Sami Nybacka

MUSIIKIN KUVAUSKIELET

Tietotekniikan
pro gradu -tutkielma
7.5.2001

Jyväskylän yliopisto
Tietotekniikan laitos
Informaatioteknologian maisteriohjelmat
Digitaalinen media
Jyväskylä

TIIVISTELMÄ

Nybacka, Sami Petteri
Musiikin kuvauskielet
Jyväskylä: Jyväskylän yliopisto, 2001
71 s.
Tietotekniikan pro gradu -tutkielma

Musiikin kuvaamiseen elektronisessa muodossa on olemassa lukuisia erilaisia kuvauskieliä. Kukin näistä on yleensä keskittynyt käsittelemään musiikkia ainoastaan oman sovellusalueensa näkökulmasta. Näin ollen kukin musiikin kuvauskieli kykenee esittämään musiikista hieman eri asioita.

Tämä tilanne on aiheuttanut monia erilaisia ongelmia esimerkiksi tiedonsiirron ja musiikin pitkäaikaisen säilytyksen kohdalla. Musiikille ei vieläkään ole yleisesti hyväksyttyä, yleiskäyttöistä standardia kuvauskieltä.

Tässä työssä tarkastellaan minkälaisia ongelmia ja mahdollisuuksia liittyy musiikin kuvaamiseen elektronisessa muodossa, ja mitä mahdollisuuksia voidaan odottaa kattavalla yleiskäyttöisellä musiikin kuvauskielellä saavutettavan. Erityisesti tässä työssä tutkitaan yleisempien käytössä olevien, sekä joidenkin uusien yleiskäyttöisiksi suunniteltujen musiikin kuvauskielten ongelmia ja mahdollisuuksia.

Tarkastelujen tuloksena nähtiin, että kattava ja yleiskäyttöinen kuvauskieli ratkaisisi monia nykyisiä ongelmia. Samalla kuitenkin todettiin, että kaikki tässä työssä käsitellyt kuvauskielet ovat tällaiseksi yleiskäyttöiseksi sopimattomia. Ainoastaan SMDL olisi ollut tehtävään tarpeeksi laaja ja kattava, mutta sen ei arvioitu olevan käytännössä käyttökelpoinen vielä pitkään aikaan.

Avainsanat Musiikki, kuvauskielet, SMDL.

ABSTRACT

Nybacka, Sami Petteri
Music description languages
Jyväskylä: University of Jyväskylä, 2001
71 p.
Master's thesis in department of mathematical information technology

There is large amount of computer languages for describing musical information. Each of these music description languages is usually directed to one specific application domain such as sound production, notation or analysis. Therefore languages often silently ignore those musical attributes that are not essential for their main purpose. This raises many problems related to, for example, data interchange and long-term preservation of musical work. At the moment there is no widely accepted method for the interchange of musical information or general-purpose music description language.

In this thesis the objective is to survey problems and possibilities related to music description languages and to examine what benefits could be gained with general-purpose music description language. Secondly, to survey the most popular music description languages in use today and also few newer languages that are designed especially to be more extensive. Problems and possibilities of these reviewed languages are examined.

The conclusion of this study is that all the languages reviewed in this thesis are not adequate as a general-purpose music description language. Only SMDL is extensive enough, but it is not at all practical yet.

Keywords Music, description language, SMDL.

Sisältö

LYHENTEET

DARMS Digital Alternate Representation of Musical Scores. Musiikin kuvauskieli nuotinkirjoitusohjelmistojen käyttöön. Useita eri murteita.

DTD Document Type Definition. SGML- ja XML-dokumenttien rakennekuvaus, dokumenttityyppimäärittely.

HTML Hypertext Markup Language. SGML-sovellus WWW-dokumenttien rakenteen merkkaamiseen.

HyTime Hypermedia/Time-based Structuring Language. SGML-sovellus hypermediaa varten.

ISO International Organization for Standardization. Kansainvälinen standardointiorganisaatio.

MIDI Musical Instrument Digital Interface. Tiedonsiirtokieli elektronisten musiikkilaitteiden reaaliaikaiseen ohjaukseen.

MPEG Moving Picture Experts Group. ISO:n alainen, digitaalisen äänen ja liikkuvan kuvan standardeihin keskittynyt ryhmä.

MML Music Markup Language. XML:ään perustuva musiikin kuvauskieli.

NIFF Notation Interchange File Format. Erityisesti nuotinkirjoitusohjelmistojen väliseen tiedonsiirtoon tarkoitettu musiikin kuvauskieli.

RIFF Resource Interchange File Format. Microsoftin kehittämä ja käyttämä tiedostojen tietosisällön rakennemäärittely.

SAOL Structured Audio Orchestra Language. MPEG-4 -standardiin kuuluva äänisynteesikieli.

SGML Standard Generalized Markup Language. Metakieli dokumenttien rakenteen merkkaukseen.

SMDL Standard Music Description Language. SGML- ja HyTime -standardeihin perustuva erittäin laaja musiikin kuvauskieli.

XML Extensible Markup Language. SGML:stä kehitetty, SGML:ää hieman yksinkertaisempi, erityisesti Internet-käyttöön soveltuva metakieli.

1 Johdanto

Musiikin esittämiseen elektronisessa, eli tietokoneella käsiteltävissä olevassa muodossa on olemassa lukemattomia erilaisia musiikin kuvauskieliä ja niitä syntyy jatkuvasti lisää. Eri tyyppiset, eri tarkoitusta varten rakennetut ohjelmistot käyttävät kukin erilaisia kuvauskieliä oman sovellusalueensa painopisteiden mukaan. Yleensä myös saman sovellusalueen yksittäiset ohjelmistot käyttävät kukin omaa ohjelmistokohtaista kuvauskieltään.

Musiikille ei siis vielä ole olemassa yleisesti hyväksyttyä, eri ohjelmistojen tarpeet täyttävää ja tarpeeksi kattavaa standardia kuvauskieltä. Tästä aiheutuu lukuisia erilaisia ongelmia mm. tiedonsiirron ja pitkäaikaisen säilytyksen kannalta.

Näiden monien nykytilanteen ongelmien vuoksi tällaista yleistä ja monipuolista standardia on kuitenkin alettu etsiä. Yleisellä kuvauskielellä tavoitellaan pitkälti samoja hyötyjä kuin elektronisen ja rakenteisen tekstin tutkimuksen yhteydessä tekstidokumenteille haetuilla standardeilla.

Ensinnäkin halutaan parantaa laitteisto-, järjestelmä- ja ohjelmistoriippumattomuutta. Tämä tarkoittaa mm. materiaalin parempaa siirrettävyyttä eri sovellusohjelmistojen välillä sekä parempaa tiedon säilyvyyttä esimerkiksi arkistoinnin yhteydessä.

Toiseksi halutaan parantaa dokumentin sisällön monikäyttöisyyttä ja sisällön eri osasten merkitysten automaattista tunnistamista. Nämä seikat mahdollistavat mm. dokumentin eri käsittelyvaiheiden automatisoinnin.

Musiikin kuvaaminen on kuitenkin huomattavasti vaikeampaa kuin esimerkiksi tekstin rakenteistaminen. Tekstidokumenttien on yleensä helppo noudattaa tiettyä tarkkaa ennaltamäärättyä muotoa. Musiikki taas on varsin abstrakti, aikariippuvainen asia, jonka tarpeita on vaikea ennakoida.

Tämä työ lähtee liikkeelle kysymyksestä, millaisia ominaisuuksia vaaditaan yleiskäyttöiseksi sopivalta, eli mahdollisimman monia muita ohjelmistoriippuvaisia

kuvauskieliä korvaamaan kykenevältä musiikin kuvauskieleltä. Tähän liittyy läheisesti kysymys siitä, onko jokin lukuisista olemassa olevista kuvauskielistä jo sellaisenaan yleiskäyttöiseksi sopiva.

Koska musiikki on jo itsessään valtavan laaja ja monimutkainen aihe, ja lisäksi olemassa olevia kuvauskieliä on valtavasti, käsittelee tämä työ musiikin kuvauskielien nykytilannetta vain katsauksen omaisesti. Tämä on kuitenkin perusteltua, koska suomeksi tämän tyyppistä selvitystä ei ole aikaisemmin tehty.

Tämän työn tarkoituksena on selvittää katsausluonteisesti

- minkälaisia ongelmia musiikin kuvaamisessa on yleensä ja erityisesti elektronisessa muodossa
- mitä mahdollisuuksia voidaan odottaa elektronisessa muodossa esitettävän musiikin tarjoavan, erityisesti jos käytössä on jokin yleiskäyttöinen standardi
- mitä ongelmia nykyisissä käytössä olevissa musiikin kuvauskielissä on havaittavissa
- mitä yleiskäyttöisiksi suunniteltuja kuvauskieliä on jo olemassa, sekä mitä ongelmia ja mahdollisuuksia näissä on havaittavissa.

Aluksi luvussa 2 tarkastellaan musiikkia yleisellä tasolla ja tämän jälkeen erityisesti elektronisessa muodossa. Luvussa luodaan katsaus erilaisiin musiikin kuvaamiseen liittyviin ongelmiin sekä mahdollisuuksiin.

Seuraavissa luvuissa 3 – 6 luodaan katsaus erilaisiin suosituimpiin olemassa oleviin musiikin kuvauskieliin ja sovellusaluekohtaisiin kuvauskielityyppeihin. Tarkasteltavat kuvauskielet on jaettu lukuihin sovellusalueittain luvussa 2 esiteltyjen kategorioiden mukaan.

Luvussa 7 esitellään yleiskäyttöisiksi suunniteltuja kuvauskieliä. Johtopäätöksiä tarkastellaan luvussa 8 ja yhteenveto on luvussa 9.

Tässä työssä esiteltävät musiikin kuvauskielet on valittu pääasiassa kielen suosion perusteella, mutta myös kuvauskielen kattavuuteen on kiinnitetty huomiota. Kuvauskielen suosio, eli levinneisyys ja käyttömäärä on arvioitu erityisesti käyttö-

määriä selvittäneiden tutkimusten perusteella (Survey of Music Publishers 1996; Survey of Music Libraries 1996; Correia 1993–94), sekä yleisesti kaikkien aiheeseen liittyvien artikkelien antamien tietojen ja vihjeiden perusteella.

Saman sovellusalueen kuvauskielet eivät perusperiaatteiltaan juurikaan eroa toisistaan. Näin ollen kuvauskieliä esittelevissä luvuissa pyritäänkin ennen kaikkea selvittämään eri kielityyppien yleisiä periaatteita. Erityisesti yleiskäyttöisiksi suunniteltuja kuvauskieliä tarkastellaan muita tarkemmin. Kaikki esiteltävät kuvauskielet tallennetaan ASCII-muodossa, ellei toisin ole mainittu.

Kaikki käsitteet ja termit määritellään tekstissä samassa yhteydessä kun niitä ensimmäistä kertaa käytetään. Määrittelyn yhteydessä termit on kursivoitu (esim. *termi*). Ohjelmistojen nimet on kirjoitettu kallistettuna (esim. *ohjelmisto*). Kuvauskielten koodinäytteissä ja koodeihin oleellisesti kuuluvissa ilmaisuissa ja viittauksissa käytetään tasalevyistä kirjasinta eli ns. kirjoituskonetyyliä (esim. `koodi`).

Tässä työssä käytetään sanaa *elektroninen* sanan *digitaalinen* synonyyminä viittaamaan johonkin tietokoneella käsiteltävissä olevaan materiaaliin, kuten esimerkiksi ilmauksissa elektroninen muoto ja elektroninen dokumentti. Periaatteessa termien merkityksellä on eroa, mutta käytännössä kirjallisuudessa usein käytetään molempia ilmauksia tarkoittamaan samaa asiaa.

Uudet vieraskieliset termit on pyritty ainakin laajemmin käsiteltyinä aina suomentamaan. Vakiintumattomissa ja omissa suomennoksissa alkuperäinen termi on mainittu suluissa esittelyn yhteydessä. Suomentamisessa apua on haettu muun muassa ATK-sanakirjasta (1997).

2 Musiikki informaationa

Tässä luvussa tarkastellaan käsitteen *musiikki* sisältöä monilta eri suunnilta. Aluksi mietitään mitä eri asioita musiikilla itse asiassa tarkoitetaan, missä eri olomuodoissa musiikkia voi olla olemassa ja erityisesti missä muodoissa sitä voidaan tallentaa. Samalla esitellään lyhyesti monia musiikkiin ja musiikin eri esitysmuotoihin liittyviä peruskäsitteitä. Aiheen laajuuden takia esimerkeissä ja käsitteiden valinnoissa on musiikkia tarkasteltu pääasiassa länsimaisen taidemusiikin näkökulmasta.

Tämän jälkeen tutkitaan tarkemmin musiikin esittämistä erityisesti elektronisessa muodossa ja esitellään yleisellä tasolla erilaisia musiikkia käsitteleviä tietokoneohjelmistotyyppejä. Lopuksi tutkitaan mitä erilaisia ongelmia musiikin elektronisessa muodossa käsittelyssä on, ja mitä mahdollisuuksia elektroninen muoto voisi vielä tarjota.

2.1 Musiikin erilaiset olomuodot

Ensisijaisesti musiikki on ääntä. Mutta musiikilla voidaan ajatella olevan myös muita tärkeitä olomuotoja, joiden esittelemiseksi seuraavassa tarkastellaan lyhyesti sävelletyn soitinmusiikin elinkaarta hyvin yleisellä tasolla.

Äänenä kuultava musiikki syntyy muusikoiden soittamista soittimista. Musiikilla on myös säveltäjä, jonka ohjeiden mukaan muusikot soittavat. Sävelletty musiikki on siis tavallaan olemassa jo ennen kuin se kuullaan äänenä. Säveltäjän on täytynyt ajatuksissaan konstruoida musiikki ja välittää se edelleen muusikoille soitettavaksi. Abstraktin sävellyksen välittäminen säveltäjän ajatuksista eteenpäin sellaisenaan ei tietenkään onnistu, vaan sävellys on kuvailtava muusikoille jollakin ymmärrettävällä tavalla, jonkinlaisella yhteisellä kielellä.

Musiikilla on siis olemassa ainakin kolme erilaista olomuotoa:

Kuva 2.1: Musiikin eri olomuotojen (musiikillinen sisältö, koodattu esitys ja auditiivinen esitys) suhde toisiinsa yksinkertaisessa tilanteessa, jossa säveltäjä säveltää soitinmusiikkia muusikon soitettavaksi.

- Äänenä kuultava musiikki eli musiikin *auditiivinen esitys*.
- Täydellisesti säveltäjän näkemyksen mukainen, vielä abstraktina mielikuvana oleva musiikki, josta käytetään jatkossa termiä *musiikillinen sisältö*.
- Jonkinlaisella musiikin kuvaamiseen sopivalla kielellä eli *musiikillisella koodilla* (engl. musical code, Selfridge-Field 1997d, 4) koodattu musiikillinen sisältö, eli musiikin *koodattu esitys*.

Vastaavaa jaottelua ovat käyttäneet mm. Tiensuu (1991b, 264) ja Selfridge-Field (1997d, 7).

Edellä kuvatut musiikin kolme olomuotoa on kuvassa 2.1 sijoitettu yksinkertaisen esimerkkitapauksen yhteyteen. Esimerkissä säveltäjä luo uuden teoksen eli hän käsittelee ajatuksissaan musiikillista sisältöä. Tämän hän kuvailee paperille sopivaa musiikillista koodia käyttäen. Näin saatu paperille tallennettu musiikin koodattu esitys annetaan muusikolle, joka ymmärtää säveltäjän käyttämää musiikillista koodia. Muusikko tulkitsee koodatun esityksen ja soittaa tämän perusteella säveltäjän luoman teoksen ääneksi.

Musiikkia on mahdollista tallentaa kahdessa näistä olomuodoista. Musiikillinen sisältö abstraktina mielikuvana ei tietenkään ole sellaisenaan tallennettavissa. Sen sijaan auditiivinen esitys voidaan tallentaa erilaisina äänitallenteina ja koodattu esitys voidaan esimerkiksi kirjoittaa paperille, kuten jo edellisessä esimerkissä mainittiin.

Kuitenkaan nämä kaksi mahdollista tallennusmuotoa eivät tee toisiaan tarpeettomiksi. Koodatun esityksen ensisijaisena tehtävänä on pyrkiä tallentamaan musiikki sellaisessa muodossa, josta se on edelleen soittajien tulkittavissa ja ääneksi soitettavissa (vrt. Tiensuu 1991b, 264; Oksala 1971, 27). Esimerkiksi mikäli säveltäjä yksin ei soita säveltämäänsä musiikkia itse, on esimerkiksi paperille kirjoitettu koodattu esitys käytännöllinen välivaihe jonka avulla säveltäjä voi välittää sävellyksensä muusikoille. Tallennettu koodattu esitys on siis eräänlainen musiikin säilyttämisen, välittämisen ja levittämisen mahdollistava, erityisesti musiikin soittajille tarkoitettu esitysohje. Ennen äänentallennusmenetelmien keksimistä tällainen ohje olikin perimätiedon lisäksi ainoa tapa tallentaa musiikkia. (Tiensuu 1991b, 264.)

Äänitallenne taas on selvästi ensisijainen musiikin kokemisen ja nauttimisen kannalta, mutta se ei yleensä yksinään ole sopiva uusien auditiivisten esitysten tuottamisen ohjeeksi. Esimerkiksi suuren orkesteriteoksen soittaminen äänitteen perusteella on selvästi mahdotonta, tai ainakin äärimmäisen vaivalloista. Lyhyesti ja yksinkertaistaen voidaankin sanoa, että tallennettu koodattu esitys on olemassa muusikoita varten ja äänitallenne kuulijoita varten.

Tässä kohtaa on tarpeellista rajata jatkossa käsiteltävää aiheistoa, koska äänitallenteet eivät kuulu tämän työn välittömään aihepiiriin. Näin ollen tässä työssä ei enää tarkemmin käsitellä auditiivista esitystä musiikin tallentamisessa, vaan jatkossa keskitytään yksinomaan koodattuun esitykseen. Lisäksi kun jatkossa puhutaan yleisesti musiikista tai musiikin tallentamisesta, sillä viitataan aina nimenomaan musiikin koodattuun esitykseen ellei selvästi toisin ole mainittu.

Tässä kappaleessa annettu kuva musiikin eri olomuodoista ja elinkaaresta on tietysti varsin yksinkertaistettu, mutta tässä hahmotellut periaatteet ja käsitteet toimivat kuitenkin hyvin myös laajemmassa kontekstissa.

Esimerkiksi sävellyksen soittajana voi nykyään toimia myös kone, esimerkiksi tietokone tai syntetisaattori. Mutta tässäkin tapauksessa säveltäjän ja soittajan (koneen) välissä tarvitaan musiikillista koodia. Säveltäjän täytyy edelleen välittää sävellys soittajalle, tässä tapauksessa esimerkiksi ohjelmoimalla sävellys tietokoneeseen. Musiikillisella koodilla tarkoitetaan tässä siis musiikin kuvaamiseen ja välittämiseen tarkoitettua koodia hyvin yleisesti. Sitä ei tule ajatella pelkästään muusikoiden ymmärtämänä kielenä, vaan musiikillinen koodi voi olla myös esimerkiksi tietokoneessa käytetyn soitto-ohjelman ymmärtämä kieli, mikäli tällä määritellään jokin sävellys.

Musiikin käsitteellistämisen mutkikkuutta valaisevina ja ajattelua virkistävinä esimerkkeinä mainittakoot tässä vielä kaksi tapausta, joissa säveltäjän ja soittajan väliin sijoittuvat olomuodot alkavat hämärtyä. Ensinnäkin täysin vapaassa improvisaatiossa ei soitettua musiikkia ainakaan perinteisessä mielessä konstruoida mielessä ennen soittamista. Toiseksi mm. säveltäjä Alvin Lucier on tehnyt musiikkia siten, että hänen aivosähkökäyränsä muutetaan konsertti- tai äänitystilanteessa suoraan korvin kuultavaksi musiikiksi (Tiensuu 1991a, 90).

2.2 Nuottikirjoitus

On selvää, että muusikot vaativat ymmärrettävältä musiikilliselta koodilta erilaisia ominaisuuksia kuin esimerkiksi tietokoneet tai syntetisaattorit. Soittaakseen sävellyksen muusikon täytyy pystyä lukemaan ja seuraamaan koodattua esitystä nopeasti ja sujuvasti. Tietokoneen soitettavaksi tarkoitettu sävellys taas voidaan koodata suoraan jollakin soitto-ohjelmiston ymmärtämällä formaalilla kielellä, jonka sellaisenaan ei tarvitse olla ihmisille helppolukuista.

Jatkossa käytetään termiä *nuottikirjoitus* kun tarkoitetaan erityisesti sellaista musiikillista koodia, joka on tarkoitettu ihmisen tulkittavaksi. Implisiittisesti tämä merkitys yleensä sisältyykin termin sanakirjamääritelmään. (Tiensuu 1991b, 264; Oksala 1971, 27.)

Nuottikirjoitus on visuaalinen tai esimerkiksi sokeilla kosketukseen perustuva merkkijärjestelmä, jonka avulla musiikkia pyritään merkitsemään muistiin.

Nuottikirjoituksella, kuten yleensäkin musiikillisilla koodeilla osoitetaan erilaisia musiikin ääneksi muuttamiseen liittyviä seikkoja, kuten soitettava sävelen säveltaso eli korkeus, soitettavan sävelen tai tauon kesto, äänenvoimakkuus, sointiväri ja niin edelleen. (Tiensuu 1991b, 264; Oksala 1971, 27.) Näistä seikoista yleensä puhuttaessa käytetään jatkossa termiä *musiikillinen käsite*. Musiikillisista käsitteistä joilla on aikaulottuvuus (esimerkiksi soitettava sävel ja tauko) voidaan käyttää erityistä nimitystä *musiikillinen tapahtuma*.

Nuottikirjoitusta tavataan useissa eri kulttuureissa, mutta erityisen keskeinen sija sillä on länsimaisessa taidemusiikissa, joka on pääsääntöisesti aina sävelletty ja määritelty nuottikirjoituksen avulla. Käytetyin nuottikirjoituksen *notaatiojärjestelmä* länsimaisessa taidemusiikissa on *viivastonotaatio*. (Tiensuu 1991b, 264.) Viivastonotaatiosta käytetään usein myös nimitystä *perinteinen länsimainen nuottikirjoitus* (engl. common music notation, toisinaan myös common practice notation). Lisäksi yleisessä kielenkäytössä nuottikirjoituksesta puhuttaessa tarkoitetaan yleensä juuri viivastonotaatiota.

Myös joitakin viivastonotaatiota edeltäneitä vanhempia notaatiojärjestelmiä on edelleen käytössä. Esimerkiksi *mensuraalinotaatiota* käytetään vielä ns. vanhan musiikin käsittelyssä. Mensuraalinotaatiota käytettiin erityisesti 1200 – 1500 -luvuilla, ja se on nykyisen viivastonotaation välitön edeltäjä (Taitto 1991, 152).

Erilaisia notaatiojärjestelmiä elää rinnakkain lähinnä siitä syystä, että kukin järjestelmä kuvaa musiikkia hieman eri näkökulmasta. Esimerkiksi soitinkohtainen *tabulatuurikirjoitus* koodaa musiikkia merkitsemällä tietoa siitä, missä kohtaa sormia soittimella tulee pitää, esimerkiksi mistä kohtaa kitaran otelautaa pitää milläkin hetkellä painaa (Tiensuu 1991b, 265).

Jonkin nuottikirjoituksen avulla koodatusta musiikista käytetään tässä työssä termiä *notaatioesitys* erotukseksi esimerkiksi vain tietokoneen tulkittavaksi tarkoitetusta koodatusta esityksestä. Samaan tapaan puhutaan erityisesti *viivastonotaatioesityksestä*, *mensuraalinotaatioesityksestä* ja niin edelleen. Painetuista tai paperille tulostetuista notaatioesityksistä käytetään nimitystä *nuottijulkaisu*.

2.3 Viivastonotaation peruskäsitteitä

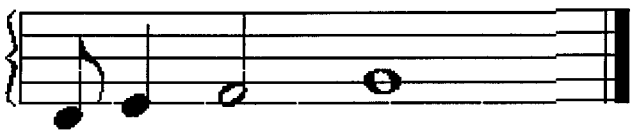
Seuraavassa esitellään hyvin lyhyesti joitakin tärkeimpiä viivastonotaation peruskäsitteitä, joita tullaan tarvitsemaan jatkossa myös muita musiikillisia koodeja käsiteltäessä. Tarkemmin viivastonotaatiosta ja sen yleisestä olemuksesta voi lukea esimerkiksi Oksalan teoksesta (1971) tai McLanen artikkelista (1996, 227–231).

Yksittäisen sävelen symbolina viivastonotaatiossa on *nuotti*, joka kirjoitetaan viisiviivaiselle *nuottiviivastolle*. Nuotti voi sijaita viivastolla viivan päällä tai viivojen välissä. Viivastoa voidaan tarvittaessa myös jatkaa ylös tai alas ns. apuviivoilla. (Oksala 1971, 28.) Nuottiviivastolla nuotit on ryhmitelty sävellyksen rytmiikan mukaan *tahteihin*, jotka erotetaan toisistaan *tahtiviivalla* (Oksala 1971, 101–102).

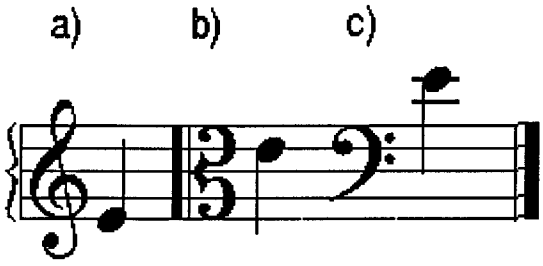
Nuotin asema viivastolla pystysuunnassa ilmaisee sävelen korkeuden. Sävelen keston ilmaisee ensisijaisesti nuotin ulkomuoto, esim. käytetäänkö nuotissa nuotinvartta vai ei, onko nuotti piirretty avoimeksi vai suljetuksi ja niin edelleen. Nuotit kirjoitetaan vaakasuunnassa vasemmalta oikealle aikajärjestyksessä, ja nuottien väliin jäävä tila mitoitetaan havainnollisuuden takia vastaamaan nuotin kestoa. Eri kestoisille tauoille on lisäksi omat symbolinsa. (Oksala 1971, 27–28.) Kuvassa 2.2 (s. 10) on nuottiviivastolla neljä eri kestoista ja eri korkuista nuottia.

Nuottien kestot eivät suoraan kerro soitettavan sävelen absoluuttista tosiaikaista kestoa, vaan vain suhteellisen keston toisiinsa nähden (Oksala 1971, 68). Kuvan 2.2 neljä nuottia ovat *aika-arvoiltaan* 1/8, 1/4, 1/2 ja 1/1. Sävelten todellinen kesto määräytyy vasta *tempomerkinnän* perusteella, joka kertoo kuinka kauan esimerkiksi neljäsosanuotti tosiajassa kestää.

Samaan tapaan nuotin sijainti viivastolla ei suoraan kerro sävelen soivaa korkeutta. Viivastolla ennen nuotteja kirjoitettu, kulloinkin voimassa oleva *klaavi* määrittelee mikä viivaston kohta tarkoittaa mitäkin todellista sävelkorkeutta (Oksala 1971, 31–32). Tästä on esimerkki kuvassa 2.3 (s. 10). Kuvassa viivastolla olevat kolme nuottia määrittelevät kaikki saman sävelkorkeuden, vaikka kukin niistä sijaitsee viivastolla eri korkeudella. Tämä johtuu siitä, että kutakin nuottia edeltää eri klaavi (kuvassa symbolit a, b ja c ovat klaaveja).



Kuva 2.2: Nuottiviivasto, jolla (vasemmalta oikealle) kahdeksasosanuotti (1/8-nuotti), neljäsosanuotti (1/4-nuotti), puolinuotti (1/2-nuotti) ja kokonuotti (1/1-nuotti).



Kuva 2.3: Kolme eri klaavia ja kolme saman korkuista (e-sävel) nuottia.

Viivastonotaatiolla esitettävissä olevat *juurisävelet* ovat nimeltään (alemmasta ylempään) c, d, e, f, g, a ja h. Sävelten nimet toistuvat *oktaavialoittain*, eli jokainen oktaaviala sisältää sävelet c – h, seuraava oktaavi jälleen sävelet c – h ja niin edelleen. Sävelten etäisyys toisistaan ei ole yhtäsuuri, vaan sävelten c – d, d – e, f – g, g – a ja a – h etäisyyden sanotaan olevan *kokoaskel* ja sävelten e – f ja oktaavialan vaihteen h – c etäisyyden *puoliaskel*. (Oksala 1971, 21, 24.)

Edellä mainittuja säveliä voidaan myös ylentää ja alentaa, jolloin tulokseksi saadaan sävelen *kromaattinen* muunnos. Ylennys ja alennus merkitään viivastolla nuotin eteen tulevalla *etumerkillä*. Yksi ylennys tai alennus vaikuttaa sävelen korkeuteen yhden puoliaskelen verran. Esimerkiksi ylennetty c-sävel osuu juuri c- ja d-sävelen väliin. Puoliaskel on perinteisessä länsimaisessa taidemusiikissa yleensä pienin käytetty sävelten etäisyys, eli *intervalli*. Näin yhteen oktaaviin mahtuu 12 erikorkuista säveltä. (Oksala 1971, 44.)

Partituuri sisältää useampiäänisten sävellysten kaikkien äänien nuotit päällekkäisillä viivastoilla. Esimerkiksi kapellimestari seuraa teoksen esitystilanteessa juuri

partituuria, koska siitä näkee helposti koko sävellyksen etenemisen. Yksittäiset soittajat lukevat yleensä vain oman soittimensa nuotteja. Soitinkohtaisista nuoteista käytetään nimitystä *stemma*. Samaan aikaan soitettavat nuotit esitetään aina päällekkäin viivaston vaakasuunnan samalla kohdalla, niin yksittäisellä viivastolla kuin partituuritasollakin. (Oksala 1971, 38.)

2.4 Viivastonotaation ongelmia

Viivastonotaatio on ehdottomasti yleisin tapa merkitä länsimaista musiikkia muistiin. Se on kuitenkin epäyhtenäinen, epätarkka ja jatkuvasti muuttuva järjestelmä. Monia viivastonotaatiolla esitetyn musiikin soittamiseen liittyviä asioita ei ole tapana merkitä viivastonotaatiolla, eikä vakiintuneita merkintätapoja edes välttämättä ole olemassa. Esimerkiksi äänenvärin määrittelyyn ei viivastonotaatiossa ole juurikaan kiinnitetty huomiota. Näin ollen musiikin soittamiseen vaikuttavat monet kirjoitetun sävellyksen ulkopuoliset seikat, mm. tieto sen syntyajan esityskäytännöstä sekä soittajan vapaa tulkinta. (Selfridge-Field 1997d, 3–4; Selfridge-Field 1997b, 30.)

Lisäksi viivastonotaation väljiä sääntöjä myös rikotaan jatkuvasti, jolloin monitulkintaisuus kasvaa entisestään. Byrd esittää nuottiesimerkkejä hyvin tavallisista tällaisista tilanteista (1994, 18). Näissä mm. kaksi eri klaavia on voimassa yhtä aikaa samalla viivastolla, samalla hetkellä soitettavat nuotit sijaitsevat viivaston vaakasuunnassa neljässä eri paikassa ja niin edelleen. Näin pelkästään viivastonotaation perussääntöjen perusteella ei viivastonotaatioesityksestä saadakaan alkuperäistä sävellystä vastaavaa kuvaa. Byrd tutkii asiaa tarkemmin väitöskirjassaan (Byrd 1984), jossa nuottiesimerkkejä rikotuista säännöistä esitetään noin sata (artikkelin Dannenberg 1993, 23 mukaan). Väitöskirjan nuottiesimerkit on valittu hyvin perinteisiltä klassisilta säveltäjiltä, joiden sävellyksiä esitetään jatkuvasti paljon.

Viivastonotaatio on siis alunperinkin ollut hieman epätarkka ja summittainen. Sen syntyaikoihin ei tosin tarkempaa notaatiojärjestelmä kaivattukaan, mutta nykyäveltäjät taas pyrkivät merkitsemään sävellyksensä muistiin mahdollisimman yk-

siselitteisesti ja mahdollisimman suurella merkintöjen tarkkuudella. Monet heistä ovat päätyneet kuormittamaan viivastonotaatiota vakiintumattomilla, usein partituurikohtaisilla symboleilla sekä näiden pitkillä selityksillä. (Stone 1980, xix.) Myös kokonaan uusia tarkempia notaatiojärjestelmiä on aina silloin tällöin kokeiltu (Stone 1980, xv–xvi; Dannenberg 1993, 24–25; ks. myös Maegaard 1964, 113–122). Mikään näistä ei ole kuitenkaan vielä edes horjuttanut viivastonotaation valta-asemaa.

Epätarkkuus ei siis ole viivastonotaation ainoa ongelma. Viivastonotaatio alkaa muuttua epäkäytännölliseksi ja tukalan ahtaaksi kun sillä yritetään koodata nykyajan monimuotoista musiikkia mahdollisimman tarkasti ja yksityiskohtaisesti, alati uusien symbolien avustuksella. Näitä ongelmia on käsitellyt mm. Stone (1980, xvi). Eräs suuri, erityisesti nykymusiikin parissa esiintyvä ongelma on se, että viivastonotaatiolla on alunperin suunniteltu esitettäväksi vain sellaisia sävelkorkeuksia, jotka perustuvat seitsemän juurisävelen asteikolle ja näiden kromaattisille muunnoksille. Tämä merkitsee sitä, että mikäli sävellyksessä on käytetty esimerkiksi puoliaskelta pienempiä sävelaskelia (tällöin puhutaan usein *mikrointervalleista*), viivastonotaatioesityksen luettavuus alkaa vakavasti kärsiä. Osaksi tämä johtuu yksin ylimääräisten selventävien symbolien paljoudesta. Osaksi siitä, että koska nuottien korkeuserot mikrointervallien tapauksessa näkyvät enää vain nuottien eteen kirjoitettavista uusista symboleista, sävelten korkeus ei enää ole visuaalisesti niin helposti hahmotettavissa viivastolta.

Lisää viittauksia viivastonotaation puutteisiin ja ongelmiin on mm. Dannenbergin (1993, 23–24), Tiensuun (1991b), Maegaardin (1964, 113–115) ja Granden ja Belkinin (1997, 35) artikkeleissa ja teksteissä.

2.5 Musiikki elektronisessa muodossa

Nykyään tietokonetta voidaan käyttää musiikin käsittelyn apuvälineenä tekstinkäsittelyn tapaan. Mutta koska musiikki on tekstiä moniulotteisempaa ja siten vaikeammin koodattavissa, on musiikin tallentaminen elektronisessa muodossa huomattavasti esimerkiksi tekstin tallentamista ongelmallisempaa.

Musiikin koodattu esitys on mahdollista tallentaa elektronisessa muodossa kahdella tavalla. Visuaalinen notaatioesitys voidaan tallentaa suoraan digitaalisena kuvana. Toinen vaihtoehto on koodata musiikki jonkinlaisella erityisesti tietokoneohjelmistojen käyttöön tarkoitetulla koodilla, jonkinlaisella *kuvauskielellä*. (Report on SMDL evaluation 1996, 2–4.)

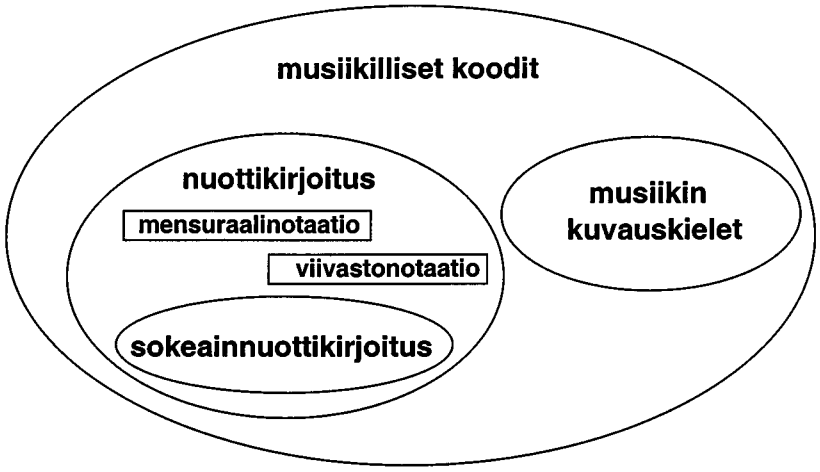
Kuvauskieli on ATK-sanakirjan (1997, 79) mukaan rakenteisen kokonaisuuden kuten tietojärjestelmän, tiedon tai ohjelman kuvaamiseen käytettävä määrämuotoinen esitystapa. Tämän perusteella tässä työssä käytetään termiä *musiikin kuvauskieli* (engl. music description language) tarkoittamaan erityisesti musiikin kuvaamiseen tarkoitettua kuvauskieltä. Musiikin kuvauskieli on siis tietokoneohjelmistojen käsiteltäväksi tarkoitettu musiikillinen koodi. Jatkossa puhutaan usein lyhyesti pelkästään kuvauskielestä tarkoitettaessa juuri musiikin kuvauskieltä.

Jostakin musiikin kuvauskielellä kirjoitetusta kokonaisuudesta, sävellyksestä tai sen osasta käytetään nimitystä musiikin *kuvauskielinen esitys* tai *musiikkidokumentti* (engl. music document). Elektronisessa muodossa kuvana esitettävästä visuaalisesta nuottikirjoituksesta puhuttaessa taas käytetään käsitettä musiikin *visuaalinen esitys*.

Musiikin kuvauskielellä, samoin kuin nuottikirjoituksella tai yleensä jollakin musiikillisella koodilla esitetään erilaisia musiikillisia käsitteitä. Yksittäisiä musiikillisia käsitteitä koodataan kuvauskielellä tarpeiden mukaan mahdollisimman tarkasti niin, että nämä koodatut käsitteet on mahdollista tunnistaa ohjelmallisesti. Näin kuvauskielellä koodattua musiikkia voidaan käytetyn kuvauskielen kattavuudesta riippuen käsitellä, manipuloida ja editoida hyvin monipuolisesti.

Visuaalisen esityksen käyttömahdollisuudet sen sijaan ovat ilmeisen rajalliset, jolloin elektroninen muoto ei tuo juurikaan lisäarvoa musiikin käsittelyyn verrattuna esimerkiksi paperilla työstettävään nuottikirjoitukseen. Yksittäisiä musiikillisia käsitteitä ja erityisesti niiden välisiä suhteita on digitaalisesta kuvasta hyvin vaikeaa, ellei peräti mahdotonta enää saada ohjelmallisesti selville (Report on SMDL evaluation 1996, 2; ks. myös Blostein ja Haken 1991).

Tarpeeksi tarkalla musiikin kuvauskielellä koodatusta kuvauskielisestä esityksestä taas voidaan sopivalla ohjelmistolla tuottaa sekä visuaalinen esitys, että syn-

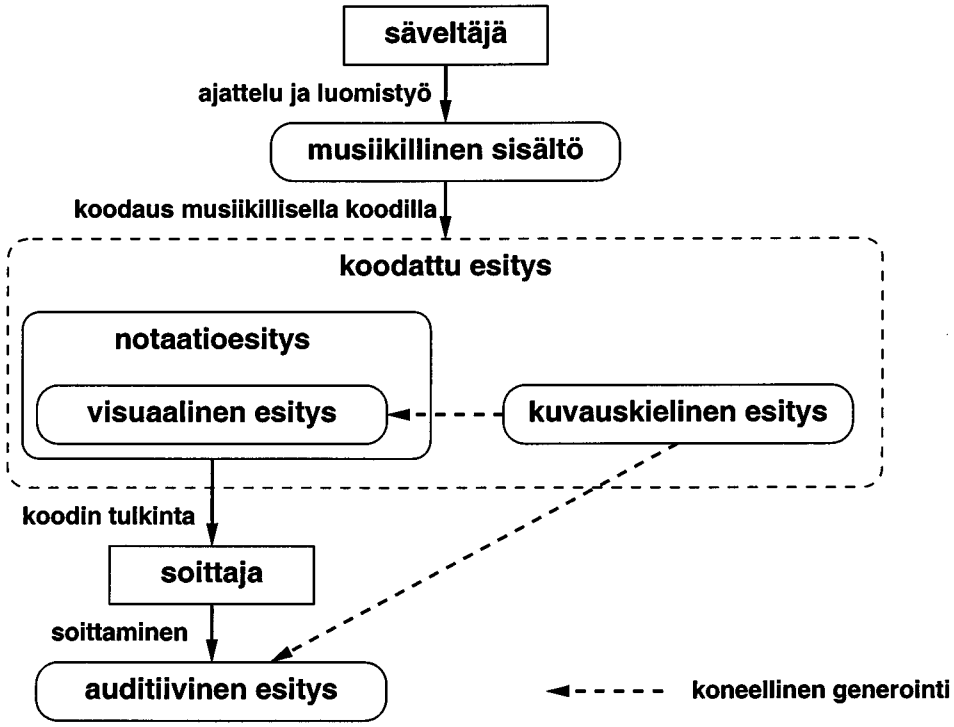


Kuva 2.4: Erilaisia musiikillisia koodeja.

teettinen auditiivinen esitys. Näitä esityksiä voidaan itse asiassa tuottaa lukuisia erilaisia aina tarpeen mukaan. Kuvauskielisestä esityksestä voidaan esimerkiksi tuottaa visuaalinen esitys useilla eri notaatiojärjestelmillä.

Näistä seikoista johtuen ei visuaalista esitystä enää tässä työssä tämän tarkemmin käsitellä ensisijaisena musiikin tallennusmuotona.

Kuvassa 2.4 on esitetty tähän mennessä esiin tulleet erilaiset musiikillisten koodien kategoriat suhteessa toisiinsa. Näiden lisäksi kaavioon on yksittäisinä koodiesimerkkeinä sijoitettu viivastonotaatio ja mensuraalinotaatio. Näitä vastaavasti on kuvassa 2.5 (s. 15) havainnollistettu, kuinka tässä kappaleessa esitellyt visuaalinen ja kuvauskielinen esitys suhtautuvat kappaleessa 2.1 (s. 4) kuvattuihin musiikin kolmeen erilaiseen olomuotoon (vrt. kuvaan 2.1, s. 5). Kuvassa säveltäjän luomistyön tuloksena syntynyttä musiikillista sisältöä voidaan merkitä muistiin koodaamalla se jollakin musiikillisella koodilla. Käytetty koodi voi olla jokin nuottikirjoitus tai jokin musiikin kuvauskieli. Kuvauskielisestä esityksestä voidaan ohjelmallisesti tuottaa tietokoneella esitettävissä oleva visuaalinen esitys sekä auditiivinen esitys.



Kuva 2.5: Musiikin eri olomuotojen (musiikillinen sisältö, koodattu esitys ja auditiivinen esitys) suhde erilaisiin musiikin elektronisiin tallennusmuotoihin (visuaalinen esitys ja kuvauskielinen esitys).

2.6 Erilaisia musiikkiohjelmistoja

Seuraavassa on lyhyesti esitelty yleisimpiä ohjelmistotyyppejä, joissa kaikissa käsiteltävän musiikin koodaukseen käytetään jonkinlaista musiikin kuvauskieltä. Vastaavaa ohjelmistotyyppien jakoa on käytetty mm. teoksessa Beyond MIDI (Selfridge-Field 1997a; tarkemmin Selfridge-Field 1997d, 28 – 34). Tärkeimpien tässä mainittujen ohjelmistotyyppien kuvauskielten tyypillisiä ominaisuuksia tarkastellaan tarkemmin seuraavissa luvuissa.

Vanhimpia musiikkiohjelmistoja ovat *äänisynteesiohjelmistot*. Niitä käytetään nimensä mukaisesti musiikin äänisynteesiin, eli synteettisen auditiivisen esityksen tuottamiseen. Tuotettava auditiivinen esitys voi olla tarkoitettu esimerkiksi konserttitilanteessa reaaliajassa kuunneltavaksi, tai se voidaan suoraan tallentaa digi-

taaliseksi äänitallenteeksi esimerkiksi CD-levylle tai DAT-nauhalle. Äänisynteesiohjelmistoja voidaan usein käyttää myös reaaliaikaiseen äänen käsittelyyn, esimerkiksi kaiun tai muun efektin lisäämiseen konserttitilanteessa soittimella soitettuun ääneen. Ohjelmistoissa on yleensä erotettu toisistaan musiikillisten käsitteiden määrittelyyn tarkoitettu musiikin kuvauskieli ja itse äänentuottamisen määrittelevä signaalitasolla operoiva synteesikieli.

Erilaisten elektronisten musiikkilaitteiden ohjaukseen ja lukemiseen on olemassa ohjelmistoja, joita jatkossa kutsutaan *kontrolliohjelmistoiksi*. Näillä voidaan kontrolloida reaaliajassa esimerkiksi efektilaitteita, syntetisaattoreita tai muita musiikkiohjelmistoja. Yleensä aluksi on luotu erityisesti reaaliaikaiseen ohjaukseen sopiva kuvauskieli, jota sitten on pyritty käyttämään erilaisissa laitteissa ja ohjelmistoissa.

Nuotinkirjoitusohjelmistoja käytetään musiikin viivastonotaatioesityksen tuottamiseen ja nuottijulkaisujen valmistamiseen. Usein näillä ohjelmistoilla voidaan käsitellä myös muita viivastonotaatiota muistuttavia notaatiojärjestelmiä, kuten esimerkiksi mensuraalinotaatiota.

Nuotintunnistusohjelmistoilla luetaan ja tunnistetaan nuottijulkaisuja kuvanlukijan avulla. Luettu informaatio tallennetaan jollakin musiikin kuvauskielellä esimerkiksi nuotinkirjoitusohjelmistoissa muokattavaksi. Visuaalinen notaatioesitys voidaan siis nuotintunnistusohjelmiston avulla muuttaa kuvauskieliseen muotoon, joskin informaatiota yleensä hukkuu tunnistuksessa paljon. Nuotintunnistus on viivastonotaation epämääräisyydestä johtuen huomattavasti esimerkiksi tekstin tunnistusta vaikeampaa.

Analyysiohjelmistoja käytetään lähinnä musiikkitieteellisiin teosanalyyseihin. Musiikkitieteilijät käyttävät analyysiohjelmistoja esimerkiksi sävellyksen sointurakenteiden tutkimiseen tai tietyn melodian erilaisten variaatioiden hakemiseen. Koska analyysiohjelmistoilla tutkitaan ja käsitellään musiikillista materiaalia ja näiden suhteita hyvin monipuolisesti, ovat myös näiden käyttämät kuvauskielet yleensä hyvin tarkkoja ja yksityiskohtaisia.

Sävellysohjelmistoja käytetään sävellyksen tai sen joidenkin osien tai rakenteiden määrittelyyn erilaisten sääntöjen ja algoritmien avulla. Näiden ohjeiden pe-

rusteella sävellysohjelmisto tuottaa valmiin sävellyksen kuvauskielisen esityksen. Kuvauskielisessä muodossa sävellys voidaan siirtää esimerkiksi johonkin nuotinkirjoitusohjelmistoon viivastonotaatioesityksen ja edelleen nuottijulkaisun tuottamista varten.

2.7 Nykytilanteen ongelmia

Musiikin kuvauskieliä käytetään musiikin esittämiseen hyvin erityyppisissä ohjelmistoissa. Koska kullakin sovellusalueella on hieman erilainen näkökulma musiikkiin ja sitä kautta erilaiset tarpeet, on luonnollista että nämä painopisteet näkyvät myös kuvauskielissä. Näin kullakin sovellusalueella käytetään perusperiaatteiltaan hieman erityyppisiä kuvauskieliä.

Lähes poikkeuksetta myös saman sovellusalueen yksittäiset ohjelmistot käyttävät kukin omaa kuvauskieltä. Jokaisen uuden ohjelmiston mukana on usein syntynyt myös uusi kuvauskieli. Tämä johtuu lähinnä siitä, että aina viime vuosiin saakka ei ole ollut olemassa edes sopivaa ehdokasta sellaiseksi yleiseksi musiikin kuvauskieleksi, joka olisi sopinut ohjelmistojen kaikkiin tarpeisiin. Tietysti useissa tapauksissa on mukana myös kaupallisia syitä: Tietyn valmistajan ohjelmistoissa pyritään käyttämään valmistajan omia kuvauskieliä, jotta asiakkaat olisivat pakotettuja jatkossakin käyttämään tämän valmistajan ohjelmistoja. Toisaalta oman, juuri sovelluksen tarpeisiin riittävän kielen suunnittelu ja käyttöönotto voi olla valmistajalle halvempaa kuin jonkin valmiin standardin implementointi.

Lukemattomista kuvauskielistä johtuen tiedonsiirto eri ohjelmistojen välillä on varsin vaivalloista. Konversio-ohjelmia toki on olemassa useiden ohjelmistojen kuvauskielten välille, mutta nämä ovat harvoin täydellisiä. Ensinnäkin ohjelmistokohtaiset kuvauskielet esittävät yleensä niin vähän musiikillisia käsitteitä kuin vain ohjelmiston tarpeisiin nähden on mahdollista. Jos ohjelmisto ei suoraan tarvitse tietoa joistakin musiikillisista käsitteistä, ei niiden esittämistä edes mahdollisteta kuvauskielessä. Muun muassa tästä syystä yhdellä kuvauskielellä koodataan usein sellaisia musiikillisiä käsitteitä, joita toisella kuvauskielellä ei lain-

kaan kyetä esittämään. Näin informaatiota hukkuu konversiossa. (Selfridge-Field 1997b, 31.)

Kuvauskielten paljous on ongelmallinen seikka myös musiikin pitkäaikaisen säilytyksen kannalta, kun musiikkia halutaan arkistoida elektronisessa muodossa. Tämä johtuu muun muassa juuri konversio-ongelmista. Edelleen laite- ja ohjelmistokanta, jota käyttäen musiikki on elektronisessa muodossa tallennettu, saattaa esimerkiksi muuttua tai kokonaan poistua markkinoilta. Näistä samoista ongelmista puhuu esim. Salminen (1995, 7), mutta lähinnä elektronisen tekstin yhteydessä.

Kaikkien näiden ongelmakohtien merkitys vain kasvaa ajan myötä, koska musiikkia ja nuottijulkaisuja käsitellään, tuotetaan ja tallennetaan yhä suuremmassa määrin pääasiassa tietokoneympäristöissä. Lisäksi suurin osa elektronisessa muodossa säilytettävästä musiikista on tällä hetkellä kuvattu edellä mainituille ongelmille erityisen taipuvaisilla kaupallisten ohjelmistojen epästandardeilla kuvauskielillä (Survey of Music Publishers 1996, 4; Survey of Music Libraries 1996), jotka usein vieläpä tallennetaan haavoittuvassa binäärimuodossa.

2.8 Yleisen musiikin kuvauskielen tarjoamia mahdollisuuksia

On mielenkiintoista huomata, että elektronisen tekstin tutkimuksessa on tunnistettu samanlaisia ongelmia kuin mitä musiikin kuvauskielien parissa nykyään esiintyy. Salminen (1995, 3–9) kirjoittaa artikkelissaan mm. tekstin eri esitysmuotojen yhteensopimattomuudesta ja ohjelmisto- ja laitesidonnaisuudesta johtuvista ongelmista. Nämä ovat pitkälti juuri samoja kuin edellä mainitut musiikin kuvauskielten ongelmat.

Rakenteiseksi tekstiksi kutsutaan elektronista tekstiä jonka rakenne noudattaa niin tarkkoja muodollisia sääntöjä, että sitä voidaan näiden rakenteiden avulla ohjelmallisesti käsitellä. Esimerkiksi sähköposti löytää tiensä perille, koska sitä käsittelevät ohjelmistot tunnistavat viestistä vastaanottajan osoitteen. (Salminen 1992, 6.) Tekstidokumenttien käsittely juuri rakenteisena tekstinä jotakin kansainvälistä

stanrdardia noudattaen näyttääkin ratkaisevan monia elektronisen tekstin ongelmia (Salminen 1995, 9).

Voidaan huomata, että rakenteisella tekstillä tavoitellut hyödyt ovat kiinnostavia myös musiikin kuvauskielten kannalta. Salminen (1995, 12–16) kirjoittaa tekstin rakenteistamisen (artikkelissa käsitellään erityisesti SGML-standardia; ks. kappale 7.2) tukevan

- materiaalin monikäyttöisyyttä
- materiaalin käsittelyvaiheiden automatisointia
- ohjelmisto- ja laiteriippumattomuutta
- tiedon pitkäaikaista säilytystä
- dokumenttien siirrettävyyttä
- yleistä tiedon hallintaa.

Musiikin kuvauskielet ovat yleisestikin hyvin rinnastettavissa rakenteiseen tekstiin. Musiikki on tekstiä abstraktimpana ja moniulotteisempana huomattavasti vaikeammin kuvattavissa ja hallittavissa, kuten jo edellisten kappaleiden perusteella voidaan huomata. Musiikin käsittelyssä on mukana aina esimerkiksi aikaulottuvuudesta syntyvät kuvausongelmat mm. synkronisoinnin ilmausten muodossa. Musiikki syntyy usein esimerkiksi yksittäisten äänten välisistä suhteista ja jännitteistä, joten yksittäisten musiikillisten käsitteiden suhde toisiinsa ja kokonaisuuteen on voitava jotenkin ilmaista. Näin pakostakin musiikin kuvauskielet ovat rakenteisia. Edelleen useat tutkijat, mm. Dannenberg (1993, 20–21), ovat sitä mieltä että erityisesti monipuoliset hierarkkiset rakenteet ovat musiikin määrittelyssä erittäin tarpeellisia.

Kun dokumentin eri rakenneosat ovat ohjelmallisesti tunnistettavissa, on mahdollista esimerkiksi tulostaa dokumentti erilaisissa muodoissa tarpeiden mukaan. Dokumentin sisältö voidaan siis erottaa sen ulkoasusta, ulkoasusta voidaan päättää myöhemmin ja dokumentti voidaan esimerkiksi tulostaa hyvin erilaisissa ulkoasuissa. Tämä vähentää mm. tarvetta kirjoittaa samoja asioita yhä uudelleen, vain hieman eri muodoissa. (Salminen 1995, 13.) Musiikin tapauksessa tämä voi tar-

koittaa esimerkiksi sitä, että kun musiikkidokumentti ei sinällään ole sidoksissa mihinkään tiettyyn ulkoasuun, voidaan dokumentin sisältämä sävellys tulostaa useita erilaisia notaatiojärjestelmiä käyttäen.

Tämän mahdollisuuden tärkeys korostuu käytössä olevien notaatiojärjestelmien ja erityisesti viivastonotaation ongelmien takia (ks. kappale 2.4). Olisi erittäin toivottavaa, että sävellyksiä voitaisiin tarkastella helposti perinteisen viivastonotaation lisäksi myös muiden notaatiojärjestelmien mukaisina. Muutamat tutkijat ovat sitä mieltä, että sävellyksen ensisijaisena määrittelymuotona olisi järkevintä käyttää jotain standardia, kattavaa musiikin kuvauskieltä epätarkkojen notaatiojärjestelmien sijaan (Hudak, Makucevich, Gadde ja Whong 1996, 1–2; vrt. Wiggins, Miranda, Smaill ja Harris 1993). Tällöin sävellys olisi alusta saakka tallennettu tarkkana kuvauskielisenä esityksenä, josta soittajien tarpeisiin voitaisiin generoida erilaisia notaatioesityksiä, vaikkapa jollakin tulevaisuudessa käytetyllä viivastonotaatiota tarkemmalla notaatiojärjestelmällä.

Tarpeeksi yksityiskohtainen, tarkka rakenteistus mahdollistaa myös esimerkiksi sen, että dokumentti voidaan konvertoida ohjelmallisesti jollekin tietylle ohjelmistolle sopivaan muotoon. Tiedonhakuun voidaan käyttää tehokkaita, vain haluttuihin dokumentin rakenteisiin kohdistuvia hakumenetelmiä. Dokumentin sisällön rakenteiden järjestystä tai sisältöä voidaan muuttaa ohjelmallisesti. (Salminen 1995, 13.) Esimerkiksi jos nuotit on määritelty musiikkidokumentissa erillisinä rakenteina vaikkapa soitinkohtaisesti, voidaan ne tunnistaa ohjelmallisesti analyysitarkoituksessa, niiden sävelkorkeuksia voidaan automaattisesti muuttaa, sointurakenteita ja harmonioita voidaan automaattisesti hakea ja niin edelleen. Lisäksi yhdestä samasta musiikin kuvauskielisestä esityksestä voidaan ohjelmallisesti tuottaa useita erisisältöisiä visuaalisia esityksiä, esimerkiksi sävellyksen partituuri sekä yksittäiset stemmat.

Yleiskäyttöiselle standardille musiikin kuvauskielelle on siis selvästi tarvetta, kun ajatellaan kuvauskielien nykytilanteen ongelmia sekä kattavalla standardilla saavutettavissa olevia hyötyjä. Mutta on erittäin vaikeaa määritellä minkälainen tällaisen yleiseksi standardiksi kelpaavan kuvauskielen pitäisi olla, toisin sanoen mitä erilaisia rakenteita ja musiikillisia käsitteitä sen tulisi voida esittään jne. Voim-

me vain toivoa, että kieli olisi niin ilmaisuvoimainen ja kattava, että sillä voisi esittää mahdollisimman monenlaista musiikkia kohtuullisella vaivalla.

Tämä johtuu tietenkin musiikista itsestään: on vaikea ennakoida kaikkia mahdollisia tapauksia mitä musiikki voi olla, miten ääni halutaan tuottaa, miten ääntä halutaan varioida, minkälaisia rakenteita musiikin määrittelyssä on hyödyllistä käyttää jne. Musiikki uudistuu sävellys sävellykseltä ja täysin uusia käsitteitä saattaa ilmaantua kuvaamaan musiikkia. (Wiggins et al. 1993, 41; Dannenberg 1993, 20.)

Joitakin erityisesti yleiskäyttöisiksi suunniteltuja kuvauskieliä on kuitenkin aika ajoin kehitelty. Näissä hankkeissa lähtökohtana ei enää ole ollut jokin tietty kuvauskieltä tarvitseva ohjelmisto vaan tällöin on suunniteltu alusta asti pelkkää mahdollisimman yleistä ja kattavaa kuvauskieltä, jota voitaisiin tulevaisuudessa käyttää uusissa sovelluksissa yhteisenä kuvauskielenä. Vain muutama näistä kehitellyistä kielistä on saavuttanut edes jonkinlaista yleisempää kiinnostusta. (Ks. Wiggins et al. 1993.) Mitään kattavaa, yleiskäyttöistä musiikin kuvauskieltä ei siis vieläkään ole levinnyt laajaan yleiseen käyttöön.

2.9 Yhteenveto

Musiikki on hyvin laaja, moniuloitteinen ja käsitteellisestikin vaikeasti hahmotettavissa oleva kokonaisuus. Tämä vaikeus heijastuu myös musiikin koodaamiseen.

Viivastonotaatio on käytetyin musiikillinen koodi, mutta siinä on vakavia puutteita. Tietokoneilla käsiteltäviä musiikillisia koodeja, eli musiikin kuvauskieliä on valtavasti. Nämä kaikki ovat yleensä vahvasti ohjelmisto- ja sovellusalueriippuvaisia, mikä seikka aiheuttaa monia ongelmia. Muun muassa tiedonsiirto eri ohjelmistojen välillä käy hyvin vaikeaksi. Lisäksi esimerkiksi musiikin pitkäaikainen säilyttäminen elektronisessa muodossa vaatisi jonkin vakaan ja ohjelmistoriippumattoman kuvauskielen.

Kuitenkin voidaan nähdä mm. elektronisen tekstin tutkimukseen vertaamalla, että musiikin kuvauskielisen esityksen käyttö musiikin määrittelyyn ja tallentamiseen mahdollistaisi monia tärkeitä hyötyjä. Musiikkia voitaisiin tällöin esimerkiksi en-

tistä tehokkaammin ja automaattisemmin käsitellä elektronisessa muodossa. Jotta hyötyjä olisi enemmän kuin ongelmia, tarvittaisiin käyttöön jokin yleinen, standardi musiikin kuvauskieli.

Seuraavissa luvuissa tarkastellaan tarkemmin erilaisia yleisimpiä olemassa olevia musiikin kuvauskieliä. Aluksi tarkastellaan sovellusalue- ja ohjelmistokohtaisia kieliä siten, että tarkasteltavat kuvauskielet on jaettu kategorioihin tämän luvun kappaleessa 2.6 esiteltyjen ohjelmistotyyppien mukaan. Kunkin sovellusalueen kielet ovat yleensä perusperiaatteiltaan hyvin samanlaisia, joten näitä voidaan tarkastella ryhmänä hyvin yleisellä tasolla näiden yhteisistä piirteistä käsin. Ainoastaan sävellysohjelmistojen kieliä ei tässä työssä käsitellä. Tämä johtuu siitä, että sävellysohjelmistoilla tuotetaan yleensä vain suoraan muihin ohjelmistoihin käsiteltäväksi sopivia kuvauskielisiä esityksiä, ja sävellysohjelmistojen sisäisten ohjelmointikielityyppisten kuvauskielten esittely ei kuulu tämän työn aihepiiriin.

Näiden lisäksi luvussa 7 tarkastellaan myös joitakin tähän mennessä tutkituimpia ja lupaavimmiksi havaittuja yleiskäyttöisiksi suunniteltuja kuvauskieliä. Näihin kieliin paneudutaan yksilötasolla edellisiä tarkemmin jo niiden kiinnostavan lähtökohdan eli yleiskäyttöisyyden takia.

3 Kontrolliohjelmistojen kuvauskielet

Kontrolliohjelmistoissa käytettävistä kielistä tässä esitellään vain MIDI, koska se on käytännöllisesti katsoen sovellusalueensa ainoa yleiseen käyttöön levinnyt kuvauskieli.

Muutamia muitakin kieliä musiikkilaitteiden reaaliaikaiseen kontrollointiin on toki kehitelty, mutta käytännöllisesti katsoen mikään näistä ei ole säilynyt hengissä suunnitteluvaihetta kauemmin. Esimerkiksi joitakin erikoislaitteita varten kehitellyt kielet on useimmiten lopulta korvannut juuri MIDI. Lisäksi kyseisiä erikoislaitteita on käytössä lähinnä alan tutkimuslaitoksissa ja laboratorioissa. Eräs tällainen tapaus on Max V. Mathewsin kehittämä *Radio Baton* -niminen kapelimestarin tahtipuikkoa muistuttava ohjauslaite. Alunperin laite ja siihen liittyvät ohjelmistot keskustelivat omalla kontrollikielellään, mutta nykyään kielenä käytetään MIDIä. (Mathews 1997, 153.)

3.1 MIDI

MIDI eli Musical Instrument Digital Interface on tiedonsiirtokieli elektronisten musiikkilaitteiden reaaliaikaiseen ohjaukseen. Musiikillisia tapahtumia tai muuta informaatiota lähetetään *MIDI-viesteinä* ohjattavalle laitteelle jonkinlaista *MIDI-ohjainta* käyttäen. MIDI-ohjain voi olla esimerkiksi koskettimisto tai tietokoneseen tai johonkin kielisoittimeen asennettava lisälaite, joka osaa tuottaa MIDI-viestejä. Näin esimerkiksi MIDI-koskettimistolla voidaan soittaa musiikkia tavallisen kosketinsoittimen tapaan. Tällöin kuitenkin soittoa vastaavan äänen tuottaakin esimerkiksi jokin MIDI-viestejä ymmärtävä syntetisoijamoduli, jolle MIDI-koskettimisto syöttää tiedot koskettimien painalluksista MIDIä käyttäen.

MIDIä käytetään paljon myös erilaisten studiolaitteiden kuten miksereiden ja tallentimien ohjaukseen. Tällöin MIDI-viesteissä ei välitetä niinkään musiikillista informaatiota, vaan muita laitteiden kontrolliviestejä, esimerkiksi tietoa mikserin

tietyn kanavan äänenvoimakkuudesta. (Hewlett, Selfridge-Field, Cooper, Field, Ng ja Sitter 1997, 42.)

MIDI-viestejä voidaan tallentaa ja editoida *sekvensserillä*, joka voi olla yksittäinen erillislaite tai toteutettu esimerkiksi tietokoneohjelmistona. Sekvensserin avulla voidaan esimerkiksi studiolaitteita ohjelmoida ennakkoon, tai soittaa ääntä tuottavia MIDI-laitteita sekvensserille tallennettujen MIDI-viestien perusteella ilman tosiaikaista MIDI-ohjaimen soittajaa.

MIDI-viestejä voidaan myös tallentaa binäärimuotoiseen *MIDI-tiedostoon*. Paremman standardin puuttuessa MIDI-tiedostosta on tullut de facto -standardi muun muassa nuotinkirjoitusohjelmien väliseen tiedonsiirtoon. Lähes kaikki muutkin musiikkiohjelmistot osaavat lukea ja tuottaa MIDI-tiedostoja. Erityisesti kiinteästi monien MIDI-laitteiden yhteydessä käytettäviä sekvensseriohjelmistoja on olemassa valtava määrä. (Hewlett et al. 1997, 43.)

Musiikin kuvaukseen alkuperäinen MIDI on kuitenkin toivottoman yksinkertainen. Mitään musiikillisia rakenteita tai musiikillisten käsitteiden välisiä suhteita ei MIDIllä voida esittää. Koska MIDI on kehitetty juuri reaaliaikaiseen tiedonsiirtoon, MIDI-viestit yksinkertaisesti kertovat laitteelle vain sen, mitä sen juuri sillä hetkellä täytyy tehdä. MIDI-viestit ovat kielellisesti ilmaistuna esimerkiksi muotoa "aloita nuotin X soitto laitteella Y" ja "lopeta nuotin X soitto laitteella Y". Muutenkin MIDI kykenee esittämään vain muutamia perustavinta laatua olevia musiikillisia tapahtumia, ja näitäkin varsin suppeasti. Esimerkiksi äänen korkeutta kuvaavia MIDI-viestejä on rajallinen määrä, ja näiden määrittelemät soitettavat sävelkorkeudet on ennalta kiinnitetty. MIDIllä ei myöskään voida esittää mitään suoraan äänen tuottamiseen liittymätöntä informaatiota, esimerkiksi visuaalisia yksityiskohtia nuottijulkaisujen tuottamista varten. (Hewlett et al. 1997, 68; Wiggins et al. 1993, 34–35; McLane 1996, 248–249; ks. myös Dannenberg 1993, 20–21.)

MIDI mahdollistaa omien, standardissa määrittelemättömien ohjelmisto- tai laitekohtaisten *erikoisviestien* (engl. system exclusive) määrittelyn. Näitä käyttäen on syntynyt noin sata erilaista epästandardia MIDIn laajennusta, jotka pyrkivät korjaamaan alkuperäisen MIDIn puutteita. Jotkut pyrkivät laajentamaan MIDIä mm.

yleisemmän musiikin kuvauksen suuntaan, toiset taas esimerkiksi lisäävät erikoisviestejä nuotinkirjoitusohjelmistoissa tarvittavien visuaalisten yksityiskohtien kuvaamiseen.

Näistä laajennuksista vain noin tusinaa on koskaan sovellettu käytäntöön, ja yleensä vain juuri niissä laitteissa tai ohjelmistoissa, joiden yhteyteen ne alunperin suunniteltiin. Laajempaan käyttöön ne eivät ole levinneet. (Hewlett et al. 1997, 69.)

4 Äänisynteesiohjelmistojen kuvauskielet

Äänisynteesiohjelmistoja on valtava määrä ja uusia syntyy jatkuvasti lisää. Ensimmäiset kehitettiin jo viisikymmentäluvulla, joten äänisynteesiohjelmistot ovat musiikkiohjelmistoista vanhimpia. Kuuluisin näistä lienee Max V. Mathewsin jo vuonna 1957 kehittämä *Music V* (Selfridge-Field 1997a, 564), jonka käsitteellisiin periaatteisiin monet nykyisetkin äänisynteesiohjelmistot perustuvat, tosin huomattavasti jalostetummassa muodossa. Niin perustuu myös tässä luvussa esiteltävä Csound.

Muita äänisynteesiohjelmistoja ei tässä työssä käsitellä. Tämä johtuu siitä, että kaikki käytetyimmät äänisynteesiohjelmistot ovat periaatteiltaan ja kuvauskieliltään lähes täysin vastaavia Csoundin kanssa. Eräs uusimmista tällaisista kielistä on *MPEG-4* -standardin sisältämä äänisynteesikieli *Structured Audio Orchestra Language* (SAOL) (Scheirer ja Vercoe 1999; The MPEG Home Page [online] 1999). MPEG-standardina tämä saattaa tulevaisuudessa muodostua hyvinkin kiinnostavaksi.

4.1 Csound

Csound on tällä hetkellä suosituin äänisynteesiohjelmisto. Se julkaistiin vuonna 1991 ja sitä kehitetään edelleen aktiivisesti. Csound on saatavissa lähes kaikille tietokonejärjestelmille, mm. useimmille UNIX-ympäristöille sekä Mac- ja PC-tietokoneille. Lisäksi Csoundin lähdekoodi on saatavissa ilmaiseksi, mikä tietysti lisää ohjelmiston suosiota. Suuri määrä erilaisia apuohjelmia, konversio-ohjelmia, instrumentin kehitysympäristöjä jne. on myös saatavilla.

Seuraavassa esittelyssä mainituista tiedoista voi lukea tarkemmin mm. Bainbridgeltä (1997), Boulangerilta (2000) tai Csoundin WWW-sivulta (The Csound Front Page [online] 2000).

Musiikin auditiivisen esityksen tuottamiseen Csound käyttää kahta eri kuvauskieltä, jotka molemmat tallennetaan yleensä eri tiedostoihin. Näistä kielistä ja tiedostoista käytetään nimitystä *orchestra* ja *score*.

Orchestra-tiedostossa määritellään äänentuotto signaalitasolla ohjelmoimalla orchestra-kielellä erilaisia *instrumentteja*, joita käyttäen score-tiedostossa koodattu musiikki soitetaan.

Score-tiedostossa luetellaan kukin sävellyksen musiikillinen tapahtuma yhdellä rivillä. Kullakin rivillä voi olla mielivaltainen määrä arvoja, joista kukin sijaitsee omalla sarakkeella. Nämä arvot ovat arvoja sarakenumeroa vastaavalle *p-muuttujalle* `p1`, `p2`, `p3` jne. Sarakkeiksi katsotaan välilyönnein tai tabulaattorimerkein erotetut merkkijonot.

Kolmen ensimmäisen p-muuttujan merkitys on kiinteästi määrätty. Ne ovat

- orchestra-tiedostossa määritellyn instrumentin numero jolle kaikki rivin arvot välitetään, ts. millä instrumentilla rivin kuvaama musiikillinen tapahtuma soitetaan
- riviä vastaavan musiikillisen tapahtuman alkuaika sekunteina kappaleen alusta
- riviä vastaavan musiikillisen tapahtuman kesto sekunteina

Näitä seuraavien p-muuttujien arvojen tulkinta riippuu valitusta instrumentista ja sen ohjelmoinnista. Kaikki p-muuttujat välitetään ao. orchestra-tiedoston instrumentille, joka voi käyttää niitä ohjelmointinsa mukaan erilaisten signaalitasolla operoivien funktioiden parametreinä, esimerkiksi äänen voimakkuuden säätämiseen, vibraaton lisäämiseen ääneen jne. Instrumentissa p-muuttujien arvoihin voidaan viitata muuttujan nimellä, eli esim. `p3`.

Kuvassa 4.1 (s. 29) on lyhyt esimerkki orchestra- ja score-kielistä. Orchestra-tiedostossa luetellaan ensin erilaisia alkuarvoja, esim. tuotettavan äänitiedoston näytteenottotaajuus ja kanavien lukumäärä. Tämän jälkeen alkaa instrumenttien listaukset. Instrumentin ohjelmointi määritellään `instr` ja `endin` -merkintöjen välissä. Esimerkin ainoassa instrumentissa `instr 1` on `oscil`-nimiselle oskillaattorifunktiolle välitetty p-muuttujat `p5` ja `p4`. `oscil`-funktiolle annettu ensim-

mäinen parametri tarkoittaa äänenvoimakkuutta, joka siis tässä tapauksessa saadaan muuttujasta p5. Toinen oscil-funktion parametri tarkoittaa äänenkorkeutta, joka tässä saadaan muuttujasta p4 muuntamalla se ensin funktiolla cpspch.

oscil-funktion tuottama äänisignaali tallennetaan muuttujaan asig. Ääni tuotetaan kuuluvaksi funktiolla outs, jolle annetaan parametreinä muuttujat, joista äänisignaali saadaan. Tässä esimerkissä tuotetaan kaksi kanavaista ääntä eli stereoääntä, joten outs saa kaksi parametriä, vasemman ja oikean kanavan signaalilähteen. Esimerkissä kuitenkin kumpaankin kanavaan syötetään sama signaali, eli asig.

Myös score-tiedostossa määritellään aluksi joitakin alkuasetuksia. Esimerkissä ensimmäisellä rivillä määritellään aaltomuototaulukko, jota jotkut instrumenteissa käytettävät funktiot hyödyntävät. Tämän jälkeen tiedostossa on koodattu neljä musiikillista tapahtumaa neljällä i1 -alkuisella rivillä. i1 on tässä p1-muuttujan arvo, joka siis tarkoittaa, että rivillä määritelty musiikillinen tapahtuma soitetaan orchestra-tiedostossa määritellyllä instrumentilla numero yksi.

Ensimmäinen musiikillinen tapahtuma alkaa hetkessä 0, eli heti kappaleen alusta. Seuraava kohdassa 0.5, eli 0.5 sekuntia kappaleen alusta, jne. Ensimmäisen musiikillisen tapahtuman kesto on 3 sekuntia, toisen 0.5 sekuntia ja molempien seuraavien 2 sekuntia.

Seuraavien p-muuttujien merkitys selviää vasta katsomalla riviä vastaavaa instrumenttimäärittelyä orchestra-tiedostosta. Tässä neljännen sarakkeen eli p4-muuttujien arvot tarkoittavat äänen korkeutta instrumentissa käytetylle cpspch-funktiolle sopivassa muodossa. Tässä muodossa ennen pistettä määritellään oktaavi, pisteen jälkeen sävel tämän oktaavialan sisältä. Pisteen jälkeinen luku voi olla mitä tahansa väliltä 0 – 12. Viimeisen sarakkeen eli p5-muuttujan saamat arvot tarkoittavat tässä äänenvoimakkuutta.

Perinteisempiin nuotti- ja aika-arvoihin perustuvaa musiikkia voidaan Csoundissa helpoiten koodata score-kielen korvaavalla *scot*-kielellä. Tämä pyrkii tuomaan viivastonotaation käsitteitä paremmin Csoundin piiriin. Scot-kielessä mm. nuottiarvoja voidaan merkitä sen sävelnimellä (c, d, e jne.) ja aika-arvoja jakajalla (kokonuotti on 1, puolinuotti on 2 jne.). Instrumenteille voi numeron sijasta antaa

Orchestra

```
sr = 44100
kr = 44100
ksmps = 1
nchnls = 2

        instr 1
asig    oscil   p5, cpspch(p4), 1
        outs    asig,asig
        endin
```

Score

```
f1 0 256 10 1

i1    0      3     8.00   4000
i1    0.5    0.5   9.00   8000
i1    1      1     8.02   10000
i1    2      1     8.04   10000
```

Kuva 4.1: Lyhyt, mutta sellaisenaan toimiva esimerkki Csoundin score- ja orchestra-kielistä.

myös nimen, mikä sekin helpottaa kirjoitustyötä, ja ennen kaikkea hahmottamista. (Bainbridge 1997, 129.)

4.2 Yhteenveto

Jo pintapuolisen vilkaisun jälkeen score-tyyppinen äänisynteesiohjelmistojen kieli vaikuttaa sopivan varsin huonosti yleiseen musiikin kuvaukseen.

Ensinnäkin score-kieli on hyvin kankea soitinmusiikin kuvaamiseen vaikeasti hahmotettavine parametrilistoineen. Csoundin Scot helpottaa tilannetta vain hieman, koska pohjimmiltaan se on vain kirjoittamista helpottamaan luotu hieman uusi käyttöliittymä alla olevaan score-kieleen.

Vakavampi, kielen monikäyttöisyyttä oleellisesti vähentävä ongelma on se, että score-kielellä koodattujen musiikillisten käsitteiden merkitys syntyy - tai paljastuu - vasta tulkittaessa score-tiedoston rivien sarakkeiden arvoja orchestra-kielen instrumentissa. Instrumentit puolestaan operoivat lähinnä hyvin lähellä signaalitasoa erilaisten synteesitekniikoiden parissa. Tästä syystä score-kielellä koodattujen musiikillisten käsitteiden merkitystä on varsin mahdotonta tulkita ohjelmallisesti.

Lisäksi score-kieli ei mitenkään tue musiikillisten rakenteiden kuvaamista, koska musiikilliset tapahtumat vain yksinkertaisesti luetellaan riveittäin. Näin voidaan sanoa, että score-tyyppiset äänisynteesiohjelmistojen kuvauskielet eivät kovin hyvin sovellu yleiseen musiikin kuvaukseen.

5 Nuotinkirjoitusohjelmistojen kuvauskielet

Nuotinkirjoitusohjelmistot voidaan jakaa kahteen eri tyyppiin sen mukaan, käyttävätkö ne graafista käyttöliittymää vai eivät. Ennen graafisten käyttöliittymien yleistymistä musiikki kirjoitettiin yleensä käsin suoraan ohjelmiston käyttämällä musiikin kuvauskielellä tavallista tekstieditoria käyttäen. Tällaisen kuvauskielen tulee siis koostua tavallisista aakkosnumeerisista merkeistä. Itse nuotinkirjoitusohjelmisto toimii tällöin hieman ohjelmointikielten kääntäjien tapaan. Ohjelmisto ensin lukee ja tulkitsee kuvauskielisen esityksen ja generoi sitten sen perusteella valmiin partituurin esimerkiksi kuvana, tai PostScript-muodossa tulostusta varten.

Koska nuotinkirjoitusohjelmistot on tarkoitettu yleensä pelkästään viivastonotaatioesitysten käsittelyyn, koodataan näiden käyttämissä kuvauskielissä lähinnä vain viivastonotaation käsitteitä eikä niinkään yleisempiä, abstraktimpia musiikillisia käsitteitä. Kunkin kuvauskielen ilmaisuvoima, eli kuvauksen kattavuus on siis täysin kieltä käyttävän ohjelmiston implementoiman notaatiosymbolivalikoiman varassa. Uusia, esimerkiksi sävellyskohtaisia rakenteita ja käsitteitä ei näin voida ottaa kuvauksessa käyttöön. (Dannenberg 1993, 24.)

Seuraavassa on esitelty hyvin lyhyesti muutamia tunnetuimpia ja käytetyimpiä ASCII-muodossa tallennettavia nuotinkirjoitusohjelmistojen musiikin kuvauskieliä. Tyypillisin näistä lienee ensimmäisenä esiteltävä DARMS.

5.1 DARMS

Jo 1960-luvulla kehitetty *DARMS* (Digital Alternate Representation of Musical Scores) on vanhin vielä käytössä oleva musiikin kuvauskieli. Kieli on alunperin syntynyt samannimisen ohjelmiston yhteydessä. DARMS koostuu merkin tai parin pituisista koodimerkinnöistä, joilla kuvataan viivastonotaation symboleja (esim. nuotti, tauko) ja rakenteita (esim. tahteja, kaaria, nuottiryhmiä, sointuja). DARMS oli ensimmäinen ohjelmisto ja kuvauskieli, jolla voitiin esittää myös va-

kiintuneimpia viivastonotaatiossa käytettyjä erikoismerkkejä ja symboleita, kuten esimerkiksi artikulaatiomerkkejä, kaaria ja dynamiikan vaihteluja ilmaisevia kuvioita. (McLane 1996, 242.)

DARMsin koodit vastaavat suoraan tulostettavia symboleja. Samoin kielellä esitettävissä olevat rakenteet vastaavat viivastonotaatioesityksissä esitettäviä visuaalisia ryhmiä, esim. peräkkäisten nuottien yhdistämistä kaarella. DARMSin painottuneisuus nuottijulkaisuihin näkyy tämän lisäksi muun muassa siinä, että itse sävelkorkeuksia ei kielessä esitetä, vaan pelkästään nuotin paikka viivastolla määritellään suhteessa viivaston ensimmäiseen viivaan. Tämä vaikeuttaa huomattavasti säveltasojen ohjelmallista tunnistamista. Erityisen ongelmallista on ns. enharmonisten sävelten tunnistaminen, joissa säveltasoon vaikuttaa nuotin sijainnin lisäksi erilaiset etumerkit. (Selfridge-Field 1997c, 165.)

DARMS-kielestä on kehitetty useita erilaisia murteita ja laajennuksia, joitakin erityisesti esimerkiksi luuttutabulatuurien ja mensuraalinotaation käsittelyyn ja julkaisuun. Näitä kaikkia ei ole kuitenkaan implementoitu ohjelmistoiksi asti. *Note Processor* -nuotinkirjoitusohjelmiston *Note-Processor DARMS* (Dydo 1997) lienee tällä hetkellä DARMS:n murteista levinnein. (Selfridge-Field 1997c.)

Kustannusyhtiö A-R Editions tuottaa nuottijulkaisunsa yhtiön omalla DARMS-murteella ja ohjelmistolla nimeltään *A-R DARMS*. Vuosina 1981 – 1995 he julkaisivat yli 150 000 sivua A-R DARMS:lla valmistettuja nuottijulkaisuja. Tässä symbolivalikoimaa on laajennettu Note-Processor DARMS:iin nähden, ohjelmistoon on lisätty automatiikkaa päättelemään sopivia symbolien asetteluja, ja kuvauskoodeja on vaihdettu toisiin tai lyhennetty entisestään koodin käsin kirjoittamisen nopeuttamiseksi. (Hall 1997, 193–194.)

5.2 SCORE

SCORE on kaupallinen, alunperin MS-DOS-käyttöjärjestelmässä toimiva nuotinkirjoitusohjelmisto ja kuvauskieli. SCORE-kuvauskieli on julkinen. SCORE:lla tuotetaan nuottijulkaisuja PostScript-tiedostoina tulostusta varten. Monet merkit-

tävät musiikin julkaisijat käyttävät edelleenkin SCORE:a nuottijulkaisujen valmistamiseen. (Smith 1997, 252.)

SCORE käsittää kaksi erilaista kuvauskieltä. Toista käytetään tiedon syöttämiseen (*syöttökieli*, engl. input code) ja toista tiedon tallentamiseen (*tallennuskieli*, engl. parametric file format). Joitakin partituurin yksityiskohtia voidaan määritellä vain tallennuskielessä. (Smith 1997, 253.) SCORE toimii niin, että syöttökieli konvertoidaan ensin tallennuskieleksi, ja vasta tästä voidaan muodostaa lopullinen partituuri.

Syöttökieli muistuttaa DARMS-kieltä, eli se koostuu muutaman merkin koodimerkinnöistä, jotka kuvaavat viivastonotaatioesityksen symboleita ja rakenteita. Tallennuskieli taas koostuu riveistä, joista kukin määrittelee yhden partituurille tulostettavan *graafisen objektin*. Jokaisella tällaisella rivillä on joukko numeerisia parametreja välilyönneillä erotettuna. Neljä ensimmäistä parametriä ovat kaikille graafisille objekteille samoja. Ne ovat objektin tyyppi, siihen liittyvän viivaston numero sekä objektin sijainti vaaka- ja pystysuunnassa. Näitä seuraavat parametrit riippuvat valitusta objektin tyypistä. (Smith 1997, 253–257.)

SCORE:n tallennuskieli on siis periaatteeltaan samanlainen kuin esimerkiksi Csoundissa käytetty musiikin kuvauskieli. On selvää, että pelkistä numeerisista riveistä koostuva tallennuskieli on äärimmäisen vaikealukuinen.

5.3 Common Music Notation

Common Music Notation (CMN) on Lisp-ohjelmointikieleen perustuva kuvauskieli ja nuotinkirjoitusohjelmisto, joilla tuotetaan PostScript- ja Quickdraw-tiedostoja tulostamista varten.

CMN-ohjelmistolla voidaankin tehdä erilaisia varsin mutkikkaita operaatiota käsiteltävälle musiikkidokumentille. Esimerkiksi kokonaisen partituurin kuvauksesta voidaan erottaa yksittäisten soittajien stemmat sekä eri äänien osuuksille voidaan tehdä tarvittaessa transponointeja automaattisesti

CMN-kielessä musiikilliset käsitteet ja rakenteet kuvataan Lisp-kielen lausekkeil-

la, eli periaatteessa sisäkkäisinä listoina. Näin ollen ainakin musiikillisten käsitteiden konteksti pysyy hyvin selvillä, koska eräänlainen hierarkkisuus sisältyy itse kielen rakenteeseen. (Schottstaedt 1997, 217–219.)

Muiden tässä mainittujen nuotinkirjoitusohjelmistojen kielten tavoin CMN on kuitenkin liian sidoksissa viivastonotaation käsitteistöön, jotta sitä voitaisiin pitää hyvänä pohjana yleiskäyttöiselle kuvauskielelle. Vaikka Lisp-kielen listarakenne sellaisenaan tukeekin esim. materiaalin monikäyttöisyyttä ja automaattista käsittelyä, voidaan CMN:n Lisp-lausekkeilla määritellä vain hyvin yleisiä viivastonotaation symboleja ja rakenteita.

5.4 TEX-ladontaohjelmiston makropaketit

Suositulle ja laajaan käyttöön levinneelle TEX-ladontaohjelmistolle on olemassa useita erilaisia viivastonotaatioesitysten tuottamiseen tarkoitettuja makropaketteja. Näiden käyttäjällä on TEX-ladontaohjelmiston normaalikäytöstä tuttu valta yksityiskohtien tulostukseen. Esimerkiksi erilaisia symboleita ja omaa grafiikkaa voidaan sijoittaa partituurin sivulla minne tahansa tarpeen mukaan.

Varhaisin makropaketti *MuTEX* kehitettiin yksiäänisen musiikin kirjoittamiseen. Sen jatkokehitys on jo lopetettu. Uudemman *MusicTEX*:n avulla voidaan kirjoittaa myös moniäänistä musiikkia ja suuria partituureja.

MusiXTEX on edelleen MusicTEX:n laajennus ja nykyään suositeltavin makropaketti. Siinä on valmiiksi määritelty monia erikoisviivastoja mm. perkussiosoittimille. Symbolivalikoima on kattava ja se sisältää mm. mensuraalinotaatiosymboleja. Käyttäjällä on suuri valta moniin pieniin yksityiskohtiin, esimerkiksi viivastojen viivojen lukumäärä on vapaasti valittavissa. Kuvauskieli koostuu jälleen muutaman merkin pituisista koodimerkinnöistä, joista kukin merkitsee yhtä graafista symbolia. (Icking 1997, 222–224.)

Näiden makropakettien kuvauskielien ongelma on saman tyyppinen kuin monilla muillakin nuotinkirjoitusohjelmistoilla: kuvauskielessä musiikilliset käsitteet tallennetaan ja kirjoitetaan ladontaohjelmiston näkökulmasta, tavallaan sijoitte-

lemalla symboleja tulostettavalle sivulle, eikä niinkään yleisempinä abstrakteina käsitteinä.

5.5 Graafiset nuotinkirjoitusohjelmistot

Graafisten käyttöliittymien yleistyttyä kuvauskielestä tuli usein käyttäjälle täysin näkymätön. Lisäksi nuotit ja muut musiikillisia käsitteitä kuvaavat nuottikirjoituksen symbolit voitiin nyt sijoittaa paikoilleen kätevästi hiirellä. (Selfridge-Field 1997d, 32.) Enää ei siis ollut välttämättä tarpeen määritellä kuvauskielessä musiikillisia rakenteita, käsitteitä ja näiden suhteita läheskään niin tarkasti kuin ennen. Kun symbolit vain saatiin oikeille paikoilleen partituurin sivulla, se riitti mainiosti nuottijulkaisujen tekemiseen.

Tämä voi tarkoittaa esimerkiksi sitä, että tällaisella kuvauskielellä tallennetaan vain notaatiosymbolien sijainti partituurilla, mutta ei enää tietoa sen suhteesta sitä ympäröiviin musiikillisiin käsitteisiin tai niitä kuvaaviin symboleihin. (Selfridge-Field 1997b, 31; Dannenberg 1993, 24.)

Musiikkidokumentin yleiskäyttöisyyden kannalta tällainen kuvauskieli on luonnollisesti ongelmallinen. Tällaisessa muodossa tallennetun musiikin erilaiset käyttömahdollisuudet rajoittuvat yleensä juuri notaatioesitysten tuottamiseen. Jotta musiikkidokumenttia voitaisiin monipuolisesti käsitellä ohjelmallisesti, täytyy sen sisältämien musiikillisten käsitteiden merkitys ja konteksti olla selvästi tiedossa.

Graafista käyttöliittymää käyttäviä nuotinkirjoitusohjelmistoja on nykyään valtavasti. Niiden kuvauskieliä ei tässä tämän enempää käsitellä. Osaksi näiden ohjelmistojen kuvauskielten edellä mainittujen puutteiden takia ja osaksi siksi, että monien suosituimpien - kaupallisten - ohjelmistojen kuvauskielten määritykset ovat salaisia tai niiden käyttöoikeus on tarkasti kontrolloitu. Tällainen on esimerkiksi erittäin suositun *Finale*-nuotinkirjoitusohjelmiston kuvauskieli *Enigma*.

5.6 Yhteenveto

Viivastonotaation valta-aseman takia nuotinkirjoitusohjelmistojen käyttämät musiikin kuvauskielet ovat tällä hetkellä yleisin tapa tallentaa musiikkia elektronisessa muodossa. Tämä on todettu mm. CANTATE-projektin (ks. kohta 7.2.3 sivulla 53) tekemissä kyselytutkimuksissa (Survey of Music Publishers 1996, 4; Survey of Music Libraries 1996). Samojen tutkimusten mukaan nuotinkirjoitusohjelmistoista edelleen suosituimpia ovat graafista käyttöliittymää käyttävät, kuten esimerkiksi kaupalliset *Finale* ja *Encore*.

Edellä viitattiin useisiin viivastonotaation käsitteille perustuvien kuvauskielten ongelmiin. Tärkein näistä lienee kuvauksen ilmaisuvoiman rajoittuminen käytettävissä olevaan notaatiosymbolivalikoimaan. Kaikki edellä esitellyt kuvauskielet määrittelevät koodeillaan vain viivastonotaatioesitykseen tulostuvia symboleja.

Erityisen ongelmallisia ovat kuvauskielet, jotka tallentavat notaatiosymbolit pääasiassa graafisena tietona. Tällöin ollaan kuvauskielen yleiskäyttöisyyden kannalta varsin lähellä visuaalista esitystä, eli pelkkänä kuvana tallennettua notaatioesitystä.

6 Analyysiohjelmistojen kuvauskielet

Automaattisen tietojenkäsittelyn mahdollisuudet huomattiin aikanaan myös musiikin parissa, ensin äänisynteesi- ja nuotinkirjoitusohjelmistojen muodossa. Hieman myöhemmin alettiin kiinnostua myös itsessään musiikillisen tiedon esittämisestä tietokoneella käsiteltävässä muodossa, jotta koodattuja sävellyksiä voitaisiin koneellisesti tutkia.

Analyysiohjelmistot tutkivat musiikillista informaatiota monesta eri näkökulmasta ja voivat myös tarvittaessa manipuloida sävellystä hyvin monipuolisesti. Sävellyksestä halutaan ehkä tehdä hakuja lyhyen melodian tai jonkin teeman perusteella, kartoittaa siinä käytettyjä harmonioita ja sointurakenteita, tehdä tarvittavia transponointeja eri soittimille jne. Ohjelmistojen käyttämän kuvauskielen tulee siis olla melkoisen laaja ja kattava, jotta tällaisia seikkoja voitaisiin ohjelmallisesti kuvauskielisestä esityksestä tunnistaa.

6.1 Humdrum ja Kern

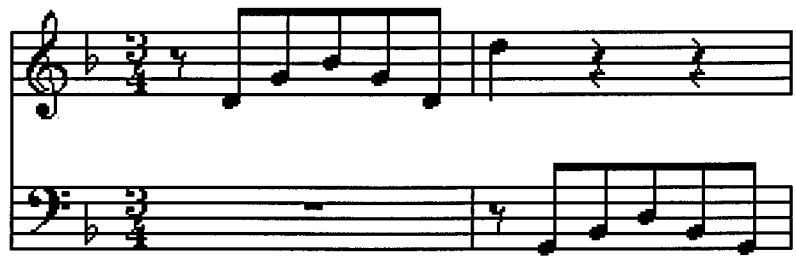
Humdrum on kuvauskieli, jota siihen liittyvän ohjelmiston yhteydessä käytetään laajasti musiikin ja yleensä musiikkiin liittyvän tiedon analysointiin. Lähes millaista informaatiota tahansa on mahdollista esittää Humdrumilla, esimerkiksi tanssiaskeleita, tunnetiloja, MIDI-viestejä jne. Humdrum ei määrittele mitään kiinteää tiedon esitystapaa, vaan käyttäjä voi itse määritellä uusia esitystapoja erityyppisille tiedoille tarpeen mukaan. (Huron 1997, 375.)

Kern on Humdrumin erityinen esitystapa tavallisimmille viivastonotaatiossa käytetyille musiikillisille käsitteille. Kern-esitystavassa näistä voidaan koodata esimerkiksi äänen korkeutta (soivan sävelen nimen mukaan), musiikillisen tapahtuman kestoa, artikulaatiomerkintöjä jne. Muu musiikillinen informaatio, kuten esimerkiksi dynamiikka, äänen korkeus jollain muulla tavalla kuin sävelen nimellä merkittynä, musiikin visuaalinen esitys sekä äänisynteesiohjelmistoille tarkoitettu

musiikin auditiivisen esityksen tuottamiseen tarkoitettu tieto voidaan esittää muilla Humdrumin esitystavoilla. (Huron 1997, 377.)

Humdrum-kuvauskielessä kaikki yhtäaikaiset musiikilliset tapahtumat esitetään samalla rivillä. Sävellyksen eri äänet ja eri esitystapaa käyttävät tiedot ovat omilla sarakkeillaan. Sarakkeet erotetaan toisistaan tabulaattorimerkillä. Kuvassa 6.1 on lyhyt esimerkki Humdrum-tiedostosta. (Huron 1997, 376.)

```
!! J.S. Bach, Praeambulum BWV 930
**kern  **kern
*staff2 *staff1
*k[b-]  *k[b-]
*M3/4   *M3/4
=1-     =1-
2.r     8r
.       8d/L
.       8g/
.       8b-/
.       8g/
.       8d/J
=2      =2
8r      4dd\
8GG/L   .
8BB-/   4r
8D/     .
8BB-/   4r
8GG/J   .
=3      =3
```



Kuva 6.1: Alkua Humdrum-kuvauskielellä esitetystä kappaleesta sekä sama kappaleen osa viivastonotaatioesityksenä. Esimerkki on peräisin Humdrum Toolkitin WWW-sivulta (Humdrum Toolkit [online] 1998).

Humdrum Toolkit on ilmainen Humdrum-kuvauskielen käyttöön tarkoitettu ohjelmistopaketti. Se sisältää noin 70 erilaista pientä työkalua Humdrumilla esitettyjen tietojen manipulointiin ja tutkimiseen. Nämä sisältävät mm. konversio-ohjelman kern-tiedon ja MIDI-tiedon välille. Erikseen on olemassa myös ohjelmia, joilla kern-esitystavan mukaista tietoa voidaan tuottaa MuseData-, SCORE- ja Plaine and Easie Code -kuvauskielistä. PostScript-muotoisen viivastonotaatioesityksen tuottamiseen on olemassa PC- ja UNIX-ympäristöihin ohjelma nimeltä *Mup*. Graafinen käyttöliittymä Humdrumin kern-esitystavalle on saatavilla Windows-käyttöjärjestelmille sekä UNIX-maailman X-Window Systemille. (Huron 1997, 398–399.)

Humdrumia on käytetty lukuisissa erilaisissa tärkeissä musiikkitieteellisissä tutkimuksissa, ja noin 9000 sävellystä ja arviolta noin 10 000 katkelmaa on kirjoitettu Humdrumilla, lähinnä kern-esitystapaa käyttäen (Huron 1997, 399).

Humdrumia on käytetty oleellisena osana monissa uraa uurtavissa musiikkitieteellisissä tutkimuksissa, ja tämän perusteella Humdrum työkaluineen on varsin vaikuttava kokonaisuus. Näin on arvioinut mm. Wild artikkelissaan (Wild 1996). Humdrumin suurimpana ongelmana Wild pitää sen vaikeaa käyttöä: Humdrumin työkalut koostuvat komentoriviltä käsin ajettavista komennoista ja Humdrum-tiedostoja kirjoitetaan lähinnä käsin tekstieditorilla. Tiedostojen käsittely onkin näin vaivalloista ja erityisesti hyvin virhealtista, koska musiikillisten käsitteiden merkitys määräytyy rivi- ja sarakesijainnin perusteella (ks. kuva 6.1, s. 38).

6.2 MuseData

Myös *MuseData* on suunniteltu tarkkaan musiikillisten käsitteiden kuvaamiseen, mutta kuitenkin Humdrumia enemmän juuri viivastonotaation käsitteisiin perustuen. MuseData on kehittäjänsä mukaan tarkoitettu juuri partituurien sisältämän informaation koodaamiseen (Hewlett 1997, 402). MuseDataan sisältyy myös erityisiä rakenteita MIDI-viestien koodaamiseen.

MuseDatan käyttöön on kehitetty mm. konversio-ohjelmia, joilla MuseData-kielestä voidaan tuottaa DARMS-, SCORE-, Humdrum kern- sekä MIDI-

tiedostoja. Myös partituurien ja erillisten stemmojen tulostamiseen on olemassa ohjelmistoja, sekä hakuohjelmia rytmin, melodian ja harmonian hakuun MuseData-tietokannasta. (Hewlett 1997, 402–404.)

MuseData on tarkoitettu pääasiassa lähdekieleksi, joka voidaan tarvittaessa kääntää jollekin tiettyyn tarkoitukseen paremmin sopivalle kuvauskielelle. Tämä kohdekieli voi esimerkiksi olla jonkin nuotinkirjoitusohjelmiston oma kuvauskieli. MuseData ei yritäkään kuvata täydellisesti suoraan musiikillisiin käsitteisiin liittymätöntä tietoa kuten esimerkiksi nuottijulkaisujen tekoon tarvittavia visuaalisia aspekteja, kuten nuotin absoluuttisia vertikaalisia paikkoja viivastolla. Ideana on ollut, että vasta MuseData-kielisestä esityksestä generoidussa kohdekielissä esityksessä sävellystä voidaan tarkemmin muokata MuseDatan puutteiden osalta. (Hewlett 1997, 402–403.)

MuseDatan kuvauskielisessä esityksessä musiikilliset käsitteet ja muut tiedot esitetään kukin omalla rivillään aikajärjestyksessä. Yksi rivi kuvaa yhtä *tietotyyppiä* (engl. *record*), joita on erilaisia eri käsitteitä varten. Esimerkiksi nuottitiedolle ja tahtimerkeille on olemassa omat tietotyyppinsä.

Yhdellä rivillä esitettävien musiikillisia käsitteitä vastaavien koodien merkitys määräytyy sen mukaan, millä sarakkeella ne sijaitsevat. Joillakin sarakkeilla kuvataan esimerkiksi vierekkäisten rivien erityistä suhdetta toisiinsa, eli MuseData kykenee koodaamaan myös musiikillisten käsitteiden välisiä suhteita itse kuvauskielessä.

Sarake MuseDatan tapauksessa tarkoittaa yhtä rivin merkkiä rivin alusta lukien, eikä kuten muissa tähän mennessä esitellyissä kuvauskielissä yhtä esimerkiksi välilyönnein eroteltua merkkijonoa. (Hewlett 1997, 410.) Tällainen esitysmuoto on luonnollisesti varsin altis virheille ja erittäin vaikea tulkita sellaisenaan. MuseData-kuvauskielinen musiikkinäyte on kuvassa 6.2 (s. 42). Esimerkissä näkyy mm. rakenteita kuvaavat aukeavat ja sulkeutuvat erilaiset sulkumerkit. Näillä osoitetaan tiettyjen rakenteiden alku- ja loppukohtaa.

6.3 Yhteenveto

Analyysiohjelmistojen kuvauskielet ovat jo varsin kunnianhimoisia sen suhteen, kuinka tarkasti ne pyrkivät musiikillisia käsitteitä ja näiden suhteita kuvaamaan. Kuitenkin esimerkiksi tässä esiteltyjen kuvauskielten sisäinen esitysmuoto on varsin kömpelö. Se, että koodausten merkitys määräytyy rivi- tai sarakesijainnin perusteella, johtunee pitkälti kuvauskielten iästä. Näitäkin kieliä on pääasiassa suunniteltu kirjoitettavaksi käsin, ja tällaisessa muodossa kuvauskieli on pienessä tilassa kohtuullisesti hallittavissa.

Esitysmuodon virhealttius on kuitenkin ilmeinen. Lisäksi merkityksen antaminen esimerkiksi rivien järjestykselle johtaa mm. siihen, että kuvauskielessä kuvattavissa olevien musiikillisten rakenteiden määrittelyt joutuvat mukautumaan tähän järjestykseen. Esimerkiksi juuri MuseDatan koodiesimerkissä 6.2 (s. 42) sulkumerkeillä osoitetut rakenteet voivat rajata vain yhden joukon peräkkäisiä rivejä. Toisaalta viivastonotaatiossakin rakenteet määritellään juuri tähän tapaan.

```
04/16/93 E. Correia
WK#:581       MV#:3c
Breitkopf & Härtel, Vol. 13
Clarinet Quintet
Trio II
Clarinet in A
1 0
Group memberships: sound, score
sound: part 1 of 5
score: part 1 of 5
$  K:0   Q:6   T:3/4   X:-11   C:4
C5     3        e     d  [     (&0p
E5     3        e     d  ]
measure 1
G5     3        e     d  [
E5     3        e     d  ]
C6     6        q     d        )
G5     3        e     d  [     (
E5     3        e     d  ]
measure 2
D5     3        e     d  [
F5     3        e     d  ]
A5     6        q     d        )
F5     3        e     d  [     (
D5     3        e     d  ]
measure 3
C5     3        e     d  [
B4     3        e     d  =
E5     3        e     d  =
D5     3        e     d  =
G5     3        e     d  =
F5     3        e     d  ]     )
measure 4
D#5    6        q #   d        (
E5     6        q     d        )
C5     3        e     d  [     (
E5     3        e     d  ]
measure 5
G5     3        e     d  [
E5     3        e     d  ]
C6     6        q     d        )
G5     3        e     d  [     (
E5     3        e     d  ]
```

Kuva 6.2: Alkua MuseData-kuvauskielellä koodatusta katkelmasta erään Mozartin teoksen klarinettiosuudesta (Hewlett 1997, 434).

7 Yleiset kuvauskielet

Tässä luvussa esitellään kuvauskieliä, jotka ovat syntyneet ohjelmistoista riippumatta, ja jotka on suunniteltu mahdollisimman yleiskäyttöisiksi kuvauskieliksi. Tässä esiteltävät kuvauskielet on valittu lähinnä sen mukaan kuinka paljon niitä on kirjallisuudessa, artikkeleissa ja projekteissa käsitelty. Mukana on myös joitakin uudehkoja kieliä, joista ei vielä ole juuri alan julkaisuissa kirjoitettu.

7.1 Notation Interchange File Format

Notation Interchange File Format (NIFF) on standardi, joka suunniteltiin musiikin viivastonotaatioesityksen kuvaukseen ja erityisesti tiedonsiirtoon eri nuotinkirjoitus- ja nuotintunnistusohjelmistojen tarpeisiin. NIFF on ensimmäinen laajalti huomiota saanut hanke, jossa pyrittiin luomaan kattava yleinen standardi musiikin tallentamiseen.

NIFF:n kehitystyössä on alusta asti ollut mukana monia merkittäviä keskenään kilpailevia kaupallisia tahoja. Standardin kehitystyö alkoi vuonna 1994 kuuden suuren musiikkiohjelmiston (*Encore*, *Score*, *Finale*, *MidiScan*, *SightReader* ja *NoteScan*) valmistajan yhteisestä päätöksestä. Ohjelmistoja olivat jo pitkään vaivanneet tiedonsiirto-ongelmat yhteisen kelvollisen kuvauskielen puuttuessa. Varsinkin nuotintunnistusohjelmistojen markkinoille tulon jälkeen asian korjaamiseen päätettiin ryhtyä. (NIFF 6a.3 1998, 2–4.) Nuotintunnistusohjelmistot eivät yleensä sisällä omaa editoria, joten tiedonsiirron nuotinkirjoitusohjelmiston kanssa on toimittava mahdollisimman hyvin.

Myöhemmin kehitystyöhön osallistui myös muita ohjelmistovalmistajia sekä tutkijoita ja tutkimuslaitoksia. Standardin historiasta ja suunnitteluun osallistuneista on kerrottu tarkemmin NIFF:n WWW-sivulla (Belkin 1998) ja standardin määrityksessä (NIFF 6a.3 1998, 2–3).

Kaupallisista tarpeista johtuen NIFF suunniteltiin ensisijaisesti käytännölliseksi tiedonsiirtokieleksi, joka voitaisiin ottaa käyttöön mahdollisimman lyhyessä ajassa. Tavoitteena ei koskaan ollut kehittää NIFF:stä kaiken kattavaa yleistä musiikin kuvauskieltä. (Grande 1997, 492.)

7.1.1 Rakenteet

Musiikillinen tieto jaetaan NIFF:ssä *loogiseen* (engl. logical), *graafiseen* (engl. graphical) ja *esitystietoon* (engl. performance). Ainoastaan loogisen tiedon kuvaaminen on NIFF-esityksessä pakollista. Esitystieto on NIFF:ssä aina MIDI-tietoa. MIDI-viestejä voi esitystietona littää mihin tahansa muihin NIFF:n rakenteisiin. (Grande 1997, 493.)

NIFF-esityksessä musiikki kuvataan ensisijaisesti loogisena tietona. Tämä tieto koostuu yleisimmistä viivastonotaation käsitteistä kuten nuoteista ja artikulaatiomerkeistä, samaan tapaan kuin esimerkiksi nuotinkirjoitusohjelmistojen kuvauskielissä. NIFF:ssä näiden eri käsitteiden väliset riippuvuudet voidaan kuitenkin määritellä hyvin tarkasti (NIFF 6a.3 1998, 31).

Kuvatut musiikilliset käsitteet tallennetaan NIFF:ssä loogisia rakenteita vastaavina ryhminä siinä järjestyksessä missä ne on tarkoitettu tulostettavaksi (NIFF 6a.3 1998, 9–11). Sisäiseltä rakenteeltaan NIFF noudattaa Microsoftin *Resource Interchange File Format* -määrityksiä (RIFF). NIFF-kielinen esitys tallennetaan tämän mukaisesti binäärimuodossa. (NIFF 6a.3 1998, 6.)

Graafiseksi tiedoksi käsitetään musiikillisiin käsitteisiin sellaisenaan liittymättömät erilaiset graafiset yksityiskohdat, kuten viivastonotaatioesityksen symbolien tarkkaan asetteluun liittyvät seikat. Graafisena tietona voidaan määritellä myös omia erikoissymboleita ja muuta grafiikkaa tulostettavaksi sävellyksen visuaaliseen esitykseen.

Loogiset rakenteet perustuvat viivastonotaation käsitteisiin ja rakenteisiin, ja niistä puhuttaessa käytetään vastaavia viivastonotaation nimityksiä. Näiden lisäksi rakenteita nimitetään myös nuottijulkaisumetaforan mukaisesti, esimerkiksi ilmaisun "partituurin sivu" mukaan. Näin loogiset rakenteet ovat tiettyyn rajaan saakka

hierarkkisia, viivastonotaation hierarkkisuuden mukaan. Näin NIFF:ssä ylin rakenteiden taso on partituuri, joka jaetaan edelleen sivuihin. Sivut puolestaan sisältävät viivastoja ja viivastoryhmiä, ääniä jne. *Aika-annos* (engl. time-slice) määrittää eri viivastoilla ja eri äänissä sijaitsevien musiikillisten tapahtumien järjestyksen ajan suhteen. Yhtäaikaisille musiikillisille tapahtumille määritellään sama aika-annos. (Grande 1997, 494–495.)

Koska NIFF standardina määrittelee vain yleisimmät viivastonotaation käsitteet ja rakenteet, siitä on tehty helposti laajennettava. Käyttäjä tai ohjelmisto voi standardin mukaan määritellä kieleen omia rakenteita varsin vapaasti. Standardin mukaan ohjelmistojen tuleekin yksinkertaisesti jättää huomiotta rakenteet, joita ne eivät tunnista. Vääriä tai laittomia rakenteita NIFF-standardissa ei siis ole. (Grande 1997, 493.)

Kuten edellä jo mainittiin, NIFF:iä voidaan laajentaa myös graafisesti. Omia kirjasimia, symboleita ja grafiikkaa voidaan tallentaa mukaan NIFF-tiedostoon ja sijoittaa osaksi partituuria. Näin lähes mitä tahansa voidaan tulostaa partituuriin, vaikka loogisia rakenteita on vain perinteisen viivastonotaation mukaisesti. (NIFF 6a.3 1998, 24; Grande 1997, 501.)

7.1.2 Ohjelmistot

NIFF-standardin päätavoite olla käytännöllinen tiedonsiirtoformaatti lyhyessä ajassa näyttää jo osittain toteutuneen, sillä kieltä tukevia ohjelmistoja on jo muutamia. Useat merkittävät nuotinkirjoitus- ja nuotintunnistusohjelmistot tukevat jo NIFF:iä. Tämä ei ole yllättävää, sillä juuri näiden ohjelmistojen valmistajat aloittivat standardin kehittämisen. Toisaalta on olemassa myös erittäin suosittuja ja laajalle levinneitä nuotinkirjoitusohjelmistoja, jotka eivät toistaiseksi ole ottaneet NIFF:iä käyttöön. Luonnollisesti näiden valmistajat eivät myöskään osallistuneet standardin kehitystyöhön.

Tällä hetkellä jonkinasteisen NIFF-tuen lupaavat seuraavat ohjelmistot. Nuotinkirjoitusohjelmistot *Igor*, *Encore* (NIFF-tuki luvattu tulevaan versioon), *LIME*, *PROSCORE* ja *SmartScore*. Nuotintunnistusohjelmistot *MIDISCAN*, *PianoScan*

ja *SharpEye Music Reader*. NIFF:n konvertointiin sokeinkirjoitukselle on lisäksi olemassa ohjelmisto nimeltä *Concert-O-Braille*.

Osaltaan NIFF:n käyttöönottoa on vauhdittanut standardin mukana valmistunut ohjelmistokehityspaketti *NIFF Software developer's kit*, jonka kuka tahansa voi hakea ilmaiseksi NIFF:n WWW-sivulta (Belkin 1998).

7.2 Standard Music Description Language

Standard Music Description Language (SMDL) on SGML- ja HyTime-standardeihin pohjautuva yleinen musiikin kuvauskieli.

Standard Generalized Markup Language (SGML) on laitteisto-, ohjelmisto- ja järjestelmäriippumaton metakieli, jolla määritellään hierarkkisten rakenteisten dokumenttien kuvaamiseen tarkoitettuja kuvauskieliä. Nämä kielet määritellään SGML:n mukaisen rakennemäärittelyn eli *dokumenttityyppimäärittelyn* avulla. Tätä kutsutaan alkuperäisen termin mukaisesti lyhenteellä DTD (engl. document type definition). Jonkin DTD:n mukaisen dokumentin eri rakenneosia eli *elementtejä* koodataan eli *merkataan* elementtien *tunnisteilla* (engl. tag). DTD määrittelee dokumentin loogisen rakenteen määrittelemällä mm. sallitut elementit, niiden tunnisteiden nimet sekä elementtien keskinäiset, hierarkkiset suhteet. Esimerkiksi WWW-dokumenttien merkkaamiseen tehty *HyperText Markup Language* (HTML) on SGML:n sovellus, ja täten jokainen HTML-dokumentti noudattaa HTML:n määrittelevää DTD:tä. SGML on kansainvälinen ISO-standardi (ISO 8879:1986). (Salminen 1995, 10; ks. myös Goldfarb 1990.)

SGML:ää on toisinaan pidetty turhan raskaana yleiseen käyttöön, ja erityisesti verkkoympäristöön. Sitä varten onkin kehitetty yksinkertaisempi ja helpommin käyttöön otettava metakieli *Extensible Markup Language* (XML). XML perustuu SGML:ään, ja on oikeastaan yksinkertaistettu versio SGML:stä. XML:stä on esimerkiksi karsittu joitakin vaikeasti implementoitavissa olevia SGML:n ominaisuuksia. XML on SGML:n osajoukko, joten jokainen XML-dokumentti on myös SGML-dokumentti.

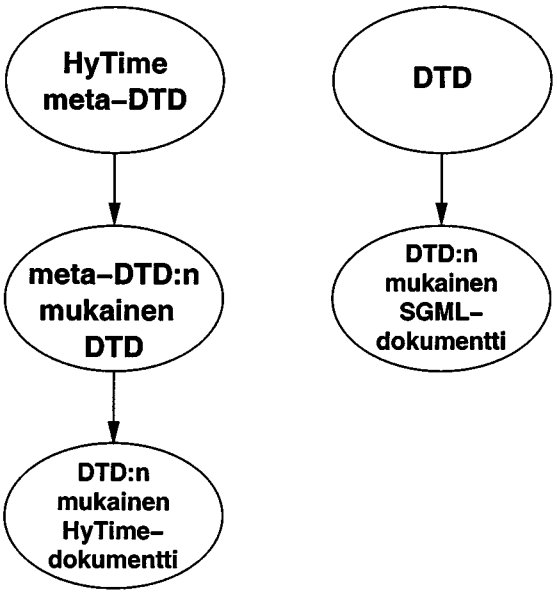
Hypermedia/Time-based Structuring Language (HyTime) on SGML:ään perustuva multimediainformaation hallintaan tarkoitettu standardi (ISO/IEC 10744). Sen avulla voidaan hallita aika- ja tilariippuvuuksia, luoda linkkejä ja viittaussuhteita mistä tahansa dokumenteista ja dokumenttien kohdista toisiin.

Vaikka HyTime on kirjoitettu SGML:llä, se ei määrittele vain yhtä tiettyä HyTime DTD:tä, jota kaikkien HyTime-dokumenttien tulisi noudattaa. Näin se eroaa esimerkiksi HTML-kielestä. Normaalin DTD:n mukaisten kiinteiden rakennemääritysten sijasta HyTime-standardissa määritellään näitä yleisempiä *arkkitehtuurimuotoja* (engl. architectural forms). Nämä määrittelevät sääntöjä elementtityyppien luontiin samaan tapaan kuin DTD:t määrittelevät sääntöjä dokumenttien luontiin. Arkkitehtuurimuotojen ansiosta HyTime-dokumenttien ja -sovellusten ei tarvitse käyttää yhtä tiettyä HyTime DTD:tä ja kiinteää joukkoa ennalta nimettyjä elementtejä hyödyntääkseen HyTimen ominaisuuksia. (Norrila 1995, 16.)

Arkkitehtuurimuotoja määrittelevää DTD:tä sanotaan *meta-DTD*:ksi (ISO/IEC 10744 1997, 217). Näin siis esimerkiksi HyTimen määrittelevä DTD on meta-DTD. HyTime meta-DTD määrittelee siis sääntöjä tavallisen DTD:n luontiin. Olioparadigmaan verraten voidaan ajatella, että tietty yksittäinen HyTimea hyödyntävä DTD perii ominaisuuksia HyTime meta-DTD:stä. Toisin sanoen jonkin yksittäisen dokumentin elementit perivät elementtien ominaisuuksia määrätyiltä arkkitehtuurimuodoilta. Kuvassa 7.1 (s. 48) vasemmalla on kuvattu, kuinka HyTime-dokumentti muodostuu HyTimen meta-DTD:n mukaisesta DTD:stä. Samassa kuvassa oikealla on kuvattu tavallisen SGML-dokumentin muodostuminen.

SMDL käyttää edelleen HyTimen arkkitehtuurimuotoja omien arkkitehtuurimuotojensa koostamiseen. SMDL määritellään siis HyTimen tapaan meta-DTD:llä, josta edelleen voidaan johtaa uusia DTD:itä (kuvan 7.2 vasen osa, s. 49). Näin esimerkiksi erityisesti oopperamusiikin kuvaukseen voitaisiin luoda oma erityinen SMDL:ää hyödyntävä DTD.

Kun tavallisten SGML-dokumenttien käsittelyyn riittää dokumentin rakenteen selvittävä *jäsennin* (engl. parser), niin HyTime-rakenteiden käsittelyyn tarvitaan vielä lisäksi *HyTime-kone* (engl. HyTime engine). Tämä huolehtii mm. monimutkaisten osoitusten hallinnasta. Samalla tavalla SMDL meta-DTD:stä johdettujen



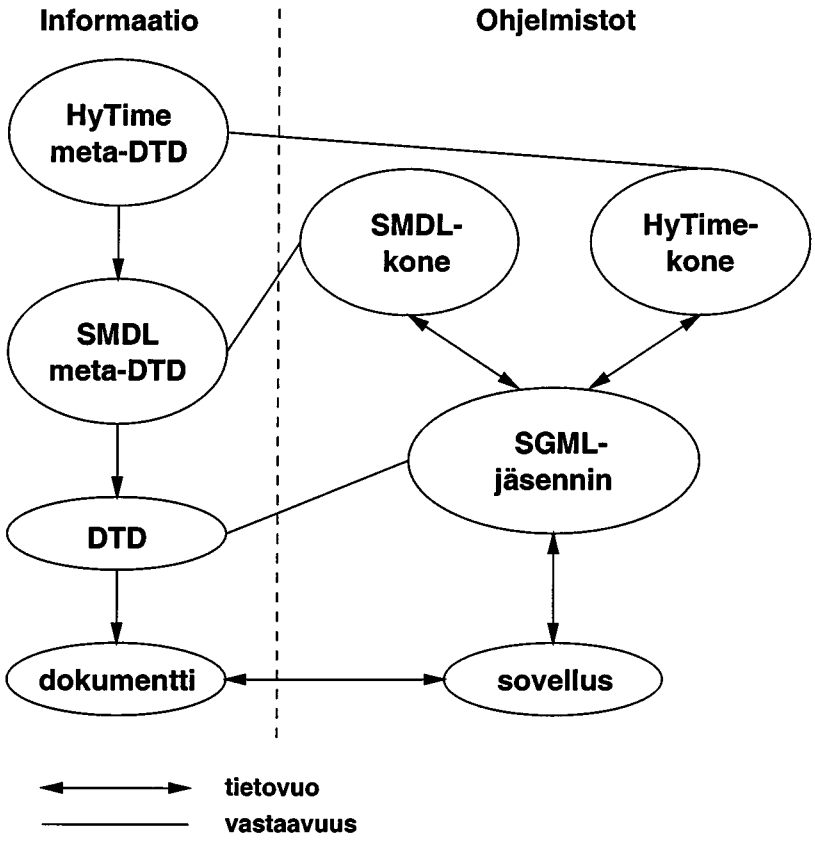
Kuva 7.1: Vasemmalla HyTime-dokumentin ja oikealla tavallisen SGML-dokumentin muodostuminen noudatetuista rakennemäärittelyistä.

dokumenttien hallintaan tarvitaan SGML-jäsentimen ja HyTime-koneen lisäksi vielä *SMDL-kone* (engl. SMDL engine). Kuvassa 7.2 (s. 49) on havainnollistettu näiden kaikkien SMDL-dokumenttien prosessointiin tarvittavien ohjelmistojen suhteita, ja näiden ohjelmistojen suhteita vastaaviin tietorakenteisiin.

On huomattava, että vaikka nykyään HyTime-koneita on jo implementoitu, on SMDL-kone vielä kaukaista tulevaisuutta. Tästä syystä SMDL:ää tutkitaan usein pelkästään suoraan HyTimesta johdetun yleisen SMDL DTD:n, eikä siis SMDL meta-DTD:n kautta. (Report on SMDL evaluation 1996, 28.) Jäljenpänä mainittu CANTATE-projektissa (kohta 7.2.3 sivulla 53) kehitetty SMDL DTD on juuri tällainen.

7.2.1 Yleiset periaatteet

Eräs johtoajatus SMDL:n suunnittelussa on ollut se, että mikä tahansa musiikillinen käsite tai ilmiö täytyy olla mahdollista kuvata SMDL:n avulla. HyTimen viittaustekniikka mahdollistaa standardin tavan esittää asioita, joita SMDL-



Kuva 7.2: HyTimen ja SMDL:n rakenteiden, sekä niiden käsittelyyn tarvittavien ohjelmistojen suhteet (Report on SMDL evaluation 1996, 27).

standardissa ei ole määritelty. Näin SMDL:n yhteydessä on mahdollista käyttää esim. tulevaisuuden musiikin kuvauskieliä, notaatiojärjestelmiä ja sellaisiakin tietoformaatteja, joista vielä emme tiedä mitään. (Sloan 1993, 52.)

SMDL-kielessä musiikillinen informaatio jaetaan neljään eri *alueeseen* (engl. domain). Nämä ovat (Sloan ja Newcomb 1997, 470)

- *looginen alue* (engl. logical domain)
- *esitystapa-alue* (engl. gestural domain)
- *visuaalinen alue* (engl. visual domain)
- *analyysialue* (engl. analytical domain).

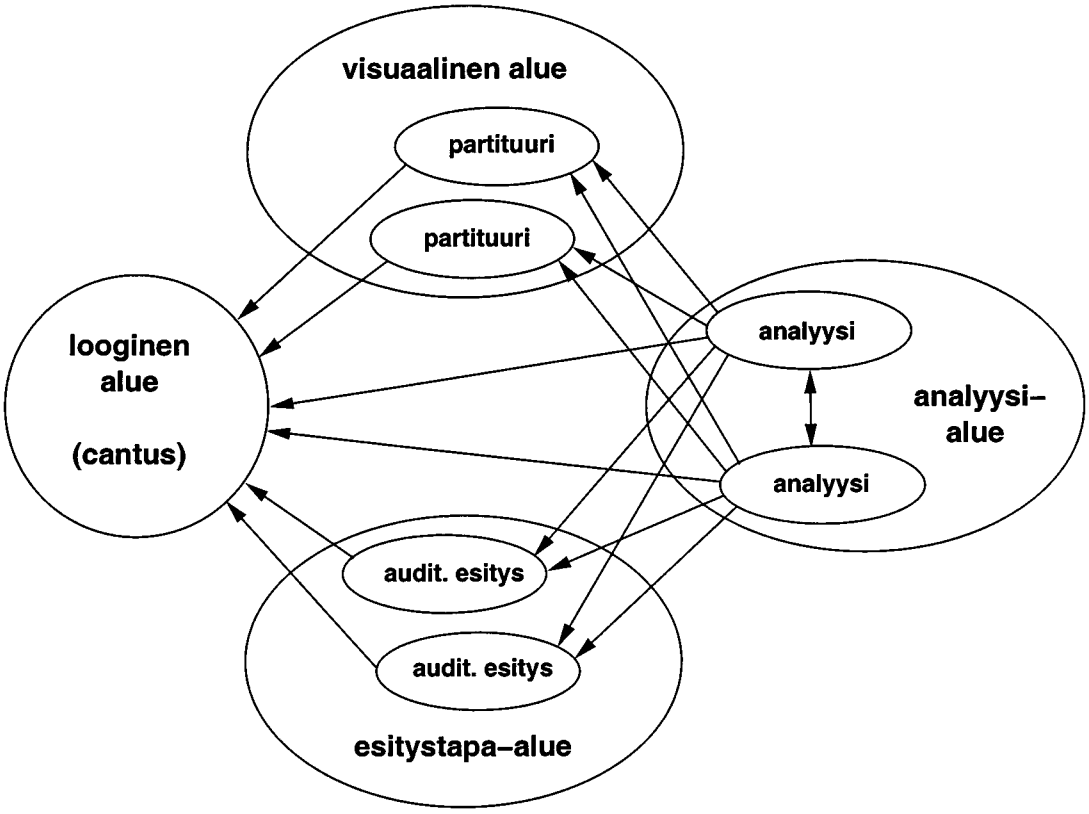
Loogisesta alueesta käytetään myös nimitystä *cantus*, joka tulee suoraan loogisen alueen tiedon arkkitehtuurimuodon nimestä (ISO/IEC DIS 10743 1995, 5, 12).

Itse musiikin kuvaus tapahtuu loogisessa alueessa. Se sisältää musiikin visuaalisesta ja auditiivisesta esityksestä riippumattoman loogisen kuvauksen. Näin ollen SMDL:n loogisessa aluessa koodattu musiikki ei ole sidoksissa mihinkään tiettyyn notaatiojärjestelmään, esimerkiksi viivastonotaatioon. Esimerkiksi sävel c voidaan loogisesti käsittää vain yhdellä tavalla, eli sillä on tietty soiva sävelkorkeus. Kuitenkin tämä sama sävel voidaan kuvata useilla eri notaatiojärjestelmillä, ja usein myös monilla eri tavoilla saman notaatiojärjestelmän puitteissa. (Sloan ja Newcomb 1997, 470.)

Cantuksessa kuvatun musiikin auditiivinen esitys voidaan sisällyttää esitystapa-alueeseen. Tämä esitys voi olla esimerkiksi elektronisessa muodossa oleva äänitiedosto. Esitystapa-alueeseen voidaan sisällyttää myös auditiivisen esityksen tuottamiseen tarvittavaa materiaalia, esimerkiksi cantuksessa kuvatun sävellyksen MIDI-kielinen esitys. Esityksiä voi olla myös useita. Esitysten eri osat voivat viitata vastaaviin cantuksessa määriteltyihin loogisiin rakenteisiin. (Sloan ja Newcomb 1997, 470.) Näin esimerkiksi yksittäinen sävelkorkeutta ilmaiseva MIDI-viesti voi osoittaa sitä vastaavaan sävelen määrittelyyn cantuksessa.

Samaan tapaan visuaaliseen alueeseen voidaan sisällyttää erilaisia kuvatun sävellyksen visuaalisia esityksiä. Esimerkiksi teosta kuvaava partituuri voidaan sisällyttää elektronisessa muodossa olevana kuvana tai jonkin nuotinkirjoitusohjelmiston kuvauskielenä. Kukin kuvan osa tai nuotinkirjoitusohjelmiston kuvauskielellä määritelty elementti voi viitata cantuksen sisältöön. Mainittakoon tässä, että NIFF-standardia on jo pitkään ja monella taholla suunniteltu käytettäväksi visuaalisen alueen virallisena nuotinkuvauskielenä (Grande ja Belkin 1997, 33; ISO/IEC DIS 10743 1995, 3).

Analyysialue puolestaan on tarkoitettu kuvattua sävellystä käsitteleville teosanalyyseille ja kommentaareille. Analyysialueen sisältö voi viitata minkä tahansa muun alueen sisältöön. Teosanalyysissä voidaan siis käsitellä cantuksen rakenteiden lisäksi myös esimerkiksi visuaalisen esityksen erityispiirteitä ja auditiivisen esityksen tiettyjä kohtia.



Kuva 7.3: SMDL:n alueet ja niiden väliset viittaussuhteet (Sloan ja Newcomb 1997, 471).

Looginen alue on ehdottomasti SMDL:n alueista tärkein. Standardiluonnoksessa kaikkien muiden alueiden käsittelyyn onkin käytetty yhteensä vain kolme sivua. Luonnoksessa muista alueista määritellään vain HyTimesta johdetut mekanismit, joilla kunkin alueen sisältämät tiedot voivat osoittaa cantuksen sisältämiin rakenteisiin ja elementteihin (ISO/IEC DIS 10743 1995, 38–40). Kuvassa 7.3 on havainnollistettu SMDL:n eri alueiden välisiä viittaussuhteita.

7.2.2 Cantuksen rakenteet

Cantuksen sisältämät musiikilliset käsitteet voidaan jakaa rinnakkaisiksi *säikeiksi* (engl. thread), jotka vastaavat viivastonotaation käsitettä ääni (Sloan ja Newcomb

1997, 473). Esimerkiksi polyfonisessa musiikissa kunkin soittimen soittamat tapahtumat on hyödyllistä kuvata soitinta vastaavissa säikeissä. Säikeitä voidaan käyttää myös muihin saman tyyppisiin asioihin, esimerkiksi erottamaan toisistaan eri lähteistä saadut saman sävellyksen keskenään ristiriitaiset kohdat koodattaessa SMDL-kielellä vanhoja löytyneitä sävellyksiä.

Sävelkorkeus voidaan määritellä hyvin monella eri tavalla, esimerkiksi kiinteänä taajuutena (esim. 440 Hz) tai asteikon sävelenä (esim. sävel d), jolloin sävelen soiva taajuus riippuu käytetystä asteikosta. Myös käytetyt asteikot voidaan määritellä vapaasti itse.

Asteikon sävelten todellinen soiva sävelkorkeus voidaan määritellä (ISO/IEC DIS 10743 1995, 25–26; Newcomb 1991, 78)

- *pitch gamut* -taulukon avulla, jonka mukaan jokainen oktaavi jaetaan osiin (kuten esimerkiksi tasavireisessä viritysjärjestelmässä)
- suhteessa johonkin taajuuteen tai muuhun säveleen (kuten esimerkiksi luonnonpuhtaassa viritysjärjestelmässä)
- käyttäjän omana funktiona.

Kiinnitetyn asteikon suhteen voidaan määritellä uusia sävellajeja määrittelemällä sävelten kromaattiset siirtymät *ficta gamut* -taulukossa (Sloan ja Newcomb 1997, 481). Näin SMDL:ssä esimerkiksi enharmoniset sävelet voidaan koodata eksplisiittisesti.

Musiikillisen tapahtuman kesto voidaan määrätä tosiajassa (esim. sekunteina tai millisekunteina) tai *virtuaaliajassa* (*virtual time*). Tempo (*baton*-arkkitehtuurimuoto) määrää virtuaaliajan aikayksiköiden suhteen tosiaikaan. (Sloan ja Newcomb 1997, 472; ISO/IEC DIS 10743 1995, 22.) Virtuaaliaika käsitteenä vastaa siis esimerkiksi viivastonotaation aika-arvo -konseptia.

Yleinen, vapaasti määriteltävä väline musiikillisen tapahtuman tai tapahtumaryhmän modifiointiin on arkkitehtuurimuoto *wand* (Sloan ja Newcomb 1997, 473). Tällä voidaan määrätä musiikillisiin käsitteisiin vaikuttava yksittäinen tai jatkuva muutos ajan suhteen esim. jonkin käyttäjän määrittelemän funktion mukaan. Yksittäinen muutos on esimerkiksi viulun kielille asetettavan sordiinon asettaminen

ja poistaminen. Jatkuva muutos on esimerkiksi viulua jousella soitettaessa jousen ja kielen kosketuskohdan siirtäminen hitaasti tallalta kaulalle soiton aikana, tai esimerkiksi jonkin äänenmuokkauslaitteen säätäminen.

Erilaisia sävelten soittamisen erityispainotuksia ilmaisevia artikulaatio- ja ornamentointimerkintöjä voidaan SMDL:ssä määritellä vapaasti joko yksittäin tai taulukon kautta. Dynamikka voidaan määritellä samaan tapaan, tai esimerkiksi wand-arkkitehtuurimuodon avulla. (Sloan ja Newcomb 1997, 473.)

7.2.3 Kehitys ja nykytila

SMDL:n suunnittelu alkoi vuonna 1985 American National Standards Instituten (ANSI) alaisessa Music in Information Processing Standards (MIPS) -työryhmässä.

Mielenkiintoista on se, että HyTime syntyi alunperin SMDL:n kehitystyön sivutuotteena. SMDL:ään suunnitellulla yleisellä viittaustekniikalla (arkkitehtuurimuodot) huomattiin olevan sovelluskohteita muuallakin, ja vuonna 1989 se eriytettiin HyTime-kieleksi. (Newcomb, Kipp ja Newcomb, 1991, 82.) HyTime standardoitiin vuonna 1992.

SMDL:n viimeisin standardiluonnos (ISO/IEC DIS 10743 1995) julkaistiin vuonna 1995. Samoihin aikoihin moni projekti ympäri maailmaa – mm. Thesaurus Musicarum Italicarum (Thesaurus musicarum italicarum 1998) ja The Music Library of the Future (Pennycook 1997) – kiinnostui SMDL:n mahdollisuuksista, mutta vain harva lopulta alkoi työskennellä sen kanssa standardin monimutkaisuuden ja luonnosvaiheessa olemisen takia.

Intensiivisimmin SMDL:ää on tutkittu Computer Access to Notation and Text in Music Libraries -projektissa (CANTATE) vuosina 1995 – 1996. CANTATE kuului EU:n Telematics for Libraries -hankkeeseen. Projektin tavoitteena oli tutkia, kuinka sävellyksiä voitaisiin siirtää verkkoympäristössä esim. julkaisijoilta kirjastoille.

Projektissa arvioitiin SMDL:ää ensin vuoden 1992 alustavan luonnoksen perus-

teella, kunnes vuoden 1995 stardardiluonnos ilmestyi. Evaluoinnin tulokset julkaistiin raportissa Report on SMDL evaluation (1996). Loppuraportissaan (Final report 1996) projekti päätyi suosittelemaan SMDL-kielen käyttöä ja jatkotutkimusta, vaikka SMDL todettiinkin vielä kehittyväksi, eikä mitään sitä käyttäviä sovellusohjelmistoja ollut lähitulevaisuudessa odotettavissa.

Steve Mounce kehitti CANTATE-projektissa tiettävästi ensimmäisen käyttökelpoisen SMDL DTD:n. Yksinkertaisuuden vuoksi tämä DTD ei määrittele SMDL:n standardiluonnoksen mukaisia arkkitehtuurimuotoja, vaan se on tavallinen SGML:n mukainen DTD. Mouncen DTD:hen perustuen CANTATE:ssa ohjelmoitiin käytännön kokeiluja varten yksinkertainen konversio-ohjelma, jolla NIFF-dokumentista tuotettiin SMDL-dokumentteja (NIFF to SMDL translator 1996). Ohjelman tuottamia SMDL-dokumentteja jouduttiin poikkeuksetta vielä korjaamaan käsin.

Keväällä 1999 käynnistettiin Glasgow'n yliopistossa Music Tagging Type Definition -projekti (MuTaTeD!), jonka tavoitteina oli mm. kehittää CANTATE-projektin SMDL DTD:tä edelleen meta-DTD:n suuntaan, ja integroida SMDL- ja NIFF-kuvauskielien käyttöä siten, että itse musiikin kuvauksessa käytettäisiin SMDL:n loogista aluetta ja visuaalisessa esityksessä NIFF:iä (Böhm 1999; Böhm 2000).

Projektin päätyttyä tammikuussa 2000 työ jatkuu MuTaTeD! II -projektin yhteydessä. Projektin tuloksena syntyi uusi konversio-ohjelma SMDL:n ja NIFF:n välille. Tämä on CANTATE-projektin vastaavaa konversio-ohjelmaa huomattavasti edistyneempi. (Böhm 2000.)

Vuonna 1996 ISO hyväksyi SMDL:n standardiksi (ISO/IEC 10743), mutta lopullista standardimääritystä ei vieläkään ole julkaistu. Julkaisu viivästyi aluksi pääasiassa siksi, että tekijät joutuivat odottamaan uuden HyTime-määrityksen (ISO/IEC 10744 1997) julkaisua.

Muun muassa Gerd Castanin kanssa käytyjen sähköpostikeskustelujen (11.1.2000) perusteella uusi, piakkoin julkaistava SMDL tulee olemaan XML-yhteensopiva, ja se hyödyntää uutta HyTimen XML-yhteensopivaa versiota. Tämä todennäköisesti vauhdittaisi standardin käyttöönottoa huomattavasti, onhan

XML noussut hyvin suosituksi metakieleksi varsin lyhyessä ajassa. Castanin tiedot perustuivat hänen mm. SMDL-standardin nykyisen toimittajan, Eliot Kimberin kanssa käymiinsä keskusteluihin.

Käyttökelpoisia SMDL-ohjelmistoja ei MuTaTeD! -projektin konversio-ohjelman lisäksi ole vielä tarjolla. Tämä johtuu pääosin standardin keskeneräisyydestä, osaksi myös ohjelmoinnin vaativuudesta. Standardin hitaasta kehityksestä huolimatta SMDL kuitenkin edelleen kiinnostaa tutkijoita. Muun muassa MuTaTeD! -projekti jatkoprojekteineen on tästä tuoreena osoituksena.

SMDL perustuu arvostettuun kansainväliseen järjestelmä-, laitteisto- ja ohjelmistoriippumattomaan SGML-standardiin, ja on näiltä osin hyvin vahva standardi. SGML:n eduista on kirjoitettu paljon elektronisen ja rakenteisen tekstin tutkimuksen yhteydessä (Salminen 1995, 12). Monissa projekteissa SMDL:n eräänä suurimmista eduista onkin pidetty juuri sen vahvaa SGML-taustaa, ja tietysti myös asemaa ISO-standardina (Böhm 1999, 4).

Musiikin kuvauksen laajuudessa SMDL on selvästi ylivoimainen muihin kuvauskieliin verrattuna. Tulevaisuudessa SMDL voikin olla musiikille sitä, mitä SGML ja XML nyt ovat tekstidokumenteille. Ainoa ongelma tällä tiellä näyttää olevan se, että ohjelmistovalmistajat eivät ole vielä innostuneet panostamaan SMDL:n kehitykseen tai käyttöönottoon. MuTaTeD! -projektin loppuraportissa (Böhm 1999, 4) tilannetta verrattiin siihen, että meillä olisi käytössä HTML:n tapainen mullistava standardi, mutta ei yhtäkään selainta joka sitä osaisi käyttää.

7.3 Music Markup Language

Music Markup Language (MML) on vielä kehitysvaiheessa oleva XML:ään perustuva kuvauskieli. Kieltä kehitetään professori Jacques Steynin johdolla Pretorian yliopistossa Etelä-Afrikassa.

Tavoitteina on mainittu, että MML-kielistä musiikkidokumentti voitaisiin helposti muuttaa muuhun muotoon, esimerkiksi erilaisiksi ASCII-tekstinä esitettäviksi esityksiksi, viivastonotaatioesitykseksi ja MIDI-muotoon. Sen tulisi olla käyttö-

kelpoinen myös verkkoympäristössä, esimerkiksi SMIL-dokumenttien yhteydessä. MML:n elementtien nimeämiseen on kiinnitetty erityistä huomiota, jotta kieli pysyisi sellaisenaan kohtuullisen luettavana ja käsitettävänä esimerkiksi HTML:n tapaan.

MML koostuu erilaisista moduleista, joita musiikkidokumenteissa voi ottaa käyttöön tarpeiden mukaan. Kukin moduli tuo musiikin kuvaukseen uusia näkökulmia ja rakenteita. Esimerkiksi aikamäärityksiin (äänen kesto), taajuusmäärityksiin (äänen korkeus), tekstiin (sävellyksen sanoitukset), MIDI:n käyttöön ja nuottikirjoitukseen liittyvät kielen rakenteet ovat omina moduleinaan. Näistä vain aika- ja taajuusmääritysmodulien käyttö on kuvauksessa pakollista. Näiden pakollisten modulien rakenteissa MML keskittyy musiikillisten käsitteiden ja rakenteiden kuvaamiseen viivastonotaation käsitteiden pohjalta. Esimerkiksi nuottiarvot kirjoitetaan aina tahtia kuvaaviin ryhmiin.

Pretorian yliopiston opiskelijaprojekteissa suunnitteilla joitakin MML:ään liittyviä sovelluksia. Ainakin jonkinlainen konversio-ohjelma MIDI:n ja MML:n välille on jo olemassa. MML:llä koodattua musiikkia kaksi- ja kolmiulotteisesti visualisoiva sovellus on kehitteillä. Suunnitteilla on myös WWW-sivulle upotettava viivastonotaatiota tulostava Java-sovelma, sekä jonkinlainen mahdollisesti MIDI:ä hyödyntävä rajapinta musiikkilaitteiden ja MML:n välille.

MML tuntuukin olevan painottunut eniten käytännön musiikintekoon mm. MIDI:n avulla, sekä verkkojulkaisemiseen liittyviin asioihin. Standardin johdannossa MML:n sanotaan syntyneen siksi, koska SMDL on liian vaikeatajuinen yleiseen käyttöön ja muut modernit kuvauskielet taas ovat liian suppeita. Erityisesti MML:n käyttökohteena nähdään toimiminen käyttöliittymän ja MIDI:n välillä. (Steyn 1999.)

MML:n etu muihin melko kattaviin viivastonotaation käsitteille perustuviin kuvauskieliin nähden on ainakin sen perustuminen XML-standardille. Jacques Steynin kanssa käydyn sähköpostikeskustelun (16.8.1999) perusteella MML:lle yritetään tulevaisuudessa saada World Wide Web Consortiumin (W3C) suositus.

7.4 GUIDO Notation Format

GUIDO Notation Format (GUIDO) on varsin uusi viivastonotaation käsitteisiin perustuva kuvauskieli. GUIDOn käyttämä kieli on suunniteltu yksinkertaiseksi lukea, ASCII-muodossa tallennetavaksi sekä järjestelmäriippumattomaksi.

Vaikka GUIDO onkin vahvasti sidoksissa viivastonotaatioon, keskittyy se kuitenkin itse musiikillisten käsitteiden kuvaamiseen, eikä pelkästään graafiseen nuottikirjoituksen latomiseen.

GUIDOn johtoajatus on se, että yksinkertaiset musiikilliset käsitteet ja rakenteet on pystyttävä kuvaamaan mahdollisimman yksinkertaisesti, jolloin kompleksisempia kuvaustapoja tarvitaan vain kompleksisten käsitteiden kuvaamisessa. (Hoos et al. 1998.)

GUIDOa voidaan käyttää kolmella tasolla kuvattavan tapahtuman kompleksisuuden mukaan. *Basic GUIDO* käsittää yksinkertaisen perusnuotinkirjoituksen. *Advanced GUIDO* sisältää mm. mahdollisuuden vaikuttaa symbolien sijoitteluun partituurissa nuottijulkaisujen tuottamista ajatellen. *Extended GUIDO* laajentaa symbolivalikoimaa perinteisen nuottikirjoituksen ulkopuolelle ja sisältää mm. mahdollisuuden määritellä kieleen omia rakenteita.

GUIDOa on tekijöiden mukaan helppo ja nopea käsitellä, jolloin myös erillaisia sovelluksia on helppo tehdä. Joitakin ohjelmia onkin jo valmistunut. *GUIDO NoteServer* generoi GUIDO-kielisen kuvauksen perusteella esimerkiksi JPEG-, GIF- tai PostScript-formaatissa olevan partituurin. Näin NoteServeriä voidaan käyttää esimerkiksi nuottijulkaisujen tekemiseen tai WWW-julkaisuun. (Renz ja Hoos 1998.)

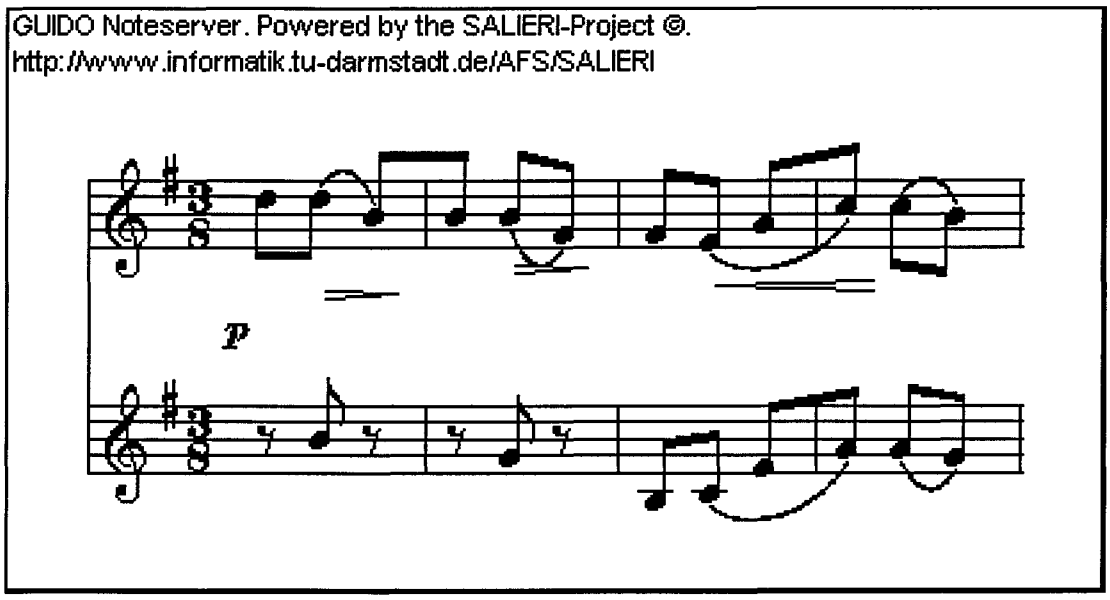
GUIDOn kotisivulla on kokeiltavana NoteServer, jolle voi syöttää GUIDO-kieltä lomakkeen välityksellä. Symbolien asettelu ei mutkikkaampia esimerkkejä kokeiltaessa aina oikein NoteServeriltä onnistunut, mutta ohjelmasta onkin julkaistu vasta versio 0.22. Kuvassa 7.4 (s. 59) oleva viivastonotaatioesitys on tuotettu NoteServerillä.

GUIDOn tulostamiseen ja editointiin on olemassa *GUIDO NoteViewer*-ohjelma

Windows-ympäristöön. Myös konversio-ohjelma GUIDOn ja MIDI-formaatin välille on tehty. *GUIDO-jäsennin* C-kielisenä lähdekoodina on lähiaikoina saatavilla GUIDOn kotisivulta (GUIDO Music Notation Format Page [online] 1999).

GUIDO-määrityksistä on tällä hetkellä (20.12.2000) julkaistu vasta Basic GUIDO (Hoos ja Hamel 1997), joten GUIDOn mahdollisuuksia yleisenä ja laajempana musiikin kuvauskielenä on vielä vaikea arvioida.

```
{ [\title<"No.3"> \tempo<"Andantino">
  \staff<1> \clef<"g"> \key<+1> \meter<"3/8">
  \i<"p"> d2/8 |
 \sl(\dim(d h1)) h
 \sl(\dim(h g)) g
 \sl(\cresc(f# a c2))
 \sl(c h1) ],
[ \staff<2> \clef<"g">  \key<+1> \meter<"3/8">
  _/8 |
  h1 _ _
  g _ h0
  \sl(c1 f# a)
  \sl(a g)]
}
```



Kuva 7.4: GUIDO-kielellä koodattua musiikkia, sekä sama viivastonotaatioesityksenä.

8 Johtopäätökset

Järjestelmä-, laitteisto- ja ohjelmistoriippumattomina SGML- ja XML-standardeihin perustuvat kuvauskielet ovat kilpailijoihinsa nähden vahvoilla. SGML ja XML ovat molemmat laajalle levinneitä standardeja. Niitä on tutkittu ja edelleen kehitetty valtavasti, niitä tukevia ja käsitteleviä ohjelmistoja kehitetään aktiivisesti ja niiden oheisstandardeihin panostetaan paljon. Näiltä osin siis ainakin SMDL ja MML ovat erittäin tukevalla pohjalla.

Tämän hetken musiikin kuvauskielistä SMDL on selkeästi laajin. Se näkyy muun muassa SMDL:n standardiluonnokseen kirjoitetussa tavoitteessa kyetä koodaamaan mitä tahansa musiikkia. Lisäksi SMDL pyrkii koodaamaan musiikkia musiikin itsensä ehdoilla muodostettuja rakenteita käyttäen, eikä niinkään suoraan viivastonotaatiosta otetuilla käsitteillä kuten lähes kaikki muut kuvauskielet. Tästä on sekä etua että haittaa. Viivastonotaatio on käytetyin ja monille musiikin kanssa työskenteleville tutuin musiikillinen koodi. Näin ollen viivastonotaation käsitteistö olisi SMDL:n käyttäjällekin tuttu vertailukohta, johon jo sinänsä mutkikasta kuvauskieltä voisi suhteuttaa. Toisaalta irrottautuminen viivastonotaatiosta jo käsitteiden ja rakenteiden tasolla mahdollistaa monien täysin uusien asioiden ja rakenteiden kuvaamisen. Tämä on varmasti merkittävä seikka etenkin tulevaisuudessa, sillä onhan viivastonotaatiota jatkuvasti ja yhä enenevässä määrin kritisoitu riittämättömäksi ja epätarkaksi. Näin SMDL voisi ehkä vielä joskus tarjota tukevapohjaisen ja kattavan vaihtoehdon viivastonotaation käytölle musiikin määrittelyssä.

SMDL on siis kaikin puolin hyvin vahva standardi. Mutta valitettavasti vain teoriassa, sillä mitään käyttökelpoisia SMDL-ohjelmistoja ei ole vielä olemassa. Lisäksi kielen kehitys on muutenkin ollut varsin hidasta viime aikoina. Nämä ongelmat johtuvat pääasiassa siitä, että SMDL on niin laaja ja monimutkainen, ja sitä kautta vaikeasti implementoitavissa. SMDL:n käyttöönottoon liittyvä valtava

työ sekä muut täysin uuden standardin käyttöönottoon liittyvät riskit ovat pitäneet ohjelmistovalmistajat toistaiseksi tehokkaasti loitolla SMDL:stä.

Tietoa hukkaamattomaksi tiedonsiirtoformaatiksi suunniteltu NIFF kuvaa vain viivastonotaatioesityksiä, mutta tallentaa myös eri käsitteiden ja symbolien keskinäiset riippuvuudet varsin tehokkaasti. Näin ilmaisuvoimaisempana kuvauskielenä NIFF tulee todennäköisesti syrjäyttämään monia yksittäisten nuotinkirjoitusohjelmistojen omia ohjelmistosidonnaisia kuvauskieliä. Tulevaisuudessa NIFF voisikin olla yleinen standardi nimenomaan nuottijulkaisujen valmistamiseen ja käsittelyyn tarkoitetuissa ohjelmistoissa. Monia NIFF:iä tukevia nuotinkirjoitus- ja nuotintunnistusohjelmia onkin jo nyt olemassa. NIFF:n binäärimuotoinen tallennustapa on standardin heikko lenkki. Ainakin pitkäaikaissäilytyksen kannalta binäärimuoto on hieman arveluttava.

Viivastonotaatiosidonnaisuutensa takia NIFF ei kuitenkaan koskaan tule olemaan yksinään riittävä yleinen musiikin kuvauskieli. Näin ollen esityisesti CANTATE- ja MuTaTeD! -projekteissa suunniteltu SMDL:n ja NIFF:n yhteiskäyttö vaikuttaa lupaavalta ja kiinnostavalta. SMDL kykenee kuvaamaan sävellyksen loogisen sisällön tarkasti ja riippumattomasti esimerkiksi viivastonotaatioesityksestä. NIFF taas kuvaa standardilla tavalla sävellystä vastaavan visuaalisen esityksen. Näyttää siis siltä, että NIFF jo jonkin verran yleisempää suosiota saaneena standardina ja SMDL kattavana, hierarkkisena ja SGML-taustaisena ISO-standardina tulevat tulevaisuudessa muodostamaan vahvan liiton, mikäli näiden standardien edes nykyisen kaltainen suosio eri tutkimusprojekteissa jatkuu. Liitolle tarvittaisiin hyvin pian myös kaupallisten tahojen tuki.

Uudet, yleiskäyttöisiksi suunnitellut ja vielä hieman keskeneräiset kuvauskielet MML ja GUIDO vaikuttavat nekin varsin kiinnostavilta. Mutta tarkemmin tarkasteltuna ne eivät kuitenkaan ole todella yleiskäyttöisiä esimerkiksi samalla tavalla kuin SMDL. MML ja erityisesti GUIDO ovat molemmat edelleen varsin tiukasti kiinni viivastonotaatiossa, eivätkä näin ollen kilpaile SMDL:n kanssa kaikenkattavan kuvauskielen tittelistä. MML:n selkeä hyvä puoli on sen perustuminen voimakkaasti tuetulle ja edelleen kehittyvälle XML-standardille. GUIDOn kuvauskielen esitysmuoto taas vaikuttaa valitettavasti jo hieman vanhanaikaiselta, var-

sinkin kun nykyään on tarjolla saman asian ajava XML:n kaltainen hierarkkinen metakieli.

Tässä työssä esiteltyjen analyysiohjelmistojen kuvauskielet, Humdrum ja MuseData ovat molemmat musiikin kuvaukselta varsin kattavia, mutta sisäiseltä esitysmuodoltaan valitettavasti kovin vanhanaikaisia, kryptisiä ja haavoittuvaisia. Samaan tapaan myös äänisynteesiohjelmistojen kuvauskielet ovat ongelmallisia. Omalla sovellusalueellaan ne kuitenkin toimivat hyvin ja kykenevät esittämään musiikkikappaleen täydellisesti, mutta niiden käyttämä signaalitasolle menevä kuvaustapa on aivan liian kankea soitinmusiikin kuvaamiseen.

Vanhojen nuotinkirjoitusohjelmistojen kielet taas merkitsevät koodeillaan suoraan visuaalisia symboleja, jolloin ongelmana on jälleen mm. rajoittuneisuus viivastonotaatioon ja kielen koodeilla esitettävissä olevaan ja ohjelmiston implementoivaan symbolivalikoimaan. NIFF ajan myötä syrjäyttäneekin hämmästyttävän kauan käytössä pysyneet vanhemmat ohjelmistoriippuvaiset nuotinkirjoitusohjelmistojen kuvauskielet. Myös rajoittuneen ja suppean MIDIn käyttö nuotinkirjoitusohjelmistojen välisessä tiedonsiirrossa tulee varmasti vähenemään erikoisesti tiedonsiirtokäyttöön suunnitellun NIFF:n yleistymisen ansiosta.

Monien vanhojen kuvauskielten lukemattomiin erilaisiin esitysmuotoihin liittyvien ongelmien perusteella voidaankin suositella jonkin yleisemmän standardin, kuten SGML tai XML käyttöä musiikin kuvauskielen pohjalla. Näin saadaan kuvauskieltä tukemaan heti valmiiksi suuri joukko erilaisia ohjelmistoja. Omien esitysmuotoratkaisujen eräs suurimmista ongelmista onkin juuri se, että esimerkiksi ohjelmistovalmistajat ovat varsin haluttomia opiskelemaan ja implementoimaan alati uusia ratkaisuja.

9 Yhteenveto

Tämän tutkimuksen alussa luvussa 2 tarkasteltiin musiikin kuvauksen nykytilannetta, ongelmia ja mahdollisuuksia. Tarkastelussa todettiin muun muassa se, että musiikin koodaaminen on ongelmallinen asia jo itsessään, jo ennen kuin yritämme koodata musiikkia tietokoneella käsiteltävään muotoon. Myöhemmin musiikin kuvauskielistä puhuttaessa todettiin, että nykytilanteen suuri ongelma on erilaisten ohjelmistoriippuvaisten kuvauskielten valtava määrä. Tämä taas on aiheuttanut ongelmia esimerkiksi tiedonsiirrossa ja arkistoinnissa. Mitään yleiskäyttöistä musiikin kuvauskieltä ei edelleenkään ole laajassa käytössä. Tarkastelussa nähtiin, että sellainen kuitenkin auttaisi nykyisissä ongelmissa sekä avaisi uusia mahdollisuuksia musiikin käsittelyssä tietokoneympäristöissä, esimerkiksi ohjelmistoriippumattomuuden ja materiaalin monikäyttöisyyden muodossa. Edelleen nähtiin, että nämä uudet mahdollisuudet ovat pitkälti samoja, mitä elektronisen tekstin tutkimuksessa tekstin rakenteistamisella haetaan.

Seuraavissa luvuissa siirryttiin tarkastelemaan yleisimpiä nykyisin käytössä olevia kuvauskieliä sovellusalueittain. Näitä sovellusaluekohtaisia kuvauskieliä tarkasteltaessa kävi ilmi, että kaikki käsitellyt yksittäiset kielet olivat sopimattomia yleiskäyttöiseksi musiikin kuvauskieleksi. Näin ollen mikään tarkastelluista kielistä ei olisi sopiva esimerkiksi korvaamaan jotain toista, eri sovellusalueen kuvauskieltä. Esimerkiksi nuotinkirjoitusohjelmistojen kielet laiminlöivät tai olivat kokonaan käsittelemättä joitakin, mahdollisesti toisella sovellusalueella tarvittavia tietoja, jotka eivät suoraan olleet viivastonotaatioesityksessä näkyviä asioita. Usein myös jo kielen rakenne sinällään oli yleisempää käyttöä rajoittava tekijä, kuten Csoundin ja erityisesti MIDIn tapauksessa. Myös MuseData ja Humdrum olivat ongelmallisia rakenteen ja sisäisen esitysmuodon alueella.

Sovellusaluekohtaisten kuvauskielten jälkeen luvussa 7 tarkasteltiin muutamia kiinnostavimpia, erityisesti yleiskäyttöisiksi suunniteltuja kuvauskieliä. Kävi ilmi, että näistäkin NIFF, GUIDO ja pitkälti myös MML olivat rajoittuneet viivastono-

taation käsitteistöön. Jo tästäkin syystä näitä kolmea ei voinut pitää todella yleiskäyttöisinä. Ainoastaan SMDL oli viivastonotaatiosta riippumaton. Muutoinkin SMDL osoittautui muita luvussa käsiteltyjä kieliä laajemmaksi ja kattavammaksi.

Kaikkien tässä työssä käsiteltyjen kuvauskielien etuja, ongelmia sekä tulevaisuuden näkymiä tarkasteltiin vielä erikseen luvussa 8. Musiikin kuvauksen kattavuuden, joustavuuden ja laajennettavuuden puolesta SMDL oli kaikista tässä työssä käsitellyistä kuvauskielistä ainoa yleiskäyttöiseksi sopiva. SMDL ei kuitenkaan ei ole vielä lainkaan käyttökelpoinen, sillä mitään siihen liittyviä julkisia sovelluksia ei ole vielä olemassa. Tähän on syynä erityisesti kielen monimutkaisuus, joka johtaa myös sovellusten kehittämisen vaikeuteen. Näyttää siis siltä, että kuvauskielen kattavuus ja yleiskäyttöiseksi sopivuus johtaa aina samalla myös kielen monimutkaisuuteen ja edelleen implementaatio-ongelmiin. Näin ollen käytännöllinen ja todella laaja-alainen yleinen musiikin kuvauskieli lienee utopiaa – tai ainakin ainoastaan teoriaa – vielä kauan aikaa jatkossakin.

LÄHTEET

ATK-sanakirja. Suomen Atk-kustannus Oy, Espoo, 9. painos, 1997.

Bainbridge D., Csound. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 111–142.

Belkin A., Notation Interchange File Format (NIFF) [online], 1998. Saatavilla WWW-muodossa: <URL: http://mistral.ere.umontreal.ca/~belkina/NIFF.doc.html>, [viitattu 12.9.1999].

Böhm C., MuTaTeD! - Music Tagging Type Definition [online], 1999. Saatavilla WWW-muodossa: <URL: http://www.pads.ahds.ac.uk/musicMutatedAbstract.html>, [viitattu 25.4.1999].

Böhm C., Report of MuTaTed! [online], luku Systems for Music Representation and Retrieval, 2000. Saatavilla Word-muodossa: <URL: http://www.pads.ahds.ac.uk/mutatedProjectReportChapter2.doc>, [viitattu 17.12.2000].

Blostein D., Haken L., Justification of printed music. Communications of the ACM, Vol 34, No. 3, 1991, 88–99.

Boulanger R., Introduction to Sound Design in Csound. Teoksessa: The Csound Book (toim. Boulanger R.), MIT Press, Cambridge, 2000, 5–64.

Byrd D., Music Notation by Computer. Väitöskirja, Indiana University Computer Science Department, 1984.

Byrd D., Music Notation Software and Intelligence. Computer Music Journal, Vol 18, No. 1, 1994, 17–20.

Correia E. J., Music Software and Data Formats: A Survey of Current Usage. Teoksessa: Computing in Musicology, Vol 9, 1993–94, 233–246.

Dannenberg R. B., Music Representation Issues, Techniques, and Systems. Computer Music Journal, Vol 17, No. 3, 1993, 20–30.

Dydo J. S., DARMS: The Note-Processor Dialect. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 175–192.

Final report. SVB/WP6/V3/F, Computer Access to Notation and Text in Music Libraries, 1996. Saatavilla Word-muodossa: <URL: http://www.svb.nl/project/download/Cantate/cant_fin.zip>, [viitattu 22.8.1999].

Goldfarb C. F., The SGML Handbook. Oxford University Press, New York, 1990.

Grande C., The Notation Interchange File Format: A Windows-Compliant Approach. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 491–512.

Grande C., Belkin A., The Development of the Notation Interchange File Format. Computer Music Journal, Vol 20, No. 4, 1997, 33–43.

GUIDO Music Notation Format Page [online], 1999. Saatavilla WWW-muodossa: <URL: http://www.informatik.th-darmstadt.de/AFS/CM/GUIDO/>, [viitattu 27.7.1999].

Hall T., DARMS: The A-R Dialect. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 193–200.

Hewlett W. B., MuseData: Multipurpose Representation. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 402–447.

Hewlett W. B., Selfridge-Field E., Cooper D., Field B. A., Ng K. C., Sitter P., MIDI. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 41–70.

Hoos H. H., Hamel K. A., The GUIDO Music Notation Format Version 1.0. Specification Part 1: Basic GUIDO. TI 20/97, Fachbereich Informatik,

Technische Universität Darmstadt, Darmstadt, 1997. Saatavilla PS-muodossa: <URL: http://www.informatik.th-darmstadt.de/AFS/CM/GUIDO/doc.html>, [viitattu 9.8.1999].

Hoos H. H., Hamel K. A., Renz K., Kilian J., The GUIDO Music Notation Format - A Novel Approach for Adequately Representing Score-level Music. Teoksessa: ICMC'98 Proceedings, 1998, 451–454. Saatavilla PS- ja PDF-muodossa: <URL: http://www.informatik.th-darmstadt.de/AFS/CM/GUIDO/doc.html>, [viitattu 9.8.1999].

Hudak P., Makucevich T., Gadde S., Whong B., Haskore music notation - an algebra of music. Journal of Functional Programming, Vol 6, No. 3, 1996, 465–483.

Humdrum Toolkit [online]. University of Virginia, 1998. Saatavilla WWW-muodossa: <URL: http://www.lib.virginia.edu/dmmc/Music/Humdrum/>, [viitattu 23.9.1999].

Huron D., Humdrum and Kern: Selective Feature Encoding. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 375–401.

Icking W., MuTEX, MusicTEX, and MusiXTEX. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 222–231.

ISO/IEC 10744, Information processing – Hypermedia/Time-based Structuring Language (HyTime) - 2d edition. ISO/IEC JTC1/SC18 WG8 N1920, International Organization for Standardization, 1997. Saatavilla HTML-, PDF- ja SGML-muodossa: <URL: ftp://ftp.ornl.gov/pub/sgml/wg8/document/n1920/index.htm>, [viitattu 13.9.1999].

ISO/IEC DIS 10743, Information Technology – Standard Music Description Language (SMDL). International Organization for Standardization/International Electrotechnical Commission, Geneva, 1995.

Maegaard J., Musiikin modernismi. Suom. S. Heikinheimo. WSOY, Porvoo, 1964.

Mathews M. V., The Radio Baton Conductor Score File. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 153–159.

McLane A., Music as Information. Teoksessa: Annual Review of Information Science and Technology (toim. Williams M. E.), Information Today, Medford, NJ, USA, Vol 31, 1996, 225–262.

Newcomb S. R., Standard Music Description Language complies with hypermedia standard. IEEE Computer, Vol 24, No. 7, 1991, 76–79.

Newcomb S. R., Kipp N. A., Newcomb V. T., The "HyTime"Hypermedia/Time-based Document Structuring Language. Communications of the ACM, Vol 34, No. 11, 1991, 67–83.

NIFF 6a.3, Notation Interchange File Format, 1998. Saatavilla Word-muodossa: <URL: http://esi24.ESI.UMontreal.CA:80/~belkina/N/NIFF6a3.zip>, [viitattu 18.9.1999].

NIFF to SMDL translator. WP3/A3-2/D3/2, Computer Access to Notation and Text in Music Libraries, 1996.

Norrila P., HyTime rakenteisten dokumenttien standardina. Pro gradu, Jyväskylän yliopisto, Tietojenkäsittelytieteiden laitos, Jyväskylä, 1995.

Oksala Y., Musiikin perusteet. Osa 1: Nuottikirjoitus. Fazer, Helsinki, neljäs painos, 1971.

Pennycook B., The Music Library of the Future: A Pilot Project [online], 1997. Saatavilla WWW-muodossa: <URL: http://www.music.mcgill.ca/newHome/mlfProject/html/mlfProposal.html>, [viitattu 25.4.1999].

Renz K., Hoos H. H., A Web-based Approach to Music Notation Using GUIDO. Teoksessa: ICMC'98 Proceedings, 1998, 455–458. Saatavilla PS- ja PDF-muodossa: <URL: http://www.informatik.th-darmstadt.de/AFS/CM/GUIDO/doc.html>, [viitattu 9.8.1999].

Report on SMDL evaluation. UBR/WP3/D3-3, Computer Access to Notation and Text in Music Libraries, 1996. Saatavilla Word-muodossa: <URL: http://www.svb.nl/project/download/Cantate/cant-3-3.zip>, [viitattu 22.8.1999].

Salminen A., Rakenteisen tekstin hallinta. Tietojenkäsittelytieteen julkaisuja, Opetusmonisteita OM-3, Jyväskylän yliopisto, Jyväskylä, 1992.

Salminen A., Elektroninen teksti: mitä se on? Teoksessa: SGML-seminaari. Eduskunnan kirjaston seminaari 12.12.1994 ja 14.12.1994 (toim. Kangas S., Karjalainen L.), Eduskunnan kirjaston tutkimuksia ja selvityksiä 2, Eduskunnan kirjasto, Helsinki, 1995.

Scheirer E. D., Vercoe B. L., SAOL: The MPEG-4 Structured Audio Orchestra Language. Computer Music Journal, Vol 23, No. 2, 1999, 31–51.

Schottstaedt B., Common Music Notation. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 217–221.

Selfridge-Field E. (toim.), Beyond MIDI. The Handbook of Musical Codes. MIT Press, Cambridge, 1997a.

Selfridge-Field E., Computer Musicology: Accomplishments and Challenges. Computer Music Journal, Vol 20, No. 4, 1997b, 29–32.

Selfridge-Field E., DARMS, Its Dialects, and Its Uses. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997c, 163–174.

Selfridge-Field E., Describing Musical Information. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997d, 3–38.

Sloan D., Aspects of Music Representation in HyTime/SMDL. Computer Music Journal, Vol 17, No. 4, 1993, 51–59.

Sloan D., Newcomb S. R., HyTime and Standard Music Description Language: A Document-Description Approach. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 469–490.

Smith L., SCORE. Teoksessa: Beyond MIDI. The Handbook of Musical Codes (toim. Selfridge-Field E.), MIT Press, Cambridge, 1997, 252–280.

Steyn J., Music Markup Language [online], 1999. Saatavilla WWW-muodossa: <URL: http://is.up.ac.za/mml/>, [viitattu 11.8.1999].

Stone K., Music Notation in the Twentieth Century: A Practical Guidebook. Norton, New York, 1980.

Survey of Music Libraries. OBA/WP1/D1-1/V3/F, Computer Access to Notation and Text in Music Libraries, 1996. Saatavilla Word-muodossa: <URL: http://www.svb.nl/project/download/Cantate/cant-1-1.zip>, [viitattu 22.8.1999].

Survey of Music Publishers. SVB/WP2/D2-1/V2.0/F, Computer Access to Notation and Text in Music Libraries, 1996. Saatavilla Word-muodossa: <URL: http://www.svb.nl/project/download/Cantate/cant-2-1.zip>, [viitattu 22.8.1999].

Taitto I., Mensuraalinuottikirjoitus. Teoksessa: Suuri musiikkitietosanakirja, 4. osa, Weilin + Göös, Keuruu, 1991, 152.

The Csound Front Page [online], 2000. Saatavilla WWW-muodossa: <URL: http://mitpress.mit.edu/e-books/csound/frontpage.html>, [viitattu 12.12.2000].

The MPEG Home Page [online], 1999. Saatavilla WWW-muodossa: <URL: http://drogo.cselt.stet.it/mpeg/>, [viitattu 23.9.1999].

Thesaurus musicarum italicarum, A multimedial edition of Italian music-theoretical sources [online], 1998. Saatavilla WWW-muodossa: <URL: http://candl.let.ruu.nl/Research/tmi/proj_eng.htm>, [viitattu [viitattu 22.4.1999]].

Tiensuu J., Alvin Lucier. Teoksessa: Suuri musiikkitietosanakirja, 4. osa, Weilin + Göös, Keuruu, 1991a, 90–91.

Tiensuu J., Nuottikirjoitus. Teoksessa: Suuri musiikkitietosanakirja, 4. osa, Weilin + Göös, Keuruu, 1991b, 264–267.

Wiggins G., Miranda E., Smaill A., Harris M., A Framework for the Evaluation of Music Representation Systems. Computer Music Journal, Vol 17, No. 3, 1993, 31–42.

Wild J., A Review of the Humdrum Toolkit: UNIX Tools for Musical Research, created by David Huron. Music Theory Online [online], Vol 2, No. 7, 1996. Saatavilla WWW-muodossa: <URL: http://boethius.music.ucsb.edu/mto/issues/mto.96.2.7/mto.96.2.7.wild.html>, [viitattu 23.9.1999].