taustaa

HL7-organisaatio on voittoa tavoittelematon vapaaehtoisuuteen perustuva organisaatio. Se on perustettu 1987, ja siinä on jäseninä järjestelmien tarjoajia, välittäjiä, maksajia, konsultteja sekä viranomaisryhmiä, joilla on kiinnostuksen kohteena kehittää ja edistää terveydenhuollon alan standardeja [HDS04]. Nimi HL7 tulee sanoista Health Level 7. Numerolla seitsemän viitataan ISO-OSI-mallin [Zim80] ylimpään, eli sovelluskerrokseen. Sovelluskerros määrittelee erilaisten tietokonejärjestelmien sovellustason kommunikaation eikä ota kantaa käytettäviin siirtotekniikoihin [Ben10]. HL7-organisaation standardi versio 3 määrittelee sovelluskerroksen lisäksi kuljetuskerroksen eri protokollaprofiileja. Versio 2 ei määrittele pakollista kuljetuskerroksen protokollaa, mutta suosittelee TCP/IP-, RS-232-johtoperusteista ja ANSI X3.28 -perusteista protokollaa [NAD08].

Terveydenhuollon tieto asettaa vaatimuksia, joita muilla aloilla ei ole. *Tiedon vähyys* (data

sparseness) tarkoittaa, että vain pientä osaa potilaan tiedoista käytetään kerrallaan. *Tiedon moniulotteisuudella* (large number of dimensions) tarkoitetaan diagnoosien, toimenpiteiden ja laboratoriokokeiden koodistojen kymmeniä tuhansia eri arvoja, joita voi yhdistellä potilaaseen. *Ei-laskettavia muuttujia* (nonadditive variables), kuten glukoosin ja kuumeen mittoja ei voi laskea yhteen, sen sijaan tiedon analysoimiseen tulee käyttää tiedon kokoamista tai tarkentamista yksityiskohtaisempaan tietoon. Esimerkiksi kuumeen mittaamisen keskiarvo ei kerro tarvittavaa tietoa, sen sijaan hoitava henkilö joutuu tutkimaan kaikki arvot tai ääriarvot. Terveydenhuollon tieto lisääntyy uusien tutkimusten ja toimenpiteiden myötä. Siksi malliin tulee voida pystyä lisäämään tietotyyppejä jatkuvasti. *Tunnistamatonta tietoa* (deidentified data) tulee voida tuottaa yksityisyyslakien mukaisesti. Johdettu ja alkuperäinen tieto tulee voida erottaa, esimerkiksi laboratoriotutkimuksen tulos ja siitä generoitu tekstimuotoinen ohje hoitavalle henkilölle ovat eri asioita. Tietoon tulee pystyä lisäämään tarkentavaa metatietoa, kuten millä laitteella jokin mittaus on tehty. Potilaan hoito vaatii lähes tosiaikaisen tiedon saatavuuden [ETD07].

HL7 versio 2 on onnistunut terveydenhuollon alan standardi. Se on käytössä laajalti Amerikassa ja Euroopassa [HDS04], ja se on eniten käytetty terveydenalan yhteentoimivuusstandardi. Siitä huolimatta ettei se ole onnistunut tarjoamaan suoraa yhteensopivuutta potilastietojärjestelmien välille [Rya06]. Tämä johtuu siitä, etteivät sen määrittelemät sanomat perustu mihinkään malliin [EAR05]. Version 2 edeltäjä, eli versio 1 kehitettiin 1987, muutamia kuukausia aikaisemmin kuin versio 2, mutta versio 2 korvasi sen nopeasti [Ben10] eikä version 1 tarkempi esitteleminen ole järkevää.

Versiossa 2 tiedonvälityksen suunnittelu järjestelmien välillä voi olla aikaa vievää ja vaatii yksityiskohtien tarkkaa toteuttamista, koska toteutus perustuu tilannetta varten (ad hoc) tehtyihin sanomiin [Rya06]. Sanomien määrittelyt eivät ole tarkkoja ja ne ovat vapaamuotoisia, mikä tekee määrittelystä joustavan, mutta vaikeuttaa yhteensopivuutta ilman ihmisten vuorovaikutusta [EAR05]. Koska versio 2 kehitettiin 1980-luvulla, sen käyttämät teknologiat ovat vanhentuneita. Esimerkiksi viestien kentät erotetaan putkimerkillä (pipe, |) ja sirkumfleksillä (caret, ^), koska muita keinoja ei vielä ollut. Viestien rakenteesta johtuen, on tärkeää monennessako kentässä tieto sijaitsee sanomassa. Yksikin puuttuva välimerkki voi muuttaa koko sanoman merkitsemättömäksi tekstijonoksi [Rya06]. Version 2 luomisaikaan tietoja välitettiin vain sairaaloiden välisessä viestinnässä. Siitä johtuen sanomat olivat yksinkertaisia. Niillä oli vain kolme tyyppiä: sisäänkirjautuminen

(admission), kotiuttaminen (discharge) tai siirto toiseen sairaalaan (transfer). Näitä kolmea sanaryhmää kutsutaan lyhenteellä ADT [Ben10].

HL7 version 2 kehitys on ollut jatkuvaa ja uusin standardi on versiossa 2.7, joka julkaistiin vuonna 2011 [HL713]. Version kaksi sisältämä tieto terveydenhuollon alasta on kasvanut, esimerkiksi versio 2.3 määritteli yli 300 erillistä viestiä (message) ja herätetapahtumaa (trigger event), jotka käyttävät yhteensä 113 palatyyppiä (segment type), viittäkymmentä tietotyyppiä ja 1250 tietoelementtiä (data element) [Bee98]. Versio kahden kehityksessä alaspäin yhteensopivuus on ollut tärkeänä periaatteena. Siksi uusia ominaisuuksia on vain lisäilty siihen muuttamatta vanhaa rakennetta. Ideana on, että järjestelmä, joka ymmärtää uusilla ominaisuuksilla varustettuja sanomia, ymmärtää myös vanhempia sanomia. Vanhemmat versiot yksinkertaisesti vain jättävät uudet ominaisuudet huomioimatta. Vanhat ominaisuudet merkitään dokumentaatiossa vanhentuneiksi (deprecated), mutta niitä ei poisteta tai korvata. Yritykset ovat haluttomia päivittämään version kaksi sisällä, koska vanhat rajapinnat toimivat ja uusi rajapinta toisi vähän tai ei ollenkaan taloudellisesti lisää tuottoja. Tästä syystä rajapintojen kehittäjien tulee ottaa huomioon eri versioiden mahdollisuus ja toimia sen mukaisesti [Ben10]. Version 2 käyttämien teknologioiden ja rakenteen vuoksi yksinään version 2 mukaisten sanomien semantiikkaa ei koneellisesti voida tarkistaa [Rya06].

3.2 Rakenne

HL7 versio 2 -tyyppisten viestien syntaksi kuvaa viestien rakenteen. Jokainen sanoma koostuu segmenteistä (segments) tietyssä järjestyksessä. Segmentit koostuvat kentistä (fields), jotka ovat tietyssä järjestyksessä. Kentillä on lisäksi tietotyyppi, joka kertoo minkälaista tietoa se sisältää [Ben10]. Jokaisen palan tiedot liittyvät johonkin tiettyyn asiaan terveydenhuollon alalta [Bee98]. Esimerkiksi PID (Patient Identification) -pala viittaa potilaan tietoihin. Osioilla on aina kolmikirjaiminen tunniste, ja sanomissa osion tunniste aloittaa aina rivin [Ben10].

HL7 versio 2 on kehitetty olettamuksella, että herätetapahtuma aiheuttaa tiedonvälityksen kahden järjestelmän välillä. Kun herätetapahtuma tapahtuu järjestelmässä, järjestelmä kerää tietokannasta tai vastaavasta sanoman tarvitsemat tiedot ja luo version kaksi mukaisen sanoman. Esimerkiksi herätetapahtuma ADT^A01, eli potilaan hallinnollinen käsitteleminen, tapahtuu kun potilas otetaan sisään hoitolaitokseen. Tapahtuma voi aiheuttaa tietojen keräämisen ja lähettämisen yhteen tai useampaan järjestelmään [EAR05]. Muita

potilaan hallintoon liittyviä tapahtumia ovat potilaan siirto, kotiuttaminen ja rekisteröinti. Merkintä ADT^A01 koostuu kahdesta osasta. ADT on herätetapahtuma ja A01 on tarkennus. Sirkumfleksi on standardin tapa erotella kentän sisältö. Muita herätetapahtumia ovat esimerkiksi kuittaus- (ACK), tilaus- (ORM) ja tulosviestit (ORU). Herätetapahtumat kuvaavat tapahtumaa, joka käynnisti sanoman generoinnin järjestelmässä.

Standardi määrittelee missä järjestyksessä viestien osiot ovat ja mitkä niistä ovat pakollisia tai toistuvia. Esimerkiksi potilaan sisäänotosta sairaalaan kertova viesti koostuu otsakeosasta (MSH), tapahtumatyypistä (EVN), potilaan tiedoista (PID) ja potilaan saapumisesta (PV1). Viestien merkintä abstraktiin syntaksitauluun vaikuttaa miten niitä tulee käyttää. Osion koodi ilman muuta merkintää tarkoittaa, että se on pakollinen ja ei-toistuva. Vapaaehtoiset osiot merkitään hakasuluilla ja toistuvat kaarisulkeilla. Esimerkiksi taulukossa 1 on kuvattu ADT^A01 tapahtuman abstrakti syntaksi taulu, missä PD1- ja NK1-osiot ovat vapaaehtoisia osioita ja lisäksi NK1-osio voi toistua useasti.

ADT^A01	**ADT-sanoma**
MSH	otsaketieto (message header)
EVN	tapahtumatyyppi (event type)
PID	potilaan tietoa (patient identification)
[PD1]	muuta väestötietoa (additional demographics)
[{ NK1 }]	lähisukulaiset (next of kin)
PV1	sisäänottotietoa (patient visit)

Taulukko 1. Abstrakti syntaksitaulukko ADT^A01-sanoman osioista.

HL7 versio 2 standardi käyttää välimerkkejä osioiden, kenttien ja tietojen erottelemiseen. Kentät erotellaan putkimerkillä (|) ja kentän sisältö sirkumflexilla (^). Kenttien sisällä tapahtuvaa toistoa kuvataan madolla (~), koodinvaihtomerkit kenoviivalla (\), alisisältö et-merkillä (&) ja ASCII-merkistön rivinvaihto lopettaa osion.

Osioiden kentät ovat nimetty niiden järjestyksen mukaan. Esimerkiksi otsaketieto-osion yhdeksäs kenttä on nimeltään MSH-9. Jos kenttään ei tule tietoa, jätetään se tyhjäksi merkitsemällä kaksi kentän erotinta peräkkäin (||). Vastaavasti kenttien sisältämät komponentit ovat nimetty niiden järjestysnumeron mukaan kentän sisällä. Esimerkiksi MSH-9.1

viittaa otsaketieto-osion yhdeksännen kentän ensimmäiseen komponenttiin. Toisin kuin kentän erottimia, kentän sisällön erottimia ei tarvita kentän lopuksi. Esimerkiksi kenttä |123^Potilas^^| on sama kuin |123^Potilas|.

Toistomerkillä erotetaan saman kentän sisällöt toisistaan. Esimerkiksi |Mies^Potilas~Male^Patient| on kenttä, missä potilaan tieto on ensin suomeksi ja sitten englanniksi. Koodinvaihtomerkillä erotetaan tiedossa olevat standardin käyttämät merkit. Esimerkiksi putkimerkki muutetaan muotoon \F\ ja sirkumfleksi \S\. Sitä voi käyttää myös ilmaisemaan tyylittelyä, kuten \.br\, joka tarkoittaa rivinvaihtoa. Vaihdetut merkit voidaan muuttaa takaisin alkuperäisiksi tietoa tallennettaessa vastaanottajan päässä.

Kaikki HL7 version 2 sanomat alkavat otsakeosiolla. Esimerkki kuinka osa siitä määritellään, on taulukossa 2. Taulukossa 2 järjestysnumero kertoo, monesko kenttä on kyseessä osiossa. Pituus ja tietotyyppi kertovat millaista ja miten pitkää tietoa kentässä sallitaan. Pakollisuus ja toistettavuus kertovat pitääkö tieto olla jokaisessa sanomassa ja kuinka monta kertaa. Viite- ja kohdenumero kertovat HL7 version 2 sisäisen tietokannan tai taulun numeron mistä kentän tiedot voi hakea. Kentän nimi on ihmisluettava nimi kentälle. Esimerkki taulukon 2 mukaisesta sanomasta olisi MSH|^~\&|Helsingin Yliopisto||||||||2.4||||||UNICODE UTF-8 [Ben10].

Järjestysnumero	**Pituus**	**Tietotyyppi**	**Pakollisuus**	**Toistettavuus**	**Viitenumero**	**Kohdenumero**	**Kentän nimi**
1	1	merkkijono	on	ei		00001	Kentän erotin
2	4	merkkijono	on	ei		00002	Vaihdettavat merkit
3	180	merkkijono	ei	ei		00003	Lähettävä järjestelmä
12	8	tunniste	on	ei	0104	00012	Versionumero
18	6	tunniste	ei	on/3	0211	00692	Merkistöjoukko

Taulukko 2. Esimerkki otsaketieto-osion määrittelystä HL7 versio 2 standardissa [Ben10].

HL7 version kaksi alkuperäinen muoto on ylläkuvatun tyyppistä. XML-metakielen yleistyminen ja sen käyttö HL7 versiossa 3 on tuonut versioon kaksi XML-rakenteen. Jokaiselle version 2 standardin versiolle on määritelty XML-muoto ja vuonna 2003 HL7-organisaatio standardoi XML-muodot. Putkimuotoisen sanoman muuttaminen XML-muotoon on suoraviivainen prosessi. Elementtitunniste (tag) saa nimekseen segmentin ja siinä olevan kentän järjestysnumeron. Kentän sisältö sijoitetaan elementtitunnisteiden sisään. Esimerkiksi kuvassa 1 on esitetty XML-muotoinen esitys taulukon 2 mukaisesta esimerkkisanomasta.

```
<MSH>
        <MSH.1>|</MSH.1>
        <MSH.2>^~\&</MSH.2>
        <MSH.3>
                <HD.1>Helsingin Yliopisto</HD.1>
        </MSH.3>
        <MSH.12>
                <HD.1>2.4</HD.1>
        </MSH.12>
        <MSH.18>
                <HD.1>UNICODE UTF-8</HD.1>
        </MSH.18>
</MSH>
```

Kuva 1. HL7 version 2 sanoma muutettuna XML-muotoon.

Version 2 tietotyypit voidaan muuttaa XML-muotoon. Kuvassa 2 version 2 osoitetiedot ovat esitettynä CDA-arkkitehtuurin rakenteellisessa muodossa. Aloitusmerkin attribuuttina on version 2 seitsemäs kenttä, joka on osoitteen tyyppi. Se voi olla vakituinen tai väliaikainen kotiosoite, syntymäpaikka tai työosoite. Muuten kentän vastaavat XML-merkkejä, kuten osoiterivi yksi ja kaksi, kaupunki, maakunta, postikoodi, maa ja maantieteellinen alue. Kuvan 2 elementtitunnisteet ovat CDA-arkkitehtuurin määrittelemät. Niiden sisältämät tiedot ovat putkimuotoisen sanoman sisällöt. Kuvassa 2 tietojen tilalla on putkimuotoisen sanoman kenttien nimet, mutta oikeassa tapauksessa ne sisältävät oikean osoitteen.

```
<addr use='AD.7'>
   <streetAddressLine>AD.1</streetAddressLine>
   <streetAddressLine>AD.2</streetAddressLine>
   <city>AD.3</city>
   <state>AD.4</state>
   <postalCode>AD.5</postalCode>
   <country>AD.6</country>
   <county>AD.8</county>
</addr>
```

Kuva 2. HL7 v2 osoitetieto esitettynä CDA-arkkitehtuurin tietotyyppimuodossa [Boo11]

4 HL7 versio 3

HL7 version 3, kuten edeltäjänsä version 2, tavoite on luoda standardoitu tapa välittää järjestelmien välillä sanomia [YuD10], jotka ovat myös määriteltyjä ja testattavia. Vaikka HL7 versio 2 on kattava ja toimiva, se on kehityksensä päässä. Sen teknologinen rakenne ei anna mahdollisuutta HL7-organisaation kehittää sitä pidemmälle. Sillä on myös heikkouksia; sen käyttöönotto useissa järjestelmissä ei ole helppoa, jokainen rajapinta vaatii useiden viikkojen analysoinnin ja sopimisen järjestelmien tuottajien kesken. Versio 2 sisältää paljon vapaaehtoisia kenttiä ja osioita, siksi luotujen viestien testaaminen automaattisesti on hankalaa [Bee98]. Version 2 ongelmien poistamiseksi HL7-organisaatio alkoi kehittää versiota kolme vuonna 1994 [Bee98]. Yksi version 2 heikkouksista on, ettei siinä ole yhteistä kliinistä sanastoa. Tällöin samaa tarkoittava asia saattaa olla sanomissa esitetty eri tavalla. Esimerkiksi toinen henkilö voi merkitä kaliumin lyhenteellä K ja toinen lyhenteellä kal. Tällaisten sanomien semanttinen yhteensovittaminen on monimutkainen prosessi. Siksi tietomallin sisällön tulee käyttää koodistoa, josta sanoman lähettäjä ja vastaanottaja voivat tarkistaa käytettävän termin [Rya06]. Aihealuesanasto (vocabulary domains) määrittää kentän sallitut arvot. Esimerkiksi siviilisääty voi olla naimaton, naimisissa, leski, eronnut tai asumuserossa. Aihealuesanastot liitetään HL7 version 3 sanomiin koodistotietotyypin avulla, esimerkiksi CV (coded value) kertoo arvon ja sen ihmisluettavan nimen sekä koodiston (koodiston tunniste ja nimi). Jos käytetty koodisto on julkinen, sanoman vastaanottava järjestelmä ymmärtää sanoman sisällön paremmin [BCC00].

Toisin kuin versio 2, versio 3 käyttää viestin määrittelyssä oliopohjaista kehittämismallia sekä RIM (Reference Information Model) -viitemallia [YuD10, Rya06]. RIM kostuu kahdesta tasosta; normatiivinen taso ja kommunikaatiotaso. Normatiivinen taso määrittelee standardin käyttämän sanaston, ja kommunikaatiotaso viestien välityksessä käytettävät tietosisällöt [HDS04]. Version 3 käyttämä viestien välitysmuoto on XML-muotoinen, jonka elementtitunnisteet (tags) perustuvat luonnolliseen kieleen. Elementtitunnisteet ovat tällöin ihmisluettavia sekä koneellisesti käsiteltäviä [VAN11]. Versio 3 tuo mukanaan suuria uudistuksia versioon 2 verrattuna [OeB05]. Aihealueasiantuntijat ovat luoneet useita terveydenhuollon alan sanastoja. Version 3 kehitysvaiheessa kehittäjät eivät halunneet tehdä samaa työtä uudelleen. Siksi RIM-malli voi viitata ulkopuolisiin koodistoihin ja käyttää jo valmiina olevia sanastoja tietosisällössään. Esimerkki ulkopuolisesta sanastosta on SNOMED-CT-ontologia, joka on kokoelma lääketieteellisisä termejä, niiden attribuutteja ja suhteita [Rya06]. Versiossa 2 ei määritelty pakollisia kuljetusprotokollia, mutta versio 3 määrittelee kolme ISO-OSI-mallin kuljetuskerroksen protokollaprofiilia. Protokollat ovat ebXML, www-sovelluspalvelu (web service) ja TCP-IP-protokollaan perustuva profiili [NAD08]. Versio 3 muuttaa käytettyä ajatusmallia ja tekniikoita, siksi sen kehittäminen on ollut pitkän ajan prosessi. Versio 3 tarjoaa spesifikaation koko elinkaaren, aina tietosisällön määrittelystä sanoman toteutukseen ja testaamiseen asti [OeB05].

Tammikuussa 1997 version 3 työryhmä julkaisi viestien kehittämiskehyksen (Message Development Framework, MDF), jonka avulla toimikunnat (committee) ja HL7-organisaation kehittäjät voivat luoda standardin viestien luomiseksi ja lähettämiseksi. MDF on täydellinen (complete) ja täysin dokumentoitu malliajattelutapaan perustuva kehittämismalli. Se määrittelee neljä mallia, jotka tulee toteuttaa viestien välitysstandardia luodessa. Nämä neljä mallia ovat käyttötapaus (use case)-, tieto (information)-, kanssakäymis (interaction)- ja viestin rakennemalli (message design). Standardin luomisprosessi alkaa käyttötapausmallilla ja jatkuu edellisen järjestyksen mukaisesti päättyen viestin rakennemalliin. Prosessi on syklinen [Bee98].

Käyttötapausmalli on ensimmäinen toimikuntien toteutettava malli. Se määrittelee tilanteet, joissa tiedon välityksen järjestelmien välillä on tapahduttava. Siinä dokumentoidaan lisäksi molempien järjestelmien odotetut käyttäytymiset. Malli auttaa tunnistamaan ja määrittelemään avainasiat välitettävästä tiedosta.

Tietomalli on seuraava toteutettava malli. Se kerää kaikissa sanomissa käytetyt tietosisällöt yhteen malliin. Sen pohjana on RIM. RIM on viitemalli kuvattuna UML-luokkakaaviona, joka sisältää aihealueet, luokkien kaikki attribuutit sekä perimis-, luokka- ja ilmentymäsuhteet [Bee98]. RIM määrittelee viesteissä käytettävän tietosisällön ja sisällön suhteet toisiinsa [Rya06]. Se sisältää kaiken sanomissa käytetyn tiedon [YuD10]. RIM, yhdessä attribuuttien rajoittamisen kanssa, määrittelevät HL7 version 3 tietosisällön [Rya06]. Tietomallissa määritellään aihealueen luokat, esitellään toimikunnan määrittelemät uudet materiaalit ja liitetään ne osaksi RIM-viitemallia. Aihealueen luokilla on herätetapahtumia (trigger events), jotka määritellään seuraavan vaiheen kanssakäymismallissa. Luokkia käytetään aktiivisesti potilastietojärjestelmässä. Esimerkiksi potilas tai tutkimuksen tilaus ja tulos ovat aihealueen luokkia. Jokaiselle aihealueen luokalle tehdään tiladiagrammi, joka kuvaa luokan koko elinkaaren. Jokainen tilasiirros tiladiagrammissa on mahdollinen herätetapahtuma, joka käynnistää sanoman luomisen. Esimerkiksi tutkimuksen tilaus aktivoi tutkimuksen tilaus –sanoman luomisen, ja tutkimuksen valmistuminen aktivoi tutkimuksen tulos –sanoman luomisen järjestelmässä.

Kanssakäymismalli on kolmas toteutettava malli. Se määrittelee HL7-sanomien käyttävien järjestelmien käyttäytymistä. Toimikunnat määrittelevät herätetapahtumia, kanssakäymisiä (interaction) ja järjestelmien rooleja. Herätetapahtumat saadaan käyttötapauksista ja RIM-viitemallista. Kanssakäyminen on yksittäinen ja yhdensuuntainen tiedon välitys, joka sisältää viestin, herätetapahtuman ja vähintään kaksi järjestelmän roolia. Toinen välitettävistä rooleista on lähettäjä ja toinen vastaanottaja. Kanssakäymiset vastaavat HL7 version 2 herätetapahtumasta luotuja sanomia. Järjestelmien roolit ovat joukko vastuita, jotka kohdistuvat sanomaan. Järjestelmällä voi olla useita rooleja yhdessä sanomassa, esimerkiksi tutkimuksen tilauksen lähettäjän rooleja voivat olla: lähettäjä, potilaan seuranta (tracker), tilauksen hallinta ja tutkimustuloksen seuranta.

Viimeinen toteutettava malli on viestin rakennemalli. Se määrittelee viestin formaatin RIM-viitemallista, ja sen tulee täyttää kanssakäymismallin vaatimukset jokaiselle kanssakäymiselle. Prosessi etenee niin, että toimikunta ottaa RIM-mallista ne luokat, attribuutit ja suhteet joita tarvitaan määriteltävissä sanomissa. Sen jälkeen niistä luodaan viestin tietomalli (message information model, MIM). MIM on RIM-mallin osajoukko, minkä lisäksi se voi rajoittaa viitemallia lisäämällä attribuutteihin kardinaliteetteja ja poistaa turhia attribuutteja luokasta. MIM-mallista luodaan viestin oliokaavio (message object dia-

gram), josta työstetään, läpikäymällä jokainen luokka, hierarkkinen viestin kuvaus (hierarchial message description, HMD). HMD on perusta sanoman toteuttamiselle. Itse toteuttaminen määritellään toteutettavan viestin määrittelyissä (Implementable Message Specifications, IMS). Määrittelyjä yhdelle viestille on useita [Bee98], mutta yleisin niistä on XML-metakieli.

Aihealuesanastokomitea vastaa yleisistä, julkisista HL7 versioon 3 liittyvistä sanastoista. Kaikkien sanoman vastaanottajien ymmärtämä sanasto parantaa yhteensopivuutta. HL7-organisaatio pyrkii käyttämään ennalta määrättyjä sanastoja aina kun mahdollista. Komitea on luonut joukon periaatteita, joita HL7 versio 3 sanastoa luodessa tulee noudattaa.

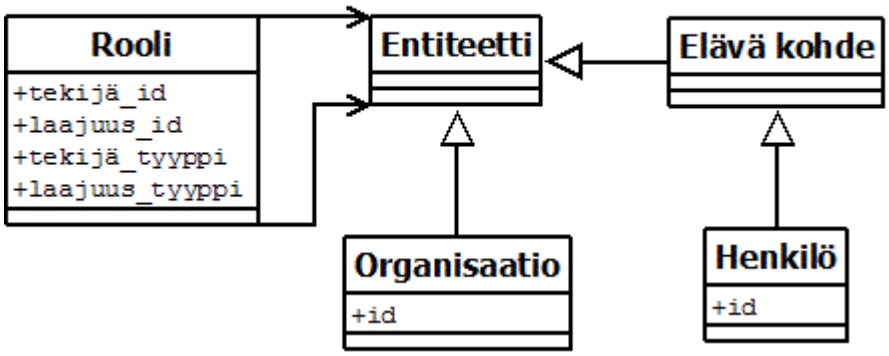
- Sanaston tulee noudattaa HL7 version 3 semantiikkaa.
- Sanaston kehittäjän tulee olla halukas osallistumaan HL7-organisaation hyväksymiin yhteentoimivuusmuutoksiin.
- Sanaston kehittäjän organisaation tulee sitoutua ylläpitämään ja päivittämään sitä.
- Sanaston lisenssimaksujen tulee olla suhteellisia sanaston tarjoamaan arvoon käyttäjille, mutta suhteutettuna sanaston kehittämisen ja ylläpidon kuluihin.
- Sanaston tulee olla kattava siinä aihealueessa mihin se on kehitetty.
- HL7-organisaatio voi ottaa sanaston haltuunsa, jolloin siitä tulee osa HL7-standardia.

Aihealuesanastokomitea on listannut toimenpiteet, joilla se tekee yhteentoimivuusmuutoksia. Ensimmäiseksi sanaston kehittäjän tulee voida julkistaa käsitteellinen malli sanastosta, ja HL7-organisaatiolla tulee olla lupa, ilman tekijänpalkkiota, toteuttaa sanasto HL7-ohjelmiin sekä liittää sanaston osia HL7-standardiin. Toiseksi sanaston kehittäjän tulee luovuttaa sanasto kirjallisen sopimuksen puitteissa HL7-organisaation hyväksymälle sanaston integraatio organisaatiolle (terminology integration organization, TIO), joka sen haluaa. Viimeiseksi TIO-organisaation tulisi tehdä kirjallinen suostumus sanaston kehittäjän kanssa sanaston sisällön luotettavuudesta ja oikeellisuudesta [BCC00]. Suomessa sähköisen lääkemääräyksen tapauksessa käytettyjä sanastoja ylläpitää Terveyden ja hyvinvoinnin laitos (THL) [THL13].

HL7 versio 3 –standardi ei ota kantaa tiedon tallennukseen tai hallinnointiin, mutta se tukee keskitettyä ja hajautettuja tietokantoja. Se tarjoaa perustan viestien siirrolle organi-

saatiosta toiseen heterogeenisessä järjestelmäympäristössä. CDA (Clinical Document Architecture) määrittelee HL7 versiota 3 apuna käyttäen kliinisiä dokumentteja talletukseen tai tiedon siirtoon. Viestien kehittämiskehys (MDF) määrittää viestien kuljetusformaatin. Se määrittelee useille eri järjestelmille viestien rakenteen. Siirrettäviä viestejä ovat esimerkiksi kyselyt (kuten sähköisten lääkemääräysten haku), vastaukset (esimerkiksi lista potilaan sähköisistä lääkemääräyksistä) sekä tilaukset (orders, esimerkiksi sähköisen lääkemääräyksen määrääminen) [HDS04].

HL7 version 3 RIM-viitemallia ei ole suunniteltu tietokannan pohjaksi, mutta se soveltuu siihen. Eggebraaten ja kumppanit artikkelissaan [ETD07] luovat tietovaraston, jonka tietomallina on RIM. He yhdistävät RIM-mallin luokat tietokantaan niin, että kaikki RIM-mallin luokat ovat omana taulunaan ER-tekniikkaa (entity-relationship) käyttäen. Poikkeuksena on havaintoluokka. Luokkiin on lisätty viiteavaimet RIM-mallin mukaisten suhteiden tyydyttämiseksi. Esimerkiksi RIM-mallin pääluokat ja aliluokat muodostavat tällaisen suhteen. Jos suhde on pääluokkaan, jolla on aliluokkia, tauluun on lisätty kenttä, joka kertoo luokan nimen. Nimi tulee lisätä, koska viittauksen kohde voi olla mikä tahansa pääluokan aliluokka. Esimerkiksi rooliluokassa voi olla kaksi suhdetta entiteettiluokan minkä tahansa aliluokan kanssa. Nämä suhteet ovat tekijä ja laajuus. Viiteavaimien, tekijä_id ja laajuus_id, lisäksi roolitauluun tulee kentät tyypeille, tekijä_tyyppi ja laajuus_tyyppi, jotka kertovat minkä nimiseen aliluokkaan suhde viittaa. Esimerkiksi kuvassa 3 tekijä_tyyppi tai laajuus_tyyppi voivat olla entiteetti, organisaatio, elävä kohde tai henkilö.

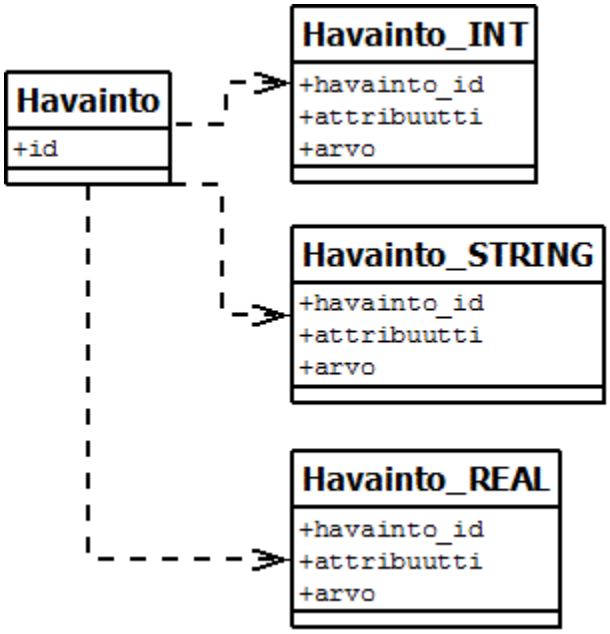


Kuva 3. Rooliluokan viiteavaimet.

Luokkien attribuuttien perimisen Eggebraaten ja kumppanit ovat selvittäneet luomalla jo-

kaiselle erillisen taulun, joka sisältää luokan ja luokan vanhemman kaikki tarvittavat attribuutit. Kentät, joissa tietotyypin arvona voi olla koodiston arvo, viittaavat omaan tauluunsa, jossa on koodin arvo, koodisto, versio ja näytettävä nimi. Ratkaisu ei tarkista tietokannan eheyttä, ja siksi se vaatii sisään luettavan tiedon oikeellisuuden tarkistusta ohjelmallisesti ennen tietokantaan syöttämistä.

Eggebraaten ja kumppanit ovat toteuttaneet havaintoluokan eri tavalla. Havaintoluokka koostuu kolmikosta tunniste, koodi ja arvo. Koodi erittelee havainnon, esimerkiksi glukoosi. Arvo kertoo havaitun mitan, esimerkiksi 35 mg. Arvon tietotyyppi RIM-mallissa on kaikki (ANY), ja vaikka mallia profiloidessa sen tietotyyppiä rajoitetaan tiettyyn HL7 version 3 tietotyyppiin, tarvitsee tietovaraston pystyä tallentamaan siihen minkä tyyppistä tietoa tahansa. Eggebraaten ja kumppanit ovat ratkaisseet tämän luomalla jokaiselle tietotyypille oman taulun, joka viittaa havaintoon. Kaikki havainnon attribuutit tallennetaan omana rivinään tietotyypin mukaiseen attribuuttitauluun. Kyselyssä pitää yhdistää kaikki taulut kaiken tiedon saamiseksi [ETD07]. Esimerkiksi kuvassa 4 on havaintoluokasta suhteet omiin attribuuttitauluihin kokonaisluvuille, merkkijonoille ja reaaliarvoille.



Kuva 4. Havaintoluokan attribuuttitaulut.

HL7 versiossa 3 käytetään luomuksien (artifact) nimeämiseen omaa tyyliä. Nimi on muoto SSDD_AAnnnnnnRRVV. Ensimmäiset neljä kirjainta (SSDD) kuvaavat aliosiota ja aihealuetta. Mahdollisia arvoja ovat esimerkiksi:

- COCT – Yleiset viestielementit (Common Message Elements)

- COMT – Yleinen viestisisältö (Common Message Content)
- FIAB – Kirjanpito ja laskutus (Accounting and Billing)
- FICR – Vahingonkorvaukset ja hyvitykset (Claims and reimbursement)
- MCAI – Viestiaktin infrastruktuuri (Message Act Infrastructure)
- MCCI – Viestihallinnan infrastruktuuri (Message Contol Infrastructure)
- MFMI – Päätiedoston hallinta infrastruktuuri (Master File Management Infrastructure)
- POLB – Laboratorio (Laboratory)
- PORX – Apteekki (Pharmacy)
- PRPA – Potilaan hallinnointi (Patient Administration)
- PRPM – Henkilökunnan hallinta (Personnel Management)
- PRSC – Aikataulutus (Scheduling)
- QUQI – Kyselyinfrastruktuuri (Query Infrastucture)
- RCMR – Sairaskertomus (Medical Records)

Ensimmäisten kirjaimien jälkeen tulee alaviiva (_), jota seuraavat kaksi kirjainta kertovat luomuksen tyypin. Tyyppejä on esimerkiksi DM (DMIM), HD (HMD), MT (viestityyppi, message type), RM (RMIM) tai TE (herätetapahtuma). Tyypin jälkeiset kuusi numeroa ovat tekijän vastuulla oleva tunniste. Tunnisteen jälkeen tulee maakoodi, kuten FI (Suomi) tai UV (universaali). Viimeiset kaksi numeroa ovat versionumerot väliltä 00-99 [Ben10]. Esimerkiksi Suomen sähköisessä lääkemääräyksessä käytetty XML-skeema on nimetty POCD_MT000040FI, jossa POCD on CDA-arkkitehtuurin käyttämä aliosio ja aihealue lyhenne CDA-arkkitehtuurista. Tyyppi on viestityyppi, tunniste 000040 ja maakoodi Suomi. Versionumeroa ei käytetä.

5 RIM-viitemalli

5.1 Yleistä

RIM (Reference Information Model) on luokkakaavio, joka esittää HL7-viesteissä käy-

tettyjen kenttien semantiikat ja yhteydet [Ben10]. Se on tärkeä osa yhteensopivuutta järjestelmien välillä. Yhteensopivuutta ja standardeja on käytetty kirjallisuudessa lähes samana asiana. Protokollia käytetään alhaisen tason viestin välitykseen kuten tietoliikenteessä ja väliohjelmistoissa. Sovelluskerroksen ohjelmien heterogeenisyys aiheuttaa kerroksen yhteentoimivuusongelmia viestien välityksessä. RIM-viitemalli on yritys tuoda yhteensopivuutta sovelluskerrokselle tekemällä kattavan tietomallin, johon viitataan sanomista [SaD08].

Sartipi ja Dehmoobad artikkelissaan [SaD08] esittävät ohjeen kuinka mistä tahansa aihealueesta voi luoda semanttisesti yhteensopivan mallin, kuten RIM-mallissa on tehty. Heidän ohjeensa perustuu RIM-mallin muuntamiseen muiden kuin terveydenhuollonalan käyttötarkoitukseen. Ohje sisältää kaksi haaraa: palvelut ja informaatio. Palveluiden haaran tehtävä on muodostaa standardoituja viestejä. Sen haaran ensimmäinen kohta on vaatimusten analysointi (requirement analysis). Siinä määritellään toiminnalliset vaatimukset, jotka muodostavat liiketoimintasääntöjä sovellusaihealueeseen. Seuraava kohta on skenaarioiden luominen (scenario extraction). Liiketoimintasääntöjä tarkennetaan muokkaamalla niitä aihealueen organisaatioiden tärkeiden ja yleisten käyttötapausten mukaan. Kolmas kohta on puhdistaminen (refinement). Geneeristen liiketoimintasääntöjen puhdistaminen luo yrityskohtaisia standardoituja tapahtumia, joista sovelluskehittäjät voivat luoda standardoituja yhteyksiä. Viimeinen kohta on interaktioiden analysoiminen (interaction analysis). Siinä tapahtumista luodaan standardoituja viestejä, jotka kuljettavat tietoa järjestelmien välillä. Viestien luomisprosessi voi olla iteratiivinen. Toisen, eli informaatio, haaran tehtävä on järjestää aihealueen tieto saataville ja lähetettäväksi. Sen ensimmäinen kohta on esitys (representation). Siinä joukko aihealueen henkilöitä määrittää aihealueen laajuuden. Lopputuloksena on tietomalli joko UML- tai ER-kaaviona. Toinen vaihe on osittaminen (partitioning). Aihealueella on erilaisia toisiinsa liittyviä erikoistapauksia. Nämä tapaukset kerätään omiksi alueiksi, joista on viittauksia toisiin alueisiin. Seuraava vaihe on puhdistaminen, jossa jokaisen alueen tiedot käydään läpi ja vain siirrettäväksi tarpeelliset tiedot jätetään tietomalliin. Viimeinen vaihe on tulkkaus (interprepation). Tässä vaiheessa eri organisaatioiden käyttämät termit yhdistetään yhdeksi yleisesti käytettävään terminologiaan. Palveluiden haara luo standardoidun viestirakenteen ja informaatiohaara tarjoaa sisällön viesteihin.

HL7 RIM-luokkakaavio on esitetty UML-tekniikalla (Unified Modeling Language) [Boo11]. Sen on luonut HL7-organisaation määräämä ja rahoittama projekti [Bee98]. Se

on keskeinen osa HL7 version 3 ajattelumaailmaa. RIM määrittelee version 3 käyttämän kieliopin; käytetyt termit, niiden sallitut suhteet ja tietotyypit. Se ei mallinna terveydenhuoltoalaa tai viestiä, vaan se on apuväline viestien luomiseen [Ben10]. Malli sisältää kaiken tarvittavan viestien lähettämiseen ja prosessointiin vastaanottavassa päässä [KiC09]. RIM-mallin alkuversioon HL7-komitea kokosi useasta lähteestä luokkia ja asioita. Ne tiivistettiin yhdeksi malliksi, jota he alkoivat harmonisoida vuonna 1997 [Bee98]. Ensimmäisessä vaiheessa RIM-malliin koottiin jatkuvasti uusia luokkia ja attribuutteja, mutta tämä teki siitä huonosti ylläpidettävän ja laajennettavan. Toisessa vaiheessa malliin jätettiin vain muutama pääluokka ja ajattelumallia muutettiin oliopohjaisesta palvelupohjaiseen malliin, jonka keskiössä on aktit eli tapahtumat [OeB05].

RIM-viitemallin selkäranka on kolme pääluokkaa; akti (Act), rooli (Role) ja entiteetti (Entity). Näiden lisäksi pääluokkien välisiä suhteita kuvaavat kolme väliluokkaa; aktien suhde (ActRelationship), osallistuminen (Participation) ja roolin linkitys (RoleLink). Vain RIM määrittelee luokille sallitut yhteydet. Omia yhteyksiä ei saa lisätä HL7 versio 3 muotoiseen viestiin, mutta niitä saa jättää pois ja monistaa [Ben10]. Onnistuneen viestinvälityksen takeena on, että kaikki viestit perustuvat kaikkien saatavilla olevaan RIM-viitemalliin ja käytössä on yhteinen sanasto ja tietotyypit [OeB05]. Viestiä luodessa RIM-viitemallin määrittelemistä tietotyypeistä tehdään mahdollisimman erikoistuneita juuri määriteltävää viestiä varten.

Jokainen tapahtuma on akti, ja se kuvataan aktiluokan avulla. Aktilla voi olla useita osallistujia, jotka liitetään aktiin osallistumisluokan yhteydellä rooliluokkaan. Roolin haltijoita voivat olla entiteetit, esimerkiksi henkilö voi olla roolissa potilas tai hoitohenkilökuntaan kuuluva [Ben10].

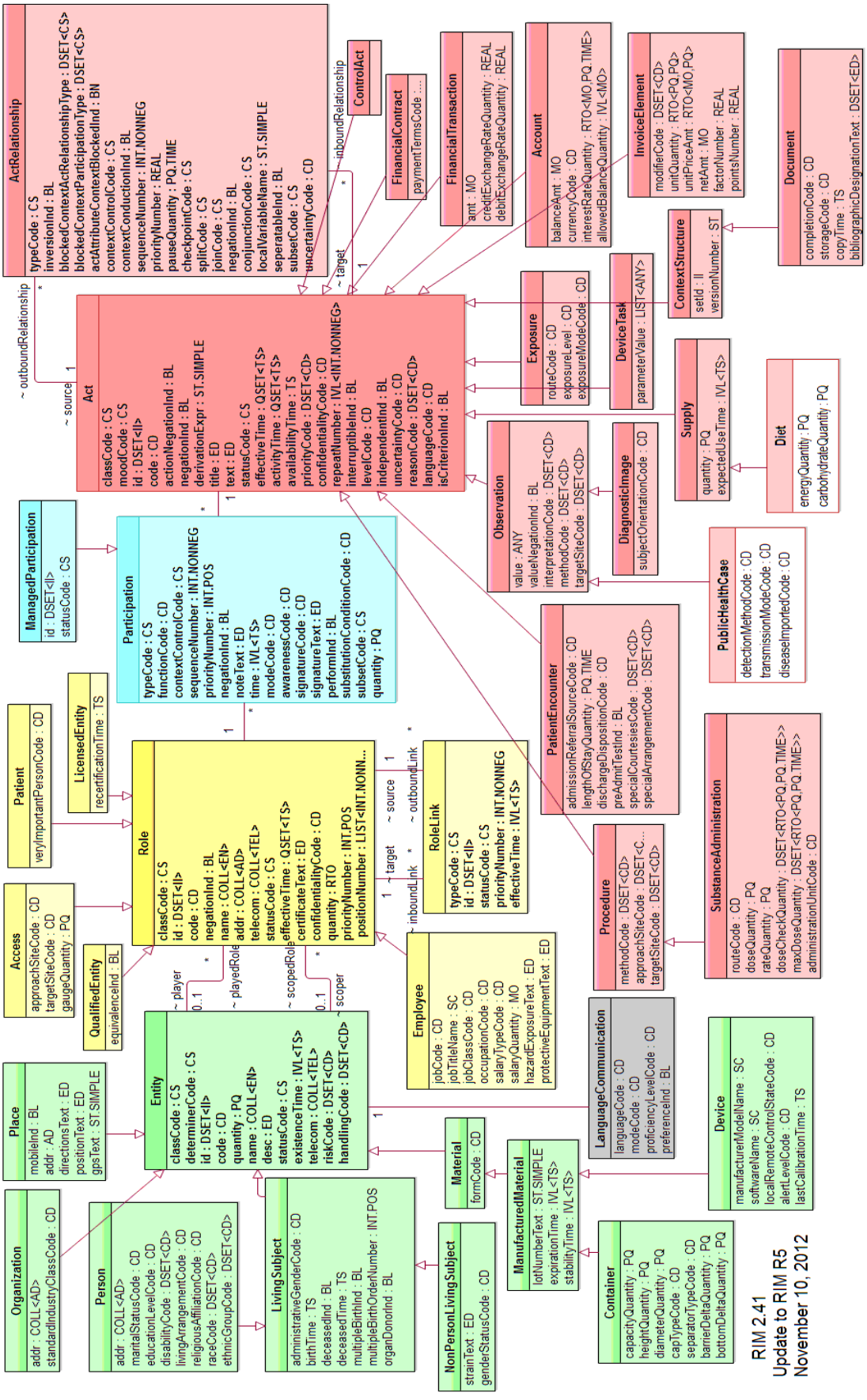
RIM-mallissa on kaksi aihealuetta. Normatiivinen aihealue kuvaa kieliopin ja kommunikaatioaihealue kuvaa viestien ja rakenteellisten dokumenttien muodon, jotta järjestelmät pysyvät välittämään tietoa keskenään [HDS04]. Kuvassa 5 on RIM-viitemallin normatiivinen sisältö. Siinä esiintyy edellä mainitut pääluokat ja näiden väliluokat. Eri väreillä on erotettu pääluokkien aliluokat. Esimerkiksi punaisella on aktiluokka ja sen aliluokat kuten havainto (observation) ja toimenpide (procedure) ovat vaaleammalla punaisella. Kuten olio-ohjelmoinnissa yleisesti aliluokka perii yläluokkansa kaikki attribuutit ja voi sisältää attribuutteja, joita yläluokalla ei ole. Tästä seuraa, että kaikki aliluokat ovat erikoistumisia yläluokasta [Ben10]. Luokkien ja attribuuttien nimet ovat niin sanotulla kamelityylillä

(camelcase). Tyylissä ihmisluettavan sanojen väli on korvattu yhdistämällä sanat ja aloittamalla jälkimmäinen sana isolla kirjaimella. Näin on tehty, koska XML-metakieli, johon RIM-mallista johdettu sanoma lopulta yleensä muutetaan, ei hyväksy väliä attribuuttien tai merkkien nimissä. Esimerkki kamelityylistä on aktiluokan aliluokka LääkkeenMääräänäminen (SubstanceAdministration) [Boo11].

RIM-mallin käytöstä on esimerkki potilaan pulssin mittaaminen. Se käyttää monia luokkia viitemallista. Potilas on henkilö, eli entiteettiluokan aliluokka. Potilas on ilmentymä potilasluokasta, joka on rooliluokan aliluokka. Henkilöluokalla on suhde potilasluokkaan, toisin sanoen henkilö on roolissa potilas. Lääkärin vastaanottokäynti on aktiluokan aliluokka. Pulssin mittaaminen on sen sijaan havainto, joka on sekin aktiluokan aliluokka. Potilaalla on suhde osallistumisluokan kautta lääkärin vastaanottoa kuvaavaan luokkaan. Havainto on linkitetty samaan vastaanottoon aktisuhdeluokan avulla [ETD07]. Jokaisella RIM-viitemallin pääluokalla on joukko aliluokkia. Ne tarkentavat pääluokkaa lisäämällä pääluokassa olevien attribuuttien lisäksi erikoistuneita attribuutteja. Esimerkiksi potilasluokalla on kaikki rooliluokan attribuutit, ja lisäksi sen itse määrittelemä attribuutti tärkeän potilaan koodi (veryImportantPersonCode) [Ben10].

RIM-mallin luokilla on rakenteellisia attribuutteja. Niiden avulla RIM on saatu kutistettua vain kuuden tärkeimmän luokan ympärille. Jokaisella aktiluokalla on attribuutit luokkakoodi (classCode) ja tilakoodi (moodCode). Ne kertovat millainen luokan ilmentymä on ja missä tilassa se on. Esimerkiksi havaintoluokalla luokkakoodi on OBS ja tilakoodi esimerkiksi EVN. Tämä kertoo, että ilmentymä on havainto, joka on tapahtunut. Rakenteellisten attribuutit auttavat viestien suunnittelussa, jotka toteutetaan myöhemmin tietokoneohjelmassa. Suunnittelijat määräävät rakenteellisten attribuuttien arvot, joita toteuttajat eivät saa muuttaa. Yleisesti erikoistumisluokat nimetään niiden rakenteellisten attribuuttien mukaan. Esimerkiksi aikaisemman havaintoesimerkin tapauksessa luokan nimeksi tulisi havaintoTapahtuma (observationEvent). Taulukossa 3 on esitetty kaikki pääluokkien rakenteelliset attribuutit.

HL7 versio kolmessa suunnittelija käyttää graafista esitystapaa muodostaessaan viestiä. Graafinen esitys on nimeltään RMIM (Refined Message Information Model), ja se näyttää värikoodatulla tavalla viestin rakenteen. RMIM-kaaviot ovat yleensä yhden sivun kokoisia, joten ihminen näkee yhdellä silmäyksellä mitä HL7-viesti sisältää [Ben10].



Kuva 5. RIM-viitemallin normatiivinen sisältö [RIM13].

Akti	luokkakoodi, tilakoodi, negaatio (negationInd) ja tasokoodi (levelCode)
Entiteetti	luokkakoodi ja määrääjäkoodi (determinerCode)
Rooli	luokkakoodi ja negaatio
Aktisuhde	tyyppikoodi (typeCode), käänteinen (inversionInd), sisältöhallintakoodi (contextControlCode), sisällön johtuminen (contextConductionInd) ja negaatio
Osallistuminen	tyyppikoodi ja sisältöhallintakoodi
Roolilinkki	tyyppikoodi

Taulukko 3. RIM-mallin luokkien rakenteelliset attribuutit [Ben10].

RIM-luokkakaavioon piirretään vain ne luokat, joilla on erikoistuneita attribuutteja pääluokkiin verrattuna. RMIM-kaaviota suunniteltaessa voi käyttää luokkia joita ei ole RIM-kaaviossa, mutta niillä ei saa olla RIM-mallin ulkopuolisia attribuutteja. Attribuutit, kuten tunniste (id), koodi (code) ja tilakoodi (status code) esiintyvät useissa luokissa. Sekaannusten välttämiseksi attribuuttien etuliitteenä käytetään luokan nimeä, esimerkiksi Rooli.id tai Akti.id. Tunnistetta käytetään luokan tunnistamiseen ja se voi olla olio- tai universaalitunniste (OID tai UUID). Yhdessä koodin kanssa, tunniste estää viestien moniselitteisyyden. Tunniste viittaa ihmiseen, organisaatioon tai asioihin, jokaisella näistä on yksiselitteinen tunniste.

RIM-mallissa käytettyjä koodeja on HL7-organisaation määrittelemiä ja ulkopuolisesta koodistosta saatavia. HL7-organisaation määrittelemät koodit määrittelevät viestin rakenteellisia ominaisuuksia, esimerkiksi luokkakoodi (classCode) määrittelee luokan nimen (akti, rooli tai entiteetti). Sen sijaan esimerkiksi koodiattribuutti määrittelee tarkasti millainen luokka on. Sen arvo tulee ulkopuolisesta koodistosta, joka määritellään omassa attribuutissaan. Koodi ja sen määrittelemä koodisto ovat yksiselitteisiä. Koodistoille on RIM-mallissa omat tietotyyppinsä. Näitä ovat CNE (coded with no exceptions), joka rajaa arvon takana olevan koodiston määrittelemäksi, eikä salli muutoksia siihen sanomassa. CWE (coded with exceptions) sen sijaan antaa mahdollisuuden lisätä lokaalikoodi tai

tekstiä tietotyyppiin [Ben10]. Esimerkiksi koodisto, eli aihealuesanasto, voi listata potilaan sukupuolet. Tällöin HL7 version 3 sanoman kenttä, jossa tietotyyppinä on CWE tai CNE, saa arvonsa suoraan sanastosta. CWE-tietotyypin tapauksessa vapaa teksti voi poiketa sanaston tekstistä, esimerkiksi koodiston tekstin male voi suunnittelija suomentaa viestissä mieheksi [BCC00]. Molemmat tietotyypit koostuvat yhdeksästä attribuutista: tunniste, teksti, koodiston nimi, vaihtoehtoinen tunniste, vaihtoehtoinen teksti, vaihtoehtoinen koodiston nimi, koodiston versiotunniste, vaihtoehtoinen koodiston versiotunniste ja alkuperäinen teksti. Tosin usein vain kolmea ensimmäistä attribuuttia käytetään. Vaihtoehtoiset koodit ovat tarpeellisia, jos vastaanottajat käyttävät eri koodistoja. Tällöin lähettäjä voi ilmaista saman asian kahdella eri koodistolla, ja kukin vastaanottaja voi valita ymmärtämänsä koodin [Ben10].

Esimerkiksi koodiattribuutti xml-muotoisena esityksenä näyttää seuraavalta: <code code="1" codeSystem="1.2.246.537.5.40105.2006" codeSystemName="Reseptisanoman tyyppi" displayName="Lääkemääräys"/>, jossa koodi on numeroarvo yksi, koodiston oliotunniste on 1.2.246.537.5.40105.2006, koodiston nimi on reseptisanoman tyyppi ja ihmisluettavanimi on lääkemääräys [Ere13]. Jokainen luokka voi sisältää vain yhden koodin. Jos viestiin tarvitsee lisätä useita koodeja, tulee jokaisen koodin olla omassa luokassaan ja luokkien linkitetty toisiinsa aktisuhdeluokan avulla [Ben10]. Toinen esimerkki ulkopuolisesta koodista on akti, entiteetti ja rooli luokan koodiattribuutti, joka voi saada arvonsa esimerkiksi SNOMED-CT-ontologiasta. Esimerkiksi potilaan verenpaineen mittaamista kuvaava SNOMED-CT-koodi on “O/E - Blood Pressure Reading 163020007”, jossa O/E tarkoittaa tutkimuksessa tehtyä (on examination) ja numerosarja koodin perässä SNOMED-CT-sanaston konseptitunnistetta (concept id). Verenpaineen mittauksella on useita attribuutteja, joista yksi on mittauskohta (finding site). Attribuutti saa arvonsa toisesta konseptista, esimerkiksi käsivarresta (structure of brachial artery, 17137000). Ulkopuolinen kaikkien vastaanottajien ja lähettäjän saatavilla oleva koodisto laajentaa HL7 version 3 tietosisältöä ja parantaa semanttista yhteensopivuutta järjestelmien välillä [Rya06].

HL7 RIM-viitemallissa on puhetoimituksia (speech act), jotka muuttavat asiantiloja. Ne kuvaavat mikä rooli terveydenhuollon informaatiolla on viestinvälityksen käynnistyksessä. Puhetoimitus koostuu ehdollisesta sisällöstä ja illokuutiosta, eli puhujan aikeesta. Ehdollinen sisältö kertoo millainen puhetoimitus on, kun sen sijaan illokuutio kuvaa mitä

puhetoimituksella yritetään saada aikaan. Puhetoimituksia ovat esimerkiksi pyyntö, kysymys ja lupaus. Puhetoimituksen illokuutio riippuu sen tekijän ja vastaanottajan suhteesta. Jos suhde ei ole pätevä, ei illokuution vaikutus tapahdu. Esimerkiksi potilas ei voi määrätä lääkärille lääkettä, mutta toisin päin tämä on mahdollista. RIM-mallissa ehdollinen sisältö sisältyy aktiluokan kuvaaviin attribuutteihin tai osallistumisluokan viittauksiin. Illokuutio sen sijaan sisältyy reagoimattomiin (inert) ominaisuuksiin, esimerkiksi tilakoodiin tai tekijän osallistumissuhteeseen [SMW06].

5.2 Viitemallin pääluokat

Pääluokkia ovat siis akti, rooli ja entiteetti. Tässä luvussa käsitellään niitä tarkemmin. Jokaisen pääluokan voi erikoistaa tarkemmaksi niiden aliluokkien avulla [KiC09].

Aktiluokka on jokin tapahtuma, joka on tapahtunut tai voi tapahtua [Ben10]. Akti on tekemistä eli se on kuin verbi. HL7 on määritellyt noin kaksikymmentä eri aktia RIM-malliin [Boo11]. Täydellinen akti kertoo millainen tapahtuma on, kuka sen on tehnyt ja mikä on sen kohde. Esimerkiksi lääkäri havaitsee painon potilaasta, missä lääkäri on tekijä, kohde potilas ja tapahtuma on painon mittaaminen. Tarkentavia tietoja ovat missä, milloin, miten, miksi ja mitä varten. Esimerkiksi painon mittaaminen tapahtui lääkärin vastaanotolla Testi Terveysasemalla (missä) 3.3.2013 kello 12:00 (milloin). Mittaaminen tehtiin vaa’alla (miten), jotta saataisiin tietää potilaan paino (miksi), koska lääkäri halusi tietää onko potilaalla ylipainoa (mitä varten).

Aktiin voi liittyä useita muita akteja, ja ne liitetään aktiin aktisuhdeluokan avulla. Akteja voi olla useita erilaisia. Akteja ovat tapahtumat, havainnot, ilmoitukset, lääkkeiden tai muiden käytettävien asioiden määräämiset ja toimitukset sekä taloudellisia toimenpiteitä. Tapahtumat ovat esimerkiksi kohtaamisia, käyntejä tai tapaamisia. Havaintoja sen sijaan ovat tutkimukset, diagnoosit ja tutkimustulokset [Ben10]. Aktiin osallistuvat henkilöt tai asiat liitetään aktiin osallistumisluokan avulla. Osallistumisia voi olla nollasta ylöspäin yhtä aktia kohden. Osallistumisluokka on aktin ja roolin yhdistävä luokka [Boo11].

Luokkakoodi (classCode) määrittää mihin kategoriaan akti kuuluu [Ben10]. Esimerkiksi Suomen sähköisessä lääkemääräyksessä luokkakoodi on SBADM (substance administration) ja toimituksessa SPLY (supply) [Ere13]. Aktiluokan toinen rakenteellinen attribuutti on tapaluokkakoodi (moodCode). Tapaluokka on verbin taivutuskategoria, ja se kertoo onko verbin esitys tositapa (indikatiivi), käskytapa (imperatiivi), ehtotapa (konditionaali)

tai mahtotapa (potentiaali). Tapaluokkakoodi määrittää onko tapahtuma tapahtunut, pyydetty tapahtumaan, tavoite tai kriteeri tapahtumalle. Esimerkiksi paino on 100 kg, on tapahtunut havainto. Painon mittaaminen päivittäin on pyyntö tapahtumalle. Tiputa tai nosta painoa 20 kg on tavoite. Tiputa tai nosta painoa, jos paino ei ole tavoitepainossa on kriteeri. Tapaluokkakoodilla erotellaan jo tapahtuneet asiat (tapahtuma) tapahtuvista (pyyntö). Määritelmä on yleisohje tapahtumalle, jota ehdotus voi muokata potilaalle sopivaksi, mistä lopuksi tulee tapahtuma, eli se mitä oikeasti on tehty. Kaikki tapaluokkakoodit on listattu taulukossa 4.

EVN	tapahtuma (event)	Tapahtunut tapahtuma, joka voi olla meneillään oleva tai historiatieto.
RQO	pyyntö (request)	Suora pyyntö tai tilaus tapahtumalle tekijältä toteuttajalle.
PRMS	lupaus (promise)	Lupaus tapahtuman täyttämiseksi.
PRP	ehdotus (proposal)	Ei pakollinen tilaus tapahtumalle.
DEF	määritelmä (definition)	Tapahtuman määritelmä.

Taulukko 4. Tapaluokkakoodin arvot [Ben10].

Tilakoodi määrittelee aktin tilan. Niitä ovat uusi, aktiivinen, valmis, peruutettu ja keskeytetty. *Uusi* on tapahtuma, joka on vasta luotu ja sille ei ole vielä tehty mitään. *Aktiivinen* on suorituksessa oleva tapahtuma. *Valmis* tapahtuma on suoritettu. *Peruutettu* tapahtuma on hylätty ennen kuin sille on ehditty tehdä mitään. *Keskeytetty* tapahtuma on lopetettu ennen sen valmistumista.

Aktilla on kaksi aikaa kuvaavaa attribuuttia; toiminta-aika (activityTime) ja tosiasiallinen aika (effectiveTime). Esimerkiksi potilaan ajanvarauksessa toiminta-aika on aika, jolloin ajanvaraus on tehty ja tosiasiallinen aika on aika, jolloin tapaaminen itse on. Yleisesti toiminta-aika on aika, jolloin tapahtuma luodaan ja tosiasiallinen aika se, jolloin tapahtuma tulee suorittaa.

Aktin erikoistumisia ovat esimerkiksi havainto (observation), toimenpide (procedure), aineen määrääminen (substance administration), toimittaminen (supply) ja potilaan tapaaminen (patient encounter). *Havainto* on kohteesta tiedon huomaamista ja tunnistamista. Siihen voi liittyä mittaamista, tutkimusvälineitä tai se voi olla vain toteamus, kuten diagnoosi. *Toimenpide* on tapahtuma, jonka välitön jälkitila on kohteen fysikaalinen muutos. Esimerkiksi leikkaus. *Aineen määrääminen* on aineen käyttöönottoa kohteelle, kuten lääkemääräyksen kirjoittaminen. Lääkemääräyksessä tapaluokkakoodi on lupaus ja lääkehistorian listaamisessa se on tapahtuma. *Toimittamisella* tarkoitetaan konkreettisen materiaalin siirtämistä toiselta entiteetiltä toiselle. *Potilaan tapaaminen* on potilaan ja hoitoa tarjoavan tahon kohtaaminen tarkoituksenaan tarjota terveyteen liittyviä palveluita. Ajanvaraukset kuuluvat tähän kategoriaan, jolloin niiden tapaluokkakoodi on lupauksia, ja tapahtuneiden käyntien tapaluokkakoodi on tapahtuma.

Entiteettiluokka kuvaa kaikkea elävää ja ei-elävää asiaa [Ben10], ne ovat fyysisiä tietoolioita, jotka esittävät rooliluokan määrittelemiä luokkia [OeB05]. Esimerkiksi henkilöt, eläimet, organisaatio tai muut olemassa olevat asiat kuuluvat tähän ryhmään. Entiteetillä on kaksi rakenteellista attribuuttia luokkakoodi ja määrittäjäkoodi (determiner code). Luokkakoodi määrittää entiteetin, eli kertoo mikä se on. Määrittäjäkoodia käytetään erottamaan kaksi samantyyppistä entiteettiä toisistaan. Entiteetit voivat olla joko suoraan rooliluokan toteuttajia tai ne voivat määritellä laajuuden, jossa rooli on voimassa. Esimerkiksi lääkäri Virtanen on suoraan roolissa lääkäri, mutta Virtasen työnantajaorganisaatio määrittelee roolin laajuudeksi ammatinharjoittajan organisaation sisällä.

Entiteetin neljä pääaliluokkaa ovat elävä kohde, materiaali, paikka ja organisaatio. Henkilö on elävän kohteen aliluokka ja siten myös entiteetin aliluokka. Henkilö perii attribuutteja niin elävältä kohteelta kuin entiteetiltäkin. Entiteettiluokalla on attribuutteja, jotka ovat määritelty myös rooliluokalle. Esimerkiksi tällaisia attribuutteja ovat tunniste, koodi, nimi, osoite, puhelin, tilakoodi ja määrä. On viestin suunnittelijan tehtävä määritellä kumman luokan attribuutteja viestissä tulee käyttää. Sääntönä on, että pysyvämmät tiedot ovat entiteettiluokassa ja vain viestin aikana voimassa olevat rooliluokassa, koska entiteetti kuvaa olemassa olevaa asiaa ja rooli sen hetkistä tilaa [Ben10].

Entiteetit toimivat eri rooleissa. Rooli antaa entiteetille pätevyyden suorittaa jokin tapahtuma ja sen mukana määrittää puhetoimituksen illokaation vaikutuksen. Entiteetti voi olla roolin kautta esimerkiksi potilas, palveluntarjoaja, työntekijä, määrääjä tai ammatinhar-

joittaja [OeB05]. Rooli kuvaa kahden entiteetin suhdetta. Toinen on tekijä (player) ja toinen konteksti (scoper). Tekevä entiteetti osallistuu tapahtumaan kontekstin antavan roolin antamassa laajuudessa. RIM määrittää viisi roolityyppiä. Perustyypin lisäksi niitä ovat työntekijä, potilas, pätevä tekijä ja pääsy (access) [Boo11].

6 Viitemallin profilointi RMIM-kaavioksi

HL7 version 3 RIM-viitemalli on kattava luokkakaavio [Ben10] ja se sisältää kaiken mitä HL7 versiolla 3 voidaan sanoa [Boo11]. Ideana versiossa 3 on rajoittaa ja tarkentaa viitemallia luomalla alijoukko mallin määrittelemistä luokista ja attribuuteista [Ben10]. Kanssakäymismallit (interaction models) määrittelevät HL7-viestien käyttämän järjestelmän käyttäytymistä. Ne käynnistyvät herätetapahtumien käynnistämänä [Bee98]. Alijoukon luominen on yleinen tapa standardien maailmassa, ja alijoukkoa kutsutaan profiiliksi [Ben10]. Profiili on RIM-mallista rajoitettu tietomalli tiettyä käyttötapausta varten [PuP12], ja se takaa käyttötapauksensa yhteensopivuuden tarjoamalla valmiin alijoukon viestin toteuttajille. Standardeissa on iso joukko valinnaiseksi määriteltyjä asioita, joita toteuttajat ottavat käyttöön vaihtelevasti. Jos toimittajat toteuttavat eri alijoukon, ei toteutusten yhteentoimivuus ole taattu [Ben10].

HL7 versiossa 3 profiiliksi rajoittaminen alkaa RIM-viitemallista ja jatkuu hierarkkista puuta alaspäin aina tarkempaan profiiliin. RIM-mallista seuraavat mallit ovat laajimmasta suppeimpaan: DMIM (Domain Message Information Model), RMIM (Refined Message Information Model), HMD (Hierarchical Message Description) ja MT (Message Type). Näistä käytetyin on RMIM, joka on viestin kuvaava kaavio. Siinä on yksi aloituskohta ja se esitetään sarjallistuvassa muodossa [Ben10]. RMIM-kaaviossa esitetään väliaikaisia ja toiminnallisia ehtoja kanssakäymiskaaviolla [OeB05].

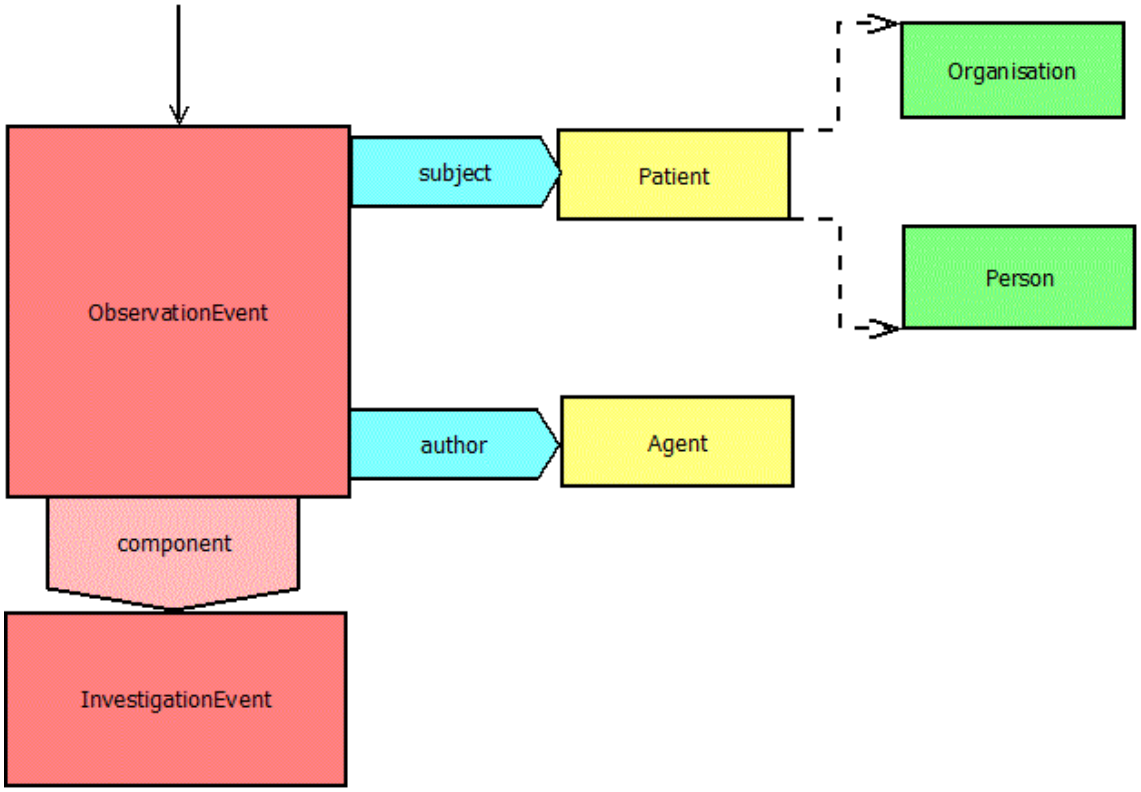
RIM-mallia suunnittelija voi rajoittaa poistamalla tai monistamalla. Suunnittelija voi jättää tuomatta RMIM-kaavioon luokkia ja attribuutteja RIM-mallista. Viitemallin rakenteellisten attribuuttien, kuten luokka- ja tapaluokkakoodien, on pakko olla mukana kaaviossa, mutta muuten on suunnittelijan valinta mitä luokkia ja attribuutteja hän jättää pois. Suunnittelija voi monistaa saman RIM-luokan useaan kertaan kaavioon. Luokkaa kutsutaan kaaviossa klooniksi. Kloonaamisen mahdollistamisella itse RIM on saatu mahdollisimman pieneksi. Poistamisen ja kloonaamisen jälkeen suunnittelija voi määritellä luokkien ja attribuuttien määrän ja vapaaehtoisuuden, esimerkiksi suunnittelija voi määritellä

yhden pakollisuuden asettamalla attribuutin määräksi 1..1. Vapaaehtoinen attribuutti esiintyy 0..1 kertaa. HL7 versio 3 käyttää pakollisuudesta termiä tulee (shall), suosituksesta termiä pitäisi (should) ja vapaaehtoisuudesta termiä voi (may). Standardi tunnistaa näiden lisäksi termien negaatiot, ei tule, ei pitäisi ja ei voi.

Suunnittelija voi rajoittaa attribuuttien tietotyyppejä. HL7 version 3 tietotyypit ovat määritelty hierarkkiseen tyyliin yksinkertaisesta monimutkaiseen. Hierarkkinen tietotyyppi on esimerkiksi CS (yksinkertainen, code simple), CV (arvollinen, code value), CE (vastaavainen, code with equivalents) ja CD (käsitteellinen, concept descriptor). Monimutkaisemman tietotyypin voi rajoittaa yksinkertaisempaan. Suunnittelija voi määrittää koodauksen, eli minkä koodiston arvoja käytetään kaaviossa. RMIM-kaavion luomisessa mukaan otetaan vain ne RIM-viitemallin ominaisuudet, joita tarvitaan käyttötapauksessa. Tavoite on saada kaavio mahdollisimman yksinkertaiseksi yhteensopivuuden varmistamiseksi [Ben10].

Kuvassa 4 on esitetty yksinkertaistettu RMIM-kaavio. RMIM-kaavio käyttää HL7-organisaation omaa UML-kuvauskieltä. Siinä jokaisella luokalla on oma värinsä ja muotonsa. RIM-mallin pääluokkien välisiä luokkia kuvataan UML-kielen yhteysluokalla (association class), joka kuvaa kahden luokan välistä suhdetta tarkalla tasolla [Boo11]. Kaavion luku aloitetaan nuolen osoittamasta sisääntulopisteestä. Kaavion väritys on vastaava RIM-viitemallin määrittelemien luokkien värien kanssa. Sisääntulopisteen osoittaman aktin nimi voi olla mikä tahansa, mutta käytäntö on nimetä se luokka- ja tapaluokkakoodien mukaan [Ben10]. Koska RMIM-kaavio myöhemmin muutetaan XML-skeemaksi, on nimeäminen pakollista. XML-skeemassa samannimisten elementtien rakenne tulee olla samanlainen, siksi esimerkiksi kaikkia eri rakenteellisia aktiluokan aliluokkia ei voida nimetä aktiksi. Vaikka RMIM-kaaviossa olevat luokat ovat kopioita, eivät ne ole kuitenkaan täysin samanlaisia viitemallin vastineidensa kanssa. Tämä johtuu profiloinnin sallimasta rajoittamisesta [Boo11]. Esimerkki nimeämiskäytännöstä on havainto, joka on tapahtunut (luokka OBS ja tapaluokka EVN). Kuvassa 6 se on nimeltään ObservationEvent. Havainnon pakolliset attribuutit ovat tunniste, koodi sekä tapahtuma-aika, niitä ei ole piirretty kuvaan. Havainto on esitetty kuvassa aktien värityksen mukaisesti tummanpunaisella. Havainnolla voi olla useita aktiluokan tutkimustapahtumia (investigation event). Tutkimustapahtumat liitetään havaintoon komponenttityyppisellä aktisuhteella, kuvassa 6 vaaleanpunaisella. Havainnolla voi lisäksi olla osallistumisia, jotka liitetään

havaintoon RIM-mallin osallistumisluokan avulla, kuvassa vaaleansininen. Osallistumisluokalla on suhde rooliluokkaan, kuvassa keltainen. Osallistuminen voi olla esimerkiksi tekijä (author) tai kohde (subject) ja roolina sillä on potilas tai terveydenhuoltohenkilökuntaan kuuluva. Rooliluokalla voi olla suhde entiteettiluokkaan, joka määrittelee esimerkiksi potilaan olevan henkilö tai palveluita tarjoavan organisaation. Entiteettiluokkaa kuvaava väri on vihreä [Ben10].



Kuva 6. Yksinkertaistettu RMIM-kaavio.

Kaikilla RMIM-kaavion luokilla voi olla attribuutteja. Attribuutteja on kolmea eri tyyppiä; vapaaehtoisia, vaadittuja ja pakollisia. Vapaaehtoiset attribuutit voi jättää pois muodostetusta sanomasta tai ne voivat olla sanomassa arvonaan tyhjäarvo (null). Vaadittu attribuutti tulee olla aina luokan mukana, mutta sen arvo voi olla tyhjäarvo. Pakollinen attribuutti on kuten vaadittu, mutta sen arvo ei saa olla koskaan tyhjäarvo. RMIM-kaaviossa vaadittujen attribuuttien perässä on tähti ja pakolliset on sen lisäksi lihavoitu. Attribuutin nimen perässä on sen sallittu tietotyyppi, sallitut esiintymiskerrat sekä oletusarvo. Esimerkiksi kuvassa 7 functionCode on vapaaehtoinen attribuutti, jonka sallittu tietotyyppi on CE (coded with equivalents), esiintymiskerta on nollasta yhteen ja oletusarvo ParticipationFunction. ContextControlCode sen sijaan on pakollinen attribuutti ja Time-attribuutti on vaadittu. FunctionCode ja contextControlCode attribuuttien tietotyyppien CE ja

CS perässä on suositeltu käytettävä sanasto. CWE-akronyymi tarkoittaa koodattua poikkeusten kanssa (coded with exceptions) ja CNE ilman poikkeuksia (coded no exceptions) [Boo11].

functionCode: CE CWE [0..1] <= ParticipationFunction **contextControlCode***: CS CNE [1..1] <= ”OP” time*: TS [1..1]

Kuva 7. RMIM-kaavion luokan attribuuttien vaihtoehdot [Boo11].

Koodistoista ei ole hyötyä, jos niitä ei ole tallennettu ja jaettu julkisesti kaikkien saataville. Koodistot tallennetaan esimerkiksi tietokantaan, mistä sanomia käsittelevä ohjelma pystyy tarkistamaan ja viittaamaan sanoman tietosisällön tietoihin. Esimerkiksi potilaan sukupuoli, sanomaa luodessa, muutetaan koodiston mukaiseksi tietojärjestelmän arvosta [BCC00]. Suomessa käytetty Koodistopalvelu [THL13] tarjoaa Suomen sisällä käytetyissä tiedonvälityksissä, kuten sähköinen resepti ja potilasarkisto, käytettävät koodistot. Koodistopalvelusta saa ladattua koodistot potilasjärjestelmään eri muodoissa, esimerkiksi XML-tiedostona, tekstitiedostona tai taulukkotiedostona. Potilasjärjestelmän vastuulle jää tiedon tallennus ja päivittäminen koodistopalvelusta uusimpaan versioon.

6.1 Muut RIM-mallin rajoitetut mallit

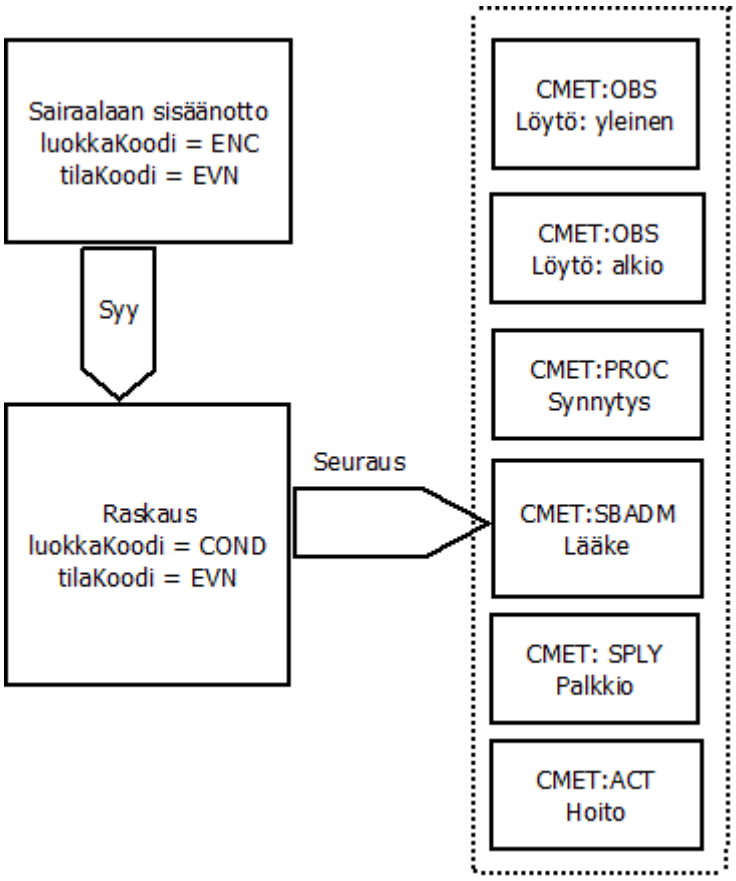
HL7 versio 3 malli koostuu neljästä kategoriasta. Ne ovat RIM-viitemalli, DMIM-aihealuemalli, RMIM-kaavio ja HMD-kuvaus [KKL12]. Niiden toteutus alkaa UML-käyttötapausten tai tekstimuotoisen kuvauksen, jota kutsutaan HL7-kuvakäsikirjaksi (storyboard), kehittämisellä ja loppuu valmiiseen sanomaan [BEP06].

RIM-viitemallia rajoittamalla suunnittelija saa tarkemman kuvauksen sanomasta ja aihealueesta. DMIM (domain message information model) on kuvaus aihealueesta HL7-notaatiolla [Ben10]. Se esittää RIM-viitemallin alijoukon yhdestä aihealueesta [KKL12], ja se sisältää RIM-mallin luokkien klooneja, attribuutteja, tiloja sekä suhteita [VAN11]. Kaavioon valittavat luokat ja attribuutit riippuvat lakien, organisaation, toimintojen ja tekniikoiden määrittelemistä vaatimuksista, kuten organisaation menettelytavoista, säännöistä ja tietämyksestä [BEP06]. DMIM voidaan rakentaa joko ylhäältä alas (top-down) tai alhaalta ylös (bottom-up). Ensimmäisessä tietosisältö johdetaan määrittelyjoukosta ja jälkimmäisessä tietosisältö määräytyy yksittäisten sanomien sisällöstä. DMIM-kaavion

luomiseen pätee samat rajoitussäännöt kuin RMIM-kaavioon ja se voidaan piirtää graafiseksi kaavioksi käyttäen samoja tekniikoita kuin RMIM-kaaviossa [Ben10] kuvassa 4. Vaatimusten mukaiset luokat valitaan, ja niiden attribuutit asetetaan vastaamaan asetettuja vaatimuksia valitsemalla pakolliset attribuutit, jättämällä pois tarpeettomat ja lisäämällä tarpeelliset [BEP06]. DMIM-kaaviosta johdetaan RMIM-kaavio [VAN11]. RMIM-kaavio mahdollistaa useiden viestien luomiseen yhdestä aihealueesta [KKL12]. DMIM ei ole hierarkkinen eikä sarjallistuva, eli sitä ei voi käyttää suoraan sanoman toteutuksessa. DMIM-kaavion luominen ei ole pakollista viitemallin profiloinnissa, sen sijaan suunnittelija voi luoda suoraan RMIM-kaavion [Ben10]. Jos se luodaan, RMIM-kaaviota luodessa siitä otetaan tarvittava alijoukko luokkia. Näistä luokista muodostetaan sarjallistuva kaavio, jonka attribuuttien koodistot ovat sidottuina. Koodistojen sitomisen voi tehdä aikaisemminkin, mutta tässä kohtaa viimeistään [BEP06].

Alankomaiden kansallinen yhteentoimivuusprojekti NICTIZ [Goo04] on luonut aihealuetietomalleja perinatologian, eli vaikeiden synnytysten, alalle. Lisäksi he ovat käyttäneet siitä saatua tietomallia muiden aihealueiden pohjana. Heidän aihealuetietomallin luontinsa on kolmivaiheinen: aihealueen tiedon analysointi, mallinnus ja toteutus. *Tiedon analysointi* koostuu useista erillisistä osista, joihin kuuluu lääketieteenammattilaisten haastatteluja hoitotapausten selvittämiseksi. Tapauksista tehdään kuvakäsikirjoja, jotka kuvaavat reaalimaailman tapahtumia. Kuvakäsikirjoista tehdään interaktiotaulu, joka kuvaa lähettäjän ja vastaanottajan käymää viestinvälitytystä. Se sisältää herätetapahtumat ja siirrettävän tiedon. Päällekkäisyyksien välttämiseksi samankaltaiset viestit ryhmitellään yhteen ja niiden sisältöä verrataan RIM-viitemalliin. Toisessa vaiheessa ensimmäisen vaiheen tietomallia kloonataan ja tarkennetaan RIM-mallista. Reaalimaailman tapausten konteksti pysyy samana suhteiden ja luokkien nimeämisen avulla. Tuloksena on neljä kerroksinen malli. Mallin toisessa kerroksessa on stereotyyppinen malli, jota tarkennetaan yksityiskohdilla myöhemmässä vaiheessa. Stereotyyppinen malli kuvaa DMIM-tyylisesti yhteyksiä luokkien välillä, niiden attribuuttien arvoja ja uudelleen käytettäviä CMET-elementtityyppejä (common message element type). Nelikerroksista mallia voi käyttää muiden aihealueiden vastaavien tekemiseen. Sen avulla voi havainnollistaa mitä eroja aihealueilla on ja mihin muutokset tulevat [Goo04]. Kuvassa 8 on osa nelikerroksista mallia. Siinä sairaalaan sisäänoton syy on raskaus, jonka seurauksena käytetään jotakin CMET-elementtityyppiä tapauksesta riippuen. CMET-tyypit ovat kuvassa havainto (OBS), toimenpide (PROC), lääkkeenmääräämineen (SBADM), toimitus (SPLY) ja akti

(ACT). CMET-tyyppien nimet tulevat RIM-viitemallin aktiluokan aliluokkien nimien lyhenteistä.

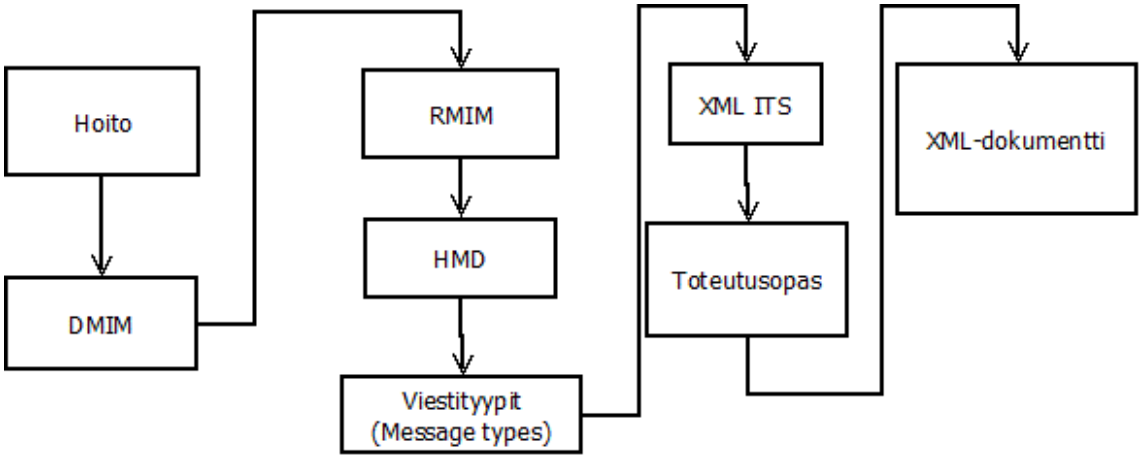


Kuva 8. Perinatologian nelikerroksisen mallin osa [Goo04].

Luokat ja niiden attribuutit ja suhteet voivat olla yleiskäyttöisiä. Tällöin DMIM- tai RMIM-kaaviosta tehdään yleinen viestin elementtityyppi, CMET. CMET on käytettävissä kaikissa aihealueissa tarpeen mukaan, ja se on komponenttina poistettava tai korvattava paikallisesti. Muokattavuus tekee spesifikaatiosta ylläpidettävän ja skaalautuvan. CMET-komponentit mallintavat käsitteitä ja tietämystä, mikä mahdollistaa järjestelmien välisen yhteensopivuuden käsitetasolla [BEP06].

DMIM-mallista saatavasta RMIM-mallista johdetaan HMD-kuvaus, joka kuvaa RMIM-mallin tietosisällön rakenteellisessa ja järjestetyssä muodossa [VAN11]. RMIM ja HMD sisältävät saman tietosisällön, mutta RMIM on tiedosta graafinen esitys ja siksi ihmisluettavampi [Ben10]. HMD-kuvauksen voi siirtää järjestelmästä toiseen, vaikka järjestelmien alusta on erilainen. HMD-kuvaus muuttaa kaksiulotteisen RMIM-mallin kuvauksen yksiulotteiseksi sarjoittuvaksi kuvaukseksi [VAN11]. HMD-kuvauksesta voidaan johtaa yhtäläinen XML-skeema [Bee98] käyttäen HL7-standardiin sopivia työkaluja [BEP06].

Kuvassa 9 on yksinkertaistettuna sanoman luomiseen vaadittava prosessi. Prosessi alkaa tarpeesta tehdä kliinisen tapahtuman mukainen sanoma. RIM-viitemallia käyttäen tapahtumasta tehdään DMIM-aihealuekaavio, josta johdetaan RMIM ja siitä HMD-kuvaus. HMD-kuvauksesta johdetaan XML-skeema viestityyppien kautta. Viestityypit (message type) ovat viestin tarkka kuvaus, jota voidaan käyttää viestin siirrossa [Ben10]. XML-skeeman tueksi sanoman suunnittelijat tekevät toteutusoppaan tai mallinteet, jotka tarkentavat XML-dokumentin toteutusta. Lopuksi saadaan järjestelmien välinen yhteensopiva XML-dokumentti.



Kuva 9. HL7-standardin komponenttien yhteydet [BEP06].

6.2 RMIM-kaaviosta XML-skeemaan

HL7 versiossa 2 siirrettävä tieto ei alun perin ole XML-muotoista tietoa. Skeeman luomiseksi on kuitenkin kehitetty keinoja muuttaa HL7 versio 2 muotoinen sanoma XML-skeemaksi.

Ensin jokainen version 2 segmentti irrotetaan omaksi osiokseen ja niille määritellään pakollisuus ja kardinaalisuusmääritykset. Sen jälkeen jokaisen osion kentän kohdalla tehdään samat määritykset osion sisällä [DRB98]. Prosessi etenee siis rekursiivisesti. Kuvassa 10 on esimerkki miten HL7 versiossa 2 määritellyt osion kentät muutetaan XML-skeeman mukaiseksi.

HL7 version 3 mukaiset sanomat perustuvat käyttötapauksiin tai skenaarioihin. RIM määrittää tietosisällön semantiikan niin, että lähettäjä ja vastaanottaja ymmärtävät sisällön samalla tavalla. Versiossa 3 on kehitetty algoritmi, joka muuttaa hierarkkisen viestinkuvauksen (HMD) XML-skeemaksi [DRB98]. Hierarkkinen viestinkuvaus on RMIM-kaa-

vion taulukkomainen esitysmuoto [Ben10]. Valmiita työkaluja RMIM-kaavion muuttamiseksi HMD-taulukoksi tarjoaa esimerkiksi RoseTree [Ros13]. Siinä käyttäjää ohjataan graafisen prosessin läpi, minkä lopputuloksena on HMD-kuvaus [OeB05].

<!ELEMENT WDN (WidgetId, WidgetName+, WidgetNickName*, WidgetProd?)>	Kentät HL7 versiossa 2. WidgetId, pakollinen, ei-toistuva WidgetName, pakollinen, toistuva WidgetNickName, ei-pakollinen, toistuva WidgetProd, ei-pakollinen, ei-toistuva

Kuva 10. Esimerkki HL7 versio 2 mukaisesta XML-skeemasta [DRB98].

Ensimmäinen vaihe HMD-taulukon luomiseksi on oikeanpuoleinen osuus taulukosta. Se kerää RMIM-kaavion luokat, attribuutit ja suhteet. Siinä kuvataan suhteen tyyppi, kuten perimis-, omistamis- tai tyyppisuhde. Osion tunniste on luokan tunniste, esimerkiksi potilas luokkaa kuvaa lyhenne PT ja henkilöluokkaa PER. Olionäkymä ja attribuutit on sisennetty kuvaussanoman sisältämistä luokan nimistä, niiden attribuuteista ja tietotyypeistä. Esimerkiksi henkilöllä voi olla attribuutteina nimi, joka on tyyppiä PN (person name), ja syntymäaika tietotyyppiä DT (datetime). Kuvaus tarkentaa mihin luokkaan luokalla on suhde, esimerkiksi henkilöluokka, joka on potilaan yläluokka. Rivinumerot selkeyttävät rivien järjestystä. Taulukossa 5 on alkua oikeanpuoleisesta HMD-taulukosta.

Suhde	Osion tunniste	Nimike (label)	Olionäkymä ja attribuutit	Kuvaus	Rivinumero
juuri isa	PT PER		Patient Person[Pt] PN name DT birthd	Potilas	1 2 3 4

Taulukko 5. Oikeanpuolinen osa HMD-taulukkoa [Bee98].

HMD-taulukon vasemmalle puolelle lisätään sanoman muoto. Jokaiselle riville lisätään tieto onko rivi pakollinen, vapaaehtoinen ja kuinka monta kertaa se voi esiintyä. Lisättävät kentät oikeanpuoleiseen HMD-taulukkoon ovat milloin ehto on voimassa, vaadittava arvo, pakollisuus ja kardinaalisuus. Esimerkki vasemmanpuoleisesta osasta HMD-taulukkoa on taulukossa 6 [Bee98].

Ehto voimassa	Vaadittu arvo	Pakollisuus	Kardinaalisuus
		kyllä	1..1
	IPT	ei	0..1

Taulukko 6. Vasemmanpuoleinen osa HMD-taulukkoa [Bee98].

SGML (Standard Generalized Markup Language) on metakieli, jonka alijoukko XML-metakieli on. Siitä saadut tulokset ovat siis verrannollisia XML-kieleen. Dolin ja kumppanit ovat artikkelissaan [DRB98] tutkineet miten hierarkkisen viestinkuvauksen tiedot voidaan esittää SGML-kielen määrittelemällä skeemalla. Heidän tuloksensa on, että muut paitsi ehdollinen olemassaolo on mahdollista esittää suoraan skeemassa. Taulukossa 7 on esitetty tarkemmin heidän saamiaan tuloksia. Niistä selviää, että SGML-metakielen määrittelemä skeema ja siis näin ollen XML-kielen määrittelemä skeema vastaa HDM-kuvauksen vaatimuksia.

-Kardinaalisuus: SGML-kielessä voi määritellä esiintymiskerrat. -Tietotyyppi: SGML-elementti mallinnetaan siten, että sen sisältömalli on tietotyyppi komponentti.
-Sisäkkäisyys: Määrittää SGML-kielen elementtien hallinnan. -Rakenne: Valinnat, listat ja ryhmät mallinnetaan säiliöelementteinä. -Osioiden ja attribuuttien nimet: Näistä tulee SGML-elementtejä. -Viittaukset muihin kenttiin: Elementin tietotyyppinä kyseinen viitattu kenttä. -Kenttien arvojen rajoitukset: SGML-kielen attribuuttien arvojen rajoitukset.
-Ehdollinen olemassaolo: Ei onnistu. -Pakollinen arvo: Voidaan mallintaa SGML-kielen attribuutin arvorajoituksilla. -Sisällyttäminen (inclusion): Määritellään SGML-kielen esiintymiskerta ilmaisimilla. -Toistot: Määritellään SGML-kielen esiintymiskerta ilmaisimilla.

Taulukko 7. HDM-kuvauksen kuvaaminen SGML-skeemalla [DRB98].

XML-dokumenttiin merkitään suhteita RIM-mallin määrittelemien luokkien välille luokissa olevien kenttien avulla. Esimerkiksi tapahtuman aiheuttaja, joka on toinen tapahtuma, voidaan esittää aktisuhdeväliluokan avulla. Aktisuhde saa tyyppikoodikseen jonkin

HL7-organisaation valmiiksi määritellyistä toteamussuhteista, jotka ovat määritelty CDA-arkkitehtuurissa [DAB06].

HL7 version 3 tuottaman XML-dokumentin tietotyypit ja rakenne määritellään XML ITS –määrittelyssä. Se noudattaa XML-standardia ja HDM-kuvauksen määrittelemää rakennetta. Jokaista HL7 version 3 viestityyppiä vastaa HDM-kuvaus ja XML-skeema, joka määrää XML-dokumentin rakenteen säännöt. XML-skeema sisältää kaiken tarvittavan tiedon XML-dokumentin luomiseksi [VAN11], mutta sen oikeellisuuden tarkistamiseen tarvitaan enemmän.

XML-skeema luodaan XML ITS –määrittelyn avulla RIM-mallista. RIM-viitemallista profiloinnin tuloksena saadusta HMD-kuvauksesta luodaan skeema algoritmisesti. Jokaisesta RIM-mallin luokasta tulee XML-skeeman tyyppi. Jokainen skeeman tietotyyppi ja HL7 version 3 aihealuesanastot esitetään omana tyyppinään skeemassa. HMD-kuvauksen viestityypit ovat elementtejä XML-skeemassa, ja niillä on kuvauksen mukaiset attribuutit. XML-skeemaan nimet otetaan suoraan HMD-kuvauksen elementin nimi –sarakkeesta.

Kuvassa 11 on aktiluokan XML-skeeman muotoinen tyyppi. Siinä määritellään aktiluokan attribuutit omina elementteinään, joilla on nimi, tietotyyppi ja esiintymiskerrat. Lisäksi skeematyypissä on lähtevien ja tulevien viittausten määrät itseensä sekä osallistumisluokan tyyppeihin. Skeemassa on osallistumisluokasta oma skeematyyppinsä ja aktiluokan viittaus saa tietotyypikseen sen. Kaikille muillekin skeeman tietotyypeille on omat skeematyyppinsä. Esimerkiksi aktiluokan Id-attribuutille tietotyyppi on II, jolle on oma monimutkainen tyyppi (compexType) määrittelynsä [HNB03].

RMIM-kaaviosta voi muodostaa muitakin tuotoksia. Yuksel ja Dogac ovat artikkelissaan [YuD10] esittäneet tavan yhdistää ISO/IEEE 11073 DIM –standardin [Dim04] ja RMIM-kaavion. DIM-standardi tarjoaa lääketieteellisten laitteiden viestinvälitykseen käytettävän abstraktin oliopohjaisen aihealuetietomallin [Dim04]. Yhdistämisen seurauksena esimerkiksi HL7 PHMR (Personal Health Monitoring Report) ja HL7 USAM (Unified Service Action Model) [SRM00] liittymiin voi syöttää tietoa suoraan mittalaitteelta. Yuksel ja Dogac luovat oman RMIM-kaavion, jota he nimittävät 11073 RMIM-kaavioksi. Se on yhteensopiva DIM-standardin tietosisällön kanssa, ja sen avulla he voivat jalostaa mittalaitetietoa eteenpäin.

HL7 PHMR on dokumentti, joka sisältää potilaan mitattua henkilökohtaista terveystietoa, kuten erilaisilta laitteilta saatua mittatietoa, muistiinpanoja, yhteenvetoja ja kaavioita.

PHMR-dokumentti on CDA-arkkitehtuurin mukainen ja se käyttää jatkuvan hoitodokumentin mallinteita (Continutity of Care Document, CCD). Sen yhdistäminen RMIM-kaavioon on RIM-viitemalli viittausten avulla suoraviivaista, koska kaikille luokille löytyy vastaavuus ja erot selviävät uudelleennimeämisellä. Esimerkiksi Yukselin ja Dogacin RMIM-kaavio sisältää mittalaitteista saadun mittausnimen (metric), jonka nimi muutetaan CDA-dokumentissa havainnoksi (observation) [YuD10].

```
<xs:complexType name="Act">
 <xs:complexContent>
  <xs:extension base="InfrastructureRoot">
   <xs:sequence>
    <xs:group ref="Act.Attrs"/>
    <xs:group ref="Act.Assocs"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:group name="Act.Attrs">
 <xs:sequence>
  <xs:element name="id" type="II" minOccurs="0" maxOccurs="unbounded"/>
   <xs:element name="code" type="Act.code" minOccurs="0"/>
   …
</xs:sequence>
 </xs:group>
 <xs:group name="Act.Assocs">
  <xs:sequence>
   <xs:group ref="Act.outboundRelationship" minOccurs="0" maxOccurs="unbounded"/>
   <xs:group ref="Act.inboundRelationship" minOccurs="0" maxOccurs="unbounded"/>
   <xs:group ref="Act.participation" minOccurs="0" maxOccurs="unbounded"/>
    …
<xs:group name="Act.participation">
  <xs:choice>
   <xs:element name="participation" type="Participation" nillable="true"/>
  </xs:choice>
 </xs:group>
```

Kuva 11. Aktiluokan XML-skeematyyppi [HNB03].

USAM on HL7-organisaation tekemä malli, joka tarjoaa potilastietojärjestelmien ja päätöksentukijärjestelmien välisen liittymän. Päätöksenteon tukena toimivat ohjeistukset (guidelines) parantavat hoidon laatua, mutta niiden käyttö on ollut vähäistä tiedon vaikean saatavuuden vuoksi. HL7-organisaation tavoite on helpottaa tiedon välitystä luomalla HL7 versiota 3 tukeva liittymä potilastietojärjestelmästä päätöksentukijärjestelmään yhdistääkseen potilaan tiedot ja lääketieteellinen tietämys. USAM on yksinkertaistettu versio RIM-viitemallista, ja se perustuu palvelutapahtuma (service action), tila (mood) ja suhde –metodologiaan. USAM-mallissa palvelutapahtumia ovat kaikki kliiniset tapahtumat. Palvelutapahtumassa on attribuutteja, jotka kertovat tapahtuman tekijän, tapahtuman, kohteen, ajan, paikan ja olosuhteet. Tila määrää tapahtuman luonteen kuten onko se määräys, suunnitelma, raportti tai tavoite. Suhteet yhdistävät kaksi palvelutapahtumaa toisiinsa. Suhteet voivat olla yleistyksiä, tarkennuksia, esiehtoja, jälkiehtoja tai viittaus korjattuun versioon [SRM00]. Yukselin ja Doagicin RMIM-kaaviossa mittaustulos on kopio RIM-viitemallin havaintoluokasta, ja sen seurauksena tiedot voidaan siirtää suoraan USAM-mallin palvelutapahtumiksi [YuD10].

7 CDA-standardi

CDA (Clinical Document Architecture) on maailmanlaajuisesti käytetty HL7 version 3 sovellus [Ben10]. Sovelluksella tarkoitetaan HL7 version 3 käyttötapaa. CDA-arkkitehtuurin alkuperäinen nimi on potilaan tiedon arkkitehtuuri (Patient Record Architecture, PRA) [EAR05], ja siitä on julkaistu kaksi versiota. Siinä käytetty ajattelumalli on tekstidokumenttien käytöstä otettu [Ben10], ja se määrittelee lääketieteellisten dokumenttien semantiikan ja rakenteen dokumenttien siirtoa varten järjestelmien välillä [EAR05]. Dokumenteilla, toisin kuin tietokannassa olevalla tiedolla, on aina tekijä sekä muuta pysyvää tietoa. Niihin on määritelty tarkkaan mitä varten ne on luotu, missä ja mistä aiheesta. Esimerkiksi reaalimaailmasta kirjeet ja laskut ovat esikuvina CDA-arkkitehtuurin määrittelemille dokumenteille. Niissä on määritelty valmiiksi aina mukana olevaa tietoa, kuten lähettäjä ja vastaanottaja [Ben10]. CDA-arkkitehtuurin julkaisun 2 (release 2) mukaiset CDA-dokumentit siirretään XML-muotoisena dokumentteina tietojärjestelmien välillä. Dokumenteissa käytetyt tietotyypit ovat HL7 versiosta 3 ja koneellisesti käsiteltävä rakenne on johdettu RIM-viitemallista [EAR05]. Nämä ominaisuudet mahdollistavat dokumenttien laajentamisen ulkoisilla koodistoilla, kuten SNOMED-CT-koodistolla tai ICD-10-diagnoosikoodistolla. CDA-arkkitehtuurin on tarkoitus olla niin laajennettava ja

joustava, että se kattaa kaikki kliinisesti tehdyt dokumentit [DAB06].

CDA-dokumenteilla on viisi ominaisuutta: pysyvyys (persistence), vastuullisuus (stewardship), todentaminen (authentication), täydellisyys (wholeness) ja ihmisluettavuus. *Pysyvyydellä* tarkoitetaan dokumentin elämänkaarta, missä se luodaan, missä luetaan ja muokataan, ja lopuksi tuhotaan. Koko elämänkaarensa aikana se pysyy yhtenäisenä muuttumattomana tietokoosteena. *Vastuullisuus* tarkoittaa, että koko elinkaarensa aikana dokumentilla on henkilö (esimerkiksi lääkäri) tai organisaatio (esimerkiksi sairaala), joka on vastuussa sen oikeellisuudesta, muokkaamisesta ja tuhoamisesta. Dokumenteista voi olla useita kopioita eri tietojärjestelmissä, ja kaikilla niillä on oltava siitä vastaava taho. Dokumenttien *todentaminen* toteutetaan tekijän allekirjoituksella (sign), jolloin lukija voi varmistua sen oikeellisuudesta [Ben10]. Esimerkkinä tästä on Suomessa käytetty toimikorttitodentaminen [Ere13]. *Täydellisyydellä* tarkoitetaan, että dokumentissa on itsessään kaikki sen sisältämä tieto yhdessä paketissa. Siihen kuuluu sisällön lisäksi kontekstitietoa, eli kuka sen on luonut, muokannut, milloin, missä, miksi ja mistä aiheesta. Yhdessä todentamisen kanssa täydellisyys varmistaa dokumentin lukijalle dokumentin tietojen paikkaansa pitävyyden ja aihealueen. *Ihmisluettavaa* dokumenttia voidaan käyttää vaikka teknologiat olisivatkin muuttuneet. Sama tieto, jonka kone käsittelee, on oltava ihmisluettavassa muodossa, koska terveydenhuollon dokumentin tulee olla olemassa vähintään sata vuotta [Ben10].

CDA-arkkitehtuurin luominen alkoi vuonna 1997 [Ben10]. Ensimmäisestä julkaisusta tuli standardi vuonna 2000 ja toisesta 2005. Se on ensimmäinen spesifikaatio, joka on johdettu HL7 version 3 RIM-viitemallista [DAB06]. Ensimmäisen julkaisun mukaisessa dokumentissa on otsake ja runko [Ben10]. Vain otsaketiedon on pakko olla RIM-mallin mukainen, runko voi sen sijaan olla mitä tahansa tietoa, esimerkiksi PDF-, kuva- tai äänitiedosto. Ulkopuolisen tiedoston voi välittää viestin sisällä, mutta sen tallennus voi olla erillinen dokumentista [DAB06]. Julkaisussa kaksi runko-osa voi olla HL7 version 3 mukainen. Takaisinyhteensopivuuden vuoksi runko-osa voi edelleen olla muutakin.

CDA-dokumentit voi jakaa kolmeen eri tasoon (level). Ensimmäisen tason dokumentti sisältää otsakkeen ja ihmisluettavan rungon. Otsakkeessa on dokumentin hakua helpottavia kenttiä, ja runko voi olla vapaata tekstiä tai esimerkiksi kuva. Toisen tason runko-osa voi olla kuten ensimmäisessä tasossa, mutta se voi koostua yhdestä tai useammasta rakenteellisesta osiosta (section). Jokainen osio sisältää yhden kerronnallisen lohkon

(block), joka on XML-formaatissa ja mahdollista muuntaa ihmisluettavaksi. CDA julkaisu 1 sisältää nämä kaksi tasoa. Kolmas taso on vain julkaisussa kaksi. Sen runko-osassa voi olla koneellisesti prosessoitavaa tietoa HL7 version 3 kliinisten toteamuksen (clinical statement) mukaisesti [Ben10]. Jokainen taso lisää enemmän merkintöjä dokumenttiin [EAR05]. Jokaisen tason dokumenttien oikeellisuus tarkistetaan geneeristä CDA-skeemaa vasten. Dokumenteille voi tehdä lisärajoituksia mallinteilla tai lisäämällä rajoitteita geneeriseen skeemaan.

CDA-dokumentin otsaketieto sisältää tietoa dokumentista; mikä se on, kuka sen on tehnyt, missä se on tehty ja milloin sekä mitä varten [Ben10]. Otsakkeessa olevat tiedot kattavat koko dokumentin, mutta joitain tietoja voi korvata rungon osioissa [DAB06]. Taulukossa 8 on esitetty tarkemmat tietokentän otsakkeen tiedoista.

Mikä	Dokumentin tunniste (document id) Dokumentin tyyppi (document type) Luottamuksellisuus (confidentiality)
Kuka	Potilas (patient) Tekijä (author)
Missä	Vastuussa oleva taho (custodian)
Milloin	Luontiaika (datetime created) Todentamisaika (datetime signed)
Miksi	Palvelutapahtuma (service event) Käynti (encounter) Täytettävä tilaus (order being fulfilled) Isädokumentti (parent document) Suostumus (consent authorization)

Taulukko 8. CDA-dokumentin otsaketiedot [Ben10].

Mikä-lohko yksilöi dokumentin. Osiossa on kolme koodia, jotka ovat pakollisia. Ne kertovat dokumentin olevan CDA-dokumentti, sen tyypin ja yksilöivän juoksevan tunnistenumeron, jonka avulla dokumentti voidaan hakea esimerkiksi arkistosta. CDA-dokumentiksi dokumentti tunnistetaan RIM-viitemallin luokka- (classcode) ja tilakoodeilla (moodcode), ja tässä tapauksessa luokkakoodi on DOCCLIN ja tilakoodi EVN. Tyyppi

ilmoitetaan koodikentässä (code), ja sen arvo on määritelty CDA-arkkitehtuurin ulkopuolisella koodistolla [Ben10], esimerkiksi Suomessa Terveyden ja hyvinvoinnin laitoksen (THL) koodistopalvelussa [THL13]. *Milloin-lohkoon* tallennetaan luonti (kenttään effectiveTime) ja hyväksymisaika (kenttään authorTime). Ne voivat olla samat. Hyväksymisaika määräytyy dokumentin tekijän päätöksellä, kun sen sijaan luontiaika on dokumentin oikea luontiaika. Jokaisella dokumentilla on ainakin kolme osallistuvaa tahoa. Kuka-lohko määrittelee niistä kaksi ja Missä-lohko yhden. *Kuka-lohkon* dokumentin kohde kertoo kenelle dokumentti kuuluu, yleensä se on potilas. Tekijä kertoo kuka on vastuussa dokumentin sisällöstä, esimerkiksi tiedot kirjannut lääkäri. *Missä-lohkossa* on vastuussa oleva taho kuten organisaatio, joka säilyttää dokumenttia ja on vastuussa, ettei sitä muuteta tai tuhota ilman asianmukaisia oikeuksia. *Miksi-lohkossa* on tietoa miten dokumentti linkitetään toisiin dokumentteihin tai tapahtumiin palvelutapahtumakentän avulla. Sen arvo voi olla esimerkiksi RIM-mallin aktiluokan ilmentymä, kuten kolonoskopia tai sairaalasta kotiuttaminen [Ben10].

Jokaisella CDA-dokumentilla on yksi otsake ja yksi runko. Sen runko voi olla joko rakenteellinen tai ei. Jos runko on rakenteellinen, se on yleensä tallennettu XML-kielellä, ja se on jaetta osioihin. Sen sijaan rakeenneton runko on vain tekstiä tai kuvia. Jokainen rakenteellisen rungon osio sisältää ihmisluettavan kerronnallisen osion, joka tallennetaan omaan kenttäänsä. Tämä täyttää vaatimuksen, jonka mukaan dokumentin tulee olla ihmisluettavassa muodossa ja käytettävissä ainakin sata vuotta. Alkuperäinen CDA-arkkitehtuurin kehittäjien idea oli sisällyttää yhteen osioon kaikki tai ainakin suurin osa rungon tiedosta. Käytännön toteutuksissa yksi osio sisältää vain yhden tekstirivin tai merkinnän (entry). Toteutettu tapa mahdollistaa dokumentista suodattamisen, uudelleenjärjestämisen ja näyttämisen eri tavoilla. Esimerkiksi osioiden järjestäminen ajan ja tekijän mukaan ja siitä vain osan osioiden näyttäminen [Ben10]. Suomen lääkemääräyksen toteutuksessa on käytetty yhtä osiota [Ere13].

Merkintä on kliininen toteamus (clinical statement), joka on koneellisesti käsiteltävässä muodossa. Dokumentin rungon osiot voivat sisältää mielivaltaisen määrän merkintöjä. CDA-dokumenteissa käytetään HL7 version 3 kliinisten tutkimusten lauserakennetta (clinical statement pattern). HL7 määrittelee lauserakenteen seuraavasti: ”Yksittäisen kliinisen tai kliiniseen asiaan liittyvän asian ilmaisu (expression), joka on tallennettu koska se on potilaan hoidon kannalta tärkeä. Kliininen dokumentti on fraktaalinen, eli sen tieto voi tarkentua, ja siksi sen laajuus ja yksityiskohtaisuus voivat vaihdella tilanteesta riippuen”

[Ben10].

Itse XML-dokumentissa CDA-dokumentin otsake ja runko sijoitetaan ClinicalDocument-xml-merkin (tag) sisään. Otsake sijoittuu ClinicalDocument-merkin ja rungon aloittaja-merkin väliin. Rungon aloittaa structuredBody-merkki. Rungon osio on section-merkin sisällä, joka on structuredBody-merkin sisällä [DAB06].

Kliinisellä toteamuksella voi olla suhteita muihin toteamuksiin, kuten kohde, tekijä, tekopaikka, suorittaja, osallistuja tai tiedottaja. Jokainen kliininen toteamus on yksi seuraavista erikoistumisista [Ben10]:

- havainto (observation), joka kattaa laajan alan toteamuksista. Se voi esimerkiksi olla historiatietoa, tutkimus, diagnoosi tai ennuste.
- toimenpide (procedure), joka voi olla laboratorionäyte tai kuva ja sitä käytetään myös leikkaustoimenpiteissä ja kuvantamisissa.
- tapaaminen (encounter) sisältää hallinnolliset tapahtumat, kuten vastaanoton ajanvarauksen ja jonottajien listan hallinnan.
- lääkkeen määräämistä tai toimitusta (substance administration or supply) käytetään tuotteiden, kuten lääkkeiden, määräämiseen, korvaamiseen ja hallintaan.
- suostumus (consent), jolla ilmoitetaan potilaan suostumus tiettyyn asiaan.

```
<ClinicalDocument>
  …CDA-dokumentin otsake…
  <structuedBody>
    <section>
      <text>Kerronnallinen osuus</text>
      <observation>…</observation>
      <substanceAdministration>
        <supply>…</supply>
      </substanceAdministration>
    </section>
    <section>…</section>
  </structuredBody>
</ClinicalDocument>
```

Kuva 12. Kliinisen dokumentin XML-rakenne [DAB06].

Kliiniset toteamukset sijoitetaan osioiden sisään niitä kuvaaviin englanninkielisiin merkkeihin. Kuvassa 12 on esitetty kliinisen dokumentin XML-muotoinen esitys tärkeimpien osien kannalta. Kerronnallinen osuus näytetään käyttäjälle sellaisenaan, mutta muut osat käsitellään koneellisesti, esimerkiksi päätöksentukijärjestelmässä. Osion kliiniset toteamukset kuvaavat yleensä kerronnallisen kentän tiedon rakenteellisessa muodossa. Kerronnallinen kenttä on aina pakollinen, koska dokumentin ihmisluettavuusominaisuuden tulee täyttyä, mutta toteamukset ovat vapaaehtoisia kenttiä. Pakollisuuksien ja vapaaehtoisuuksien seurauksena CDA-arkkitehtuurin toteuttaminen voi olla hyvin yksinkertaista tai hyvin työlästä. Yksinkertaisessa tavassa osioihin lisätään vain kerronnallinen kenttä. Tätä voi myöhemmin laajentaa lisäämällä toteamuksia, joita vastaanottajan ei heti tarvitse edes käsitellä. Heti alussa dokumentin määrittäjät voivat sopia yksityiskohtaisemmasta toteutuksesta, jossa kaikki kerronnallisen kentän tiedot on avattu rakenteellisiin toteamusosiin [DAB06].

Toteamusten suhteet voivat olla seitsemää eri tyyppiä ja niillä ilmaistaan yhteyttä erilaisten asioiden välillä. *CAUS-suhteella* (aiheutui) ilmaistaan, että suhteen lähde aiheutti suhteen kohteen. Esimerkiksi penisilliinin määrääminen aiheutti ihottumahavainnon. *COMP* (komponenttiyhteys) ilmaisee kohteen olevan lähteen osa. Esimerkiksi hemoglobiinin mittaus on osa täydellisen verenkuvan mittausta. *GEVL* (arviointi) yhdistää halutun ja tehdyn havainnon. Tällöin voidaan arvioida kuinka hyvin tavoite on saavutettu, esimerkiksi onko paino pudonnut tai noussut haluttuun arvoon. *MFST-yhteys* (luettelo) ilmaisee lähteen kuuluvan kohteen luetteloon. Esimerkiksi nokkosihottuma kuuluu penisilliinin aiheuttamiin allergisiin reaktioihin. *RSON* (syy-yhteys) ilmaisee syy-yhteyden asioiden välillä, esimerkiksi rintakipu on syynä rasitustestin tekemiseen. *SAS-suhteella* (alkaa jälkeen) ilmaistaan aktin alkamista toisen aktin jälkeen. Esimerkiksi laajamittainen hikoilu alkaa rintakipujen ilmentymisen jälkeen. *SPRT* (tukee) ilmaisee, että kohde tukee lähteen todisteita, esimerkiksi mahdollinen keuhkosyöpä saa tukea röntgenkuvasta löytyneestä suuresta massasta keuhkoissa [Ben10].

CDA-dokumenttiin voi liittää ulkopuolisia viittauksia. Näitä voivat olla toinen CDA-dokumentti, weblinkki, ääni, kuva ja muuta multimediaa. CDA-dokumentissa viittaus on omassa osiossaan nimeltä ulkopuolinen havainto (externalObservation). Viittauksen kohde sijaitsee CDA-dokumentin ulkopuolella, mutta täydellisyysvaatimuksen perusteella sen tulee olla saatavilla aina CDA-dokumentin yhteydessä. Viittauksen ansiosta potilaskertomus voi olla täydellinen [CLC07].

Kuvassa 13 on esimerkki ulkopuolisen havainnon sijoittumisesta CDA-dokumenttiin. Ulkopuolinen havainto on esimerkissä CDA-dokumentti, jonka joukkotunniste on ilmoitettu ulkopuolisen havainnon tunnistekentässä (id). Ulkopuolisesta havainnosta on ilmoitettu haettavan dokumentin tyyppi ja hakuosoite. Tässä tapauksessa kyseessä on xml-dokumentti ja se löytyy osoitteesta http://tto.arkisto/saataville/200455laho.xml?lupaxxx. Viittaukset ovat hyödyllisiä viitetietokantaympäristössä, jossa riittää alkuperäistä kyselyä tehdessä hakea vain viittaukset jatkodokumentteihin. Jatkodokumentit haetaan tarvittaessa, jos käyttäjä niin haluaa. Kuvat voidaan sijoittaa CDA-dokumentin sisälle omaan mime-osioon (Multipurpose Internet Mail Extensions) Base64-koodattuina tai niistä tehdään vain viittaus kuvatiedostoon, jonka CDA-dokumentin lukijaohjelma hakee ne käyttäjälle automaattisesti. Kuvat tulee sijoittaa verkkoon siten, että vastaanottajalla on pääsy niihin, tai lähettää CDA-dokumentin mukana arkistoitavaksi dokumentin kanssa [Ere13d].

```
<ClinicalDocument>
 <structuredBody>
  <section>
   <text>kerronnallinen teksti</text>
   <entry>
    <observation>…</observation>
    <observation>
     <externalObservation>
      <id root="1.2.246.537.10.6280613.11" extension="2004.55"/>
      <text mediaType="text/xml">
       <reference value="http://tto.arkisto/saataville/200455laho.xml?lupaxxx"/>
      </text>
      <setId root="1.2.246.537.10.6280613.11 extension="2004.55"/>
      <versionNumber value="1"/>
     </externalObservation>
    </observation>
   </entry>
  </section>
 </structuredBody>
</ClinicalDocument>
```

Kuva 13. Ulkopuolinen havainto CDA-dokumentissa [CLC07, Ere13d].

8 CDA-profiili

8.1 Profiilin määritelmä

Mallinteilla (templates) ja toteutusoppailla suunnittelija voi rajoittaa CDA-arkkitehtuuria CDA-dokumentin toteuttajien ohjeiksi ja tarjota tarkistussääntöjä, joilla dokumentin voi todeta täyttävän nämä rajoitukset [DAB06]. Mallinne on joukko toimintasääntöjä, jotka vaikuttavat sanomaan tai sen osaan [Boo11]. Mallinne määrittelee miten CDA-dokumenttien tulee käyttäytyä jossain käyttötapauksessa tai tilanteessa. CDA-profiili on joukko mallinteita, jotka määrittelevät yhden dokumenttityypin. Suunnittelija voi määritellä useita ja jopa päällekkäisiä mallinteita. Mallinnelista tarjoaa toteuttajalle mahdollisuuden valita käytettävissä olevista mallinteista tilanteeseen sopivimman [Ben10]. Suunnittelija voi luoda mallinteita HL7-organisaation työkaluilla, UML-työkaluilla tai muilla kolmannen osapuolen työkaluilla [Boo11], esimerkiksi ISO Schematronilla [Vli07]. Suosituin tapa mallinteiden tekoon on ihmisluettavat kerronnalliset mallinteet, joiden oikeellisuus tarkistetaan käyttämällä Schematronia [Boo11].

Mallinteet yksilöidään tunnisteen avulla ja ne voidaan tallettaa erilliseen säiliöön (repository). Mallinteeseen kuuluu joukko kuvaustietoa, kuten nimi, versio, kuvaus, linkki vanhempidokumenttiin ja selitys miten sitä käytetään [Ben10]. HL7-organisaation työkaluilla tehtyyn mallinteeseen tulee HL7-mallin tunniste ja dokumenttiin kirjataan HL7-mallin nimiavaruuden oliotunniste. Muilla tavoilla mallinnetta luotaessa, suunnittelija antaa itse sille oliotunnisteen [Boo11].

Mallinne on joukko lauseita, jotka kohdistuvat RMIM-kaavioon rajoittaen sitä [Ben10]. Mallinne luodaan yleensä yhden tyyppistä dokumenttia varten, mutta se voi rajoittaa usean tyyppisiä dokumentteja, jos dokumentti tai mallinne sallii sen. Mallinteet ovat joko avoimia tai suljettuja. Avoimet mallinteet rajoittavat vain sen mitä niihin on kirjattu, kaikki muu on sallittua. Tällä tavoin mallinteiden jatkokehittäminen on mahdollista. Suljetut mallinteet sallivat vain sen mitä niihin on kirjattu. Niillä voi määritellä esimerkiksi viranomaisen ja terveyspalvelujen tuottajan tiedonvälityksen ja varmistaa, että vain tarvittavat tiedot kulkevat oikeassa muodossa. Yleensä mallinteet ovat avoimia [Boo11].

Mallinteita on kolmea tyyppiä:

- Kerronnallinen mallinne, jota käytetään toteutusoppaissa kuvaamaan rajoitetta.

Esimerkiksi dokumentin todentajalla tulee olla voimassa oleva lupa toimia lääkärinä.

- Schematron [Vli07] tarkistukset. Esimerkiksi legalAuthenticator and not legalAuthenticator [@nullFlavor], missä tarkistus kohdistetaan johonkin XML-solmuun.
- Staattinen RMIM-kaavio, jossa todentaja on merkitty pakolliseksi. Staattisesta RMIM-kaaviosta tulee XML-skeema, jossa rajoitukset on toteutettu [Ben10].

Suomessa käytetyt mallinteet ovat kerronnallisia toteutusoppaita, joilla on määrittelyn lisäksi mallinteelle kuuluvat perustiedot. Lisäksi XML-skeemoilla määritellään mitä arvoja dokumentin tietty kenttä voi saada. Esimerkiksi valuutan arvo ei voi olla muuta kuin eur, fim, usd ja niin edelleen [Ere13].

CDA-dokumentissa viitataan ulkopuoliseen mallinteeseen mallinnetunnisteen (templateId) avulla. Esimerkiksi xml-rivi: <templateId root=”2.16.840.1.113883.10.20.1”/>, viittaa mallinteeseen, jonka oliotunniste (oid) on 2.16.840.1.113883.10.20.1 [Ben10]. Oliotunniste on yksilöllinen tunniste kaikkialla maailmassa. Ensimmäiset numerot kertovat tunnisteen olevan ISO OID –standardin mukainen. Niitä seuraa mallinteen luoneen organisaation tai tahon maatunniste sekä itse organisaation tai tahon tunniste. Suomessa organisaatio ilmoitetaan yrityksen y-tunnuksella ja taho lääkärin sv-numerolla, jolloin kaikilla dokumentin tekijöillä on yksilölliset tunnisteet. Näitä tunnisteita seuraa oliotunnisteessa tarkentavia numeroita, kuten tyyppi, vuosi ja organisaation sisäinen juokseva tunniste [Ben10, Ere13].

Malline voi rajoittaa koko dokumenttia tai vain osiota tai merkintää. Tällöin mallinnetunniste on osion tai merkinnän aloittavien ja lopettavien xml-merkkien sisällä. Mallinteita käytetään joko koneellisesti automaattisissa tarkistuksissa tai niitä käyttävät toteuttajat, jotka lukevat niistä miten ja millaista tietoa kenttään tai lohkoon kuuluu. Automaattisessa tarkastuksessa tarkistaja (validator) käy valmiin dokumentin läpi, ja tutkii täyttääkö XML-dokumentti XML-skeeman, ja mahdollisesti Schematronin, määrittelemät säännöt ja rakenteen. Tarkistaja ilmoittaa sen käyttäjälle mahdollisista virheistä [Ben10].

Mallinnetunniste (templateId) voi esiintyä CDA-dokumenteissa monissa yhteyksissä. Yleisellä tasolla se määrittelee minkä määrityksen mukaan dokumentti tai sen osa on toteutettu [eAr13]. Esimerkiksi sähköisen lääkemääräyksen otsaketiedoissa mallinnetunniste määritellään olemaan CDA R2 -otsakkeen voimassa oleva määrittelydokumentti

[Ere13]. Alemmilla tasoilla mallinnetunniste ohjeistaa ja rajaa toteutusta. Esimerkiksi elektronisen kansalaisarkiston riskitiedoista määritellään kentillä olevan vain sallitut luokitukset. Yksittäisen merkinnän kohdalla mallinne ohjeistaa ja rajoittaa toteuttajaa määräämällä kentän arvot, esimerkiksi verenpaineen arvot saavat olla välillä 50–250 [eAr13].

Dokumentilla voi olla useita mallinteita samaan aikaan. Tällöin kaikkien mallinteiden sääntöjen tulee toteutua. Mallinteilla voi olla riippuvuussuhteita toisiinsa, jolloin yhden mallinteen käyttö osiossa vaatii toisen mallinteen käytön samassa osiossa. Tämä logiikka on perimisestä, eli uuden mallinteen ei tarvitse toteuttaa itse kaikkia sääntöjä, vaan sen sijaan se perii vanhan mallinteen ja lisää omia sääntöjä toteutettavaksi. Tällä tavalla dokumenttien siirto on yhteensopivampaa, koska kaikkien vastaanottavien tahojen ei tarvitse toteuttaa molempia mallinteita. Molemmat mallinteet toteuttava on yhteensopivampi kuin vain toisen mallinteen toteuttavan vastaanottaja, mutta molemmat vastaanottajat saavat tärkeimmät tiedot samanlaisena, koska ne toteuttavat toisen mallinteista.

Mallinteiden periminen toteutuu avoimessa tiedonvälityksessä. Siinä lähettäjä ja vastaanottaja määrittelevät mitä tietoja lähetetään, mutta lähettäjällä on mahdollisuus lähettää enemmän kuin vastaanottaja osaa käsitellä. Esimerkiksi lähettäjä voi lisätä viestiin paikallisen koodinsa, jota vastaanottajan ei tarvitse käsitellä. Toisaalta toinen vastaanottaja voi tarvita sitä omassa käsittelyssään. Mallinteiden pilkkominen pieniin ja uudelleen käytettäviin palasiin on todettu hyödylliseksi, minkä periminen mahdollistaa.

CDA-mallinteita on toteutettu yli 500. HL7, IHE, ANSI/HITSP, epSOS ja Continua ovat toteuttaneet noin 50 dokumenttimallinnetta, 100 osiomallinnetta (section template) ja merkintämallinnetta (entry template) [Boo11]. Lisäksi esimerkiksi Suomen HL7-yhdistys on luonut useita kerronnallisia mallinteita ja kerännyt ne KanTa-hankeen sivuille.

Mallinteen toteuttamisen ensimmäinen vaihde on määritellä millaista kliinistä dokumenttia varten mallinnetta ollaan luomassa. Se vaikuttaa dokumentin code-merkin attribuutteihin ja sisältöön. Seuraavaksi määrittelijän tulee kartoittaa mitä osallistumis- ja aktisuhdeluokkia viestiin tulee ja mitä tietoa voi olla dokumentin otsakkeessa. Nämä riippuvat käyttötarkoituksesta. Viimeiseksi määrittelijän tulee hahmottaa mitä runko-osassa tulee olla ja mitä voi olla. Määrittelyn voi tehdä käyttötapauksen mukaan tai käyttää apuna toteutusoppaita, joissa on määritelty mitä tietyissä tapauksissa tarvitaan. Viimeisen vaiheen tuloksena on lista osioista, jotka ovat joko pakollisia, tarvittavia tai vapaaehtoisia.

Mallinteessa käytettyjä rajoitteita voi olla eritasoisia. *Pakollinen* (mandatory) rajoite tarkoittaa elementin ja sen sisällön pakollisuutta, kumpikaan ei saa olla tyhjäarvoinen tai puuttua. *Tarvittava* (required) sen sijaan tarkoittaa, että elementti on oltava sanomassa, mutta sen arvo voi olla tyhjäarvoinen. *Ehdollinen* (conditional) rajoite vaatii elementin olemassaolon, jos tietty ehto toteutuu. *Tarvittava, jos tiedossa* (required if known) tarkoittaa elementin olemassaolo silloin kun se on lähettävälle järjestelmälle tiedossa. *Suositeltavaa* (recommended) rajoitetta käytetään kun järjestelmä ei välttämättä pysty toteuttamaan ehtoa, mutta elementin olemassaolo on hyvää käytäntöä. *Vapaaehtoinen* (optional) rajoite antaa lähettäjälle mahdollisuuden valita lähetetäänkö tieto sanomassa. Se tarkoittaa, että asia pitää tehdä näin, jos se tehdään, mutta sen tekeminen ei ole pakollista. *Ei suositeltava* (not recommended) rajoite mahdollistaa sellaisten järjestelmien, joilla ei ole muuta mahdollisuutta kuin käyttäytyä parasta käytäntöä vastaan, toteuttaa mallinne. Tällöin rajoitetta voi käyttää, mutta sitä tulisi välttää. *Kielletty* (prohibited) rajoite on ehdoton kielto. *Yhteisesiintyminen* (co-occurance) rajoitetta käytetään yksittäisessä tiedossa määrittelemään sen suhdetta muihin tietoihin. Esimerkiksi painon tai pituuden lähettämisessä mittayksikkö on ilmoitettu rajoitteessa.

Jos osio sisältää vain kerronnallisen osan, kerronnallinen malline voi kertoa millaista tietoa osion tulee sisältää. Rakenteellisen sisällön kohdalla mallinne rajoittaa mikä koodi tunnistaa osion [Boo11], esimerkiksi KanTa-sivuilla määritelty lähete-hoitopalaute-rajapinnan kuvauksessa jokaisella osiolla on koodi, jonka attribuuttina on koodin arvo ja koodiston oliotunniste. Kuvassa 14 on lähetteen osio, missä koodiarvo on 11 ja koodiston oliotunniste on 1.2.246.537.6.13.2006. Koodisto määrittelee jokaiselle hoitoprosessin vaiheelle nimen ja koodiarvon. Koodisto löytyy Terveyden ja hyvinvointilaitoksen koodistopalvelimelta [THL13].

```
<section>
  <code code="11" codeSystem="1.2.246.537.6.13.2006" codeSystemName="Hoitoprosessin vaihe"
displayName="Tulotilanne"/>
                <title>Tulotilanne</title>
```

Kuva 14. Lähetteen ja hoitopalautteen osio [Lah10].

CDA-dokumentti sallii lokalisaation. CDA-määrittelyissä on paikallisille merkinnöille ja otsakkeille varattu omat xml-merkit (local_markup, local_header). Saksassa tehdyssä SCIPHOX-projektissa [HSD03] yhdeksi vaikeimmaksi haasteeksi osoittautui paikallisen

tarkennuksen ja globaalin yleistyksen välinen valinta. Aikaisempien yhteentoimivuusongelmien takana on ollut juuri se, että usein on tehty kaikki paikallisen tarkennuksen mukaan. Sen seurauksena järjestelmät eivät ole olleet yhteensopivia. SCHIPHOX-projektin tavoitteena oli tarjota Saksan sairaaloille yhteinen viestien välitystapa. Projektissa päädyttiin lisäämään omia pieniä tietoyksiköitä (small semantic units, SSU) globaaliin määritykseen. Tietoyksiköt sisältävät tietoa kuten potilaan vakuutustiedot, ja ne on lisätty CDA-arkkitehtuuriin lokaalilla merkinnällä (local_markup). Lokaalit lisäykset ovat omassa nimiavaruudessaan, koska ne validoidaan omalla tietoyksiköitä varten tehdyllä XML-skeemalla. Projekti käyttää CDA-määrittelyn tarjoamia XML-skeemoja, ja täydentää niitä itse tehdyllä erillisellä skeemalla [HSD03]. Suomessa käytetyn sähköisen lääkemääräyksen määrittelyssä on käytetty paikallista otsaketta, joka on omassa nimiavaruudessaan (hl7fi) [Ere13].

8.2 Schematron

CDA-arkkitehtuurin mukainen XML-dokumentti muodostetaan sille määritellystä RMIM-kaaviosta. XML-dokumentti on oikeellinen, jos se täyttää sen skeeman määräämät ominaisuudet, kuten kardinaalisuus ja tietotyyppirajoitteet. XML-skeeman yleisesti käytetty standardi on W3C XML Schema [W3C13]. CDA-arkkitehtuurin mukaan CDA-dokumentti ei ole oikeellinen, jos se ei täytä CDA-skeemansa vaatimuksia. CDA-skeemat ovat osa CDA-arkkitehtuuria ja niitä hallinnoi HL7-organisaatio. Ne sisältävät CDA-arkkitehtuurin sallimia rajoitteita. XML-skeemalla voi esittää vain osan CDA-arkkitehtuurin määrittelemistä vaatimuksista. Esimerkiksi CDA-arkkitehtuurin yleinen vaatimus on, että XML-elementillä tulee olla attribuuttina joko A-attribuutti tai B-attribuutti, mutta ei molempia yhtä aikaa. XML-skeema ei pysty tätä tarkistamaan, mutta Schematron pystyy [Boo11].

Schematron on XML-metakielen lisäys, joka tarkistaa XML-dokumentin oikeellisuuden samaan tapaan kuin XML-skeema. Sen on kehittänyt Rick Jeliffe vuonna 1999 ja se on avoin standardi. Se perustuu puumaiseen XPath-tarkistusten läpikäyntiin ja XSLT-tyylitiedostoihin [Vli07]. Schematronia on käytetty HL7 version 3 XML-skeemoissa tarkistamaan tietotyyppien rajoitukset, joiden tarkistaminen perinteisellä XML-skeemalla on hankalaa tai mahdotonta. Sitä on käytetty laajalti tarkistamaan liiketoimintalogiikkaa [Boo11].

Schematronissa määritellään joukko sääntöjä (rule), jotka kohdistuvat solmuihin XML-

dokumentissa. Erillinen prosessoija käy XML-dokumentin läpi ja tutkii pitävätkö säännöt paikkansa [Vli07]. Schematron säännöistä tehdään XSLT-tyylitiedosto ajamalla Schematron-skeema prosessoijan läpi. Tuloksena syntynyttä tyylitiedostoa voi käyttää XML-dokumentin oikeellisuuden tarkistukseen [Boo11]. Tarkistuksen lopuksi käyttäjälle ilmoitetaan virheellisistä arvoista Schematron-skeeman määrittelemien virheilmoitusten kautta.

Schematron lisää XML-dokumenttiin ilmaisuvoimaa. Sillä voi tarkistaa XML-dokumentin rakennetta, mutta sen pääasiallinen tarkoitus on lisätä tarkistuksia, joita XML-skeemalla ei voi tehdä. Schematronilla voi esimerkiksi tarkistaa, että solmujoukon kaikilla solmuilla on uniikki tunnisteattribuutti, mikä on vaikeaa tai mahdotonta muilla tarkistusvälineillä [Vli07]. Esimerkiksi kuvassa 15 tarkistetaan Schematron-säännöllä onko authenticator-solmulla id-attribuuttia, ja jos sitä ei ole ilmoitetaan käyttäjälle virheilmoitus. Context- ja test-attribuuttien sisällä on XPath-lauseke (expression), joka määrittelee mistä ja mitä tietoa tarkistetaan.

```
<rule context=”authenticator”>
  <assert test=”@id”>Todentajalla tulee olla id-tunniste</assert>
</rule>
```

Kuva 15. Schematron tarkistus onko todentajalla tunnistetietoa [Ben10].

Schematron-säännöt kohdistuvat aina kontekstiin CDA-dokumentin sisällä. Konteksti määrittää missä sääntö on voimassa. Sääntöjoukoille voi antaa nimiä ja niitä voi käyttää useassa paikassa. Schematron-säännöillä voi tarkistaa mallinteiden olemassaolon dokumentissa. Esimerkiksi sääntöjoukon ensimmäinen sääntö tarkistaa onko tarkistavan sääntöjoukon muodostaman mallinteen mallinnetunniste dokumentissa. Seuraavissa säännöissä tarkistetaan onko tarkistavan mallinteen vaatimat mallinteet dokumentissa. Esimerkiksi, jos tarkistava mallinne lisää aikaisempaan mallinteeseen liiketoimintasääntöjä, riittää sen tarkistaa vain onko aikaisempi mallinne olemassa dokumentissa sen sijaan, että se kopioisi säännöt jolloin sääntöjen ylläpidettävyys heikkenee [Boo11].

Schematron-tarkistuksissa käytetään syntaksina XPath-kieltä [ClD99]. XPath-kielessä voi viitata kontekstin attribuutteihin ät-merkillä esimerkiksi @nimi. Lapsielementteihin voi viitata suoraan lapsielementin nimellä, kuten Sukupuoli. Piste (.) valitsee käsiteltäväksi kontekstisolmun. Tähti (*) valitsee kontekstielementin lapset ja kaksi pistettä (..) valitsee kontekstielementin vanhemman. Lapsiin voi viitata tietämättä nimeä numeroinnilla, esimerkiksi ensimmäinen lapsi on *[1]. Indeksointi alkaa ykkösestä. Vertailuja voi

tehdä yhtäsuuruuden lisäksi suurempi kuin, suurempi tai yhtä suuri, pienempi kuin, pienempi tai yhtä suuri sekä eri suuri. XML-dokumentin sisällä pienempi kuin ja pienempi tai yhtä suuri kuin merkit tulee koodata <-merkillä. Tarkistuksia voi ketjuttaa liitossanoilla ja (and) sekä tai (or). Numeroiden yhteenlasku tapahtuu aritmeettisilla operaattoreilla, kuten yhteen- ja vähennyslasku. Lisäksi XPath-kieli tarjoaa joukon funktioita, kuten määrä (count) ja paikka (position), joilla tarkistuksiin saa lisää ilmaisuvoimaa [ClD99].

9 Sähköinen lääkemääräys

Tässä kappaleessa käsitellään Suomessa toteutettua sähköisen lääkemääräyksen toteutusta yleisesti. Sähköiset lääkemääräykset lähetetään potilastietojärjestelmistä kansalliseen reseptikeskukseen ja niitä voi sieltä hakea potilasjärjestelmään. Sähköisen lääkemääräyksen suunnittelu ja toteuttaminen aloitettiin vuonna 2006 [Ere13c].

9.1 Toteutus

Sähköinen lääkemääräys toteuttaa CDA R2 [DAB06] -arkkitehtuurin. Se on valittu sähköisen määräyksen toteutusarkkitehtuuriksi, koska kaikki käsiteltävät dokumentit, kuten lääkemääräys, toimitus tai korjaus, arkistoidaan mahdollista myöhempää käyttöä varten [Ere13b]. CDA-arkkitehtuurin ominaisuus dokumentin ihmisluettavuudelle ja vähintään sadan vuoden elinkaari sopii tähän vaatimukseen.

Sähköisen lääkemääräyksen herätetapahtumat jaetaan kahteen luokkaan. Toinen luokka aktivoituu kun lääkemääräysdokumentteja muutetaan ja toinen kun niitä kysellään. Muutosherätetapahtumia ovat esimerkiksi lääkemääräyksen mitätöinti, luominen, muokkaaminen tai korvaaminen toisella. Kyselyherätetapahtumia ovat pelkkien metatietojen sekä metatietojen ja dokumentin sisällön kysely ja vastaus. Toisin kuin yleisesti HL7 version 3 profiloinnissa, sähköisessä lääkemääräyksessä kaikille herätetapahtumille on yksi RMIM-kaavio ja siitä johdettu HDM-kuvaus [Ere13c].

Määrittelydokumentissa [Ere13a] määritellään pakolliset tietokentät lääkemääräyksen CDA R2 -otsakkeelle. Niitä on 19 kappaletta, esimerkiksi tunniste, mallinnetunniste, tyyppikoodi, tekijä ja omistaja. Suomen HL7 Finland –yhdistys on määritellyt kansallisia kenttiä, jotka eivät ole CDA-arkkitehtuurissa. Tämä ei kuitenkaan tarkoita, etteivät doku-

mentit olisi standardin mukaisia. CDA-standardin toteuttava käsittelijä osaa lukea dokumentit sisäänsä, mutta ei osaa käsitellä kansallisia kenttiä. Nämä kentät ovat hl7fi-nimiavaruudessa, esimerkkinä hl7fi:declaredTime, joka kertoo milloin lääkemääräys on rekisteröity reseptikeskukseen.

Otsakkeen asiakirjatunniste (id) on kahdennettuna runko-osuudessa. Tällä on varmistettu tiedon muuttumattomuus ja helppo saatavuus. Lisäksi dokumentin allekirjoitus kohdistuu vain dokumentin runko-osaan, joten ilman tunnisteen olemassaoloa rungossa ei tunnistetta voitaisi allekirjoittaa. Alkuperäinen lääkemääräys viittaa vain itseensä, mutta korjatun lääkemääräyksen alkuperäisen dokumentin tunniste viittaa alkuperäiseen dokumenttiin. Alkuperäisen dokumentin tunniste on joukkotunniste (setId). Viittaus tapahtuu tallentamalla alkuperäisen dokumentin oliotunniste (OID) joukkotunnisteeseen. Joukkotunniste yhdistää kaikki samasta dokumentista johdetut uudet dokumentit, kuten korjaukset. Usealla dokumentilla voi siis olla sama joukkotunniste, mutta niiden oma oliotunniste on yksilöllinen. Yhteensä erilaisia lääkemääräysdokumentteja on 15. Näitä ovat esimerkiksi lääkemääräys, mitätöinti, korjaus, lukitus, varaus, uusimispyyntö, toimitus ja annosjakelu [Ere13a]. Otsake on jokaiselle dokumenttityypille yhtenäinen [Ere13c].

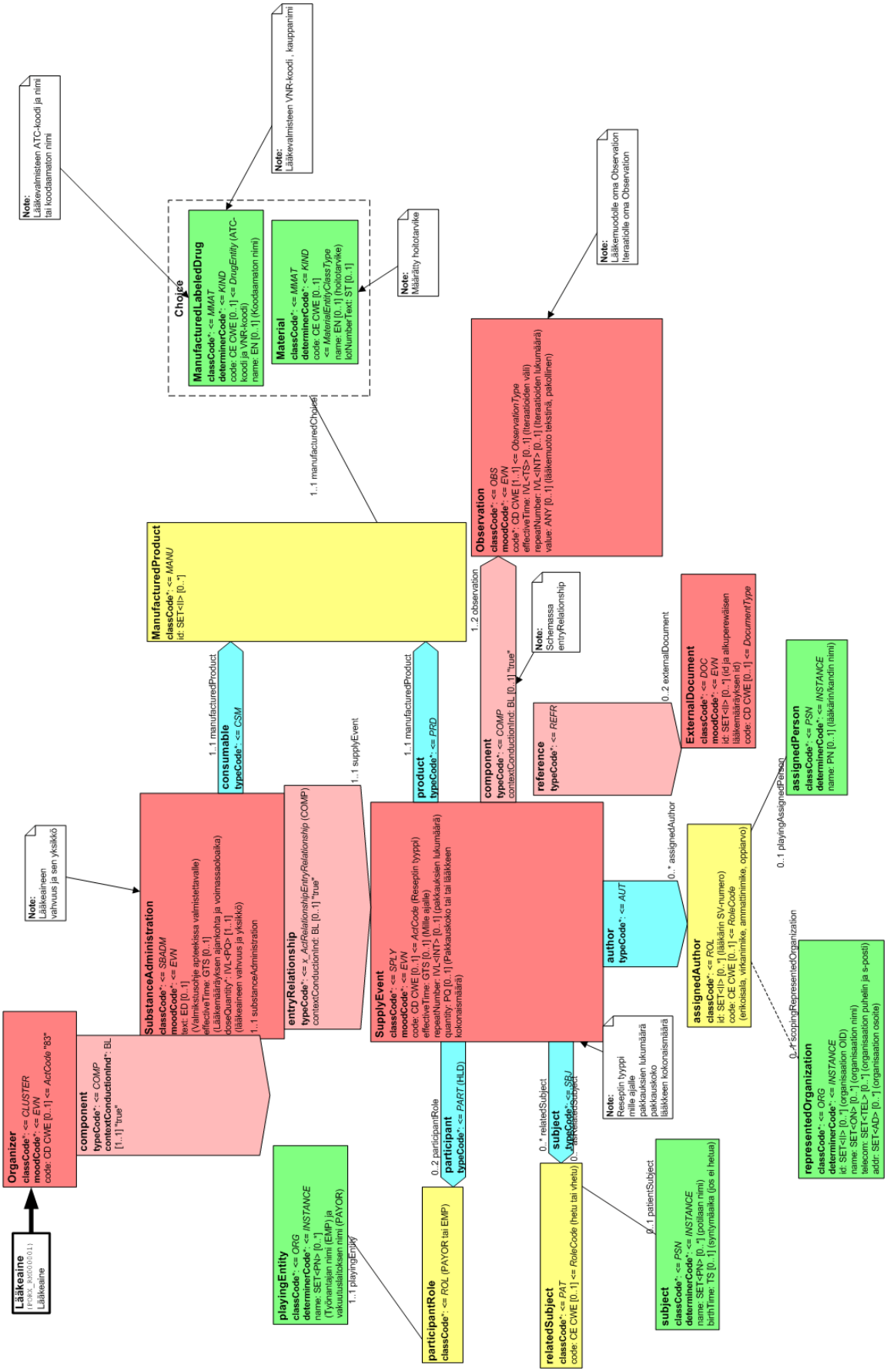
Yhdessä lääkemääräysdokumentissa on vain yhden tapahtuman tiedot. Runko-osan alussa määritellään mallinnetunniste, joka kertoo minkä kerronnallisen mallinteen eli soveltamisoppaan mukaan CDA R2 –dokumentti on toteutettu. Suomen HL7 –yhdistys yhdessä Kelan, ulkopuolisten konsulttien ja järjestelmätoimittajien kanssa ovat luoneet useita versioita sähköisen lääkemääräyksen määrittelydokumenteista. On tärkeää erottaa mitä määrittelyä sanoman toteutuksessa on käytetty, jotta sanoman vastaanottaja tietää minkä sääntöjen mukaan sanomaa tulee tulkita. Esimerkiksi lääkemääräyksellä mallinnetunniste xml-dokumenttissa on <templateId root=”1.2.246.777.11.2008.18” />, jossa root-attribuutin sisällä on määrittelydokumentin oliotunniste. Runko-osassa kaikki kliiniset merkinnät ovat yhden osion sisällä. Tämä ei vastaa Tim Bensonin kirjassaan [Ben10] mainitsemaa yleistä tapaa, missä jokaisella merkinnällä on oma osio. Sen sijaan se vastaa alkuperäistä CDA-arkkitehtuurin määritystä.

Ihmisluettavuus on yksi CDA-dokumentin ominaisuuksista, ja se on toteutettu omana tekstiosanaan osiossa. Sama tieto on sanomassa rakenteellisena, ja se on toteutettu merkinnöissä eli kliinisissä lauseissa. Jokaisella lääkemääräystä koskevalla merkinnällä on järjestäjä (organizer). Järjestäjäelementissä on joko lääkevalmisteen ja pakkauksen tiedot

sekä reseptin perustiedot, lääkkeen vaikuttavat ainesosat, lääkkeen muut ainesosat, annostus tai lääkityksen muut tiedot. Rakenteellisen tiedon pakollisuudet ja sallitut merkkimäärät määritellään taulukossa mallinetunnisteen osoittamassa määrittelydokumentissa [Ere13c]. Esimerkiksi lääkemääräyksessä on pakollinen kenttä SV-numero, jonka maksimi merkkimäärä on kahdeksan merkkiä.

Kuvassa 16 on soveltamisoppaan [Ere13b] RMIM-kaavio lääkemääräyksestä. Sama kaavio on pohjana muillekin sanomatyypeille kuin lääkkeen määräämiselle, jolloin aika, paikka ja tekijät kuvaavat kyseessä olevaa tapahtumaa, esimerkiksi lääkemääräyksen lukituksen purkua. Kaaviossa järjestäjä on lääkeaine, jonka osana on lääkemääräys (substanceAdministation). Molemmat ovat aktiluokan ilmentymiä ja niitä yhdistävä luokka on aktisuhdeluokan ilmentymä. Lääkemääräyksestä on viittaus osallistumisluokan kautta valmistettuun tuotteeseen, joka on rooli joko lääkevalmisteelle tai hoitotarvikkeelle. Lääkevalmiste ja hoitotarvike ovat entiteettiluokan ilmentymiä. Määräyksen tarkemmat tiedot esitetään toimitusluokan (supply) attribuuteilla. Sillä on sama osallistumisviittaus määrättävään lääkkeeseen kuin itse lääkemääräyksellä. Lisäksi toimituksesta on viittaus osallistumisluokan kautta määrättävän lääkkeen myyntiluvan haltijaan. Lisäksi samalla suhteella esitetään lääkemääräystä koskevat tiedot maksajasta, kuten määräyksen kohteen työnantaja tai vakuutusyhtiö. Kohde (subject) ilmoitetaan vastaavan viittausketjun avulla. Lääkemääräyksen tekijä ja organisaatio, jossa lääkemääräys on määrätty, ilmoitetaan sen sijaan tekijä (author) osallistumisluokan avulla. Toimituksella on lääkemuoto ja iterointi, jotka esitetään aktisuhteen kautta havaintoaktina (observation). Alkuperäinen tunniste saadaan runko-osaan allekirjoitusta varten lisätyn toimituksen viittausaktisuhteen avulla. Viittaus tapahtuu alkuperäiseen kliiniseen dokumenttiin, jolla on tunniste. Dokumenteilla, joita ei allekirjoiteta, ei tarvita viittausta kliiniseen dokumenttiin. Näitä ovat esimerkiksi lääkemääräyksen lukituksen purku ja uusimispyyntö.

Järjestäjän ollessa vaikuttavat aineet tai annostus RMIM-kaavio on hieman erilainen. Vaikuttavissa aineissa kuvan 16 kaaviosta on mukana vain yläosa, eli lääkemääräysluokka ja siihen liittyvä osallistumisluokka ja sen viittaus rooliin ja entiteettiin. Annostuksessa sen sijaan järjestäjällä on lääkemääräyksen lisäksi useita osaluokan (component) aktisuhteita havaintoon. Lääkemääräysluokalla on vain viittaus havaintoon.



Kuva 16. Lääkevalmisteen ja pakkauksen tiedot sekä reseptin perustiedot [Ere13b].

Soveltamisoppaassa [Ere13b] on toteuttajille tarkoitettu ohjeistus ylläkuvatun RMIM-kaavion XML-muotoisesta toteutuksesta ja kenttien maksimipituuksista. Toteutuksen tueksi on XML-skeemoja, jotka ovat generoitu käyttäen XML ITS -määrittelyä. Käytännössä dokumentin XML-toteutuksen arkkitehtuuri jää toteuttajan vastuulle, ja soveltamisopas ottaa kantaa vain tietosisällön yksityiskohtiin ja valmiin XML-dokumentin rakenteeseen, ei niinkään miten se on tuotettu.

Soveltamisoppaassa [Ere13b] kuvataan mitä XML-tiedoston tulee sisältää. RMIM-kaaviossa olevat luokat ovat omia XML-merkkejään. Niiden pakolliset kentät, kuten luokkakoodi ja tilakoodi, asetetaan XML-merkkien attribuuteiksi. Sen sijaan muut RMIM-kaaviossa luokalla olevat attribuutit sijoitetaan omiksi XML-merkeiksi, jotka ovat luokan merkkien sisäpuolella. Samoin luokasta lähtevät viittaukset toisiin luokkiin sijoitetaan alkuperäisen luokan XML-merkkien sisään. Esimerkiksi kuvan 14 RMIM-kaaviossa järjestäjäluokan (organizer) sisältää XML-dokumentissa kaikki muut RMIM-kaavion luokat. Kuvassa 17 on esimerkki järjestäjäluokan alkumerkistä. Siinä kuvan 14 RMIM-kaavion järjestäjäluokan koodiattribuutti on omana XML-merkkinään järjestäjäluokan alkumerkin jälkeen. Koodiattribuutin arvo on ulkopuolinen koodisto, joka on avattuna kuvassa 17.

```
<organizer classCode="CLUSTER" moodCode="EVN">
        <code code="83" codeSystem="1.2.246.537.6.12.2002.126" codeSystem-
Name="Lääkityslista" displayName="Lääkevalmisteen ja pakkauksen tiedot"/>
          <statusCode code="completed"/>
```

Kuva 17. Järjestäjäluokan XML-merkki [Ere13b].

Kuvassa 16 olevan lääkemääräysluokan (substanceAdministation) attribuutti lääkeaineen vahvuus (doseQuantity) toteutetaan XML-dokumenttiin omana XML-merkkinään, jolla on sisältönään käännös (translation). Käännöksellä on lisäksi sisältönä vahvuustieto tekstimuotoisena. Vahvuuden tietotyyppi RMIM-kaaviossa on tietoväli (interval ivl). Se mahdollistaa tiedon esittämisen ala- ja ylärajojen avulla. Esimerkiksi vahvuus voi olla vähintään kymmenen milligrammaa ja enintään 20 milligrammaa. Ala- ja ylärajat ilmoitetaan XML-dokumentissa omina ala (low)- ja ylä (high) –merkkeinään. Ylä- ja alarajoja ei ole toteutettu soveltamisoppaassa [Ere13b], mutta siinä on maininta keskimääräisen tiedon (center) käytöstä. Keskimääräinen tieto erottelee vahvuuden määrän ja yksikön toisistaan, toisin kuin tekstimuotoinen tiedon esitystapa.

Lääkemääräyksen voimassaoloaikaan sen sijaan on toteutettu ala- ja ylärajat [Ere13b]. Sen tietotyyppi RMIM-kaaviossa on yleinen aikamääritys (general timing specification,

GTS) [DAB06]. Se voi saada joko yhden arvon tai aikavälin. Jos tarvitaan aikavälin esittämistä, käytetään sen esittävää tietotyyppiä (interval of time, IVL_TS) [Eres13b]. Koska HL7 version 3 tietotyyppi ajan määrittämiselle on hierarkkinen, onnistuu tietotyypin vaihto toteutusvaiheessa. Yleinen aikamääritys on hierarkian alimmaisena, ja sitä voi yksinkertaistaa tarvittaessa aikavälitietotyyppiin [Ben10].

Kuvassa 16 viittaus lääkemääräyksestä osallistumisluokan kulutustavaran (consumable) kautta valmistettu tuote (manufacturedProduct) toteutetaan XML-dokumentissa merkitsemällä kulutustavaralle oma XML-merkki, jonka sisään valmistettu tuote lisätään. Valmistetulla tuotteella on XML-merkkiensä sisällä itse entiteettiluokka joko lääkevalmiste tai hoitotarvike. Kuvassa 18 on esimerkki hoitotarvikkeen XML-esityksestä.

```
<organizer>
 …
 <component>
  <substanceAdministration>
   …
   <consumable>
     <manufacturedProduct>
<manufacturedMaterial>
                <code nullFlavor="NI"/>
                <name>DUODERM EXTRA THIN 10X10CM</name>
       </manufacturedMaterial>
     </manufacturedProduct>
   </consumable>
   …
  </substanceAdministration>
 </component>
</organizer>
```

Kuva 18. Hoitotarvike XML-dokumentissa [Ere13b].

Vastaavasti kuin kulutustavara, muut liitosluokkia käyttävät viittaukset esitetään XML-dokumentissa sisäkkäisinä rakenteina. Tässä muutoksessa häviää RMIM-kaaviossa oleva semantiikka luokkien välillä, ja jäljelle jää vain sisäkkäinen tekstimuotorakenne. XML-dokumentissa tiedetään vain, että toinen merkki on toisen vanhempi, mutta suhteen laatua ei suoraan voi kuvata.

9.2 Mallinteet lääkemääräyksessä

Sähköisen lääkemääräyksen mallinteet ovat kerronnallisia ja lisäksi valmiin XML-dokumentin tarkistusta varten on XML-skeema.

Kerronnalliset mallinteet ovat soveltamisoppaita, ja niitä on omansa otsakkeelle [Ere13a] ja runko-osalle [Ere13b]. Oppaat käyvät yksityiskohtaisesti läpi kenttien maksimipituudet ja järjestykset.

XML-skeemat ovat luotu käyttäen XML ITS –määritystä. Osa XML-skeemoista ovat yleisiä CDA-arkkitehtuurin mukaisia skeemoja, joissa on kuvattu kaikki RIM-mallin luokat ja tietotyypit. Lisäksi Suomen paikalliset skeemat sisältävät Suomen HL7-yhdistyksen lisäämät ja muokkaamat kentät. Suomen määrityksissä esimerkiksi kenttien pakollisuuksia on poistettu. Skeemat rajoittavat XML-dokumenttia määrittelemällä attribuuttien esiintymiskerrat, tietotyypit ja valinnaisuudet [Ere13]. Valinnaisuudella tarkoitetaan mahdollisuutta valita jokin annetuista vaihtoehdoista, ja se on XML-skeeman vakio-ominaisuus [XML04]. Jokaiselle sanomalle on oma XML-skeemansa. Sanomia ovat esimerkiksi reseptien haku tai lähetys Reseptikeskukseen.

HL7 version 3 RIM-viitemalli määrittelee luokkien attribuutit ja niiden tietotyypit. RIM-mallista johdettu RMIM-kaavio profiloi mallia monistamalla ja rajoittamalla sitä. RMIM-kaaviossa on mukana attribuuttien ja suhteiden esiintymiskerrat, jotka siirtyvät suoraan XML-skeemaan. Vaikka XML-skeema tukee kenttien maksimipituuden määrittämistä, sitä ei RMIM-kaaviosta saada. Maksimipituudet on kerrottu kerronnallisissa soveltamisoppaissa [Ere13a] ja [Ere13b].

Schematronia [Vli07] ei ole käytetty sähköisen lääkemääräyksen XML-dokumenttien tarkistuksissa. Sillä voisi tarkistaa esimerkiksi vastaako sanomaa koskevan potilaan henkilötunnus ja syntymäaika toisiaan tai tarkistaa sanoman kenttien maksimipituudet.

Schematronilla voi tarkistaa ehtoja, joita on sovellusoppaassa [Ere13b]. Esimerkiksi myyntiluvan haltija on osallistumisluokan ilmentymä, ja sillä on aina sama tyyppikoodi sekä sen viittaamalla roolilla on aina sama luokkakoodi. Vakuutusyhtiön ja työnantajan esittämisessä on samankaltainen rakenne kuin myyntiluvan haltijalla, mutta eri luokkakoodin ja tyyppikoodin arvoilla. Toinen esimerkki on lääkeaineen määräämistä aikavälille. Tällöin määräämisen koodi on kolme ja aikamäärе on pakollinen. Kuvassa 19 on Schematron-sääntö, joka tarkistaa ajan olemassaolon, jos reseptin määräämistapa on

ajalle. Kuvassa esiintyvä koodisto on Määrätyn määrän esittämistapa, ja sen arvot löytyvät koodistopalvelimelta [THL13]. Sääntö ilmoittaa virheilmoituksen, jos määrätyn määrän esittämistapa on ajalle ja koodielementin lapsena ei ole aikaa.

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
  <title>Esimerkki Schematron-sääntö.</title>
  <pattern>
    <rule context=”code[@codeSystem=’1.2.246.537.5.40100.2006’]”>
      <report test=”@code = ’3’ and not(child::effectiveTime)”>Ajalle määrätessä aikaväli on pakollinen.</report>
    </rule>
  </pattern>
</schema>
```

Kuva 19. Schematron-sääntö aikavälin olemassaolon tarkistamiseen.

Schematronilla voi tarkistaa kuuluuko attribuutin arvo johonkin joukkoon. Esimerkiksi lääkevalmisteen, jolla ei ole VNR-koodia (pohjoismainen tuotenumero), tulee ilmoittaa asia koodielementin attribuutissa tyhjä arvo (nullFlavor) joko arvolla UNK (tuntematon, unknown) tai NI (tyhjä, nil). Schematronilla tarkistus kohdistuu koodielementtiin, jolla on koodistona VNR-koodiston oliotunniste. Sille elementille tarkistetaan onko tyhjää arvoa esittävä attribuutti olemassa, ja jos on, saako se jommankumman arvon. Virhe ilmoitetaan, jos arvona ei ole kumpaakaan yllä mainituista arvoista.

Sovellusoppaan [Ere13b] mukaan puhelinnumeron erottelu välilyönneillä on kielletty. Schematronilla voi tarkistaa elementin tai attribuutin arvon ja ilmoittaa virheestä, jos se sisältää välilyönnin. Kuvassa 20 on esimerkki tarkistuksesta.

```
<schema xmlns=”http://purl.oclc.org/dsdl/schematron”>
  <title>Esimerkki Schematron-sääntö.</title>
  <pattern>
    <rule context=”telecom”>
      <report test=”contains(.,' ')">Puhelinnumero ei saa sisältää välilyöntiä.</report>
  </rule>
  </pattern>
</schema>
```

Kuva 20. Puhelinnumeron välilyönnin tarkistus Schematronilla.

Lääkemääräyksen annostelun alkuaika ja kesto ilmoitetaan omassa aikaelementissään (effectiveTime). Se voi saada alkuajan (low) lisäksi vain joko keston (width) tai loppuajan (high). Schematronilla tarkistus tapahtuu valitsemalla kontekstiksi aikaelementin ja tarkistamalla, ettei elementillä ole kuin kaksi lasta ja, että ensimmäinen on alkuaika. Esimerkki tarkistuksesta on kuvassa 21.

```
<schema xmlns=”http://purl.oclc.org/dsdl/schematron”>
  <title>Esimerkki Schematron-sääntö.</title>
  <pattern>
    <rule context=”effectiveTime”>
      <report test=”count(*) = 2 and *[1] = low">Aikaelementillä tulee olla kaksi lapsielementtiä ja ensimmäisen lapsista tulee olla alkuaika.</report>
    </rule>
  </pattern>
</schema>
```

Kuva 21. Annostelun aikamääreiden tarkistus Schematronilla.

Lääkemääräyksen mitätöintisanomassa siirretään mitätöinnin syy omana koodistonaan. Syykoodi-elementin lapsielementissä on ilmoitettu mitätöinnin osapuoli eli joko lääkäri, apteekki tai reseptikeskus. Sovellusoppaassa [Ere13b] kerrotaan sallitut yhdistelmät syyn ja osapuolen välillä. Sallittujen yhdistelmien tarkistus onnistuu Schematroinilla, mutta tällöin yhdistelmien muutos vaatii muutoksia Schematron-skeemaan.

10 Kritiikkiä HL7-standardista

HL7 versio 3 on suosittu standardi, mutta sen käyttöönotto on ollut hidasta [SmC06]. Tässä kappaleessa käsittelen HL7 RIM ja version 3 saamaa kritiikkiä sekä miten RIM-mallin suunnittelijat vastaavat saamaansa kritiikkiin.

Smith ja Ceuster listaavat artikkelissaan [SmC06] RIM-viitemallin ongelmiksi toteutuksen monimutkaisuuden, käytettävyyden erikoisilla aihealueilla, teknisiä ongelmia, vanhanaikaisuuden, liikkuma-alan, HL7 version 3 liiketoiminta- ja johtamismallin, dokumentoinnin, opittavuuden ja markkinoinnin. Smith ja Ceuster kritisoivat itse HL7-standardin lisäksi HL7-organisaatiota. Heidän mukaansa konsultit, jotka hyötyvät rahallisesti standardin monimutkaisuudesta, osallistuvat standardin kehitystyöhön aktiivisesti. He kritisoivat version 3 muutoksien tekemisprosessia hitaaksi ja hankalaksi, ketterästi ja no-

peasti kehittyvien ohjelmistojen aikakaudella. Heidän mukaansa HL7-dokumentaatio sisältää sisäisiä ristiriitoja ja epäselvyyksiä. Esimerkkinä he esittävät aktiluokan (act), jonka englanninkielistä nimeä on käytetty isolla ja pienellä alkukirjaimella ja olioiden, luokkien ja ilmentymien etuliitteenä (Act-object, Act-instance, ActClass). Heidän mukaansa standardin opittavuus kärsii huonon dokumentaation, lukemattomien erikoistapausten ja monimutkaisten ongelmien sanoman muodostuksessa vuoksi. Smith ja Ceuster pitävät RIM-viitemallin kuutta pääluokkaa ja niiden attribuutteja riittämättöminä mukautua kaikkiin käyttötapauksiin. He kritisoivat attribuuttien korvaamista tilanteeseen sopivilla, ja vertaavat prosessia ruohonleikkurin ohjelmistosuunnitteluun, jossa poistetaan lentokoneen ohjeesta turhat attribuutit. He pitävät HL7 version 3 lähtökohtaa vääränä. Versio 3 on heidän mukaansa alun perin kehitetty Yhdysvaltojen sairaaloiden tarvitsemiin sanomiin, mikä näkyy vieläkin viitemallissa historian painolastina. Schadow ja kumppanit kuitenkin artikkelissaan [SMW06] kumoavat tämän väitteen ja pitävät sitä myyttinä. Smithin ja Ceusterin mukaan tapahtumamalli ei sovi kaikkiin tapauksiin, esimerkkinä he käyttävät perimätutkimusta, missä tapahtumien sijaan keskiössä ovat prosessit ja reaktiot.

Smith ja Ceuster käsittelevät viitemallissa käsiteltävien tapahtuman tiedon ja sen itsensä eroa. Tapahtuman tiedot ovat ne, jotka tallennetaan sanomaan, ja itse tapahtuma on asia, joka sairaalassa tapahtuu. Heidän mukaansa RIM-viitemallin dokumentaatio sekoittaa virheellisesti nämä kaksi asiaa yhdeksi, vaikka heidän mukaansa RIM-mallin puolustajat väittävät viitemallin käsittelevän vain itse tapahtumia. Smith ja Ceuster kritisoivat RIM-mallin olevan yhtä aikaa tietomalli ja viiteontologia. Tämä tarkoittaa sitä, että lääkityksen lopettamisessa käsitteellisesti lääkitys loppuu ja lääkitysolion tilaa muutetaan aktiivisesta lopetetuksi [SmC06]. Schadow ja kumppanit vastaavat RIM-mallin mallintavan reaalimaailman asioita olio-ohjelmoinnin periaatteen mukaisesti tyyppimuuntamalla tiedot tietomalliin, jossa on intuitiiviset nimet luokille, esimerkiksi Ihminen tai Toimenpide. Version 3 dokumentaatio implisiittisesti olettaa tietojen tallennuksen tapahtuvan esimerkiksi potilastietojärjestelmään, mutta tallennus ei koske koko reaalimaailman asiaa, kuten potilasta tai toimenpidettä [SMW06].

Smith ja Ceuster kritisoivat lisäksi RIM-mallin tapaa esittää tietoa, mikä ei mahdollista päättelykoneiden käyttämää automaattista saman olioiden yhdistämistä itseensä. Heidän mukaansa se onnistuu vain, jos tietomallissa käsitellään olioita suoraan, esimerkiksi kasvaimen ympärillä olevan kudoksen turpoaminen aiheutui nesteen kasautumisesta. Sen sijaan RIM-tietomalli pakottaa jakamaan esimerkkitapauksen kolmeen osaan. Havaintoon,

jonka teki lääkäri, että kasvaimen ympärillä oleva kudos on turvonnut. Havaintoon, jonka lääkäri teki, että nestettä on kasaantunut alueelle. Lisäksi tarvitaan näiden kahden havainnon yhdistävä viittaus, jonka syykoodi on aiheuttaa ja tapakoodina jo tapahtunut [SmC06]. Schadow ja kumppanit vastaavat entiteettien kuvaavan tietoa, joka on tiedossa ennen tapahtumaa, ja havaintoaktien kuvaavan prosessin aikana saatavaa tietoa tai sitä ennen tiedossa olevaa tietoa, joka ei ole yleisesti hyväksytty. Esimerkiksi lääkkeiden pitoisuus mallinnetaan entiteetissä, mutta sen väri mallinnetaan havaintoaktissa. Ilman jakoa tulisi entiteeteistä tehdä useita versioita, joita olisi vaikea ylläpitää.

Akteja Smith ja Ceuster kritisoivat väittämällä niiden olevan määritelmänsä vastaisesti itse tapahtumia, eikä kirjauksia tapahtumista. Esimerkkinä he käyttävät potilastapaamista (patientEncounter), joka heidän mukaansa on tapahtuma. Heidän mukaansa aktin ilmentymä on lausunto jostain mitä hoitohenkilö on joskus kuullut, nähnyt, ajatellut tai tehnyt. Aktiluokka, sen sijaan on koodattu, attribuutteja sisältävä luokka tällaisista ilmentymistä. Heidän mukaansa tässä ongelmana on aikaisemmin mainittu ero viiteontologian ja tietomallin välillä. Smith ja Ceuster uskovat, että ero on RIM-mallin käyttäjien tiedossa ja käyttäjät käyttävät HL7 RIM-mallia sen mukaan [SmC06]. Schadow ja kumppanit myöntävät HL7 version 3 dokumentin ristiriitaisuudet ja heidän mukaansa niihin on puututtava. Heidän mukaansa ongelma johtuu usean eri tekijän muokatessa määritelmää [SMW06].

Smith ja Ceuster ehdottavat kritiikkinsä ratkaisuksi RIM-viitemallin jakoa kahteen eri käsitemalliin. Toinen olisi terveydenhuollon viittausontologia (Reference Ontology of the Healthcare Domain), joka sisältäisi kaikki yksittäiset oliot terveydenhuollosta, kuten potilas, sairaus, tulehdus tai prosessi. Toinen osa olisi terveydenhuoltotiedon malli (Model of Healthcare Information), joka sisältäisi asioita kuten dokumentit, havainnot, kirjaukset ja sanomat. Heidän mukaansa tällä tavalla terveydenhuollon mallin ei tarvitsisi olla tapahtumakeskeinen ja ilman aktiluokkaa voisi tehdä dokumentteja. Heidän mukaansa ehdotus antaisi yhtenäisen pohjan HL7-organisaation pyrkimykselle [SmC06]. Schadow ja kumppanit kritisoivat ratkaisua ja pitävät vaarallisena tehdä toinen malli erilleen RIM-mallista. He uskovat, että ihmiset olettaisivat RIM-mallin kuvastavan tätä mallia ja kysyvät miksi malleja tulisi olla kaksi, jos toinen kuvastaa toista. Heidän mukaansa yksi oikeaa maailmaa kuvaava malli, jossa on määritellyt tiedonhallinnan funktiot, riittää. Tiedonhallinnan funktiot määrittelevät kuinka ihmiset ja ohjelmistot sovittavat näkökulmansa, viestivät aikomuksensa ja lopulta vaikuttavat reaalimaailmaan muutoksillaan [SMW06].

Schadow ja kumppanit artikkelissaan [SMW06] vastaavat myös muuhun saamaansa kritiikkiin. Esimerkiksi RIM-mallin on sanottu rikkovan UML-mallinnuksen ja olioajattelun periaatteita. Schadow ja kumppanit vastaavat RIM-mallin olevan UML-malli, ja sitä ylläpidetään HL7-organisaation työkaluilla. Aihealuemallit (domain-specific model) voidaan suunnitella ensin rajoittamattomina UML-aihealueanalyysimalleina (domain analysis models, DAM), mutta ne muutetaan profiloinnin aikana rajoitetuiksi normalisoiduiksi UML-malleiksi, jotka ovat tarkennuksia RIM-mallin luokista. Profiloinnin aikana suoritettavaa attribuuttien poistoa ja luokkien kloonausta on kritisoitu. Kriitikot väittävät attribuuttien poiston rikkovan perimisen ajattelutapaa, mutta Schadow ja kumppanit pitävät attribuuttien poistoa projektiona, jota käytetään relaatiotietokannoissa.

Schadow ja kumppanit vastaavat HL7-organisaation saamaan kritiikkiin, ettei HL7 ole käyttänyt alan käytettyjä standardeja, sen sijaan HL7 on kehittänyt omia tapojaan tehdä asioita. Schadown ja kumppanien mukaan HL7 on alusta lähtien käyttänyt standardeja, mutta kaikkea ei ole ollut saatavilla HL7 version 3 kehitystyön alkuvaiheessa, joten he ovat joutuneet kehittämään ne jo ennen kuin niistä on tullut standardeja.

Vizenor artikkelissaan [Viz04] kritisoi HL7 RIM-viitemallin käyttämiä puhetoimituksia. Hän esittää kolme kritiikkiä: puhetoimitukset eivät ole attribuuttisia akteja, RIM ei erottele tapahtuvia ja pysyviä asioita ja RIM tarvitsee enemmän perusteellisia tapahtumien syitä.

Vizenorin mukaan puhetoimitukset eivät vain kuvasta reaalimaailman asioita, vaan ne ovat niitä. Sen sijaan raportit puhetoimituksista, eli niiden tuottama tulos, ei ole reaalimaailman tapahtuma. Hänen mukaansa RIM viittaa reaalimaailman tapahtumiin puhetoimitusten välityksellä, mikä tekee attribuuttisista akteista puhumisen harhaanjohtavaksi. Vaihtoehtona hän esittää, että RIM ei viittaa reaalimaailman tapahtumiin, mikä tarkoittaa, ettei viitemallissa ole puhetoimituksia [Viz04]. Schadow ja kumppanit myöntävät, ettei esimerkiksi rokotus yksinään ole puhetoimitus, mutta heidän mukaansa RIM-viitemallissa ei käsitellä pelkkää rokotusta, vaan sen sijaan joku pyytää sitä tai raportoi siitä. Puhetoimituksen ehdollinen sisältö on rokotuksen antamisen reaalimaailman tapahtuman peilikuva (reflection). Schadown ja kumppanien mukaan ehdollinen sisältö kuuluu puhetoimitukseen, kuten kolikon molemmat puolet kuuluvat kolikkoon [SMW06].

Toisen kritiikin erottelun tapahtuvien (aktien) ja pysyvien (dokumenttien) asioiden eron

puuttumisen Vizenor perustelee niin, että RIM-mallin dokumentaation mukaan tietojoukko on joukko attribuuttisia akteja. Hänen mukaansa attribuuttisia akteja ei voi tallentaa, päivittää tai täydentää. Sen sijaan dokumenteille voi tehdä näitä toimenpiteitä. Hänen mukaansa CDA-arkkitehtuurin määrittelemistä dokumentin ominaisuuksista pysyvyys ja vastuullisuus eivät sovi attribuuttisille akteille. Hänen mukaansa tämä viittaa siihen, ettei RIM-mallissa erotella attribuuttisia akteja ja dokumentteja toisistaan. Tästä seuraa, että näiden asioiden erottamisen seuraamisesta saama informaatio hävitetään [Viz04]. Schadow ja kumppanit puolustautuvat attribuuttinen sanan käyttöä yleiseksi käytännöksi, joka sitoo olion attribuutit yhteen. Heidän mukaansa se ei kumoa illokuutiota, sen sijaan se mahdollistaa valmistavat olosuhteet illokuution olemassaololle. He lisäävät ehdollisen sisällön ja attribuuttien olevan erottamaton osa puhetoimituksia RIM-maailmassa, koska toisin kuin reaalimaailmassa puheen tekijä ei ole implisiittinen [SMW06].

Viimeiseen kritiikkiin Vizenor ottaa esimerkiksi sopimuksen, joka on RIM-viitemallissa akti. Aktilla on suhteet entiteetteihin osallistuja- ja rooliluokan kautta. Entiteetit ovat sopimuksen osapuolia. Vizenor erottelee aktin tapahtuvaksi, entiteetit pysyviksi ja itse sopimuksesta tehtävän dokumentin fyysiseksi asiaksi, joka tallennetaan. Hänen mukaansa näiden erityyppisten asioiden eroja tulisi seurata, koska puhetoimitus, sen illokuutio ja sen olemassaolon tallentama dokumentti käyttäytyvät eri tavalla [Viz04]. Schadow ja kumppanit vastaavat dokumentin olevan erikoismuoto puhetoimituksesta. RIM-viitemalli tarkastelee puhetoimituksen jälkiehtoja tiloina kommunikaatiojärjestelmässä, ja että puhetoimitus itse ylläpitää niitä [SMW06].

11 Yhteenveto

Yhteentoimivuus terveydenhuollon alan järjestelmien välillä on ongelma, koska järjestelmät on tehty vuosien mittaan ja niillä on ollut eri tekijöitä. Yhteentoimivuutta ei ole alun perin suunniteltu järjestelmiin. Yhden tai useamman järjestelmän kommunikointi keskenään vaatii yhteisten viestien suunnittelua ja toteutusta.

Yksi ratkaisu yhteentoimivuudelle järjestelmien välillä tarjoaa HL7-organisaatio. Organisaatio on tehnyt kaksi terveydenhuollon viestivälitysstandardia (HL7 versio 2 ja 3). Versio 3 perustuu RIM-viitemalliin. Se on aihealueen kattava tietomalli, josta sanoman profilointi aloitetaan.

Profiloinnilla tarkoitetaan viitemallin tarkentamista tiettyä käyttötapausta varten. Esimerkiksi potilaan laboratoriokokeen tulokset eivät vaadi kaikkea RIM-viitemallin tietosisältöä, sen sijaan viestin suunnittelija profiloi mallista tarvittavan sisällön RMIM-kaavioon. RMIM-kaavio on RIM-mallin alijoukko, jossa on käytetty RIM-mallin luokkien kloonaamista sekä attribuuttien ja luokkien poistamista. Näin vain tarvittava tieto siirretään. RMIM-kaaviosta luodaan HMD-kuvauksen perusteella XML-skeema, jota vasten tietojärjestelmän luoma XML-dokumentti voidaan tarkistaa. CDA-dokumenteilla voi XML-skeeman lisäksi olla mallinteita, jotka voivat olla kerronnallisia toteutusoppaita tai Schematron-tarkistuksia.

Schematron-tarkistukset tarjoavat XML-dokumentin oikeellisuuden tarkistamiseen uusia työkaluja. Niillä voi tehdä tarkistuksia, joita ei pelkällä XML-skeemalla voi tehdä, kuten elementin attribuuttien vaihtoehtoinen olemassaolo, eli vain elementin yhdellä attribuutilla voi olla arvo kerrallaan. Schmatron-tarkistuksia käytetään täydentämään CDA-profiloinnilla saatua XML-skeemaa eikä korvaamaan sitä.

Suomessa toteutetun sähköisen lääkemääräyksen profiloinnissa on luotu yksi RMIM-kaavio kaikille mahdollisille sanomatyypeille. RMIM-kaaviosta on johdettu XML ITS –määritystä käyttäen XML-skeemoja, jotka rajoittavat valmiin XML-dokumentin tarkistusta tietotyyppien, pakollisten elementtien ja attribuuttien osalta. Sovellusoppaat [Ere13a] ja [Ere13b] käyvät yksityiskohtaisesti läpi XML-dokumentin sisällön ja määrittävät ehtoja elementtien tekstiarvojen ja attribuuttiarvojen pituuksille. Sovellusoppaissa on rajoitettu XML-dokumentin rakennetta esimerkiksi joissain tilanteissa vain osa elementeistä saa olla olemassa ja osassa ei. Sovellusoppailla on omat oliotunnisteet, jotka on lisätty XML-dokumentin mallinnetunnisteen (templateId) arvoksi. Tällä tavoin dokumenttia luettaessa tiedetään mihin sovellusoppaaseen toteutus on perustunut. Varsinaista automaattista tarkistusta ei XML-skeemojen lisäksi ole.

HL7 versio 3 ja RIM ovat saaneet osaksi kritiikkiä, kuten olio-ohjelmoinnin periaatteiden rikkominen ja puhetoimitusten väärä käyttö. Kriitikot Smith ja Ceuster [SmC06] haluaisivat jakaa RIM-viitemallin kahtia, missä toisessa olisi tietosisältö ja toisessa viittausontologia. RIM-mallin suunnittelijat ovat vastanneet kritiikkiin ja heidän mukaansa [SMW06] kritiikki on johtunut pääasiassa väärinkäsityksistä.

HL7 versio 3 ja sen käyttötapa CDA-arkkitehtuuri tarjoavat yhteentoimivuutta kliinistä

tietoa käsittelevien tietojärjestelmien välille. Niiden avulla lähettävä ja vastaanottava järjestelmä käsittelevät siirrettävää tietoa samalla tavalla, koska tiedon sisältö on määritelty molemmille yhteisellä RIM-viitemallilla ja ulkopuolisilla koodistoilla. Mallinteilla CDA-profiilin tekijä voi ohjata viestien toteuttajia ja parantaa yhteentoimivuutta. Kerronnalliset mallinteet voivat kuitenkin olla monitulkintaisia, ja Schematron-tarkistusten tekeminen raskasta. XML-skeema ei yksinään riitä esittämään RMIM-kaavion ja CDA-arkkitehtuurin kaikkia vaatimuksia.

HL7 versio 3 on osittain semanttinen viestinvälitysstandardi. Se perustuu yhteisiin sopimuksiin, mutta siirrettäviä viestejä ei pysty ymmärtämään ilman sopimusten tai RIM-viitemallin tietosisältöä. Kokonaan semanttinen standardi vaatisi semanttisen verkon teknologioiden käyttöä, kuten OWL-ontologiaa ja RDF-tiedonvälitysstandardia. Tällöin sanomassa kulkisi mukana tarvittava tieto sanoman tulkitsemiseen missä tahansa järjestelmässä.

Lähteet

BBG10 Idoia Berges, Jesus Bermudez, Alfredo Goñi ja Arantza Illarramendi: Semantic Interoperability of Clinical Data. *Proc. of the First Internat. Workshop on Model-Driven Interoperability*, Lokakuu 2010, sivut 10-14.

BCC00 Suzanne Bakken, Keith E. Campbell, James J. Cimino, Stanley M. Huff ja W. Ed Hammond: Toward Vocabulary Domain Specifications for Health Level 7–coded Data Elements. *Journal of the American Medical Informatics Association, Vol. 7 No. 4*, Heinäkuu / Elokuu 2000, sivut 333-342.

Bee98 George W. Beeler: HL7 Version 3—An object-oriented methodology for collaborative standards development. *Internat. Journal of Medical Informatics 48*, 1998, sivut 151-161.

Ben10 Tim Benson: *Principles of Health Interoperability HL7 and SNOMED*. Springer, 2010.

BEP06 B. G. M. E. Blobel, K. Engel ja P. Pharow: Semantic Interoperability HL7 Version 3 Compared to Advanced Architecture Standards. *Methods Inf Med, Vol. 45 No. 4*, 2006, sivut 343-353.

BLD05 Veli Bicer, Gokce Banu Laleci, Asuman Dogac ja Yildiray Kabak: Providing Semantic Interoperability in the Healthcare Domain through Ontology Mapping. *Proc. of eChallenges*, Lokakuu 2005.

Boo11 Keith W. Boone: *The CDA Book*. Springer, 2011.

ClD99 James Clark ja Steve DeRose: *XML Path Language (XPath)*, W3C Recommendation, Marraskuu 1999.

CLC07 Y. J. Chang, J. S. Lai, P. H. Cheng ja Faipei Lail: Portable CDA for the exchange of clinical documents. *9th Internat. Conf. on e-Health Networking, Application and Services*, Kesäkuu 2007, sivut 1-5.

DAB06 Robert H. Dolin, Liora Alschuler, Sandy Boyer, Calvin Beebe, Fred M. Behlen, Paul V. Biron, Amnon Shabo: HL7 Clinical Document Architecture, Release 2. *J Am Med Inform Assoc. 2006 Jan-Feb; 13(1)*, 2006, sivut 30–39.

Dim04 Health Informatics - Point-of-care Medical Device Communication - Part 10201: Domain Information Model, ISO/IEEE 11073-10201:2004(E), 2004.

Dog12 Asuman Dogac: Interoperability in eHealth Systems. *Proc. of the VLDB Endowment , Vol 5, No 12*, Elokuu 2012, sivut 2026-2027.

DRB98 Robert H. Dolin, Wes Rishel, Paul V. Biron, John Spinosa ja John E. Mattison: SGML and XML as Interchange Formats for HL7 Messages. *Proc AMIA Symp*, 1998, sivut 720-724.

EAR05 Marco Eichelberg, Thomas Aden, Jörg Riesmeier, Asuman Dogac ja Gokce B. Laleci: A Survey and Analysis of Electronic Healthcare Record Standards. *ACM Computing Surveys, Vol. 37, No. 4*, Joulukuu 2005, sivut 277–315.

eAr13 eArkisto Potilastietojärjstelmien käyttötapaukset. http://www.kanta.fi/documents/10180/3437452/eArkisto+PTJ-kayttotapaukset+v2.3.pdf/2e658eea-9bfa-4e0d-bf3c-abd38d6bb357 [16.2.2013].

Ere13 Määrittelyt eReseptille. https://www.kanta.fi/maarittelyt-ereseptille [16.2.2013].

Ere13a Lääkemääräyksen CDA R2 Header, OID: 1.2.246.777.11.2011.17, versio 3.0, 1.7.2011.

Ere13b Lääkemääräyksen sanomat CDA R2-rakenteena, OID: 1.2.246.777.11.2011.18, versio 3.0, 1.9.2011.

Ere13c Lääkemääräyksen Medical Records sanomat, OID: 1.2.246.777.11.2011.19, versio 3.0, 1.9.2011.

Ere13d Linkkien käyttö, OID: 1.2.246.777.11.2004.7, 8.1.2004.

ETD07 T. J. Eggebraaten, J. W. Tenner ja J. C. Dubbels: A health-care data model based on the HL7 Reference Information Model. *IBM Systems Journal, Vol 46, No 1*, 2007, sivut 5-18.

Goo04 William Goossen: Model once, use multiple times: reusing HL7 domain models from one domain to the other. *Studies in Health Technology and Informatics Vol. 107*, Helmikuu 2004, sivut 366–370.

HDS04 Jagbir S. Hooda, Erdogan Dogdu ja Raj Sunderraman: Health Level-7 Compliant Clinical Patient Records System. *Proc. of the 2004 ACM symposium on Applied computing*, Maaliskuu 2004, sivut 259-263.

Hei95 Sandra Heiler: Semantic Interoperability. *ACM Computmg Surveys, Vol. 27, No 2*, Kesäkuu 1995, sivut 271-273.

HL713 About HL7 http://www.hl7.org/about/index.cfm?ref=nav [30.10.2012].

HMN12 M. J. Hurrell, T. G. Monk, A. Nicol, A. N. Norton, D. L. Reich ja J. L. Walsh: Implementation of a standards-based anaesthesia record compliant with the health level 7 (HL7) clinical document architecture (CDA). *Journal of Clinical Monitoring and Computing, Vol 26, No 4*, Elokuu 2012, sivut 295-304.

HNB03 Kai U. Heitman, Dale Nelson ja Paul V. Biron: HL7 XML Implementable Technology Specification for Version 3 Composite Messages. ftp://ftp.ihe.net/International/Europe/HL7%20V2.5/Version%203%20Ballot%20Cycle%204%20-%20DRAFT/v3ballot4/html/foundationdocuments/itsxml/messaging-its-xml.htm [3.7.2013].

HSD03 Kai U. Heitmanna, Ralf Schweigerb ja Joachim Dudeck: Discharge and referral data exchange using global standards the SCIPHOX project in Germany. *Internat. Journal of Medical Informatics, Vol. 70, No 2-3*, Heinäkuu 2003, sivut 195-203.

KiC09 Hwa Sun Kim ja Hune Cho: Health Level 7 Development Framework for Medication Administration. *CIN: Computers, Informatics, Nursing & Vol. 27, No. 5*, 2009, sivut 307–317.

KKL12 Wajahat Ali Khan, Asad Masood Khattak, Sungyoung Lee, Maqbool Hussain, Bilal Amin ja Khalid Latif: Achieving interoperability among healthcare standards: building semantic mappings at models level. *Proc. of the 6th Intern. Conf. on Ubiquitous Information Management and Communication, No. 101*, Helmikuu 2012.

Lah10 Lähetteen ja hoitopalautteen CDA R2-rakenne, OID: 1.2.246.777.11.2009.9, 3.8.2010 [4.5.2013].

LaS11 Jobst Landgrebe ja Barry Smith: The HL7 Approach to Semantic Interoperability. *ICBO: International Conference on Biomedical Ontology*, Heinäkuu 2011, sivut 139-146.

Mea06 Charles N. Mead: Data Interchange Standards in Healthcare IT—Computable Semantic Interoperability: Now Possible but Still Difficult, Do We Really Need a Better Mousetrap? *Journal of Healthcare Information Management, Vol. 20, No. 1*, 2006, sivut 71-78.

NAD08 Tuncay Namli, Gunes Aluc, ja Asuman Dogac: An Interoperability Test Framework for HL7-Based Systems. *IEEE Transactions On Information Technology In Biomedicine, Vol. 13, No. 3*, Toukokuu 2009, sivut 389-399.

OeB05 Frank Oemig ja Bernd Blobel: Does HL7 Go towards an Architecture Standard? *Connecting Medical Informatics and Bio-Informatics*. ENMI, 2005, sivut 761-766.

PuP12 Juha Puustjärvi ja Leena Puustjärvi: Ontology-Based Integration of Clinical Documents. *Proc. of the 14th Internat. Conf. on Information Integration and Web-based Applications & Services*, 2012, sivut 342-347

RIM13 HL7 Reference Information Model http://www.hl7.org/implement/standards/rim.cfm [10.3.2013]

Ros13 RoseTree http://gforge.hl7.org/gf/project/rose-tree/ [29.3.2013]

Rya06 Amanda Ryan: Towards Semantic Interoperability in Healthcare: Ontology Mapping from SNOMED-CT to HL7 version 3. *In AOW '06 Proc. of the second Australasian workshop on Advances in ontologies, vol. 72*. 2006, sivut 69-74.

SaD08 Kamran Sartipi ja Azin Dehmoobad: Cross-Domain Information and Service Interoperability. *Proc. of the 10th Internat. Conf. on Information Integration and Web-based Applications & Services. New York: ACM*, 2008, sivut 25–32.

SaS12 Shelly Sachdeva ja Subhash Bhalla: Semantic Interoperability in Standardized Electronic Health Record Databases. *ACM Journal of Data and Information Quality, Vol. 3, No. 1, Article 1*, Huhtikuu 2012, sivut 1-37.

SmC06 Barry Smith ja Werner Ceusters: HL7 RIM: An Incoherent Standard. *Studies in Health Technology and Informatics, vol. 124*, 2006, sivut 133–138.

SMW06 Gunther Schadow, Charles N. Mead ja D. Mead Walker: The HL7 reference

information model under scrutiny. *Studies in Health Technology and Informatics, vol. 124*, 2006, sivut 151–156.

SRM00 Gunther Schadow, Daniel C. Russler, Charles N. Mead ja Clement J. McDonald: Integrating Medical Information and Knowledge in the HL7 RIM. *Proc. AMIA Symp.*, 2000, sivut 764-768.

THL13 Terveyden ja hyvinvoinnin laitoksen koodistopalvelu http://www.thl.fi/fi_FI/web/fi/tutkimus/palvelut/koodistopalvelu [20.1.2013]

VAN11 Teeradache Viangteeravat, Matthew N Anyanwu, Venkateswara Ra Nagisetty, Emin Kuscu, Mark Eijiro Sakauye ja Duojiao Wu: Clinical data integration of distributed data sources using Health Level Seven (HL7) v3-RIM mapping. *Journal of Clinical Bioinformatics, Vol. 1, No. 32*, Marraskuu 2011.

Viz04 Lowell Vizenor: Actions in Health Care Organizations: An Ontological Analysis. *Proc. of Medinfo, vol 11*, 2004, sivut 1403-1407.

Vli07 Eric van der Vlist: *Schematron*. O'Reilly, 2007.

W3C13 W3C XML Schema http://www.w3.org/XML/Schema [1.8.2013]

XML04 XML Schema Part 0: Primer Second Edition, W3C Recommendation 28 October 2004, http://www.w3.org/TR/xmlschema-0/ [3.7.2013].

YuD10 Mustafa Yuksel ja Asuman Dogac: Interoperability of Medical Device Information and the Clinical Applications: An HL7 RMIM based on the ISO/IEEE 11073 DIM. *IEEE Transactions On Information Technology In Biomedicine, Vol. 15, No. 4*, Heinäkuu 2011, sivut 557-566.

Zim80 Hubert Zimmermann: OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection. *IEEE Transactions On Communications, Vol. Com-28, No. 4*, Huhtikuu 1980, sivut 425-432.