Jouni Turpeinen

UML-laajennusten arviointi: sovellusalueena WWW-sovellukset

Tietojärjestelmätieteen
pro gradu -tutkielma
30.7.2004

Jyväskylän yliopisto
Tietojenkäsittelytieteiden laitos
Jyväskylä

TIIVISTELMÄ

Turpeinen, Jouni Petteri
UML-laajennusten arviointi: sovellusalueena WWW-sovellukset
Jyväskylä: Jyväskylän yliopisto, 2004.
138 s.
Pro gradu –tutkielma

UML on yleiskäyttöinen mallinnuskieli, jolle on esitetty lukuisia sovellusaluekohtaisia laajennuksia. UML:n suosiosta huolimatta laajennuksia koskevia arviointeja eikä vertailuja ole juurikaan tehty. Tämän tutkimuksen tavoitteena on selvittää UML:n laajennustavat, kartoittaa laajennusten kohteena olevat sovellusalueet, kehittää laajennusten arvioitiin sopiva viitekehys sekä tehdä WWW-sovellusalueen UML-laajennusten arviointi ja vertailu.

Tutkimus on otteeltaan käsitteellisteoreettinen. Se perustuu UML:ää, kielten arviointia sekä UML-laajennuksia koskevaan kirjallisuuteen.

Tutkimuksen tuloksena osoitetaan, että on kaksi pääasiallista laajennustapaa, UML:n sisäänrakennetut laajennusmekanismit ja metamallipohjainen laajentaminen. Kirjallisuuden läpikäynnillä on valittu 20 UML:n laajennusta ennalta määriteltyjen kriteerien mukaiseen tarkasteluun. Tästä kartoituksesta käy muiden muassa ilmi, että suurin osa laajennuksista on kehitetty WWW-sovellusten, agenttipohjaisten sovellusten tai tosiaikajärjestelmien mallintamiseen. Laajennusten arviointia varten kehitetty viitekehys pohjautuu olemassa olevaan kirjallisuuteen. Viitekehyksen mukaiseen arviointiin ja vertailuun on valittu kolme WWW-sovellusten suunnitteluun kehitettyä UML-laajennusta. Arviointi osoittaa laajennusten välillä selkeitä eroja ja puutteita niin käsitteiden kuin notaation osalta. Tutkimuksen tulokset ovat hyödyllisiä UML:n kehittäjille, UML-laajennusten kehittäjille ja soveltajille.

AVAINSANAT: UML, UML-laajennus, sovellusalue, WWW-sovellus

SISÄLLYSLUETTELO

TAULUKOT

KUVIOT

1 JOHDANTO

UML (Unified Modeling Language) on yleiskäyttöinen visuaalinen mallinnuskieli, joka tarjoaa kuvaustekniikoita ohjelmistojen määrittelyyn, visualisointiin, kehittämiseen ja dokumentointiin. UML on monen vuoden kehittelyn tulos, jossa pyritään yhdistämään aikaisemmin ilmestyneiden olioperusteisten mallien parhaat puolet. UML pohjautuu pääosin kolmeen oliomallinnusmenetelmään: Booch (1991), OMT (Rumbaugh 1991) ja OOSE (Jacobson, Christerson, Jonsson ym. 1992). OMG (Object Management Group) hyväksyi UML:n standardiksi vuoden 1997 lopussa. (OMG 2003a, 1–55)

UML:ää voidaan pitää nykyään oliokeskeisten sovellusten mallintamista merkittävästi yhtenäistävänä standardina. Se tarjoaa laajan valikoiman malleja niin staattisten kuin dynaamisten piirteiden mallintamiseen, tukien näin järjestelmänkehityksen eri vaiheita aina vaatimusmäärittelystä ohjelmistotuotteen testaukseen.

UML on melko laaja kokonaisuus sisältäen muun muassa yhdeksän kaaviotyyppiä. Kuitenkin UML:n kehittäjät jättivät mallinnuskielestä tarkoituksella pois joitakin käsitteitä ja mekanismeja, joita käytetään muissa mallinnuskielissä, jotta UML:stä ei tulisi liian monimutkaista. UML:n kehittäjien yhtenä tavoitteena oli tehdä mallinnuskielestä mahdollisimman yleiskäyttöinen. Tämän lisäksi heillä oli tavoitteena tehdä UML:stä laajennettava, jotta se voidaan laajentaa ja sovittaa tiettyä menetelmää, sovellusaluetta, organisaatiota tai käyttäjää varten. UML:ään määriteltiin sisäiset laajennusmekanismit, joiden avulla käyttäjät voivat määritellä ja käyttää uusia elementtejä. (Eriksson & Penker 2000, 187–188)

Koska UML:stä on muodostunut yleisesti käytettävä standardi, sitä on alettu soveltaa hyvin monenlaisilla sovellusalueilla ja erilaisissa organisaatioissa. UML:llä pystytään mallintamaan hyvin erityyppisiä järjestelmiä. Koska

erityisalojen käsitteet eivät kuitenkaan sisälly suoraan UML:ään, on kirjallisuudessa esitetty laajennuksia UML:ään.

Kirjallisuudessa on esitetty kymmeniä UML:n laajennuksia useille eri sovellusalueille. Tässä raportissa *sovellusalueella* tarkoitetaan tiettyä ohjelmisto- tai järjestelmätyyppiä sekä tietojärjestelmän suunnitteluparadigmaa, jota yhdistää yhtenäinen joukko ongelmia. Laajennuksia on esitetty muun muassa seuraaville sovellusalueille: WWW-sovellukset (esim. Conallen 1999a, 1999b, 2000; Baresi, Garzotto & Paolini 2001), tosiaikajärjestelmät (esim. Selic & Rumbaugh 1998; Apvrille, Saqui-Sannes, Lohr ym. 2001), multimediasovellukset (esim. Sauer & Engels 1999, 2001), agentti-perusteiset sovellukset (esim. Caire, Leal, Chainho ym. 2002; Odell, Perunak & Bauer 2000) sekä tietokantasovellukset (esim. Ambler 2002; Gornik 2002).

Laajennusesitysten yhteydessä käytettävä käsitteistö ei ole kovin vakiintunut. OMG:n (2003a, 131–133) mukaan UML:ää pystytään laajentamaan kahdella tavalla. Ensimmäisessä tavassa käytetään UML:n sisäänrakennettuja laajennusmekanismeja, jotka ovat stereotyypit, rajoitukset, nimetyt arvot ja ominaisuudet. Laajennusta, jossa hyödynnetään pelkästään UML:n sisäänrakennettuja laajennusmekanismeja, kutsutaan *profiiliksi*. Edellä mainitusta laajennustavasta käytetään myös nimeä *kevyensarjan* (lightweight) laajennusmekanismi. Kehitystyön tavoitteena on ollut, että UML:n laajentaminen onnistuu pelkästään UML:n sisäisten laajennusmekanismien avulla. Toisaalta laajennuksia voi tehdä myös lisäämällä metamalliin uusia metarakenteita tai muuten muokkaamalla sitä. Tämä laajennustapa kuvataan MOF (Meta Object Facility) määrityksessä (OMG 2002) ja siitä käytetään nimeä *raskaansarjan* (heavyweight) laajennusmekanismi. (OMG 2003a, 131–133)

UML:n suosiosta huolimatta kattavaa kartoitusta, jossa selvitetään, mille kaikille sovellusalueille UML-laajennuksia on suunniteltu, ei ole tehty. Myöskään laajennuksia koskevia arviointeja eikä vertailuja ole juuri tehty.

Tämän tutkimuksen tavoitteet jakautuvat neljään osaan. Ensimmäisenä tavoitteena on tarkastella UML:ää sen laajentamisen näkökulmasta. Tämän tavoitteen tärkeimpänä tehtävänä on tutkia UML:n laajennustapoja. Tarkoituksena on lähinnä luoda taustatietoa, jota tarvitaan muiden tavoitteiden selvittämiseksi.

Toisena tavoitteena on tehdä kartoitus UML-laajennuksista sovellusalueittain. Tähän tavoitteeseen liittyvä tutkimusongelma on: ”Mille sovellusalueille UML-laajennuksia on esitetty?” Lisäksi kartoituksessa selvitetään laajennuksista erinäisiä ominaisuuksia, kuten laajennustapa sekä kaaviot, joihin laajennukset kohdistuvat. Kartoituksessa tarkastellaan periaatteessa kaikkia laajennuksia, jotka on suunniteltu jollekin nimetylle sovellusalueelle. Kartoitus ei ole kuitenkaan täysin kattava, sillä tutkielman kirjoittajalla ei ole mahdollisuutta saada käsiinsä kaikkea alasta julkaistua kirjallisuutta. Kartoituksen perusteella myös valittiin WWW-sovellusalueen laajennukset arvioitavaksi ja vertailtavaksi yksityiskohtaisemmin. Tämän sovellusalueen valintaan vaikutti muun muassa se, että sovellusalueelle on esitetty riittävästi laajennuksia, jotta niiden välistä tarkempaa vertailua voidaan tehdä. Lisäksi WWW-sovellusalue on nuori ja nopeasti kehittyvä, joten tämän sovellusalueen tutkimus on hyödyllistä.

Tutkielman kolmantena tavoitteena on muodostaa viitekehys, jonka avulla UML-laajennuksia voidaan arvioida ja vertailla sovellusaluekohtaisesti. Tähän tavoitteeseen liittyvä tutkimusongelma on: ”Miten UML-laajennuksia voidaan arvioida ja vertailla sovellusaluekohtaisesti? Viitekehyksen kehittäminen on välttämätöntä, sillä kirjallisuudessa ei ole esitetty viitekehystä, jonka avulla pystyttäisiin suoraan vertailemaan UML-laajennuksia. Tässä tutkielmassa muodostettava viitekehys pohjautuu olemassa olevaan kirjallisuuteen.

Tutkielman neljäntenä tavoitteena on arvioida ja vertailla muodostetun viitekehyksen pohjalta WWW-sovellusalueen UML-laajennuksia. Tähän tavoitteeseen liittyvä tutkimusongelma on: ”Millaisia WWW-sovellusalueen

UML-laajennukset ovat ja miten ne eroavat toisistaan? Tässä tutkielmassa arvioitavat ja vertailtavat laajennukset ovat: UWE-menetelmä (esim. Hennicker & Koch 2001a), W2000-viitekehys (esim. Baresi, Garzotto & Paolini 2001) sekä WAE-profiili (esim. Conallen 2000).

Tutkimusmenetelmä on käsitteellisteoreettinen. Aihetta lähestytään olemassa olevan kirjallisuuden pohjalta.

Tutkielma on jäsennetty kahdeksaan lukuun. Luvussa kaksi tarkastellaan UML:ää sen laajentamisen näkökulmasta. Tässä luvussa kuvataan muun muassa UML:n laajennustavat ja määritellään keskeisiä käsitteitä. Luvussa kolme esitetään UML-laajennusten kartoitus sovellusalueittain. Tämän luvun alussa määritellään käsiteltävien laajennusten rajaus ja laajennuksista kartoitettavat asiat. Luvun lopussa raportoidaan kartoituksen tulokset. Luvussa neljä muodostetaan viitekehys UML-laajennusten tarkasteluun. Tässä luvussa pääpaino on Krogstien ja Solvbergin (2000) viitekehyksen esittelyssä ja arvioinnissa. Luvun lopuksi muodostetaan muokattu viitekehys pohjautuen edellä mainittuun viitekehykseen. Luvussa viisi tarkastellaan WWW-sovelluksia ja WWW-sovellusten suunnitteluun kehitettyjä UML-laajennuksia. Luvun alussa määritellään WWW-sovelluksen käsite ja esitellään WWW-sovellusten kehitystyön erityispiirteitä. Luvun pääpaino on kuitenkin WWW-sovellusalueelle kehitettyjen laajennusten esittelyssä. Luvussa kuusi arvioidaan ja vertaillaan edellisessä luvussa esitettyjä UML-laajennuksia pohjautuen muodostettuun viitekehykseen. Luvussa seitsemän tehdään johtopäätöksiä tutkielman tuloksista ja esitetään jatkotutkimusaiheita. Luvussa kahdeksan tutkielman sisällöstä vedetään yhteenveto.

2 UML

Tässä luvussa keskitytään tarkastelemaan UML:ää sen laajentamisen näkökulmasta. Vaikka luvun alussa esitetään lyhyt yleiskatsaus UML:ään, oletetaan kuitenkin, että UML sekä oliolähestymistavan keskeiset periaatteet ovat lukijalle pääosin tuttuja. Tässä luvussa määritellään myös tutkimuksen kannalta keskeisiä käsitteitä. Tarkoituksena on luoda taustaa laajennusten vertailuun ja arviointiin. Jotta voidaan käsitellä UML:n laajentamista, on tärkeää tuntea UML:n semantiikkaa. Tämän takia luvussa esitellään UML:n nelitasoinen metamalliarkkitehtuuri keskittyen erityisesti metamallitasoon. Luvussa esitellään myös UML:n sisäiset laajennusmekanismit, jotka luovat pohjan UML:n laajentamiselle. Tämän luvun tärkeimpänä antina kuvataan lopuksi UML:n laajennustavat. Luku pohjautuu UML:n versioon 1.5, koska tämän hetkiset laajennukset perustuvat tähän tai aikaisempiin UML:n versioihin.

2.1 UML:n yleiskatsaus

UML (Unified Modeling Language) on yleiskäyttöinen visuaalinen mallinnuskieli, joka tarjoaa kuvaustekniikoita ohjelmistojen määrittelyyn, visualisointiin, kehittämiseen ja dokumentointiin. UML on monen vuoden kehittelyn tulos, jossa pyritään yhdistämään aikaisemmin ilmestyneiden olioperusteisten mallien parhaat puolet. UML:n kehittäjät ovat kolmena ”amigona” tunnetut Grady Booch, James Rumbaugh ja Ivar Jacobson. UML pohjautuukin pääosin heidän aikaisemmin julkaisemiinsa oliomallinnusmenetelmiin: Booch (1991), OMT (Rumbaugh 1991) ja OOSE (Jacobson, Christerson, Jonsson ym. 1992). Alun perin UML:stä suunniteltiin vain käytännön standardia. Vuonna 1996 OMG pyysi kuitenkin ehdokkaita olioperusteiseksi mallinnuskielistandardiksi. UML asetettiin yhdeksi ehdokkaaksi. OMG hyväksyi ehdokkaista UML:n standardiksi yksimielisesti vuoden 1997 lopussa. (Rumbaugh, Jacobson & Booch 1999, 3–6)

UML:n julkaisun jälkeenkin sen kehitystyö on ollut jatkuvaa. UML:n kehitystiimin johtaja Cris Korbyn kuvaa artikkelissaan (Korby 1999) UML:n kehitysprosessin yksityiskohtaisesti. Tähän mennessä UML:ään tehdyt korjaukset ovat olleet kuitenkin melko pieniä. Tällä hetkellä UML:n virallinen versio on 1.5. Myös versio 2.0 on julkaistu lähes kokonaan. Tämä versio tuo mukanaan hieman suurempia muutoksia.

UML:ää voidaan pitää nykyään oliokeskeisten sovellusten mallintamista merkittävästi yhtenäistävänä standardina. Sen suosioon on monia syitä. Rumbaughin, Jacobsonin ja Boochin (1999, 8) mukaan tärkeimpänä tavoitteena oli tehdä UML:stä mahdollisimman yleiskäyttöinen, jotta sitä voidaan käyttää kaikilla sovellusalueilla ja kaikissa järjestelmänkehityksen vaiheissa. UML tarjoaakin laajan mallivalikoiman niin staattisten kuin dynaamisten piirteiden mallintamiseen tukien näin järjestelmänkehityksen eri vaiheita aina vaatimusmäärittelystä ohjelmistotuotteen testaukseen. Edelleen keskeisenä tavoitteena UML:n kehittäjillä oli olla sitomatta UML:ää mihinkään tiettyyn menetelmään. UML:ää voidaankin käyttää useimpien ohjelmistonkehitystä tukevien menetelmien yhteydessä.

UML on melko laaja kokonaisuus sisältäen yhdeksän kaaviotyyppiä: käyttötapauskaavio (use-case diagram), luokkakaavio (class diagram), oliokaavio (object diagram), tilakaavio (state diagram), tapahtumakaavio (sequence diagram), yhteistyökaavio (collaboration diagram), toimintokaavio (activity diagram), komponenttikaavio (component diagram) sekä hajautuskaavio (deployment diagram). (Booch, Rumbaugh & Jacobson 1998) UML:n kehittäjät jättivät kuitenkin mallinnuskielestä tarkoituksella pois joitakin käsitteitä ja mekanismeja, jotta UML:stä ei tulisi liian monimutkaista. Sen sijaan UML:stä tehtiin laajennettava, jotta sitä voidaan laajentaa ja sovittaa tiettyä menetelmää, sovellusaluetta, organisaatiota tai käyttäjää varten. UML:ään määriteltiin sisäiset laajennusmekanismit, joiden avulla käyttäjät voivat määritellä ja käyttää uusia elementtejä. (Eriksson & Penker 2000, 187–188)

2.2 UML:n semantiikka

UML:n semantiikan eli merkityksien tunteminen luo pohjan UML:n laajennusten ymmärtämiselle. UML:n metamallit määrittelevät UML:n semantiikan. (Eriksson & Penker 2000, 188) Tämän takia seuraavaksi tarkastellaan UML:n metamalliarkkitehtuuria sekä erityisesti metamallitasoa ja elementin käsitettä.

2.2.1 Nelitasoinen metamalliarkkitehtuuri

UML:n arkkitehtuuri perustuu nelitasoiseen metamalliin (four-layer metamodel architecture). Taulukossa 1 kuvataan metamalliarkkitehtuurin rakenne ja tarkoitus. Arkkitehtuuri koostuu seuraavista tasoista: meta-metamalli, metamalli, malli sekä käyttäjäobjektit (user objects). Ylimmällä tasolla oleva meta-metamalli luo pohjan arkkitehtuurille, kun taas alimman tason käyttäjäobjektit esittävät konkreettisinta tasoa. Arkkitehtuurissa alempi taso esittää ylemmän tason ilmentymää. Esimerkiksi metamallin Luokka on meta-metamallin MetaLuokan ilmentymä. (OMG 2003a, 59–64)

TAULUKKO 1. Nelitasoinen metamalliarkkitehtuuri (vrt. OMG 2003a, 63).

Taso	**Kuvaus**	**Esimerkki**
meta-metamalli	Metamalliarkkitehtuurin perusta. Kuvaa kielen, jolla määritellään metamalli.	MetaLuokka, MetaAttribuutti, MetaOperaatio.
metamalli	Meta-metamallin ilmentymä. Kuvaa kielen, jolla määritellään malli.	Luokka, Attribuutti, Operaatio, Komponentti.
malli	Metamallin ilmentymä. Kuvaa kielen, jolla kuvaillaan sovellusalue.	Osake, kysyHinta, myyntiTasoMääräys.
käyttäjäobjektit	Mallin ilmentymä. Kuvaa sovellusalueen.	<Huippu_SW_Osake_98789>, 654.56, myynti_raja_määräys.

Nelitasoinen metamalliarkkitehtuuri on osoittautunut hyväksi perusrakenteeksi kuvaamaan tarkkaan monimutkaisten mallien semantiikkaa. Tässä lähestymistavassa on Korbyn (1999) mukaan monia hyötyjä:

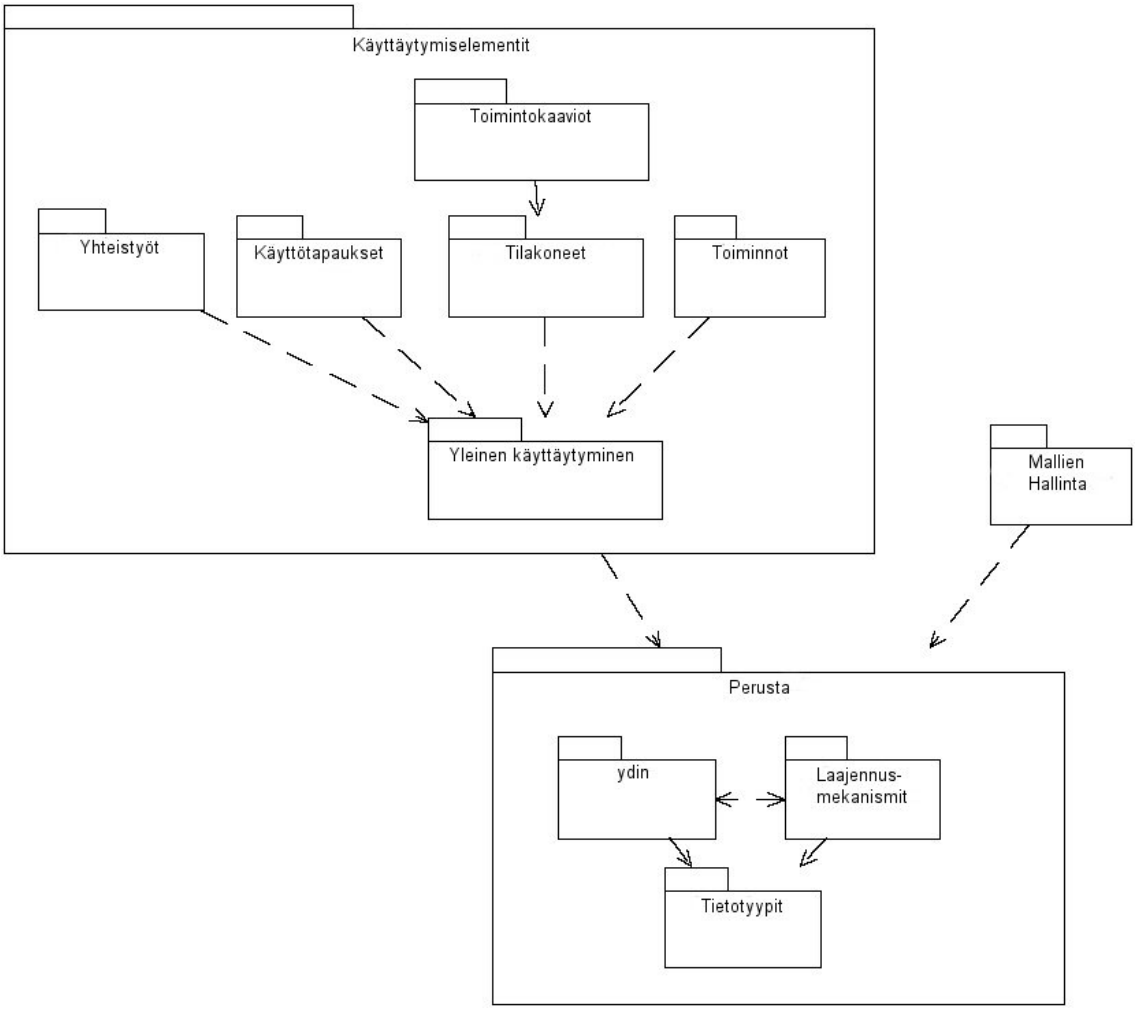
- Se tarjoaa toistuvat semanttiset käsitteet kaikille tasoille, mikä johtaa suppeampaan ja säännöllisempään semantiikkaan.
- Se tarjoaa perusrakenteen UML:n laajennusten määrittelyyn.
- Se tarjoaa perusrakenteen, jonka avulla UML:n metamalli voidaan sovittaa muihin standardeihin, jotka pohjautuvat nelitasoiseen metamalliarkkitehtuuriin.

UML:n meta-metamalli määritellään MOF (Meta Object Facility) määrityksessä (OMG 2002). UML:n metamalli on MOF meta-metamallin ilmentymä (OMG 2003a, 63–64). Seuraavassa alakohdassa käsitellään UML:n metamallia hieman tarkemmin.

2.2.2 Metamallitaso

UML:n metamalli on melko monimutkainen, tämän takia se on jaettu loogisiin paketteihin kuvion 1 mukaisesti. Paketin sisäiset metaluokat muodostavat kiinteän kokonaisuuden. Sen sijaan pakettien välisillä metaluokilla on vähän kytkentöjä. Metamalli on jaettu kolmeen pakettiin: perusta (Foundation), käyttäytymiselementit (Behavioral Elements) ja mallien hallinta (Model Management). Paketit on jaettu edelleen alipaketteihin. Perusta-paketti sisältää seuraavat alipaketit: ydin (Core), laajennusmekanismit (Extension Mechanisms) sekä tietotyypit (Data Types). Käyttäytymiselementit-paketti sisältää alipaketit: toimintakaaviot (Activity Graphs), yhteistyöt (Collaborations), käyttötapaukset (Use Cases), tilakoneet (State Machines), toiminnot (Actions) sekä yleinen käyttäytyminen (Common Behavior). Riippuvuudet pakettien välillä esitetään

nuolilla. Paketti, josta nuoli lähtee, riippuu nuolen kärjen osoittamasta paketista. (OMG 2003a, 36–65)



KUVIO 1. UML:n metamallin pakettirakenne (OMG 2003a, 36).

UML:n metamalli kuvataan tarkemmin pakettikohtaisesti käyttäen UML-kaavioita sekä luonnollista ja formaalia kieltä (Object Constraint Language). Yleisellä tasolla määrityksessä kuvataan ensin kielen syntaksi ja tämän jälkeen staattinen sekä dynaaminen semantiikka. Tarkemmin paketit kuvataan UML:n metamallissa taulukon 2 mukaisesti. Jokainen paketti on kuvattu taulukon yhden tai useamman määritelmän osan avulla. Huomioitavaa on, että nykyinen kuvaus ei ole täysin formaali. (OMG 2003a, 65–68).

TAULUKKO 2. Metamallin määritelmän osat (vrt. OMG 2003a, 67–68).

Määritelmän osa	**Sisältö**
Abstrakti kielioppi (Abstract Syntax)	Sisältää UML-notaatiolla tehdyn luokkakaavion ja kuvaukset luonnollisella kielellä luokkakaaviossa käytetyistä käsitteistä.
Muotosäännöt (Well-formedness Rules)	Sisältää formaalin (Object Constraint Language) sekä sanallisen kuvauksen UML-metaluokkien staattisista ominaisuuksista.
Semantiikka (Semantics)	Sisältää sanallisen selityksen rakenteiden merkityksistä.
Vakioelementit (Standard Elements)	Sisältää aiemmin samassa osassa määriteltyjen metaluokkien stereotyyppien sanalliset kuvaukset sekä muotosäännöt.
Huomautukset (Notes)	Sisältää sanalliset selitykset metamallin ratkaisujen syistä, käytännön neuvoja sekä esimerkkejä.

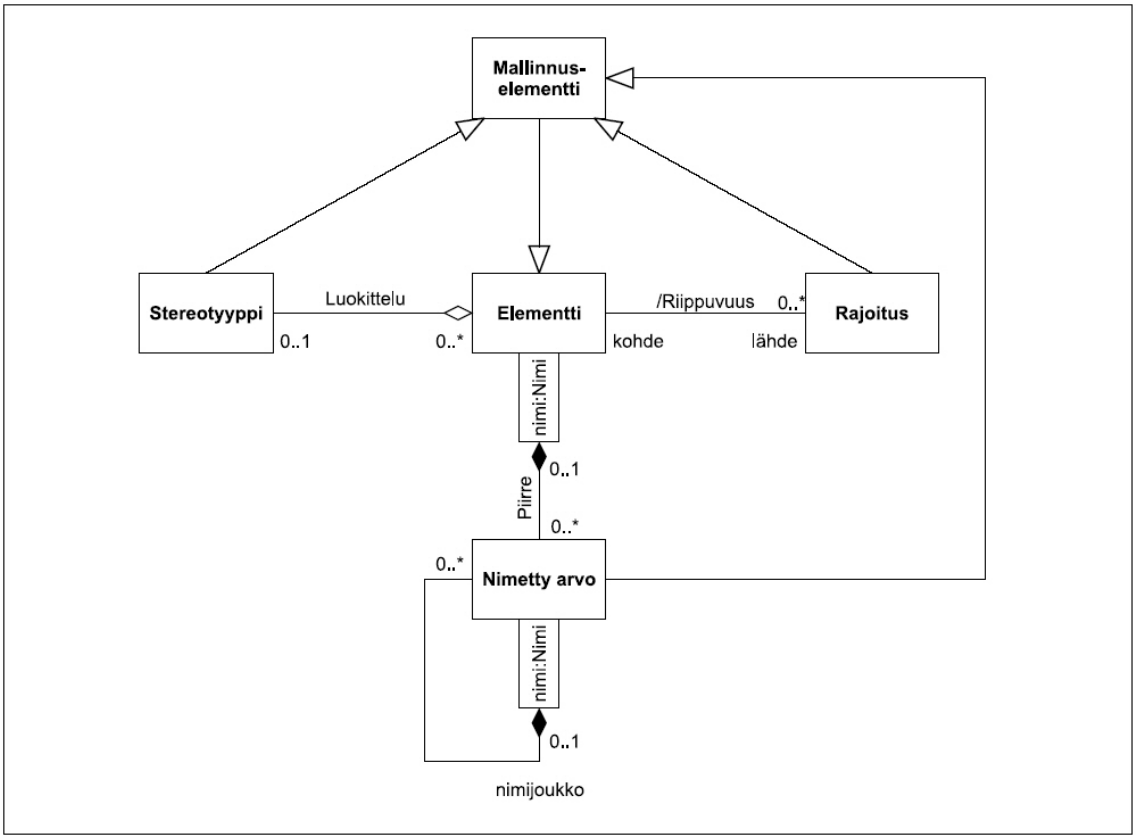
Tämän tutkimuksen kannalta olennaisin paketti on perusta-paketti ja erityisesti sen alipaketit ydin ja laajennusmekanismit. Ydin-paketti on perusta-paketin olennaisin alipaketti, se määrittelee metamallin abstraktit ja konkreetit peruskäsitteet. Tämä paketti muodostaa arkkitehtuurisen selkärangan täydentävien käsitteiden, kuten metaluokkien, meta-assosiaatioiden ja meta-attribuuttien liittämiselle. Laajennusmekanismit-paketti sisältää UML:n laajennusmekanismien kuvaukset. (OMG 2003a, 69–71) Seuraavassa alakohdassa määritellään ydin-paketin sisältämä elementin käsite.

2.2.3 Elementti

Ydin-paketti määrittelee muun muassa elementin (element) käsitteen (OMG 2003a, 70–71). *Elementti* on abstrakti kantaluokka useimmille UML:n osille. Se toimii perustana, johon voidaan kiinnittää monia eri mekanismeja. UML:n laajentamisen kannalta elementti on tärkeä käsite. Elementti erikoistuu mallinnuselementteihin (model elements), näkymäelementteihin (view elements), järjestelmään (system) sekä malliin (model). *Mallinnuselementti* on mallinnettavasta järjestelmästä johdettu abstraktio. Mallinnuselementit

erikoistuvat kaikkiin UML:n käyttämiin mallinnuskäsitteisiin, joista tärkeimmät ovat tyyppi, suhde, stereotyyppi, rajoitus ja nimetty arvo. *Näkymäelementti* on joko graafinen tai sanallinen projektio mallinnuselementistä tai mallinnuselementtijoukosta. Mallinnuselementit ovat siis käsitteitä ja näkymäelementit mallien tai kaavioiden rakentamiseen käytettyjä symboleja. Melkein kaikilla mallinnuselementeillä on elementti esitystä varten. Joillekin mallinnuselementeille, kuten mallinnuselementin käyttäytymiselle, ei ole kuitenkaan olemassa esitysmuotoa. Kaikilla elementeillä on jokin nimi. (Eriksson & Penker 2000, 188)

Kuvion 2 mukaan elementtiin liittyy nolla tai yksi stereotyyppiä, nolla tai useampia nimettyjä arvoja sekä nolla tai useampia rajoituksia. Nimetyt arvot voivat olla arvojoukkoja. Stereotyyppi, rajoitus, ja nimetty arvo periytyvät kaikki mallinnuselementistä. (Eriksson & Penker 2000, 192–194)



KUVIO 2. Laajennusmekanismit ja elementti (Eriksson & Penker 2000, 193).

Kaikkia elementtejä ei voi yleistää ja erikoistaa, se onnistuu vain yleistettävillä elementeillä (generalizable element), joita ovat stereotyyppi, paketti ja tyyppi. Yleistettävien elementtien aliluokat ovat myös yleistettäviä. Tyypillä on aliluokat alkeistyyppi (primitive type), luokka (class) ja käyttötapaus (use-case). Luokalla on edelleen joukko aliluokkia. Jos nimetty arvo, stereotyyppi tai rajoite liitetään mallinnuselementtiin, elementistä perityllä mallinnuselementillä tulee olemaan sama nimetty arvo, stereotyyppi tai rajoite. Mikäli yleistyksellä on stereotyyppi, sen erikoistumisella ei voi olla erillistä stereotyyppiä. Elementillä voi olla vain nolla tai yksi stereotyyppi, mukaan lukien perityt stereotyypit. (Eriksson & Penker 2000, 192–194)

Seuraavassa kohdassa tarkastellaan laajennusmekanismit-paketin sisältämiä käsitteitä, kuten stereotyyppiä, rajoitusta sekä nimettyä arvoa tarkemmin. Tarkoituksena on selvittää, kuinka UML:ää voidaan laajentaa sisäisten laajennusmekanismien avulla pohjautuen elementin käsitteeseen.

2.3 UML:n sisäiset laajennusmekanismit

Laajennusmekanismit-paketissa määritellään, kuinka UML:n mallinnuselementtejä voidaan räätälöidä ja laajentaa. Mallinnuselementeille voidaan antaa uusia semanttisia merkityksiä stereotyyppien, rajoitusten sekä nimettyjen arvojen ja ominaisuuksien avulla. (OMG 2003a, 131) Tässä kohdassa käsitellään UML:n sisäisistä laajennusmekanismeista ominaisuuksia ja nimettyjä arvoja sekä rajoituksia Erikssonin ja Penkerin (2000, 187–219) mukaan. Stereotyyppejä tarkastellaan Erikssonin ja Penkerin (2000, 187–219) lisäksi myös Bernerin, Glinzin ja Joosin (1999) mukaan.

2.3.1 Ominaisuudet ja nimetyt arvot

Ominaisuutta (property) käytetään yleisesti tarkoittamaan jotakin elementtiin liittyvää arvoa. Ominaisuuksilla lisätään elementteihin tietoa, jota voivat

hyödyntää sekä tietokoneet että ihmiset. Ihmisille suunnattu tieto voi olla esimerkiksi mallin tekijä tai viimeinen muutospäivä. Tietokoneet voivat hyödyntää esimerkiksi tietoa siitä, millaista koodia täytyy generoida.

Nimetty arvo (tagged value) on ominaisuuden tarkka määrittely nimi-arvo-pariksi, jossa nimeä kutsutaan nimekkeeksi (tag). Jokainen *nimike* vastaa tietyn tyyppistä ominaisuutta. Nimikkeitä voidaan soveltaa kaikenlaisiin elementteihin. Sekä nimike että arvo kuvataan merkkijonoina. Ominaisuuden merkintätapa on seuraava:

{nimike = arvo} tai {nimike1 = arvo1, nimike2 = arvo2…} tai {nimike}

Ominaisuudet kuvataan nimikkeenä ja arvona aaltosulkujen sisällä. Ominaisuuksia ja nimettyjä arvoja voidaan käyttää kaavioissa ja erillisissä dokumenteissa. Kaavioissa ominaisuudet merkitään lähelle sitä elementtiä, johon ne liittyvät.

UML:ssä on monia valmiiksi määriteltyjä nimettyjä arvoja elementeille. Standardoituja nimettyjä arvoja ovat esimerkiksi: jälkiehto (postcondition), esiehto (precondition) sekä abstrakti (abstract).

UML:ään voidaan luonnollisesti määritellä myös omia nimettyjä arvoja. Pääasiallisena tarkoituksena on tuoda jotakin lisämerkitystä elementtiin, johon nimetty arvo liitetään. Yksi esimerkki mahdollisesta käyttäjän määrittelemästä nimetystä arvosta voisi olla standardialgoritmi. Tässä tapauksessa nimike olisi standardialgoritmi ja arvo olisi algoritmin nimi. Tätä nimettyä arvoa voitaisiin käyttää koodin generoinnin yhteydessä.

2.3.2 Rajoitukset

Rajoitus (constraint) on elementtiin liittyvä semanttinen ehto tai rajoite. Rajoitus ei voi lisätä uutta semantiikkaa, se voi ainoastaan rajoittaa elementin olemassa olevaa semantiikkaa. Yksi rajoitus soveltuu vain yhden tyyppiseen elementtiin.

Rajoitus voi liittyä kuitenkin useampaan elementtiin, jos ne ovat samantyyppisiä. Rajoituksen merkintätapaa kuvaa seuraava esimerkki:

{henkilö.esimies.palkka >= henkilö.apulainen.palkka}

Rajoitukset kuvataan aaltosulkujen sisällä, joko kaaviossa tai erikseen. Jos rajoitus liitetään mallinnuselementtiin, jolla on vastaava näkymäelementti, se voidaan näyttää kaaviossa näkymäelementin vieressä. Jos rajoitukseen liittyy useampi samantyyppinen elementti, rajoitus tulisi näyttää katkoviivan vieressä, joka kulkee kyseisten elementtien yli.

UML:ssä on 18 valmiiksi määriteltyä rajoitusta. Tällaisia standardoituja rajoituksia ovat esimerkiksi: yleinen (global), täydellinen (complete), erillinen (disjoint), epätäydellinen (incomplete) sekä päällekkäinen (overlapping).

UML:ssä voidaan luonnollisesti käyttää myös omia rajoituksia. Käyttäjän täytyy määritellä, mihin elementtiin rajoitusta voidaan soveltaa. Myös rajoituksen semanttinen merkitys on määriteltävä. Yksi esimerkki itse määritellystä rajoituksesta voisi olla yksittäinen (singleton). Tätä rajoitusta voidaan soveltaa luokkiin. Tällöin yksittäisellä luokalla voi olla vain yksi olio kerrallaan. Yksittäistä luokkaa voidaan käyttää vaikka ilmentymälaskurina, jolloin kyseinen luokka toteutetaan luokkatason attribuuttina.

2.3.3 Stereotyypit

Erikssonin ja Penkerin (2000, 205) mukaan stereotyyppiä (stereotype) käytetään UML:ssä määriteltyjen elementtien yleisen semantiikan erikoistamiseen ja laajentamiseen. Ominaisuuksien lisäämisen tai rajoitusten määrittelyn sijaan olemassa olevaan mallinnuselementtiin voidaan lisätä uutta semantiikkaa. Stereotyyppien avulla voidaan laajentaa elementin semantiikkaa, mutta ei rakennetta. Berner, Glinz ja Joos (1999) määrittelevät stereotyypin hieman väljemmin. Heidän määritelmänsä mukaan *stereotyyppillä* tarkoitetaan

mallinnuskielen yhteydessä hyvin muodostettua mekanismia, jonka avulla käyttäjä voi määritellä kielen elementeille laajennuksia, parannuksia sekä uudelleenmäärityksiä muokkaamatta suoraan kielen metamallia. Tässä tutkielmassa käytetään edellä kuvattua stereotyypin määritelmää.

Kun elementtiin liitetään stereotyyppi, sen semantiikka ohittaa elementin oman semantiikan, jolloin elementtiä voidaan pitää uutena elementtinä, joka perustuu alkuperäiseen elementtiin. Stereotyypit ovat yleistettäviä elementtejä. Niitä voidaan siis erikoistaa tai yleistää. Stereotyypin merkintätapa on seuraava:

<<stereotyypin nimi>>

Stereotyypin nimen ympärillä käytetään kaksoiskulmasulkuja. Edellä kuvattu merkintätapa on itse asiassa näkymäelementti. Yleensä näkymäelementti sijoitetaan sen elementin nimen yläpuolelle tai eteen, johon stereotyyppi perustuu. Stereotyyppiin voidaan liittää myös graafinen kuvake.

Yleensä stereotyyppejä sovelletaan luokkiin, tyyppeihin, suhteisiin, solmuihin, komponentteihin ja operaatioihin. UML:ssä on yli 40 valmiiksi standardoitua stereotyyppiä. Tällaisia stereotyyppejä ovat muun muassa: toimija (actor), metaluokka (metaclass), ohjaus (control), raja (boundary) sekä tieto (entity).

Myös käyttäjät voivat lisätä uusia stereotyyppejä. Stereotyyppiä määritellessä täytyy kuvata, mihin elementtiin stereotyyppi perustuu, lisäksi täytyy kuvata stereotyypin lisäämät ja tarkentamat semantiikat. Yksi esimerkki käyttäjän määrittelemästä stereotyypistä voisi olla ohjaaja-stereotyyppi, joka soveltuu luokkiin. Ohjaajaluokka vahtii tiettyä resurssia ja sallii vain määrätyn määrän käyttäjiä resurssia käyttämään samaan aikaan.

Stereotyypit ovat hyvin tehokas laajennusmekanismi, sillä sekä nimetyt arvot että rajoitukset voidaan korvata stereotyypeillä. On myös yleistä, että stereotyyppien yhteydessä esitetään sekä nimettyjä arvoja että rajoituksia. (Eriksson & Penker 2000, 187–219)

Stereotyypit antavat mahdollisuuden muokata UML:ää tavalla, joka ulottuu vähäisistä notaatiota koskevista muutoksista aina peruskielen uudelleenmäärittelyyn. Koska stereotyypit ovat tehokas laajennusmekanismi, niiden käytöllä on hyvät sekä huonot puolensa. Hyvä puoli on se, että stereotyyppien avulla voidaan luoda mallinnuskieliä, jotka ovat joustavia sekä ilmaisuvoimaisia tietyille sovellusalueille. Huono puoli on se, että suunnittelematon ja liiallinen stereotyyppien käyttö voi johtaa ristiriitaisen, hankalakäyttöisen sekä vaikeasti ymmärrettävän kielen syntyyn. (Berner, Glinz & Joos 1999).

Bernerin, Glinzin ja Joosin (1999) mukaan stereotyyppien suunnittelu on vaikea ja haastava tehtävä. He määrittelevät neljä yleistä vaatimusta, jotka hyvin suunniteltujen stereotyyppien tulisi täyttää:

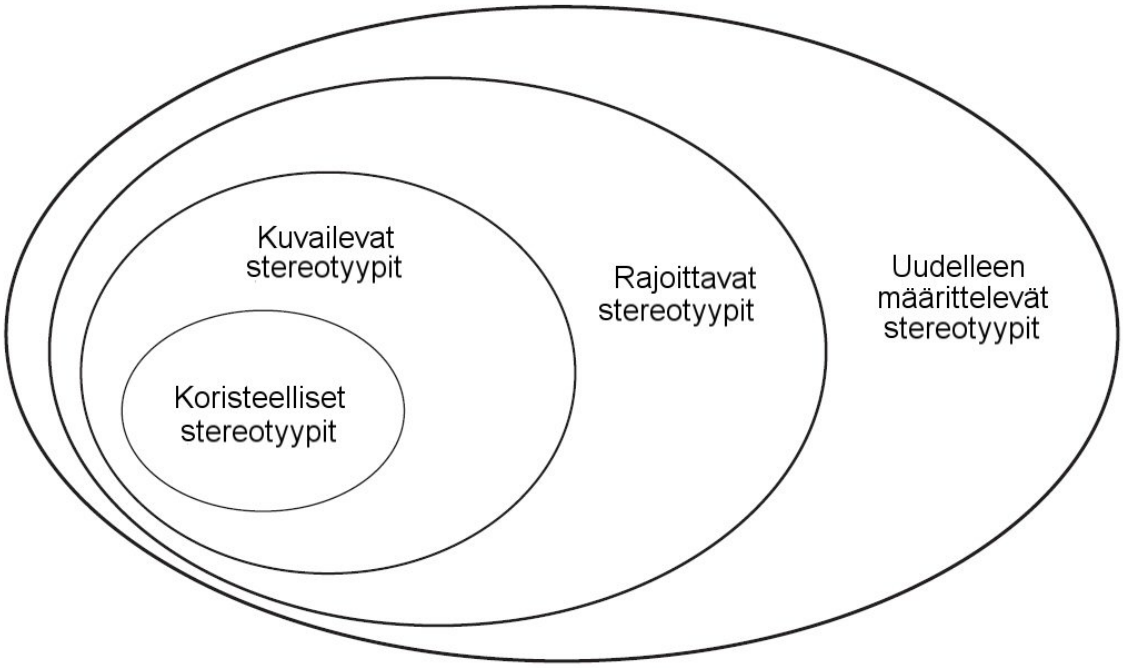
- Jokaisen stereotyypin tulisi olla hyvin määritelty. Määritelmän tulisi olla tarvittaessa formaali, mutta kuitenkin myös ymmärrettävä.

- Jokaisen stereotyypin tulisi olla hyödyllinen. Tällä tarkoitetaan sitä, että stereotyypin tulisi määrittää uusi käsite tai piirre, joka ei sisälly peruskieleen. Stereotyypin tulisi myös helpottaa mallinnuskielen käyttöä halutulla sovellusalueella.

- Stereotyyppien tulisi muodostaa johdonmukainen joukko. Stereotyyppi ei saa olla yhteensopimaton muiden stereotyyppien kanssa, ellei sitä ole selkeästi määritelty yhteensopimattomaksi joidenkin tiettyjen stereotyyppien kanssa.

- Stereotyyppien tulisi olla ortogonaalisia. Tällä tarkoitetaan sitä, että mikäli jokin stereotyyppi määrittelee tietyn käsitteen tai piirteen, tätä samaa piirrettä ei pidä pystyä mallintamaan minkään toisen stereotyypin avulla.

Berner, Glinz ja Joos (1999) luokittelevat stereotyypit neljään luokkaan. Heidän luokittelunsa käy ilmi kuviosta 3. Luokittelu perustuu siihen, kuinka voimakkaasti stereotyyppien avulla voidaan vaikuttaa mallinnuskielen syntaksiin ja semantiikkaan. Luokittelu noudattaa sisällyttämishierarkiaa, eli kuviossa 3 ulomman kategorian mukaiset stereotyypit sisältävät myös sisempien kategorioiden ominaisuudet. Seuraavassa käsitellään stereotyyppiluokitusta Bernerin, Glinzin ja Joosin (1999) mukaan. Määritelmien lisäksi joitakin stereotyyppiluokkia kuvataan myös esimerkkien avulla. Jokaisen kategorian stereotyypeistä esitetään myös hyviä ja huonoja puolia.



KUVIO 3. Stereotyyppiluokat (Berner, Glinz & Joos 1999, 252).

Koristeelliset stereotyypit (decorative stereotypes) muuntavat mallinnuskielen elementin syntaksia, eivätkä tee mitään muuta. Näiden stereotyyppien avulla voidaan muuttaa elementtien visuaalista ulkoasua, mutta ei voida lisätä tietoa tai uusia käsitteitä peruskieleen. Koristeellisten stereotyyppien hyvänä puolena voidaan pitää sitä, että niiden määritteleminen on helppoa ja vaivatonta. Koristeellisten stereotyyppien avulla voidaan parantaa, mutta myös joissain tilanteissa heikentää mallien ymmärrettävyyttä. Huono puoli on se, että

koristeelliset stereotyypit eivät tuo lisävoimaa mallinnuskieleen, ne ovat pelkkää syntaktista sokeria.

Kuvailevat stereotyypit (descriptive stereotypes) laajentavat tai muuntavat mallinnuskielen elementin abstraktia syntaksia ja määrittelevät uuden elementin pragmatiikan. Peruskielen semantiikka pysyy muuttumattomana. Tällaisten stereotyyppien avulla ei voida määritellä myöskään semanttisia rajoitteita. Kuvailevien stereotyyppien tyypillisimpiä käyttökohteita ovat olioiden luokittelut ja standardoidut huomautukset. Yksi esimerkki olioiden luokittelusta on luokkien jakaminen liiketoiminta-, ohjaus-, sekä käyttöliittymäluokkiin. Esimerkki standardoidusta huomautuksesta on stereotyyppi kokoonpanoinformaatio, joka sisältää tiedot tekijästä, luontipäivästä sekä viimeisestä muokkausajankohdasta. Huomattavaa on, että nimetty arvo vastaa yksinkertaista kuvaavaa stereotyyppiä. Tiedot tekijästä, luontipäivästä ja viimeisestä muokkausajankohdasta voitaisiin mallintaa myös käyttäjän määrittämien nimettyjen arvojen avulla. Hyvänä puolena kuvailevilla stereotyypeillä on se, että niiden avulla mallinnuskieltä voidaan rikastaa kontrolloidulla tavalla. Kuvailevien stereotyyppien määrittely on vaivatonta. Huonona puolena tällaisilla stereotyypeillä on se, että niiden muutosvoima on rajoittunut puhtaasti syntaktisiin ominaisuuksiin.

Rajoittavat stereotyypit (restrictive stereotypes) ovat muuten samanlaisia kuin kuvailevat stereotyypit, mutta niiden avulla voidaan määritellä myös uuden elementin semantiikka. Tyypillisesti semantiikan avulla määritetään rajoituksia esitetyille elementeille. Rajoittavien stereotyyppien avulla ei kumota peruskielen semantiikkaa, niiden avulla ainoastaan laajennetaan sitä. Rajoittavien stereotyyppien hyvä puoli on se, että ne tuovat todellista voimaa mallinnuskieleen. Näiden stereotyyppien kyvykkyys menee paljon edelle verrattuna kuvaileviin ja koristeellisiin stereotyyppeihin. Huonona puolena tämän luokan stereotyypeillä on niiden määrittelyn vaikeus. Rajoittavien stereotyyppien määrittely vaatii syvällistä tietämystä stereotyypin halutuista

ominaisuuksista, peruskielestä, mallinnuskielten suunnitteluperiaatteista sekä metakielestä, jonka avulla stereotyypin semantiikka määritellään.

Uudelleen määrittelevät stereotyypit (redefining stereotypes) määrittelevät mallinnuselementin muuttaen sen alkuperäistä semantiikkaa. Syntaksia koskien uudelleen määrittelevät stereotyypit käyttäytyvät samalla tavoin kuin rajoittavat stereotyypit. Koristeellisten, kuvailevien sekä rajoittavien stereotyyppien ilmentymät ovat valideja stereotyypitetyn kielen elementtejä. Uudelleen määrittelevien stereotyyppien kohdalla näin ei ole. Uudelleen määrittelevän stereotyypin avulla voidaan esitellä mallinnuselementti, joka ei koske mitään peruskielen elementtiä. Tähän luokkaan kuuluvien stereotyyppien hyvänä puolena voidaan pitää sitä, että niiden avulla mallinnuskieleen voidaan luoda sellaisia syvällisiä sekä radikaaleja muutoksia, joiden tekeminen muuten olisi mahdotonta. Uudelleen määrittelevien stereotyyppien määrittely on vielä astetta haastavampaa verrattuna rajoittavien stereotyyppien määrittelyyn. Tällaisten stereotyyppien tekemistä tulisi välttää.

Bernerin, Glinzin ja Joosin (1999) mukaan koristeellisten ja uudelleen määrittelevien stereotyyppien käyttö on hyvin ongelmallista. Syntaksin varioimista ja perustavanlaatuisen peruskielen semantiikan uudelleen määrittämistä tulisi välttää. Käytännön kannalta kaikkein käyttökelpoisimpia ovat kuvailevat ja rajoittavat stereotyypit. Berner, Glinz ja Joos (1999) painottavat sitä, että jokainen uusi stereotyyppi täytyy määritellä asianmukaisesti, olipa se sitten mihin tahansa kategoriaan kuuluva.

2.4 UML:n laajennustavat

On havaittu, että on vaikeaa kehittää yhtä mallinnuskieltä, joka tukee suoraan eri sovellusalueiden tarpeita. UML:n tavoitteena on olla kuitenkin yhtä aikaa mahdollisimman yleiskäyttöinen ja tukea eri sovellusalueiden tarpeita. Kompromissina UML:stä tehtiin laajennettava, jolloin sen käyttäjät voivat räätälöidä kieltä vaatimustensa mukaiseksi esittelemällä sovellusaluekohtaisia

mallinnuselementtejä. (Schleicher & Westfechtel 2001) Tässä kohdassa esitellään kaksi lähestymistapaa laajentaa UML:ää. Nämä tavat ovat UML:n sisäisiin laajennusmekanismeihin perustuva laajentaminen ja metamallipohjainen laajentaminen.

2.4.1 Sisäisiin laajennusmekanismeihin perustuva laajentaminen

OMG:n (2003a, 131–133) mukaan UML:ää pystytään laajentamaan kahdella tavalla. Ensimmäisen tavan mukaan käytetään UML:n sisäänrakennettuja laajennusmekanismeja, jotka on esitelty kohdassa 2.3. Laajennusta, joka on suunniteltu tietylle sovellusalueelle tai tiettyyn tarkoitukseen ja joka laajentaa UML:n metamallia käyttäen valittua joukkoa laajennusmekanismeja antamaan olemassa oleville mallinnuselementeille uusia merkityksiä, kutsutaan *profiiliksi*. Tästä laajennustavasta käytetään myös nimeä *kevyensarjan* (lightweight) laajennusmekanismi.

Dokumentissa OMG (1999) määritellään tarkemmat vaatimukset profiilille. Seuraavaksi käsitellään kyseisiä vaatimuksia:

- Profiilin täytyy erikoistaa standardia UML:n metamallia siten, että erikoistetut semantiikat eivät ole ristiriidassa standardin metamallin kanssa.

- Profiilissa voidaan käyttää vain UML:n sisäisiä laajennusmekanismeja.

- Täytyy olla mahdollista vaihtaa profiileja ja niiden mukaisia kaavioita mallinnusvälineiden välillä käyttäen olemassa olevia vaihtomekanismeja. Tämän takia profiilin tulisi organisoida elementit paketteihin ja profiilin osien keskinäiset suhteet tulisi mallintaa käyttäen UML:n suhteita.

- Profiilin täytyy kuvata UML:n metamallista osajoukko sovellettavien osien osalta.
- Profiilin täytyy viitata erikoisalaan liittyviin UML kirjastoihin, joissa mahdollisesti tarvittavat mallinnuselementit ovat ennalta määritelty.
- Profiileille täytyy voida tehdä tiettyjä toimenpiteitä, erityisesti niitä täytyy voida erikoistaa ja yhdistää.
- Täytyy olla mahdollista eritellä profiilin osia, kuten paketteja ja alipaketteja.
- Nimettyjen arvojen määritelmien tulisi olla formaaleja ja tyypitettyjä.
- Pitäisi olla mahdollista luoda UML:n laajennus, joka yhdistää profiileja ja mallikirjastoja yhdeksi loogiseksi kokonaisuudeksi.

Kuten huomaamme edellä mainittu laajennustapa asettaa tiettyjä rajoituksia, kuinka metamallia voidaan laajentaa. Esimerkiksi uusia metaluokkia tai meta-assosiaatioita ei tässä laajennustavassa voi lisätä metamalliin. Rajoitusten mukana seuraa kuitenkin myös joitakin hyötyjä, esimerkiksi UML-mallinnustyökalut tukevat yleensä profiileja (OMG 1999). Seuraavassa alakohdassa esitellään laajennustapa, jossa ei kyseisiä rajoituksia ole.

2.4.2 Metamallipohjainen laajentaminen

Kehitystyön tavoitteena on ollut, että UML:n laajentaminen onnistuu pelkästään UML:n sisäisten laajennusmekanismien avulla. Kuitenkin laajennuksia voi tehdä myös lisäämällä uusia metaluokkia tai muita metarakenteita metamalliin. Tämä laajennustapa kuvataan MOF (Meta Object Facility) määrityksessä (OMG 2002) ja siitä käytetään nimeä *raskaansarjan* (heavyweight) laajennusmekanismi. (OMG 2003a, 131–133)

Koska UML:n metamalli itsessään on MOF-metametamallin instanssi, UML:n metamallin laajentaminen raskaansarjan laajennusmekanismia käyttäen tarkoittaa siis uuden MOF-metametamallin instanssin luomista. Näin UML:n metamallia laajennettaessa itse asiassa määritellään kokonaan uusi kieli (Schleicher & Westfechtel 2001). Alhir (1998) käyttääkin raskaansarjan laajennuksesta osuvasti nimitystä *UML:n muunnelma* (variant). Tässä raportissa käytetään kaikentyyppisistä laajennuksista yhteistä termiä *UML:n laajennus*. Mikäli halutaan erikseen korostaa, että kyseinen laajennus on profiili tai UML:n muunnelma, silloin käytetään kyseisiä termejä.

Jotkut tutkijat, muun muassa Schleicherin ja Westfechtelin (2001) sekä D'Souza, Sane ja Birchenough (1999) ovat antaneet suosituksiaan siitä, kuinka UML:ää kannattaisi laajentaa. Schleichnerin ja Westfechtelin (2001) mukaan UML:n sisäänrakennetut laajennusmekanismit ovat melko rajalliset. Tämän takia on jouduttu esittämään monia laajennuksia, joissa laajennetaan UML:n metamallia lisäämällä siihen uusia metaluokkia sekä meta-assosiaatioita. Heidän mukaansa tämän laajennustavan etu on mahdollisuus käyttää luokkakaavioita määriteltäessä rakenteisia rajoitteita. Heidän mukaansa suositeltava tapa laajentaa metamallia on lisätä metamalliin vain sellaisia uusia metaluokkia, jotka periytyvät jostakin alkuperäisen UML:n metamallin luokasta. Myös uusien meta-assosiaatioiden olisi suotavaa perustua jo olemassa oleviin assosiaatioihin. Tällaisesta metarakenteiden lisäämistavasta he käyttävät nimitystä kontrolloitu (controlled). Tavasta, jossa lisätään aivan vapaasti mitä tahansa alkuperäisestä UML:n metamallista riippumattomia uusia metaluokkia sekä meta-assosiaatiota, he käyttävät nimitystä kontrolloimaton (uncontrolled).

Schleicher ja Westfechelt (2001) määrittelevät myös kaksi erillistä UML:n metamallin laajentamistapaa. Lähestymistapojen mukaisia laajennuksia he nimittävät tavallisiksi metamallilaajennuksiksi (regular metamodel extensions) ja rajoittaviksi metamallilaajennuksiksi (restrictive metamodel extensions). Kummassakin lähestymistavassa voidaan käyttää sekä kontrolloitua että

kontrolloimatonta tapaa lisätä metaluokkia sekä meta-assosiaatioita. Suurimpana erona laajennustapojen välillä on se, että tavallisissa metamallilaajennuksissa voidaan vain lisätä mallinnuselementtejä, kun taas rajoittavissa metamallilaajennuksissa voidaan myös poistaa olemassa olevia elementtejä. Tässä tapauksessa mallinnuselementeistä tehdään abstrakteja, jolloin elementtien ilmentymät voidaan kieltää.

UML:ään on esitetty lukuisia laajennuksia, ja todennäköisesti uusia laajennuksia esitetään tulevaisuudessa aina vain enemmän. Tähän on syynä muun muassa UML:n koko ajan kasvava suosio eri sovellusalueilla. Cook, Kleppe, Mitchell ym. (1999) ovatkin keksineet osuvan termin. He puhuvat syntyneestä UML-kieliperheestä. He ovat myös havainneet, että vaikka laajennuksia on esitetty lukuisia, ei ole olemassa kuitenkaan yhteistä hyväksyttyä tapaa kuvailla niiden syntaksia tai semantiikkaa. Yhteinen kuvaustapa helpottaisi huomattavasti muun muassa laajennusten mukaisten kaavioiden lukemista. Cook, Kleppe, Mitchell ym. (1999) ehdottavatkin yhden tavan kuvata UML-kieliperheen jäsenten syntaksia ja semantiikkaa. Heidän määritelmänsä sisältää pitkän listan huomioon otettavia seikkoja, ja he jakavat määrittelemänsä paketteihin. Tämän tarkemmin heidän ehdotustaan ei pystytä tässä yhteydessä käsittelemään.

2.5 Yhteenveto

Tässä luvussa tarkasteltiin UML:ää sen laajentamisen näkökulmasta. Ensiksi esiteltiin UML:n semantiikkaa, jonka käsittely aloitettiin UML:n nelitasoisen metamalliarkkitehtuurin kuvaamisella. Tämän jälkeen keskityttiin kuvaamaan metamallitasoa ja erityisesti elementin käsitettä, jonka huomattiin olevan keskeinen käsite UML:n laajentamisen näkökulmasta. Seuraavaksi kuvattiin UML:n sisäisiä laajennusmekanismeja, jotka ovat ominaisuudet ja nimetyt arvot, rajoitukset sekä stereotyypit. Laajennusmekanismien tarkastelussa erityishuomio kiinnitettiin stereotyypin käsitteeseen. Luvun lopuksi esiteltiin

UML:n laajennustavat. Havaittiin, että on olemassa kaksi pääasiallista tapaa laajentaa UML:ää. UML:n kehittäjien suosittelema tapa on käyttää pelkästään sisäisiä laajennusmekanismeja. Tällaisesta laajennuksesta käytetään nimeä profiili. Mikäli UML:n sisäiset laajennusmekanismit eivät ole riittäviä, joudutaan tekemään lisäyksiä UML:n metamalliin. Tällaisesta laajennuksesta käytetään nimeä UML:n muunnelma. Tässä luvussa havaittiin myös, että UML:n laajennusten semantiikan tai syntaksin esityksille ei ole yhtenäistä kuvaustapaa. Tämä seikka vaikeuttaa laajennusten tarkastelua.

Luvun keskeinen tarkoitus oli luoda pohjaa laajennusten kartoitukselle ja arvioinnille. Seuraavassa luvussa tehdään UML-laajennusten kartoitus, jossa laajennuksia vertaillaan karkealla tasolla. Tarkoituksena on muun muassa selvittää, mihin kaavioihin laajennukset kohdistuvat ja mitä laajennustapaa laajennuksissa käytetään.

3 LAAJENNUSTEN KARTOITUS

Tässä luvussa tehdään UML:n laajennusten kartoitus. Luvun ensimmäisessä kohdassa esitetään rajaus, jonka mukaan vertailtavat laajennukset valitaan. Seuraavassa kohdassa esitetään laajennuksista kartoitettavat seikat. Tämän luvun tärkeimpänä antina raportoidaan lopuksi kartoituksen tulokset, joiden perusteella päätetään myös mihin laajennuksiin keskitytään yksityiskohtaisessa analyysissä. Luku päättyy yhteenvetoon.

3.1 Laajennusten valinta

Kirjallisuudessa on esitetty kymmeniä, ellei satoja UML:n laajennuksia. Koska laajennuksia on esitetty näin paljon sekä hyvin erilaisille alueille, on välttämätöntä rajata osa laajennuksista tutkimuksen ulkopuolelle. Tässä luvussa esitettävä laajennusten kartoitus ei pyri olemaan siis täysin kattava. Osittain tähän on syynä myös se, että tutkielman kirjoittajalla ei ole mahdollisuutta saada käsiinsä kaikkea alasta julkaistua kirjallisuutta. Tutkimukseen etsittiin laajennuksia kaikista mahdollisista tieteellisistä julkaisuista, jotka olivat tutkijan saatavilla. Seuraavana kuvataan, millä kriteereillä laajennusten rajaus on tehty.

Tässä luvussa otetaan tarkasteluun periaatteessa kaikki laajennukset, jotka on suunniteltu koskemaan jotain nimettyä sovellusaluetta. Sovellusalueen käsite on hyvin yleisesti käytetty tietojärjestelmätieteen alalla. Sen yksiselitteinen määrittely on kuitenkin melko hankalaa. Glass ja Vessey (1998) tarkoittavat sovellusalueen kattavan yhtenäisen joukon ongelmia, jotka juontuvat yleensä ongelman luonteesta. Glass ja Vessey (1995) esittävät tekemässään tutkimuksessa joitakin mahdollisuuksia luokitella sovellusalueita. Heidän mukaansa sovellusalueita voidaan luokitella muun muassa teollisuudenalan tai sovellustyypin perusteella. Heidän mielestään ei ole kuitenkaan mahdollista tehdä yleispätevää luokittelua sovellusalueista, koska tieteenala on vielä varsin

kypsymätön. Tässä tutkimuksessa tarkoitetaan *sovellusalueella* tiettyä ohjelmisto- tai järjestelmätyyppiä sekä tietojärjestelmän suunnitteluparadigmaa, jota yhdistää yhtenäinen joukko ongelmia. Ohjelmisto- tai järjestelmätyyppejä ovat esimerkiksi reaaliaikajärjestelmä ja WWW-sovellus. Esimerkki paradigmasta on muun muassa agentti-perusteisuus.

Laajennusesityksiä etsittäessä löydettiin useita kymmeniä artikkeleita, jotka käsittelevät tietylle sovellusalueelle esitettyä laajennusta. Koska löydettyjen artikkelien määrä oli näin suuri, jouduttiin myös tätä joukkoa hieman supistamaan. Seuraavassa kerrotaan tästä tarkemmin.

Huomioitavaa laajennusesityksissä on se, että monet niistä ovat kehitteillä. Useista laajennuksista on esitetty paranneltuja versioita ja niitä kehitetään edelleen. Tähän tutkimukseen valituista laajennuksista on pyritty löytämään alkuperäisten tekijöiden mahdollisimman uudet versiot. Joihinkin laajennuksiin on esitetty myös muiden tutkijoiden toimesta parannusehdotuksia, tai joidenkin tutkijoiden esittämät laajennukset perustuvat hyvin pitkälti aiemmin esitettyihin laajennuksiin. Tällaiset laajennukset on rajattu pääosin tutkimuksen ulkopuolelle. Edellä mainitun kaltaisten laajennusten vertailu ei olisikaan kovin mielekästä. Jäljelle jääneistäkin laajennuksista on jätetty osa tutkimuksen ulkopuolelle, koska niistä ei ole saatavilla tarpeeksi tietoa. Joitakin laajennuksia on jätetty analyysin ulkopuolella myös siksi, että ne esittävät hyvin vähän uutta verrattuna alkuperäiseen UML:ään. Siten laajennukset, joissa esitetään vain muutama uusi stereotyyppi tai keskitytään vain yhteen kaaviotyyppiin, on jätetty pääosin analyysin ulkopuolelle. Esimerkiksi käyttötapauskaaviosta on tehty kymmeniä muunnelmia. Mäkinen (2003) tarkastelee ansioituneesti kyseisiä muunnelmia pro gradu-tutkielmassaan.

UML:ää kohtaan on esitetty myös yleisluonteista kritiikkiä liittyen eri osa-alueisiin. Myös tämän takia UML:ää on laajennettu useisiin tarkoituksiin, riippumatta tietystä sovellusalueesta. Laajennukset, jotka eivät koske

suoranaisesti tiettyä sovellusaluetta on rajattu automaattisesti tutkielman ulkopuolelle. Tällaisia ovat esimerkiksi laajennukset, jotka koskevat liiketoiminnan mallinnusta (esim. Marshall 2000; Bastos, Dubugras & Ruiz 2002; Tyndale-Biscoe, Sims, Wood ym. 2002), käyttöliittymänsuunnittelua (esim. Pinheiro da Silva & Patton 2001, 2003) ja arkkitehtuurin suunnittelua (esim. Medvidovic, Rosenblum, Redmiles ym. 2002; Selonen & Xu 2003). UML:ää on sovitettu myös useisiin menetelmiin yhteensopiviksi. Pääosin tällaiset laajennukset (esim. Castro, Kolp & Mylopoulos 2002; Gomez, Cachero & Pastor 2001; Yim, Cho, Kim ym. 2000) rajataan kartoituksen ulkopuolelle. Kuitenkin joitakin menetelmän ohessa esitettyjä UML:n laajennuksia on otettu analyysiin mukaan. Tällaisia ovat tapaukset, joissa menetelmä kohdistuu selvästi tietylle sovellusalueelle ja menetelmän ohessa esitetään selkeä UML:n laajennus. UML:n sisältämään OCL:ään (Object Constraint Language) on myös ehdotettu laajennuksia (esim. Rammig 2002; Flake 2002; Flake & Muller 2002). Myös nämä laajennukset rajataan tutkielman ulkopuolelle, vaikka ne koskisivatkin nimettyä sovellusaluetta.

Tutkimuksen rajaus on muodostettu edellä kuvatulla tavalla, jotta kartoitukseen saadaan mahdollisimman yhtenäinen joukko laajennuksia. Näin mahdollistetaan laajennusten välinen vertailu. Toiseksi tällä tavalla rajattuna kartoitettavien laajennusten joukko pysyy hallittavan kokoisena. Kartoitukseen valittiin 20 laajennusta edellä määritellyin kriteerein.

3.2 Kartoitettavat asiat

Kartoitukseen valituista UML:n laajennuksista pyritään selvittämään tässä kohdassa määritellyt seikat. Ensimmäisenä tavoitteena on selvittää, mille sovellusalueille laajennuksia on suunniteltu. Joitakin laajennuksia voidaan käyttää useammalla sovellusalueella, mutta tässä kartoituksessa selvitetään laajennuskohtaisesti vain pääasiallinen sovellusalue, jolle laajennus on

suunniteltu. Myös laajennuksen nimi pyritään selvittämään. Kaikkia laajennuksia ei ole kuitenkaan nimetty.

Toiseksi tutkitaan, mihin UML:n kaavioihin laajennuksessa esitetyt muutokset kohdistuvat. Useissa laajennuksissa alkuperäisiä UML-kaavioita nimetään uudelleen ja tietystä kaaviotyypistä saatetaan johtaa useampia kaavioita. Uusia kaavioiden nimiä ei kuitenkaan esitellä tässä yhteydessä. Tarkoituksena on vain osoittaa kaaviot, joita muutokset koskevat. Tarkemman tason analyysissä tarkastellaan sinne valittujen laajennusten kaavioita tarkemmin. Mikäli laajennuksessa esitellään kokonaan uusi kaavio, joka ei perustu mihinkään standardin UML:n kaavioon, se tuodaan erikseen esille.

Kolmanneksi selvitetään, millaisia uusia käsitteitä tai rakenteita laajennuksessa esitetään. Useat laajennukset perustuvat joukkoon käsitteitä ja rakenteita, jotka on esitelty kyseiselle sovellusalueelle aikaisemmin esitetyssä menetelmässä, mallinnuskielessä tai mallissa. Tällöin kyseinen menetelmä mainitaan tässä yhteydessä. Pääsääntöisesti tällöin käsitteitä ei esitellä erikseen. Joissakin laajennuksissa esitellään niin suuri joukko käsitteitä, ettei niitä kaikkia ole mahdollista tarkastella tässä yhteydessä. Tarkemman tason analyysissä tarkastellaan, sinne valittujen laajennusten, rakennetta ja käsitteitä tarkemmin.

Neljäntenä seikkana selvitetään laajennuksen laajennustavat. Tällä tarkoitetaan UML:n sisäisiä laajennusmekanismeja, metamallipohjaista laajennustapaa tai muuten kaavioihin tehtyjä muutoksia. Jotkin laajennukset saattavat käyttää useampaa laajennustapaa. Laajennustavasta, joka ei perustu UML:n sisäisiin laajennusmekanismeihin eikä esitä muutoksia metamalliin, käytetään nimitystä ”muu laajennustapa”.

Viidentenä selvitetään laajennuksen esittämisperusteet. Yleensä perusteena on käytetty jotakin tiettyä standardin UML:n heikkoutta, jota paikkaamaan laajennus on esitetty.

Viimeiseksi selvitetään, millaisia todisteita laajennuksen soveltuvuudesta käytäntöön on esitetty. Mikäli jokin taho, kuten Rational tai OMG on julkaissut laajennuksen, voidaan olettaa, että laajennus soveltuu käytäntöön. Myös tunnetun CASE-välineen mallinnustukea voidaan pitää merkkinä käytäntöön soveltuvuudesta. Useimpien laajennusesitysten soveltuvuudesta käytäntöön ei voida kuitenkaan tietää. Usean laajennusesityksen yhteydessä esitetään yksittäinen tapaustutkimus, jossa esimerkinomaisesti näytetään, kuinka laajennusta voidaan soveltaa. Tätä ei voida kuitenkaan pitää riittävänä todisteena laajennuksen soveltumisesta käytäntöön laajemmin.

Edellä kuvattua yksityiskohtaisemmin ei tässä yhteydessä ole mahdollista käsitellä laajennuksia. Mainittujen seikkojen pohjalta voidaan kuitenkin suorittaa karkean tason vertailu laajennusten välillä. Seuraavassa kohdassa raportoidaan kartoituksen tulokset ja päätetään, mihin laajennuksiin keskitytään jatkossa.

3.3 Kartoituksen tulokset

Tiivistelmä kartoituksen tuloksista esitetään taulukossa 3. Laajennukset esitetään taulukossa sovellusalueittain. Taulukon sarakkeissa kuvattavat asiat on määritelty edellisessä kohdassa. Tässä kohdassa kuvataan tarkemmin keskeisiä havaintoja kartoituksesta.

UML:n laajennuksia on esitetty useille sovellusalueille. Kartoituksen perusteella eniten laajennuksia on esitetty seuraaville alueille: WWW-sovellukset, agentti-perusteiset sovellukset sekä tosiaikajärjestelmät. Myös tietokantasovelluksille on esitetty muutama laajennus. Muille sovellusalueille laajennusesityksiä on tehty huomattavasti vähemmän. Seuraavaksi tarkastellaan laajennuksia ensin sovellusaluekohtaisesti, tämän jälkeen käsitellään laajennusten laajennustapoja ja kerrotaan laajennusten soveltuvuudesta käytäntöön. Viimeiseksi päätetään, minkä sovellusalueen laajennuksia arvioidaan tarkemmin.

TAULUKKO 3. UML:n laajennusten kartoitus.

Sovellusalue ja laajennuksen nimi	**Lähteet**	**Kaaviot, joita muutokset koskevat**	**Laajennuksen perusta, käsitteet sekä rakenteet**	**Laajennus-tavat**	**Laajennuksen esitysperusteet**	**Todisteet käytäntöön soveltuvuudesta**
WWW-sovellus, **W2000**	Baresi, Garzotto & Paolini 2001; Baresi, Garzotto & Maritati 2002.	Käyttötapauskaavio, tilakaavio, luokkakaavio, tapahtumakaavio, yhteistyökaavio.	Perustuu HDM:ään (Hypermedia Design Model)	Stereotyypit, nimetyt arvot, rajoitukset.	WWW-sovellusten suunnittelu on monimutkainen tehtävä, joka vaatii eri menetelmien ja tekniikoiden yhdistämisen.	Ei tietoa käytännön soveltuvuudesta.
WWW-sovellus, **WAE**	Conallen 1999a, 1999b, 2000.	Esiteltyjen stereotyyppien esisijaiset käyttökohteet ovat: Luokkakaavio, tapahtumakaavio sekä komponenttikaavio.	Esittelee useita käsitteitä stereotyyppien muodossa.	Stereotyypit, nimetyt arvot, rajoitukset.	WWW-sovellusten suunnittelu on monimutkainen tehtävä, jota ei voida tehdä perus UML:llä. Laajennuksessa keskitytään erityisesti WWW-sovelluksen arkkitehtuurin mallintamiseen.	Rationalin julkaisema profiili. Rational Rose-väline tarjoaa vastaavan mallinnustuen.
WWW-sovellus, **UWE**	Baumeister, Koch & Mandel 1999; Koch, Baumeister, Hennicker & Mandel 2000; Hennicker & Koch 2001a.	Luokkakaavio, oliokaavio, tapahtumakaavio.	Perustuu OOHDM-menetelmään (Object-Oriented Hypermedia Design Method).	Stereotyypit, nimetyt arvot, rajoitukset.	Mallinnettaessa WWW-sovelluksia tarvitaan myös navigointi ja käyttöliittymämalleja, joita UML ei tarjoa.	Ei tietoa käytännön soveltuvuudesta.
Multimedia, **OMMMA-L**	Sauer & Engels 1999, 2001.	Luokkakaavio, tapahtumakaavio, tilakaavio, *esityskaavio* (presentation diagram).	Perustuu MVC-malliin (Model-View-Controller).	Stereotyypit, muutokset metamalliin.	UML:n puutteet käyttöliittymän ja ajasta riippuvan toiminnan mallintamisessa.	Laajennusta on sovellettu erilaisissa teollisuusprojekteissa, esimerkiksi autoteollisuudessa.

(jatkuu)

TAULUKKO 3. UML:n laajennusten kartoitus. (jatkuu)

Sovellusalue ja laajennuksen nimi	**Lähteet**	**Kaaviot, joita muutokset koskevat**	**Laajennuksen perusta, käsitteet sekä rakenteet**	**Laajennus-tavat**	**Laajennuksen esitysperusteet**	**Todisteet käytäntöön soveltuvuudesta**
Agentti-perusteinen sovellus, **AUML**	Odell, Parunak & Bauer 2000; Bauer, Muller, Odell 2001; AUML web site.	Tapahtumakaavio, yhteistyökaavio, toimintokaavio, luokkakaavio, hajautuskaavio, tilakaavio.	Sisältää useita agentti-perusteisia mallinnuskäsitteitä.	Rajoitteet, stereotyypit, muutokset metamalliin.	UML:n puutteet mallintaa agentti-perusteisia järjestelmiä. Olio- sekä agentti-perusteinen mallintaminen eroavat perusteiltaan.	Ei tietoa käytännön soveltuvuudesta. AUML:stä työstetään parhaillaan FIPA:n standardia.
Agentti-perusteinen sovellus, **MESSAGE/UML**	Caire, Leal, Chainho ym. 2002; MESSAGE web site.	Luokkakaavio, tapahtumakaavio.	Sisältää useita agentti-perusteisia mallinnuskäsitteitä.	Stereotyypit, muutokset metamalliin.	UML:n puutteet mallintaa agentti-perusteisia järjestelmiä. Olio- sekä agentti-perusteinen mallintaminen eroavat perusteiltaan.	Ei tietoa käytännön soveltuvuudesta.
Agentti-perusteinen sovellus	Bergenti & Poggi 2000.	Luokkakaavio, yhteistyökaavio.	Sisältää muutamia agentti-perusteisia mallinnuskäsitteitä.	Stereotyypit.	Agentti-perusteisuus esittelee uuden abstraktiotason ohjelmistotuotantoon. UML:llä ei ole mahdollista mallintaa tätä abstraktiotasoa.	Ei tietoa käytännön soveltuvuudesta.
Agentti-perusteinen sovellus	Kavi, Aborizka & Kung 2002; Kavi, Kung, & Bharnbhani ym. 2003	Käyttötapauskaavio, luokkakaavio, tapahtumakaavio, toimintokaavio, tilakaavio.	Perustuu BDI (Belief Desire Intention) formalismiin. Sisältää useita agentti-perusteisia mallinnuskäsitteitä.	Muu laajennustapa.	UML:n puutteet mallintaa agentti-perusteisia järjestelmiä. Olio- sekä agentti-perusteinen mallintaminen eroavat perusteiltaan.	Ei tietoa käytännön soveltuvuudesta.

(jatkuu)

TAULUKKO 3. UML:n laajennusten kartoitus. (jatkuu)

Sovellusalue ja laajennuksen nimi	**Lähteet**	**Kaaviot, joita muutokset koskevat**	**Laajennuksen perusta, käsitteet sekä rakenteet**	**Laajennus-tavat**	**Laajennuksen esitysperusteet**	**Todisteet käytäntöön soveltuvuudesta**
Agentti-perusteinen sovellus	Flake, Geiger & Küster 2001.	Käyttötapauskaavio, luokkakaavio, oliokaavio, tilakaavio.	Perustuu BDI (Belief Desire Intention) formalismiin. Sisältää agentti-perusteisia mallinnuskäsitteitä.	Stereotyypit.	UML:n puutteet mallintaa agentti-perusteisia järjestelmiä visuaalisilta perusteiltaan. Keskitytään agentin sisäisen toiminnan mallintamiseen.	Ei tietoa käytännön soveltuvuudesta.
Ryhmätyöohjelmisto	Rubart & Dawabi 2002.	Esitettyjä laajennusmekanis-meja voidaan käyttää kaikissa kaavioissa.	Sisältää joitakin ryhmätyöohjelmistoille ominaisia käsitteitä, lähinnä stereotyyppien muodossa.	Stereotyypit, nimetyt arvot, rajoitteet.	Ryhmätyöohjelmistoille ominaisia mallinnuspiirteitä ei ole esitetty UML:ssä. Tarvitaan tukea jaetun tiedon mallintamiselle.	Ei tietoa käytännön soveltuvuudesta.
Suojattu järjestelmä, **UMLsec**	Jurjens 2002.	Toimintokaavio, tilakaavio, tapahtumakaavio, luokkakaavio, oliokaavio, hajautuskaavio.	Sisältää useita suojatuille järjestelmille ominaisia mallinnuskäsitteitä.	Stereotyypit, nimetyt arvot rajoitukset.	UML tarjoaa sopivan viitekehyksen suojaus ja turvallisuus seikkojen esiin tuomiselle.	Laajennusta on sovellettu onnistuneesti muutamissa talousalan sovelluksissa.
Tila- ja aikatietoja sisältävä sovellus, **STUML**	Price, Srinivasan & Ramamohanarao 1999a, 1999b.	Luokkakaavio.	Esitellään neljä käsitettä erityyppiselle tiedolle: tilaa koskeva (spatial), ajallinen (temporal), tilaa ja aikaa koskeva (spatiatemporal) ja ryhmä (group).	Stereotyypit.	Tila ja aikatietojen esittäminen UML:llä on ongelmallista.	Ei tietoa käytännön soveltuvuudesta.

(jatkuu)

TAULUKKO 3. UML:n laajennusten kartoitus. (jatkuu)

Sovellusalue ja laajennuksen nimi	**Lähteet**	**Kaaviot, joita muutokset koskevat**	**Laajennuksen perusta, käsitteet sekä rakenteet**	**Laajennus-tavat**	**Laajennuksen esitysperusteet**	**Todisteet käytäntöön soveltuvuudesta**
Oliorelationaalinen tietokantasovellus	Marcos, Vela & Cavero 2001.	Luokkakaavio.	Esitetyt käsitteet ja rakenteet pohjautuvat SQL:1999:n sekä Oracle8i:n.	Stereotyypit, nimetyt arvot, rajoitteet.	UML ei tarjoa mallinnustukea oliorelationaalisen tietokannan suunnitteluun.	Ei tietoa käytännön soveltuvuudesta.
Relationaalinen tietokantasovellus, **UML Data Modeling Profile**	Gornik 2002.	Komponenttikaavio, luokkakaavio.	Sisältää useita tietokantasovelluksille ominaisia mallinnuskäsitteitä.	Stereotyypit, nimetyt arvot.	UML ei tarjoa mallinnustukea relationaalisen tietokannan suunnitteluun.	Rationalin julkaisema profiili. Rational Roseen sisältyvä Data Modeler-väline tarjoaa vastaavan mallinnustuen.
Relationaalinen tietokantasovellus, **UML Profile for Data Modeling**	Ambler 2002.	Luokkakaavio.	Sisältää useita tietokantasovelluksille ominaisia mallinnuskäsitteitä.	Stereotyypit, nimetyt arvot, rajoitteet.	UML ei tarjoa mallinnustukea tietokannan suunnitteluun.	Ei tietoa käytännön soveltuvuudesta.
Tosiaikajärjestelmä	Axelson 2001.	Luokkakaavio, tilakaavio.	Portti, vuo (flow), yhteys (connection).	Stereotyypit, rajoitteet.	Mahdollistaa myös järjestelmän fyysisen ympäristön mallintamisen.	Ei tietoa käytännön soveltuvuudesta.
Tosiaikajärjestelmä, **TURTLE**	Apvrille, Saqui-Sannes, Lohr ym. 2001; Lohr, Apvrille, Saqui-Sannes ym. 2003.	Luokkakaavio, toimintokaavio.	Ei merkittäviä uusia käsitteitä.	Stereotyypit.	UML:n epätäydellinen formaali semantiikka sekä puutteet kaavioiden validoinnissa.	Laajennusta on sovellettu onnistuneesti useissa tapaustutkimuksissa.

(jatkuu)

TAULUKKO 3. UML:n laajennusten kartoitus. (jatkuu)

Sovellusalue ja laajennuksen nimi	**Lähteet**	**Kaaviot, joita muutokset koskevat**	**Laajennuksen perusta, käsitteet sekä rakenteet**	**Laajennus-tavat**	**Laajennuksen esitysperusteet**	**Todisteet käytäntöön soveltuvuudesta**
Tosiaikajärjestelmä	Björkander 2000.	Luokkakaavio, tilakaavio.	Pohjautuu SDL-kieleen (Specification and Description Language).	Stereotyypit, nimetyt arvot	Tavoitteena yhdistää UML:n ilmaisuvoima ja SDL:n johdonmukaisuus sekä semanttiset vahvuudet.	SDL:ää on käytetty jo yli 20 vuoden ajan tosiaikajärjestelmien mallintamiseen. Laajennuksen käytännön soveltuvuudesta ei ole kuitenkaan tietoa.
Tosiaikajärjestelmä, **Profile for Schedulability, Performance and Time**	OMG 2003b.	Profiili on hyvin laaja ja koostuu useista osaprofiileista, joissa esitettyjä laajennusmekanis-meja voidaan soveltaa eri kaavioissa.	Sisältää useita käsitteitä, jotka on omaksuttu aiemmin esitetyistä tosiaikalaajennuksista.	Stereotyypit, nimetyt arvot, rajoitteet.	Tavoitteena muodostaa standardi profiili tosiaikajärjestelmien mallintamiseen.	OMG:n hyväksymä profiili.
Tosiaikajärjestelmä, **UML-RT**	Selic & Rumbaugh 1998.	Luokkakaavio, yhteistyökaavio, tilakaavio.	Kotelo (capsule), port (portti), yhdistin (connector), protokolla (protocol). Käsitteet on alun perin määritelty ROOM- menetelmässä (Real-Time Object-Oriented Methodology).	Stereotyypit.	Laajennus on suunniteltu erityisesti tukemaan monimutkaisten tosiaikajärjestelmien arkkitehtuurien määrittelyä.	Rationalin julkaisema profiili. Rational Rose Real Time-väline tarjoaa vastaavan mallinnustuen.

WWW-sovellukset ovat melko nuori ja koko ajan kehittyvä sovellusalue. WWW-sovellusten suunnittelua pidetään yleisesti myös monimutkaisena tehtävänä (Baresi, Garzotto & Paolini 2001; Conallen 1999b). Perus UML:ssä on havaittu puutteita mallintaa muun muassa käyttöliittymää ja navigointia, jotka ovat olennaisia osia WWW-sovellusta (Koch, Baumeister, Hennicker & Mandel 2000). Tämän takia laajennuksiin on lainattu käsitteitä ja rakenteita muun muassa hypermedian mallintamiseen käytetyistä menetelmistä ja malleista (esim. Garzotto, Paolini & Schwabe 1993; Schwabe, Rossi & Barbosa 1996). Kaikissa analysoiduissa WWW-sovellusalueen laajennuksissa laajennukset kohdistuvat luokkakaavioon, mutta myös muihin kaavioihin on esitetty laajennuksia riippuen laajennusesityksestä. Laajennukset on toteutettu UML:n sisäisten laajennusmekanismien avulla.

Agentti-perusteisuus on vasta viimeaikoina suosiotaan nostanut mallinnusparadigma. Se eroaa lähtökohdiltaan olio-perusteisesta mallinnuksesta. (Kavi, Kung, & Bharnbhani ym. 2003) Tämän takia alueelle on esitetty useita laajennuksia (Bergenti & Poggi 2000; Caire, Leal, Chainho ym. 2002; Flake, Geiger & Küster 2001; Kavi, Aborizka & Kung 2002; Odell, Parunak & Bauer 2000). Laajennukset esittelevät useita agentti-perusteisia käsitteitä ja laajentavat monia kaavioita. Laajennukset tuntuvat keskittyvät kuitenkin erityisesti luokka-, tila- ja tapahtumakaavioihin. Agentti-perusteisissa laajennuksissa on käytetty sekä raskaansarjan laajennusmekanismia että UML:n sisäisiä laajennusmekanismeja.

Tosiaikajärjestelmien mallintamisella on huomattavasti pitempi historia kuin WWW-sovelluksilla ja agentti-perusteisilla sovelluksilla. UML:ssä ei ole kuitenkaan otettu huomioon kaikkia tosiaikasovellusten erikoisvaatimuksia. Laajennuksiin (Björkander 2000; Selic & Rumbaugh 1998) on omaksuttu käsitteitä muun muassa ROOM-menetelmästä (Selic, Gullekson & Ward 1994) ja SDL-kielestä (ITU-T 1999). Edellä mainittuja menetelmiä on käytetty pitkään mallinnettaessa tosiaikajärjestelmiä. Laajennuksissa esitetyt käsitteet ja

rakenteet kohdistuvat pääasiassa luokka- ja tilakaavioon. Tämän sovellusalueen kaikki analyysissä mukana olevat laajennukset on toteutettu UML:n sisäisten laajennusmekanismien avulla. Tällä sovellusalueella laajennusten kehittely on johtanut jo siihen, että OMG on hyväksynyt standardin profiilin (OMG 2003b).

Tietokantasovellusten mallintamisella on jo pitkä historia, silti UML ei tarjoa kunnollista tukea relationaalisen eikä myöskään oliorelationaalisen tietokannan mallintamiseen. Tälle sovellusalueelle ehdotetut laajennukset (Ambler 2002; Gornik 2002; Marcos, Vela & Cavero 2001) kohdistuvat lähes poikkeuksetta pelkästään luokkakaavioon. Laajennukset on toteutettu UML:n sisäänrakennettujen laajennusmekanismien avulla.

Edellä mainittujen sovellusalueiden lisäksi laajennuksia on ehdotettu myös seuraaville sovellusalueille: multimedia (Sauer & Engels 1999, 2001), tila- ja aikatietoja sisältävät sovellukset (Price, Srinivasan & Ramamohanarao 1999a, 1999b), ryhmätyöohjelmistot (Rubart & Dawabi 2002) sekä suojatut järjestelmät (Jurjens 2002). Huomioitavaa on, että tämä lista ei ole kuitenkaan täysin kattava. Laajennuksia on saatettu esittää myös muille sovellusalueille.

Useimmat laajennukset on toteutettu UML:n sisäisten laajennusmekanismien avulla. Selkeästi käytetyin yksittäinen laajennusmekanismi on stereotyyppi, lähes kaikissa laajennuksissa hyödynnetään stereotyyppejä. Yleistä on myös käyttää nimettyjä arvoja ja rajoituksia stereotyyppien yhteydessä. Joissakin laajennusesityksissä, jotka on suunniteltu agentti-perusteiseen mallintamiseen ja multimediasovellusalueelle, joudutaan turvautumaan myös raskaansarjan laajennusmekanismiin sekä tekemään epämääräisiä muutoksia kaavioihin (Caire, Leal, Chainho ym. 2002; Kavi, Aborizka & Kung 2002; Odell, Parunak & Bauer 2000; Sauer & Engels 1999, 2001). Joukossa oli myös jokunen laajennus, josta jäi vaikutelma, että ne on muodostettu ottamatta selvää, kuinka UML:ää oikeasti pitäisi laajentaa. Tällöin on muunneltu kaavioita käyttämättä UML:n sisäisiä laajennusmekanismeja tai tekemättä muutoksia metamalliin.

Useimpien laajennusten soveltuvuudesta käytäntöön ei ole tietoa. Monen laajennuksen yhteydessä esitetään yksittäinen tapaustutkimus, jossa laajennusta sovelletaan. Tätä ei voida kuitenkaan pitää todisteena laajennuksen soveltuvuudesta käytäntöön laajemmin. Oletettavaa on, että osa laajennuksista unohtunee tyystin kenenkään niitä laajemmin käyttämättä. Joillekin laajennuksille ei ole viitsitty keksiä edes nimeä (esim. Rubart & Dawabi 2002; Flake, Geiger & Küster 2001). Monet laajennukset näyttävät sen sijaan myös kehittyvän ajan mittaan, kun tutkijat julkaisevat niitä koskevia uusia artikkeleita (ks. esim. AUML web site). Joitakin laajennuksia voidaan olettaa käytettävän melko yleisesti. Tällaisia laajennuksia ovat ainakin Rationalin ja OMG:n julkaisemat profiilit (Conallen 2000; OMG 2003b; Selic & Rumbaugh 1998). Näille laajennuksille löytyy myös CASE-välineiden mallinnustuki, mikä luonnollisesti madaltaa laajennusten käyttökynnystä. On kuitenkin muistettava, että sellaista laajennusta, jossa käytetään pelkästään UML:n sisäisiä laajennusmekanismeja, pystytään käyttämään useimmissa CASE-välineissä.

Tämän kartoituksen tekeminen osoittautui työlääksi tehtäväksi. Ensinnäkin rajauksen mukaisten laajennusten etsiminen oli suuri urakka. Toisekseen laajennusesitysten erilaiset esitysmuodot hidastivat niihin tutustumista. Jotkin esitykset ovat myös melko vaikeatajuisia. Kaikista laajennuksista ei myöskään löydy tarpeeksi tietoa. Tämän takia jouduinkin rajaamaan joitakin laajennuksia analyysin ulkopuolelle.

Kartoituksen perusteella päädyn tarkemman tason analyysissä keskittymään WWW-sovellusalueen UML-laajennuksiin. Tälle alueelle on esitetty riittävästi laajennuksia, jotta vertailua voidaan tehdä. Esitetyistä laajennuksista löytyy myös riittävästi tietoa. WWW-sovellusalue on myös melko nuori ja tutkimaton alue. Tämän sovellusalueen tutkimus on selkeästi varhaisemmassa vaiheessa verrattaessa esimerkiksi tosiaikajärjestelmiin, joiden mallintamisen tueksi on ehditty jo määrittelemään OMG:n standardi laajennus (OMG 2003b). Myös agentti-perusteisten laajennusten vertailu olisi mahdollista, sillä tälle

sovellusalueelle on muodostettu useita laajennuksia. Agentti-perusteisuus on myös nuori ja kehittyvä sovellusalue. Tämän tutkielman puitteissa ei ole kuitenkaan mahdollista keskittyä kahdelle sovellusalueelle.

3.4 Yhteenveto

Tässä luvussa esitettiin kartoitus UML:n laajennuksista. Kartoitukseen otettiin mukaan periaatteessa kaikki laajennukset, jotka koskevat nimettyä sovellusaluetta. Kartoituksen tavoitteena oli ensinnäkin tutkia, mille sovellusalueille UML:n laajennuksia on esitetty. Toiseksi laajennuksista selvitettiin, mihin kaavioihin esitetyt laajennukset kohdistuvat. Kolmanneksi tutkittiin, mitä merkittäviä uusia käsitteitä ja rakenteita laajennukset esittävät ja perustuvatko käsitteet johonkin aikaisemmin esitettyyn menetelmään tai mallinnuskieleen. Neljänneksi otettiin selvää laajennusten laajennustavoista. Viidenneksi selvitettiin laajennusten esittämisperusteet. Viimeiseksi pyrittiin selvittämään, onko laajennusten soveltuvuudesta käytäntöön todisteita.

Havaittiin, että laajennuksia on esitetty useille sovellusalueille. Kartoituksen mukaan suurin paine on ollut kehittää laajennuksia WWW-sovellusten, agentti-perusteisten sovellusten sekä tosiaikajärjestelmien mallintamiseen. Tässä luvussa havaittiin myös, että yleensä tietyn sovellusalueen sisällä esitetyt laajennukset kohdistuvat pääosin tiettyihin kaaviotyyppeihin, vaikka vaihteluakin ilmenee aika paljon. Ylipäätään luokkakaavioon on esitetty eniten laajennuksia riippumatta sovellusalueesta. Lähes kaikissa laajennuksissa esitellään uusia käsitteitä. Käsitteet perustuvat yleensä johonkin sovellusalueelle aikaisemmin esitettyyn menetelmään. Useimmat laajennukset on toteutettu UML:n sisäisten laajennusmekanismien avulla. Selkeästi käytetyin yksittäinen laajennusmekanismi on stereotyyppi. Laajennusten esittämisperusteita on lukuisia. UML:ssä tuntuu olevan paljon heikkouksia, joita paikkaamaan laajennuksia on esitetty. Useimpien laajennusten soveltuvuudesta käytäntöön ei ole saatavilla tietoa.

4 VIITEKEHYS LAAJENNUSTEN TARKASTELUUN

Tässä luvussa muodostetaan viitekehys, jonka avulla UML:n laajennuksia voidaan arvioida ja vertailla. Viitekehyksestä laaditaan yleinen, jotta sitä voidaan soveltaa useilla sovellusalueilla. Luvun alussa tarkastellaan yleisesti mallien ja mallinnuskielten laadun arviointia ja mainitaan joitakin keskeisiä tutkimuksia. Seuraavaksi esitellään yksityiskohtaisesti Krogstien (1995) esittämä viitekehys ja arvioidaan viitekehyksen soveltuvuutta tähän tutkielmaan. Tämän jälkeen esitetään tehdyt muutokset ja täydennykset Krogstien viitekehykseen, jotta se saadaan tähän tutkielmaan paremmin soveltuvaksi. Luku päättyy yhteenvetoon.

4.1 Yleiskatsaus mallien ja mallinnuskielten laadun arviointiin

Äskettäin on julkaistu kansainvälinen standardi ISO/IEC 9126 (ISO web site), jonka avulla voidaan arvioida ohjelmiston laatua. Kuitenkaan ei ole olemassa standardia, jonka avulla voitaisiin arvioida mallien laatua. Mallien laadun arvioimista pidetään vaikeana tehtävänä. Tähän on olemassa useita syitä. Ensinnäkin käsitteellistä mallintamista voidaan pitää enemmän taiteena kuin kurinalaisena insinöörityönä. Toiseksi valmiiden tuotteiden laatua on paljon helpompi arvioida kuin loogisten spesifikaatioiden. (Moody 2003)

Mallien ja myös mallinnuskielten laadun arvioiminen on kuitenkin tärkeää, sillä lopputuotteen laatu riippuu pitkälti vaatimusmäärittelyn ja ohjelmistoprojektin aikaisissa vaiheissa tuotettujen mallien laadusta. Mallien ja mallinnuskielten laadusta ja siitä, kuinka saavuttaa mallien korkea laatu, onkin kirjoitettu kohtuullisen paljon. Kirjallisuudessa on esitetty myös joitakin viitekehyksiä, joiden avulla laatua voidaan arvioida (esim. Lindland, Sindre & Solvberg 1994; Krogstie 1995; Paige, Ostroff & Brooke 2000). Usein viitekehyksissä vain listataan mahdollisia kriteereitä, niiltä puuttuu siis selkeä rakenne. Esimerkiksi Paige, Ostroff ja Brooke (2000) ovat esittäneet listan kriteereitä, joita

mallinnuskielten tulisi noudattaa. Heidän mielestään mallinnuskielen tulisi olla ennen kaikkea yksinkertainen (simplicity). Muita heidän listaamiaan laatukriteereitä ovat ortogonaalisuus (uniqueness), johdonmukaisuus (consistency), saumattomuus (seamlessness), käänteisyys (reversibility), skaalautuvuus (scalability), tuettavuus (supportability), toimintavarmuus (reliability) sekä mahdollisimman vähäinen tilan käyttö (space economy). Vaikka joitakin viitekehyksiä laadun arvioimiseen on esitetty, todella harvojen viitekehysten käyttökelpoisuutta on tutkittu empiirisesti.

Lindlandin ym. (1994) esittämää viitekehystä pidetään uraauurtavana sekä ensimmäisenä viitekehyksenä mallien laadun tarkasteluun, jolla on selkeä rakenne (Poels, Nelson, Genero ym. 2003). Kyseinen viitekehys on hyvin yleisesti viitattu. Moody, Sindre, Brasethvik ym. (2003) ovat tehneet myös empiirisen tutkimuksen, jossa he tutkivat viitekehyksen soveltuvuutta käytäntöön. Tutkimuksen tuloksena he toteavat viitekehyksen validiksi ja käyttökelpoiseksi. Lindlandin ym. (1994) viitekehykseen on esitetty myös joitakin laajennuksia, näistä tunnetuimpana voidaan pitää Krogstien (1995) viitekehystä. Myös tähän viitekehykseen on esitetty myöhemmin joitakin laajennuksia. Krogstien (1995) esittämää viitekehystä voidaan pitää kattavimpana tähän asti esitetyistä yleisesti mallien laatua tarkastelevista viitekehyksistä. Se käsittää mallien laadun arvioimiseen käytettävien kriteerien lisäksi myös mallinnuskielen laadun arvioimiseen liittyviä kriteereitä.

Vaikka joitakin viitekehyksiä mallinnuskielten laadun arvioimiseen on esitetty, on tehty vain harvoja tutkimuksia, joissa mallinnustekniikoita vertaillaan. Gemino ja Wand (2002) esittävät tutkimuksessaan koosteen niistä harvoista mallinnustekniikoihin kohdistuneista empiirisistä tutkimuksista, mitä on tehty. Heidän koosteensa sisältää viitteet 14 tutkimukseen. UML:ää heidän listaamistaan tutkimuksista koskee vain yksi (Kim, Hahn & Hahn 2000). UML:ää on arvioitu kriittisesti kuitenkin myös muutamissa muissa tutkimuksissa (esim. Morris & Spanoudakis 2001; Opdahl & Henderson-Sellers

2002) sekä vertailtu muihin mallinnuskieliin (esim. Henderson-Sellers & Firesmith 1999; Zendler, Pfeiffer, Eicks ym. 2001). Kirjallisuutta, jossa vertaillaan UML:n laajennuksia, ei ole juurikaan olemassa. Bichler, Radermacher ja Schurr (2002) vertailevat artikkelissaan kahta tosiaikajärjestelmien mallintamiseen suunnattua UML-laajennusta. Artikkelissa keskitytään kuitenkin laajennusten vertailun sijasta lähinnä vertailemaan kahta suosittua CASE-välinettä, joihin on toteutettu tuki laajennuksille.

Tämän tutkimuksen kannalta ei ole olemassa täysin sopivaa viitekehystä, jota voitaisiin käyttää suoraan vertailtaessa UML:n laajennuksia. Krogstien (1995) viitekehys toimii hyvänä lähtökohtana. Se on kuitenkin hyvin abstrakti, eikä se ole sidottu mihinkään tiettyyn kieleen. Tämän takia Krogstien viitekehystä täytyy täydentää UML:n laajennuksiin liittyvillä käsitteillä.

4.2 Krogstien viitekehys

Krogstien (1995) väitöskirjassaan kehittämä viitekehys perustuu Lindlandin ym. (1994) esittämään viitekehykseen. Viitekehystä on edelleen hieman täydennetty teoksessa Krogstie ja Solvberg (2000). Tässä kohdassa viitekehystä tarkastellaan tämän lähteen mukaan. Esityksessä kerrotaan myös, miltä osin viitekehyksessä esitetyt käsitteet pohjautuvat Lindlandin ym. (1994) esittämään viitekehykseen.

4.2.1 Käsitteet

Krogstien ja Solvbergin (2000) viitekehyksellä on kolme ainutlaatuista ominaisuutta. Ensinnäkin siinä erotetaan laatutavoitteet ja keinot tavoitteiden saavuttamiseksi. Toiseksi se on läheisessä yhteydessä kielitieteellisiin ja semioottisiin käsitteisiin. Mallien laatua tarkastellaan semioottisilla tasoilla, kuten syntaktisella, semanttisella sekä pragmaattisella tasolla. Lisäksi viitekehys perustuu konstruktivistiseen maailmankuvaan. Tämän käsityksen

mukaan mallit luodaan osittain mallintamisessa mukana olevien henkilöiden keskustelujen avulla, joiden tietämys mallinnusalueesta muuttuu mallintamisen aikana. (Krogstie & Solvberg 2000, 94–95)

Viitekehyksen pääkäsitteet ja niiden väliset suhteet esitetään kuviossa 4. Seuraavassa esitellään kyseisessä kuviossa esiintyviä käsitteitä. Yleisön käsite ei esiinny suoraan kuviossa, mutta tämän käsitteen kuvaaminen auttaa ymmärtämään viitekehystä.

- *Yleisö* (audience) koostuu erilaisista *toimijoista* (actor), jotka liittyvät jotenkin malleihin. Yleisön yksilöitä nimitetään *osallisiksi* (participant).

- *Kieli* (language extension) on kaikkien ilmaisujen joukko, jotka voidaan ilmaista mallinnuskielen sanaston ja syntaksin mukaisesti.

- *Malli* (externalized model) käsittää kaikki ilmaisut jostakin osasta mallinnettua todellisuutta ilmaistuna jollain kielellä.

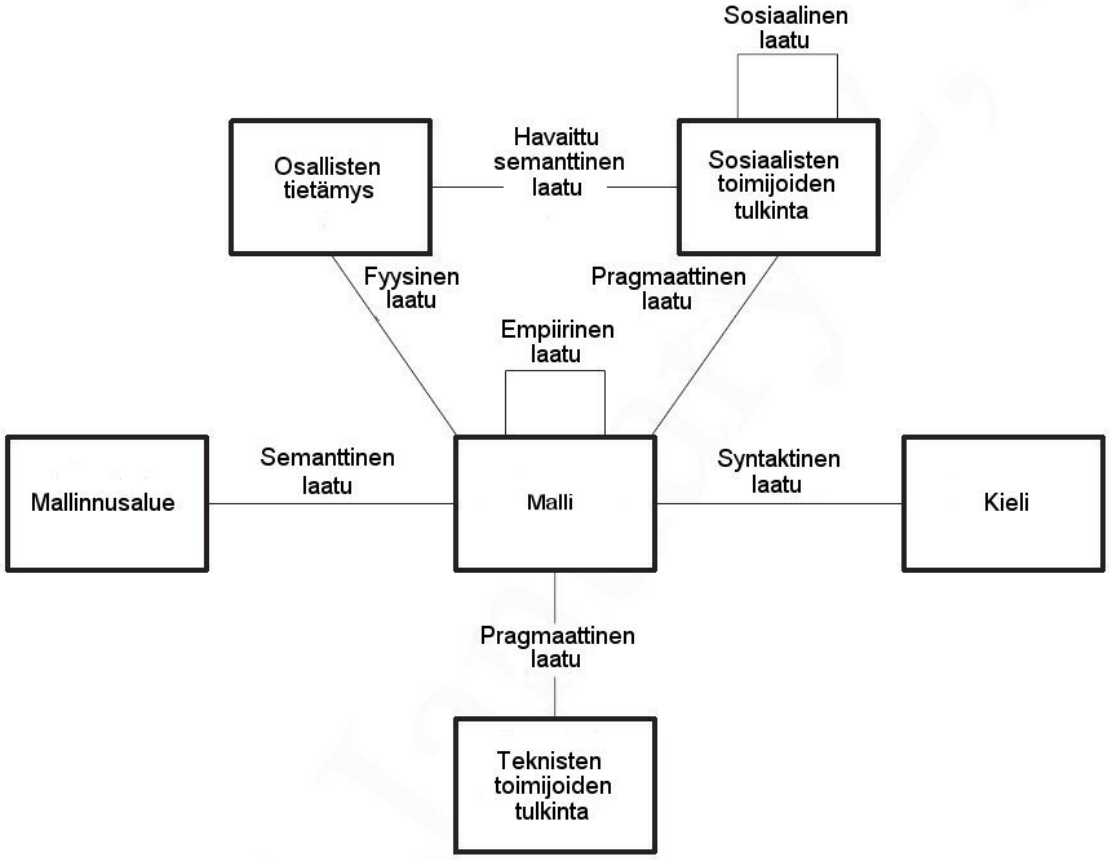
- *Mallinnusalue* (modeling domain) käsittää niiden ilmaisujen joukon, jotka voidaan esittää käsillä olevasta tilanteesta.

- *Osallisten tietämystä* (participant knowledge) tarvitaan käsillä olevan ongelman ratkaisuun ja mallien suunnitteluun. Osallisten tietämys muuttuu yleensä mallintamisen aikana.

- *Sosiaalisten toimijoiden tulkinnalla* (social actor interpretation) tarkoitetaan sosiaalisten toimijoiden ymmärrystä mallista. Sosiaalisilla toimijoilla tarkoitetaan ihmisiä.

- *Teknisten toimijoiden tulkinnalla* (techical actor interpretation) tarkoitetaan teknisten toimijoiden ymmärrystä mallista. Tekninen toimija on yleensä jokin tietokoneohjelma, kuten CASE-väline.

Lindlandin ym. (1994) esittämä viitekehys kattaa kuvioissa 4 esiintyvistä käsitteistä mallin, mallinnusalueen, kielen sekä sosiaalisten ja teknisten toimijoiden tulkinnan yhtenä käsitteenä. Muut käsitteet ovat Krogstien ja Solvbergin (2000) viitekehykseen tekemiä täydennyksiä.



KUVIO 4. Käsitteellisten mallien laatu (Krogstie & Solvberg 2000, 95).

4.2.2 Mallien laatu semioottisilla tasoilla

Viitekehyksessä mallien laatu on jaettu semioottisille tasoille. Tarkastelu voidaan jakaa kahteen alueeseen, teknilliseen sekä sosiaaliseen. Teknillinen alue kattaa fyysisen, empiirisen sekä syntaktisen tason. Sosiaalinen alue kattaa semanttisen, pragmaattisen sekä sosiaalisen tason. Seuraavassa kuvataan lyhyesti, mitä tavoitteita mallien laadulla on kullakin semioottisella tasolla. Tarkastelu pidetään hyvin suppeana, koska tutkielman kannalta on

olennaisempaa keskittyä tarkastelemaan mallinnuskielten laatua. Semioottisten tasojen lyhyt tarkastelu tässä yhteydessä on kuitenkin perusteltua, sillä näin saadaan parempi kokonaiskuva Krogstien ja Solvbergin (2000) viitekehyksestä. Tämän tutkielman osalta ei ole kuitenkaan olennaista syventyä keinoihin, joilla mallien laatua voidaan tavoitella.

Fyysinen laatu (physical quality) vallitsee mallin ja osallisten tietämyksen välillä. Vaikka tietojärjestelmämallit eivät yleensä ole fyysisiä ominaisuuksiltaan, mikä tahansa malli voidaan esittää fyysisesti jotenkin, esimerkiksi paperilla tai disketillä. Tavoitteena fyysisellä laatutasolla on sellainen ilmaisu (externalization), joka on helppo sisäistää (internalization). Ilmaisulla tarkoitetaan jonkin sosiaalisen toimijan tietämykseen pohjautuvaa ilmaisua jollakin mallinnuskielellä. Sisäistämisellä tarkoitetaan yleisön mahdollisuutta saada selvää ilmaistusta mallista. Mahdollisimman selkeään ilmaisuun päästään käyttämällä mallinnusalueelle sopivaa mallinnuskieltä.

Empiirinen laatu (empirical quality) käsittelee mallin elementeille ominaisia seikkoja, muun muassa virheiden yleisyyttä luettaessa tai tehtäessä malleja sekä ihmisen ja tietokoneen välisen vuorovaikutuksen ergonomiaa. Tällä tasolla tavoitteena on mahdollisimman vähäinen virheiden määrä (minimal error frequency). Erilaisille kaavioille ja tekstuaalisille malleille on esitetty useita periaatteita ja keinoja parantaa mallien luettavuutta sekä vähentää virheiden määrää. Tässä yhteydessä ei ole tarkoituksenmukaista esitellä näitä keinoja.

Syntaktinen laatu (syntactic quality) yhdistää mallin ja kielen, jolla malli on kirjoitettu. Tällä tasolla on vain yksi laatutavoite, syntaktinen oikeellisuus (syntactic correctness). Tällä tarkoitetaan sitä, että kaikkien ilmaisujen, jotka esiintyvät mallissa, täytyy olla mallinnuskielen syntaksin ja sanaston mukaisia.

Semanttinen laatu (semantic quality) määrittyy mallin ja mallinnusalueen välillä. Tällä tasolla on kaksi laatutavoitetta, validisuus (validity) ja täydellisyys (completeness). Validisuudella tarkoitetaan, että kaikki mallissa esiintyvät

ilmaisut ovat oikein ja merkityksellisiä ongelman kannalta. Täydellisyydellä tarkoitetaan, että malli käsittää kaikki merkitykselliset ilmaisut ongelman kannalta ja ilmaisujen tulee olla myös oikein. Normaalisti näiden tavoitteiden täydellinen saavuttaminen on hyvin työlästä, eikä se olekaan välttämättä tarkoituksenmukaista.

Havaittu semanttinen laatu (perceived semantic quality) vallitsee sosiaalisten toimijoiden tulkinnan ja osallisten tietämyksen välillä. Tällä tasolla on kaksi tavoitetta, havaittu validisuus (perceived validy) sekä havaittu täydellisyys (perceived completeness). Näillä tavoitteilla tarkoitetaan sitä, kuinka jokin yksilö kokee mallin validisuuden ja täydellisyyden.

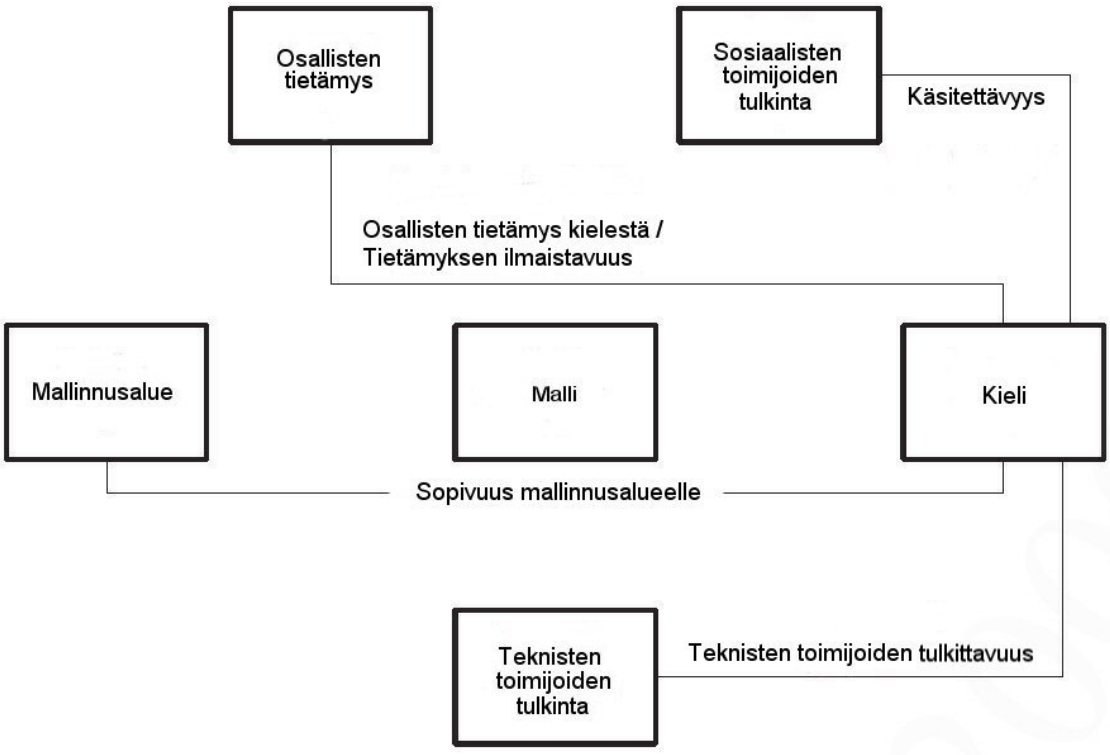
Pragmaattinen laatu (pragmatic quality) vallitsee mallin ja sosiaalisten sekä mallin ja teknisten toimijoiden tulkinnan välillä. Tällä tasolla on vain yksi laatutavoite, ymmärrys (comprehension). Tämä on tärkeä tavoite, sillä nerokkaimmastakaan tavasta ratkaista ongelma ei ole hyötyä, mikäli kukaan ei ymmärrä sitä. Täydellistä mallien ymmärrystä kaikkien osapuolten kohdalta on kuitenkin harvoin mahdollista saavuttaa, eikä tämä ole aina tarpeenkaan.

Sosiaalinen laatu (social quality) vallitsee sosiaalisten toimijoiden tulkintojen välillä. Tällä tasolla on yksi tavoite, nimeltään yksimielisyys (agreement). Tämä tavoite kattaa yksimielisyyden toimijoiden tietämyksessä ja tulkinnoissa. Täydellisen yksimielisyyden saavuttaminen kaikista seikoista on usein kuitenkin mahdotonta, eikä se ole aina tarpeenkaan. Usein on parempi valita jokin vaihtoehto kuin pyrkiä saavuttamaan täydellinen yksimielisyys.

4.2.3 Käsitteellisten mallinnuskielten laatu

Edellä on keskitytty käsittelemään mallien laatua semioottisilla tasoilla. Tässä kohdassa tarkastellaan mallinnuskielten laatuun vaikuttavia tekijöitä. Mallinnuskielten laadun tarkastelu onkin tämän tutkielman kannalta olennaisempaa kuin ilmaistujen mallien laadun tarkastelu.

Kuviossa 5 näkyvät mallinnuskielen suhteet muihin viitekehyksen käsitteisiin Krogstien ja Solvbergin (2000, 113) mukaan. Mallinnuskielen suhteet muihin viitekehyksen käsitteisiin voidaan käsittää kielen laatuluokkina. Kielen laatuluokat ovat täten sopivuus mallinnusalueeseen (domain appropriateness), osallisten tietämys kielestä (participant language knowledge appropriateness), tietämyksen ilmaistavuus (knowledge externalizability appropriateness), käsitettävyys (comprehensibility appropriateness) sekä teknisten toimijoiden tulkittavuus (tecnical actor interpretation appropriateness). Laatuluokkien sisällä laatukriteerien erottelu tehdään kahdelle tasolle. Kriteerit erotellaan kattamaan kielen käsitteellinen perusta (conceptual basis) ja kielen ulkoinen ilmaisu (external reprensentation), eli se kuinka käsitteet ilmaistaan visuaalisesti. Seuraavassa syvennytään edellä mainittuihin viiteen kielen laatuluokkaan yksityiskohtaisemmin.



KUVIO 5. Käsitteellisten mallinnuskielten laatu (Krogstie & Solvberg 2000, 113).

Sopivuus mallinnusalueeseen yhdistää kielen ja mallinnusalueen. Tällä tarkoitetaan sitä, että mallinnusalueen kaikki käsitteet pitää olla mahdollista ilmaista valitun mallinnuskielen avulla. Tässä yhteydessä puhutaan mallinnuskielen ilmaisuvoimasta (expressive power). Toiset mallinnuskielet ovat luonnollisesti sopivampia tiettyyn tilanteeseen kuin toiset.

Jotta mallinnuskieli olisi mahdollisimman sopiva ja ilmaisuvoimainen tietylle mallinnusalueelle, sen käsitteellisen perustan tulisi olla niin laaja, että sen avulla voidaan mallintaa mitä tahansa mallinnusalueelta. Toisaalta kielen ei pitäisi mahdollistaa mallinnusalueelle kuulumattomien asioiden mallintamista. Mallintamisen tulisi olla mahdollista mahdollisimman vähillä käsitteillä, jotta malli säilyy ymmärrettävänä. Tämä voidaan saavuttaa seuraavia seikkoja noudattaen:

- Käsitteiden tulisi olla ennemmin yleisiä kuin erikoistuneita.
- Käsitteitä tulisi voida luokitella luonnollisesti.
- Kielen pitäisi olla sekä tarkka että joustava. Tällä tarkoitetaan sitä, että kielen tulisi olla formaali ja yksiselitteinen, mutta sillä pitäisi pystyä mallintamaan myös epämääräisempiä asioita.

Tässä laatuluokassa ulkoiselle ilmaisulle on vain yksi vaatimus: se ei saa sekoittaa käsitteellistä perustaa. Esimerkiksi jokaiselle käsitteellisessä perustassa esitetylle ilmaukselle tulee olla yksikäsitteinen esitystapa.

Osallisten tietämys kielestä yhdistää kielen ja osallisten tietämyksen. Kielen käsitteellisen perustan tulisi vastata mahdollisimman paljon sitä miten yksilöt kokevat todellisuuden. Tämä vaihtelee luonnollisesti ihmisten välillä johtuen heidän kokemuksistaan. Osallisten tietämys kielestä ei ole myöskään muuttumaton, sillä on mahdollista kouluttaa henkilöitä käyttämään tiettyä kieltä. Käsitteiden esityksellisten vastineiden tulisi olla ymmärrettäviä ja

kohdettaan kuvaavia, jotkin symbolit kuvaavat tiettyä ilmiötä paremmin kuin toiset.

Tietämyksen ilmaistavuus yhdistää myös kielen ja osallisten tietämyksen. Tämän laatuluokan tavoitteena on, että osallisten tietämyksessä koskien mallinnusaluetta ei olisi sellaisia ilmaisuja, joita ei voida mallintaa mallinnuskielen avulla. Tätä laatuluokkaa ei viitekehyksen yhteydessä juurikaan käsitellä, eikä sitä pidetä tärkeänä.

Käsitettävyys yhdistää kielen ja sosiaalisten toimijoiden tulkinnan. Käsitettävyydellä tarkoitetaan sitä, että mallinnuskielen käyttäjän tulisi voida ymmärtää kaikki mallinnuskielen mukaiset ilmaisut. Käsitteellisen perustan kannalta seuraavat seikat ovat tärkeitä:

- Kielen käsitteiden tulisi olla helposti eroteltavissa toisistaan.
- Käsitteiden lukumäärän tulisi olla kohtuullinen. Mikäli käsitteiden lukumäärän täytyy olla iso, niiden pitäisi olla organisoitu hierarkkisesti.
- Käsitteiden käytön tulisi olla yhdenmukaista. Samojen käsitteiden käyttö usean ilmiön yhteydessä tai eri käsitteiden käyttö saman ilmiön yhteydessä tekee kielestä sekavan.
- Kielen tulisi olla joustava yksityiskohtien määrän esittämisen suhteen. Ilmauksien tulisi olla helposti laajennettavissa toisilla ilmauksilla, jotka tuovat esille enemmän yksityiskohtia. Toisaalta yksityiskohtien tulisi olla helposti piilotettavissa.
- Mallien pitäisi olla helposti jaettavissa luonnollisiin osiin.

Tarkasteltaessa mallinnuskielen soveltuvuutta mallinnusalueeseen käsiteltiin ilmaisuvoimaa, eli sitä mitä kaikkea on mahdollista ilmaista. Käsitettävyyden kannalta on tärkeää miettiä sitä, kuinka lyhyesti ja selkeästi (expressive

economy) asiat voidaan esittää. Uusia käsitteitä tulisi esitellä vain tarpeellinen määrä. Nyrkkisääntönä voidaan pitää sitä, että useimmin toistuvat sekä tärkeimmät ilmaukset tulisi pyrkiä pitämään mahdollisimman lyhyinä. Ulkoisen ilmaisun kannalta seuraavat seikat ovat tärkeitä:

- Symbolien erottaminen toisistaan pitäisi olla helppoa.
- Symbolien käytön tulisi olla yhdenmukaista.
- Symbolien tulisi olla yksinkertaisia.
- Korostuksien käytön tulisi olla yhteydessä ilmaisujen tärkeyteen mallissa.
- Symbolien sommittelu tulisi voida tehdä esteettisesti miellyttäväksi.

Teknisten toimijoiden tulkittavuus yhdistää kielen ja teknisten toimijoiden tulkinnan. Tällä tarkoitetaan mahdollisuutta hyödyntää erilaisia välineitä mallinnuksen eri vaiheissa. Tekniset toimijat vaativat yleensä kieleltä formaalia syntaksia ja semantiikkaa.

4.2.4 Viitekehyksen arviointi

Krogstien ja Solvbergin (2000) viitekehystä voidaan pitää varsin kattavana, sillä se käsittelee sekä mallien että mallinnuskielten laatua monilla semioottisilla tasoilla. Viitekehyksen hyvänä piirteenä voidaan pitää myös sitä, että mallinnuskielen laatuun liittyvissä luokissa kriteerit on jaettu käsittelemään erikseen notaatiota ja käsitteellistä perustaa. Viitekehyksen heikkous on siinä, että se on varsin abstrakti, eikä sitä ole sidottu minkään tietyn mallinnuskielen yhteyteen. Viitekehystä voidaan käyttää eri mallinnuskielten arvioimiseen, mutta tutkittavan kielen erityisominaisuudet täytyy ottaa erikseen huomioon.

UML:n laajennuksia voidaan pitää erityisinä kielinä. Tämän takia Krogstien ja Solvbergin (2000) viitekehyksestä käsitteellisten mallinnuskielten laatuluokat soveltuvat vertailun lähtökohdaksi. Mallien laadun käsittely semioottisilla tasoilla ei sovi laajennusten tarkasteluun, sillä tässä ei ole tarkoitus käsitellä yksittäisten ilmaistujen mallien laatua. Mikäli haluttaisiin vertailla yksittäisten mallien laatua, tarvittaisiin jokaisen laajennuksen notaation mukaisesti tehdyt esimerkkimallit samasta ongelmasta. Tämän tutkielman yhteydessä tällaisten esimerkkien tekeminen ei ole mahdollista. Seuraavassa tehdään viitekehyksessä esiteltyjen mallinnuskielen laatuluokkien osalta huomioita tämän tutkielman kannalta.

Mallinnuskielen sopivuus mallinnusalueeseen voidaan käsittää tämän tutkielman kannalta laajennuksen sopivuutena siihen sovellusalueeseen, johon se on suunniteltu. Tämän luokan sisällä tavoitteet asetetaan yksityiskohtaisesti sovellusalueen erikoisvaatimusten mukaisesti. Tarkoitus on tutkia, kuinka hyvin laajennus vastaa sovellusalueen vaatimuksiin. Toisin sanoin laajennuksen käsitteellisen perustan tulisi olla tarpeeksi laaja, jotta sen avulla on mahdollista mallintaa sovellusalueen erityispiirteet. Tässä yhteydessä puhutaan laajennuksen ilmaisuvoimasta. Yleisesti voidaan olettaa, että suuri käsitteiden määrä lisää ilmaisuvoimaa. Tämän luokan sisällä tarkastelu keskittyy mallinnuskielen käsitteisiin, ei niinkään notaatioon. Laajennusten voisi olettaa sopivan hyvin sovellusalueelleen, koska laajennuksilla nimenomaan pyritään vastaamaan yksittäisen sovellusalueen vaatimuksiin.

Osallisten tietämyksestä kielestä voidaan tehdä tämän tutkielman kannalta oletus, jonka mukaan osalliset tuntevat perus UML:n. Tietämystä vertaillaan siis suhteessa siihen, kuinka paljon uutta ja erilaista laajennuksissa esitetään verrattuna perus UML:n. Mikäli uudet käsitteet ja notaatiot poikkeavat kovasti perus UML:n tavasta esittää asioita, voidaan tätä pitää huonona seikkana. Mikäli laajennuksessa esitetään paljon outoja käsitteitä ja notaatioita, laajennuksen oppimiskynnys nousee.

Tämän tutkielman ohessa tietämyksen ilmaistavuutta on mielestäni mahdotonta arvioida. Ensinnäkin tämä seikka riippuu yksittäisen osallisen tietämyksestä. Toisekseen on mahdotonta arvioida, ovatko kaikki mahdolliset ideat ja ongelmanratkaisutavat mahdollista toteuttaa laajennuksen avulla.

Mielestäni käsitettävyyden tarkastelua laajennusten yhteydessä voidaan pitää erittäin tärkeänä, sillä on olennaista, että mallit ovat luettavia ja ymmärrettäviä. Tämän laatuluokan sisällä tarkastellaan sekä notaatiota että käsitteitä useiden viitekehyksessä esitettyjen kriteereiden avulla. Huomattavaa on myös, että käsitettävyyden ja ilmaisuvoiman välillä saattaa vallita ristiriita. Mikäli laajennuksessa esitetään useita uusia käsitteitä, se saattaa lisätä ilmaisuvoimaa, mutta saattaa vaikuttaa negatiivisesti käsitettävyyteen. Mikäli uusia käsitteitä esitetään todella paljon, se saattaa tehdä laajennuksesta monimutkaisen.

Teknisten toimijoiden tulkittavuuden voidaan ymmärtää tarkoittavan mahdollisuutta käyttää erilaisia välineitä, kuten CASE-välineitä mallinnuksen apuna. Laajennuksen perustuessa pelkästään UML:n sisäänrakennettuihin laajennusmekanismeihin voidaan yleisimpiä CASE-välineitä käyttää. Mikäli laajennuksessa tukeudutaan raskaansarjan laajennusmekanismiin, joudutaan CASE-välineisiin tekemään muutoksia, jotta niitä voitaisiin käyttää mallintamisen apuna.

Edellä esitetyn perusteella Krogstien ja Solvbergin (2000) esittämää viitekehystä voidaan soveltaa vertailtaessa UML:n laajennuksia. Siinä esitetyt mallinnuskielen laatuluokat soveltuvat UML:n laajennusten käsitteiden ja notaatioiden vertailuun. Viitekehystä täytyy kuitenkin tarkentaa UML:lle ominaisilla kriteereillä, muuten se on liian abstrakti. Laajennuksissa esitetyt käsitteet pohjautuvat lähes poikkeuksetta stereotyypin käsitteeseen, joten on luonnollista suunnata tarkempi tarkastelu stereotyyppeihin. Laajennuksissa käytetyt nimetyt arvot ja ominaisuudet sekä rajoitukset ovat yleensä alisteisia stereotyypeille, eli niitä käytetään lähinnä stereotyyppien yhteydessä.

4.3 Muokattu viitekehys

Tässä kohdassa kehitetään muokattu viitekehys Krogstien ja Solvbergin (2000) viitekehyksen pohjalta. Tehtävät muutokset pohjautuvat edellisessä kohdassa tehtyihin päätelmiin ja arvioihin. Kohdassa määritellään yksityiskohtaisesti tarvittavat muutokset ja lisäykset Krogstien ja Solvbergin (2000) viitekehykseen, jotta se saadaan paremmin soveltuvaksi tähän tutkielmaan. Lisäksi kuvataan Krogstien ja Solvbergin (2000) viitekehyksestä saatuja kriteereitä tarvittavilta osin.

Krogstien ja Solvbergin (2000) esittämästä viitekehyksestä UML-laajennusten arviointiin ja vertailuun soveltuvat alakohdassa 4.2.4 esitetyn perusteella seuraavat mallinnuskieltä koskevat laatuluokat: sopivuus mallinnusalueeseen, osallisten tietämys kielestä, käsitettävyys sekä teknisten toimijoiden tulkittavuus. Krogstie ja Solvberg (2000) esittelevät viitekehyksessään myös tietämyksen ilmaistavuutta kuvaavan laatuluokan, tämän luokan sisältämiä asioita ei voida kuitenkaan arvioida tämän tutkielman puitteissa. Tämän takia kyseistä luokkaa ei sisällytetä muokattuun viitekehykseen. Laatuluokasta sopivuus mallinnusalueeseen käytetään jatkossa termiä sopivuus sovellusalueeseen. Kyseinen termi on informatiivisempi tämän tutkimuksen kannalta. Kokonaan uutena osana Krogstien ja Solvbergin (2000) viitekehykseen lisätään laatuluokka nimeltään stereotyypeille ominaiset vaatimukset. Tämän luokan sisältämät kriteerit pohjautuvat Bernerin, Glinzin ja Joosin (1999) määrittelemiin hyvän stereotyypin ominaisuuksiin. Näitä asioita on tarkasteltu tämän tutkielman kohdassa 2.3.3. Edellä esitettyyn luokkajakoon pohjautuva kriteerikehikko esitetään taulukossa 4. Laatuluokkien mukaiset kriteerit on jaettu Krogstien ja Solvbergin (2000) esityksen mukaisesti kahteen sarakkeeseen sen mukaan, koskevatko ne mallinnuskielen käsitteellistä perustaa vai notaatiota. Seuraavassa kuvataan muokatun viitekehyksen kriteerit luokkakohtaisesti.

TAULUKKO 4. Kriteerikehikko UML-laajennusten arviointiin.

Laatuluokka	**Kriteerit käsitteelliselle perustalle**	**Kriteerit notaatiolle**
Sopivuus sovellusalueeseen	Käsitteellinen perustan tulisi olla niin laaja, että kaikki sovellusalueelle olennainen voidaan mallintaa. Sovellusalueeseen kuulumatonta ei tulisi voida mallintaa.	Ulkoinen ilmaisu ei saisi sekoittaa käsitteellistä perustaa.
Osallisten tietämys kielestä	Käsitteiden pitäisi vastata todellisuutta ja olla helposti opittavia. Käsitteiden tulisi olla perus UML:n mukaisia.	Käsitteiden esityksellisten vastineiden tulisi olla ymmärrettäviä sekä kohdettaan kuvaavia.
Käsitettävyys	Käsitteiden tulisi olla helposti eroteltavissa toisistaan. Käsitteiden lukumäärän tulisi olla kohtuullinen. Käsitteiden käytön tulisi olla yhdenmukaista. Kielen tulisi olla joustava yksityiskohtien määrän esittämisen suhteen. Mallien tulisi olla helposti jaettavissa luonnollisiin osiin.	Symbolien erottaminen tosistaan pitäisi olla helppoa. Symbolien käytön tulisi olla yhdenmukaista. Symbolien tulisi olla yksinkertaisia. Korostuksien käytön tulisi olla yhteydessä ilmaisujen tärkeyteen mallissa. Symbolien sommittelu tulisi voida tehdä esteettisesti miellyttäväksi.
Teknisten toimijoiden tulkittavuus	Käsitteiden tulisi pohjautua UML:n sisäisiin laajennusmekanismeihin, jotta useimpia CASE-välineitä pystyttäisiin hyödyntämään.	
Stereotyypeille ominaiset vaatimukset	Stereotyyppien tulisi olla selkeästi määriteltyjä. Stereotyyppien tulisi olla hyödyllisiä. Stereotyyppien tulisi muodostaa johdonmukainen joukko. Tulisi suosia kuvailevia ja rajoittavia stereotyyppejä ja välttää koristeellisia ja uudelleenmäärittteleviä stereotyyppejä.	

Viitekehyksen ensimmäinen laatuluokka on nimeltään laajennuksen sopivuus sovellusalueeseen. Tämä luokka sisältää kaksi kriteeriä käsitteelliselle perustalle ja yhden kriteerin notaatiolle. Nämä kriteerit ovat täysin Krogstien ja Solvbergin (2000) esityksen mukaisia. Ensimmäisen kriteerin mukaan käsitteellinen perustan tulisi olla niin laaja, että kaikki sovellusalueelle olennainen voidaan mallintaa. Toisen kriteerin mukaan sovellusalueeseen kuulumatonta ei tulisi voida mallintaa. Näiden kriteerien tarkempi sisältö määräytyy yksityiskohtaisesti kunkin sovellusalueen erityispiirteiden mukaan. Tässä kohdassa ei oteta kantaa siihen, mitä milläkin sovellusalueella tulisi voida mallintaa, koska tarkoitus on luoda viitekehys, jota voidaan soveltaa useammilla sovellusalueilla. Kolmannen kriteerin mukaan ulkoinen ilmaisu ei saisi sekoittaa käsitteellistä perustaa. Tällä tarkoitetaan sitä, että jokaiselle käsitteellisessä perustassa esitetylle ilmaukselle tulee olla yksikäsitteinen esitystapa.

Viitekehyksen toinen laatuluokka on nimeltään osallisten tietämys kielestä. Tämä luokka sisältää kaksi käsitteellistä perustaa koskevaa kriteeriä ja yhden notaatiota koskevan kriteerin. Kriteeri, jonka mukaan käsitteiden pitäisi vastata todellisuutta ja olla helposti opittavia sekä kriteeri, jonka mukaan käsitteiden esityksellisten vastineiden tulisi olla ymmärrettäviä ja kohdettaan kuvaavia, ovat suoraan Krogstien ja Solvbergin (2000) viitekehyksestä. Kriteeri, jonka mukaan laajennuksissa esitettyjen uusien käsitteiden tulisi olla perus UML:n mukaisia, on lisätty uutena kriteerinä kohdassa 4.2.4 tehtyjen pohdintojen perusteella. Voidaan olettaa, että UML:ää tuntevan on helpompi oppia käyttämään laajennusta, mikäli tehdyt laajennukset ovat UML:n mukaisia.

Viitekehyksen kolmas laatuluokka on nimeltään käsitettävyys. Tämä luokka sisältää viisi kriteeriä käsitteelliselle perustalle sekä notaatiolle. Nämä kriteerit ovat täysin Krogstien ja Solvbergin (2000) esityksen mukaisia. Suurimman osan tämän laatuluokan kriteereistä tarkoitus on varsin ilmeinen, joten niiden kuvaaminen ei ole tässä yhteydessä järkevää. Kahden kriteerin kohdalla

tarkempi kuvaus lienee kuitenkin paikallaan. Käsitteiden käytön yhdenmukaisuudella tarkoitetaan sitä, että samojen käsitteiden käyttö usean ilmiön yhteydessä tai eri käsitteiden käyttö saman ilmiön yhteydessä tekee kielestä sekavan. Kielen joustavuudella yksityiskohtien määrän esittämisen suhteen tarkoitetaan sitä, että ilmauksien tulisi olla helposti laajennettavissa toisilla ilmauksilla, jotka tuovat esille enemmän yksityiskohtia. Toisaalta yksityiskohtien tulisi olla helposti piilotettavissa.

Viitekehyksen neljäs laatuluokka on nimeltään teknisten toimijoiden tulkittavuus. Tämä laatuluokka sisältää vain yhden kriteerin käsitteelliselle perustalle. Kyseistä kriteeriä on hieman muokattu verrattuna Krogstien ja Solvbergin (2000) esittämään kriteeriin. Krogstien ja Solvebergin (2000) mukaan kielen tulisi mahdollistaa erilaisten välineiden hyödyntäminen mallinnuksen eri vaiheissa. Muokatussa viitekehyksessä tämä kriteeri suunnataan tiukemmin UML-laajennuksiin kytkeytyväksi. Muokatun kriteerin mukaan laajennuksessa esitettyjen uusien käsitteiden tulisi pohjautua UML:n sisäisiin laajennusmekanismeihin, jotta useimpia CASE-välineitä pystyttäisiin hyödyntämään.

Viitekehyksen viides laatuluokka on nimeltään stereotyypeille ominaiset vaatimukset. Tämä luokka sisältää neljä kriteeriä käsitteelliselle perustalle. Nämä kriteerit pohjautuvat Bernerin, Glinzin ja Joosin (1999) määrittelemiin viiteen hyvän stereotyypin ominaisuuteen. Näitä ominaisuuksia on tarkasteltu tämän tutkielman kohdassa 2.3.3. Heidän tekemät määritelmät ovat kuitenkin osittain päällekkäisiä käsitettävyyden laatuluokkaan liittyvien kriteerien kanssa. Bernerin, Glinzin ja Joosin (1999) mukaan stereotyyppien tulisi olla ortogonaalisia. Tämä vaatimus on päällekkäinen kriteerin kanssa, jonka mukaan käsitteiden käytön tulisi olla yhdenmukaista. Tämän takia kyseistä kriteeriä ei sisällytetä kriteerikehikkoon. Loput neljä Bernerin, Glinzin ja Joosin (1999) kuvaamaa hyvän stereotyypin ominaisuutta sisällytetään tähän laatuluokkaan. Ensimmäisen kriteerin mukaan stereotyyppien tulisi olla

selkeästi määriteltyjä. Määritelmän tulisi olla tarvittaessa formaali, mutta kuitenkin myös ymmärrettävä. Toisen kriteerin mukaan jokaisen stereotyypin tulisi olla hyödyllinen. Tällä tarkoitetaan sitä, että stereotyypin tulisi määrittää uusi käsite tai piirre, joka ei sisälly peruskieleen. Stereotyypin tulisi myös helpottaa mallinnuskielen käyttöä halutulla sovellusalueella. Kolmannen kriteerin mukaan stereotyyppien tulisi muodostaa johdonmukainen joukko, eli stereotyyppi ei saa olla yhteensopimaton muiden stereotyyppien kanssa, ellei sitä ole selkeästi määritelty yhteensopimattomaksi joidenkin tiettyjen stereotyyppien kanssa. Neljännen kriteerin mukaan tulisi suosia kuvailevia ja rajoittavia stereotyyppejä ja välttää koristeellisia ja uudelleen määritteleviä stereotyyppejä. Tämän kriteerin tarkastelu edellyttää laajennuksissa käytettyjen stereotyyppien luokittelua. Stereotyyppien luokitus saattaa olla joiltakin osin ongelmallista, sillä kaikkia stereotyyppejä ei ole välttämättä määritelty kunnolla.

4.4 Yhteenveto

Tässä luvussa muodostettiin viitekehys, jonka avulla UML-laajennuksia voidaan vertailla ja arvioida. Luvun alussa tehtiin lyhyt katsaus mallien ja mallinnuskielten laadun tarkasteluun yleisellä tasolla ja mainittiin joitakin keskeisiä tutkimuksia. Pääpaino tässä luvussa on ollut kuitenkin Krogstien ja Solvbergin (2000) viitekehyksen esittelyssä. Viitekehystä myös arvioitiin kriittisesti. Arvioinnin perusteella muokatun viitekehyksen perustaksi otettiin Krogstien ja Solvbergin (2000) viitekehyksessään esittämät mallinnuskielen laatuluokat. Esitetyistä laatuluokista poistettiin tietämyksen ilmaistavuutta kuvaava laatuluokka, koska luokan sisältämiä asioita ei voida arvioida tämän tutkielman puitteissa. Viitekehykseen lisättiin laatuluokka, jonka sisällä pyritään arvioimaan stereotyypeille ominaisia piirteitä. Myös joidenkin muiden laatuluokkien sisältämiä kriteereitä muokattiin hieman UML-laajennusten arvioitiin paremmin soveltuviksi.

Keskeisenä tarkoituksena on ollut laatia viitekehyksestä yleinen, jotta sitä voidaan käyttää useilla sovellusalueilla. Tässä tutkielmassa viitekehyksen toimivuutta testataan WWW-sovellusalueella. Huomattavaa viitekehyksessä on se, että sitä täytyy täydentää sovellusaluekohtaisesti laatuluokan sopivuus sovellusalueeseen osalta. Seuraavan luvun yhtenä tehtävänä onkin määritellä WWW-sovellusten kehittämiselle ominaiset piirteet. Näitä piirteitä käytetään hyväksi vertailtaessa laajennusten sopivuutta WWW-sovellusalueeseen.

5 WWW-SOVELLUKSET JA UML-LAAJENNUKSIA

Tässä luvussa tarkastellaan WWW-sovelluksia ja WWW-sovellusalueelle kehitettyjä UML-laajennuksia. Luvun alussa määritellään, mitä WWW-sovelluksella tarkoitetaan. Tämän jälkeen selvitetään WWW-sovellusten kehittämisen erityispiirteitä. Seuraavaksi kuvataan yksityiskohtaisesti kolme sovellusalueelle esitettyä UML-laajennusta. Luku päättyy yhteenvetoon.

5.1 Yleiskatsaus WWW-sovelluksiin

WWW-sovelluksen käsitteestä löytyy kirjallisuudesta useita erilaisia mielipiteitä, eikä yleisesti hyväksyttyä määritelmää ole olemassa. Tässä tutkielmassa käytetään Conallenin (1999b) tekemää määritelmää. Hän tarkoittaa *WWW-sovelluksella* sovellusta, jonka avulla käyttäjä voi syöttötiedoillaan vaikuttaa liiketoiminnan tilaan. Hänen mukaansa WWW-sovelluksen ja WWW-sivuston suurin ero on siinä, että WWW-sovelluksen avulla voidaan vaikuttaa organisaation liiketoimintaan, kun taas pelkkä sivusto on usein vain staattinen tarkastelun kohde.

WWW on alun perin suunniteltu yksinkertaisten sivustojen julkaisemiseen. Nykyiset WWW-sovellukset ovat usein kuitenkin hyvin monimutkaisia. Offutin (2002) määritelmä kuvaa hyvin WWW-sovellusten monimutkaisuutta. Hänen mukaansa WWW-sovellus on hajautettu järjestelmä, joka on usein toteutettu useilla ohjelmointikielillä ja tyyleillä. Lisäksi sovellus voi sisältää useita uudelleenkäytettäviä komponentteja ja on yleensä toteutettu uusilla teknologioilla. WWW-sovelluksen on yleensä myös tarjottava monia rajapintoja esimerkiksi käyttäjille, toisille sivustoille sekä tietokannoille.

Nykyään WWW-sovelluksia käytetään hyvin monenlaisiin tarkoituksiin. Myös sovellusten monimutkaisuus vaihtelee suuresti. Ginige ja Murugesan (2001, 14) luokittelevat WWW-sovellukset seitsemään kategoriaan taulukon 5 mukaisesti. Yksittäinen sovellus voi kuulua myös useampaan kuin yhteen kategoriaan.

TAULUKKO 5. WWW-sovelluskategoriat (vrt. Ginige & Murugesan 2001, 14).

Kategoria	**Esimerkkisovellukset**
Informatiiviset (informational)	Verkkosanomalehdet, tuotekatalogit, manuaalit, elektroniset kirjat.
Interaktiiviset (interactive)	Rekisteröintilomakkeet, personoidut palvelut, verkkopelit.
Tapahtumaperusteiset (transactional)	Elektroniset kauppapaikat, pankkipalvelut.
Työnkulkua edistävät (workflow)	Suunnittelu- ja aikataulujärjestelmät.
Ryhmätyöympäristöt (collaborative working environments)	Ryhmätyöskentelyn mahdollistavat sovellukset.
Verkkoyhteisöt ja markkinapaikat (online communities and marketplaces)	Keskustelupaikat, verkkokaupat, verkkohuutokaupat.
Portaalit (web portals)	Erilaiset portaalit.

WWW-sovelluksen perusarkkitehtuuri koostuu selaimesta ja WWW-palvelimesta. Tällaista arkkitehtuurimallia kutsutaan kaksitasoiseksi arkkitehtuuriksi. Tämä yksinkertainen malli toimii hyvin vain staattisten WWW-sivustojen kanssa. Nykyaikaiset WWW-sovellukset ovat yleensä monimutkaisempia ja niissä käytettävä arkkitehtuuri on usein vähintään kolmitasoinen tai N-tasoinen. Monimutkaisemmat sovellukset sisältävät selaimen ja WWW-palvelimen lisäksi usein ainakin sovelluspalvelimen ja tietokantapalvelimen. Näitä palvelimia voi olla myös useampia. Tällaisessa tapauksessa suurin osa ohjelmistoista on yleensä sijoitettu sovelluspalvelimille, jotka hakevat tarvittaessa tietoa tietokantapalvelimilta. Monitasoisemmalla arkkitehtuurimallilla saadaan parannettua WWW-sovelluksen tietoturvaa, skaalautuvuutta, luotettavuutta sekä toiminnallisuutta. (Offut 2002)

WWW-sovelluksen toiminta etenee niin, että selain pyytää sivuja WWW-palvelimelta. Mikäli pyydetty sivu on staattinen, se palautetaan sellaisenaan. Sivut voivat olla myös dynaamisia ja sisältää skriptejä. Tällöin WWW-palvelin

tai sovelluspalvelin prosessoi sivut ennen lähettämistä selaimelle. Näin sivuja voidaan muotoilla ja täydentää sisällöllisesti. Sivut voivat sisältää myös skriptejä, jotka tulkitaan selaimessa. Näiden skriptien avulla voidaan muokata muun muassa käyttöliittymää. (Conallen 1999b)

5.2 Kehittämistyön erityispiirteitä

WWW-sovellukset ovat monessa suhteessa samanlaisia verrattuna perinteisiin järjestelmiin, niillä on kuitenkin myös joitakin erityisiä ominaisuuksia. Tutkijoiden keskuudessa ei ole kuitenkaan olemassa yhteisymmärrystä siitä, mikä tarkalleen on ominaista vain WWW-sovellusten kehittämiselle. Seuraavassa tarkastellaan joidenkin tutkijoiden käsityksiä WWW-sovellusten kehittämiselle ominaisista piirteistä.

Baskerville ja Priesheje (2001) ovat tutkineet WWW-sovellusten kehittämisen erityispiirteitä. He tunnistavat tutkimuksessaan kymmenen seikkaa, jotka ovat ominaisia WWW-sovellusten kehittämiselle. Seuraavassa esitellään WWW-sovellusten kehittämistyön erityispiirteitä heidän mukaansa:

- Aikataulupaineet (time pressure): Internetissä kilpailu on yleensä kovaa. Saavutettu kilpailuetu on yleensä lyhytaikaista, sillä Internetissä ideoiden kopioiminen on helppoa. Tämän takia kehittämisaikataulut ovat yleensä hyvin tiukat.

- Epämääräiset vaatimukset (vague requirements): Vaatimukset ovat yleensä epätarkkoja tai eivät ole tiedossa ollenkaan.

- Protoilu (prototyping): Koska vaatimusmäärittely on hankalaa, käytetään usein prototyyppilähestymistapaa.

- Julkaisusuuntautuneisuus (release orientation): Sovellus pyritään julkaisemaan mahdollisimman aikaisin. Julkaisun jälkeen sovellukseen tehdään useita päivityksiä.

- Rinnakkainen kehittäminen (parallel development): Esimerkiksi tietokannan kehittämistä saatetaan tehdä samaan aikaan kuin käyttöliittymän suunnittelua. Analyysi- ja suunnitteluvaihetta voi olla vaikea erottaa.

- Vakaa arkkitehtuuri (fixed architecture): Sovellusten kompleksisuutta täytyy voida hallita. Usein käytetään kolmitasoista arkkitehtuuria, jossa tietokanta, liiketoimintalogiikka sekä käyttöliittymä erotetaan.

- Koodaus kunniaan (coding your way out): Joissakin tapauksissa voidaan joutua ratkaisemaan ongelmia nopeasti koodaamisen avulla.

- Laatu on neuvoteltavissa (quality is negotiable): Usein joudutaan päättämään, halutaanko kehittämistyö suorittaa nopeasti ja laadusta tinkien vai hitaasti ja laadukkaasti.

- Riippuvuus osaajista (dependence on good people): Internetprojektit suoritetaan yleensä kovien aikataulupaineiden alla ja pienissä ryhmissä. Tällöin yksilöiden osaaminen korostuu.

- Organisaation tarve (need for structure): Vanhat tietojärjestelmän kehittämisen organisaatiorakenteet eivät välttämättä toimi kehitettäessä WWW-sovelluksia.

Edellä esitetyssä listassa kahdeksan viimeisen ominaispiirteen voidaan nähdä johtuvan ainakin osittain aikataulupaineista sekä epämääräisistä vaatimuksista. Vidgrenin, Avisonin, Woodin ym. (2002) mukaan Baskervillen ja Prieshejen (2001) esittämät ominaispiirteet kuvaavat hyvin WWW-sovellusten kehitysprojekteja. Heidän mielestään suurin osa ominaispiirteistä sopisi kuitenkin kuvaamaan myös tavallisia tietojärjestelmäprojekteja. Vidgren ym. (2002) esittävätkin oman listansa konkreettisista eroista WWW-sovellusprojektien ja tavallisten tietojärjestelmäprojektien välillä:

- Internetaika (Internet time): WWW-sovellusten kehittämisaika on selvästi lyhyempi verrattuna tavallisiin sovelluksiin. Useat merkittävätkin projektit viedään läpi muutamissa viikoissa.

- Strategiset seuraamukset (strategic implication): Strategiset seuraamukset liittyvät suoraan liiketoiminnan tavoitteisiin, joiden avulla tuotto synnytetään.

- Painotus graafisessa käyttöliittymässä (emphasis on graphical user interface): On tarvetta taitaville graafikoille, jotka osaavat työskennellä ohjelmistoammattilaisten kanssa.

- Asiakaslähtöisyys (customer-orientation): Sovelluksen käyttäjä on asiakas eikä työntekijä.

Edellä on esitetty yleisesti WWW-sovellusten kehittämiselle ominaisia piirteitä. Tämän tutkielman kannalta on olennaista tunnistaa myös tarkemmin mallintamisen näkökulmasta, mitkä ovat erot WWW-sovellusten ja tavallisten sovellusten mallintamisen välillä. Rossi, Schwabe ja Lyardet (1999) nimeävät kolme selkeää eroa tavallisten järjestelmien ja WWW-sovellusten mallintamisen välillä. Ensimmäisenä seikkana he mainitsevat navigoinnin mallintamisen tärkeyden. Tähän seikkaan liittyen he esittävät seuraavat kysymykset:

- Mistä osista muodostuu informaatioyksikkö, joka voidaan yhdistää navigointiin?

- Kuinka määritellään, mitkä ovat mielekkäitä linkkejä informaatioyksiköiden välillä?

- Mistä käyttäjä aloittaa navigoinnin?

- Kuinka organisoida polut, joita pitkin käyttäjä voi navigoida?

- Mikäli olemassa olevaan järjestelmään lisätään WWW-käyttöliittymä, kuinka määritellään ne objektit, joista tehdään sivuilla näkyviä informaatioyksiköitä, ja mistä suhteista tehdään linkkejä?

Toisena selkeänä erona tavallisten järjestelmien ja WWW-sovellusten mallintamisen välillä Rossi ym. (1999) nimeävät käyttöliittymän mallintamisen tärkeyden. Käyttöliittymään liittyen he esittävät seuraavat kysymykset:

- Mitkä objektit tuodaan käyttäjän nähtäville? Kuinka nämä objektit liittyvät navigointiobjekteihin?
- Kuinka käyttöliittymä käyttäytyy käyttäjän tehdessä joitakin toimenpiteitä?
- Kuinka navigointioperaatiot erotetaan käyttöliittymäoperaatioista ja tiedon prosessoinnista?
- Kuinka käyttäjä saa selville sijaintinsa sivuston navigointitilassa?

Kolmantena erona tavallisten järjestelmien ja WWW-sovellusten mallintamisen välillä Rossi ym. (1999) nimeävät toteutukseen liittyvät seikat. Toteutukseen liittyen he esittävät seuraavat kysymykset:

- Kuinka informaatioyksiköt toteutetaan sivuilla?
- Kuinka navigointioperaatiot toteutetaan?
- Kuinka muut käyttöliittymäobjektit toteutetaan?
- Kuinka olemassa olevat tietokannat integroidaan WWW-sovellukseen?

Edellä esitetyn perusteella voidaan yhteenvetona todeta seuraavaa. Lähtökohtaisesti suuri ero tavallisen- ja WWW-sovelluksen välillä on, että WWW-sovelluksen käyttäjä on yleensä satunnainen Internetsurffaaja, kun tavallisilla järjestelmillä on yleensä tietty käyttäjäjoukko. Tästä syystä WWW-

sovellukselle on yleensä vaikea tehdä vaatimusmäärittelyä. Tästä taas seuraa, että WWW-sovelluksen suunnittelu on yleensä hyvin iteratiivista. Järjestelmästä pyritään tekemään nopeasti prototyyppi, jota parannellaan ajan mittaan. WWW-sovellusten kohdalla käyttöliittymän tärkeys näyttäisi olevan korostunut tavallisiin järjestelmiin verrattuna. Mikäli sovelluksen käyttöliittymä ei tyydytä asiakasta, on hänen todella helppo siirtyä käyttämään kilpailijan sovellusta. Mallintamisen osalta kirjallisuudessa painotetaan ehkä eniten navigoinnin mallintamisen tärkeyttä. Tämä on selkeä ero verrattuna tavallisiin järjestelmiin, joissa ei yleensä mallinneta navigointia. Koska WWW-sovellukset ovat yleensä monimutkaisia, arkkitehtuurin mallintamista voidaan myös pitää tärkeänä seikkana.

Eroavaisuuksien lisäksi Pastor, Schwabe, Rossi ym. (2002) huomauttavat, että WWW-sovellusten mallintamisessa on myös yhtäläisyyksiä verrattuna tavallisiin järjestelmiin. WWW-sovellusten yhteydessä täytyy mallintaa sovelluksen tietosisältö ja sen rakenteet sekä toiminnallisuus samaan tapaan kuin tavallisten järjestelmien yhteydessä.

5.3 UML:n laajennuksia WWW-sovellusalueella

Koska WWW-sovellusten mallintaminen eroaa tavallisten sovellusten mallintamisesta, tutkijat ovat kehittäneet erityisiä menetelmiä ja mallinnuskieliä tukemaan WWW-sovellusten kehittämistä. Esimerkkejä tunnetuimmista menetelmistä ovat OOHDM (Schwabe, Rossi & Barbosa 1996; Schwabe, Esmeraldo, Rossi ym. 2001) ja WebML (Ceri, Fraternali, Bongio ym. 2003). Muutamat tutkijat ovat havainneet UML:n hyväksi lähtökohdaksi mallintaa WWW-sovelluksia, koska se on laajennettava ja yleisesti käytetty standardi. Tästä syystä UML:ään on esitetty joitakin laajennuksia WWW-sovellusalueelle. Seuraavissa alakohdissa esitetään kolme UML:n laajennusta, jotka on suunniteltu tukemaan WWW-sovellusten mallinnusta. Kunkin laajennuksen osalta alussa kerrotaan lyhyesti laajennuksen perustasta ja mahdollisista

mallinnusvaiheista. Tämän jälkeen syvennytään tarkastelemaan laajennusten notaatiota ja käsitteitä. Seuraavassa tarkasteltavat laajennukset ovat nimeltään UWE (esim. Hennicker & Koch 2001a), W2000 (esim. Baresi, Garzotto & Paolini 2001) sekä WAE (esim. Conallen 2000).

5.3.1 UWE

UWE (UML-based Web Engineering) on UML-profiiliin pohjautuva menetelmä WWW-sovellusten suunnitteluun. Menetelmä sai alkunsa, kun Baumeister, Koch ja Mandel (1999) esittelivät menetelmän hypermedian mallintamiseen. He ovat käyttäneet kehittämistyössään lähtökohtana OOHDM-menetelmää (Schwabe, Rossi & Barbosa 1996). Huomattavaa on, että UWE-menetelmä oli alun perin tarkoitettu hypermedian mallintamiseen. Menetelmään on esitetty parannuksia muun muassa artikkeleissa Koch, Baumeister, Hennicker ym. (2000), Hennicker ja Koch (2001a) sekä Koch ja Kraus (2002). Samalla menetelmä on suunnattu selkeästi WWW-sovellusten mallintamiseen, ja se on nimetty UWE-menetelmäksi. UML-profiilin metamalli, johon UWE pohjautuu, on esitetty raportissa Kraus ja Koch (2003). Tässä kohdassa keskitytään käsittelemään UWE-menetelmään sisältyvää UML-profiilia Hennickerin ja Kochin (2001a) mukaan. Erityisesti syvennytään tarkastelemaan sekä profiilissa käyttäviä käsitteitä että notaatiota. Seuraavaksi kuvataan kuitenkin lyhyesti UWE:n askeleita, jotta saadaan yleiskuva menetelmästä.

UWE kattaa ohjelmistotuotannon vaiheista analyysin ja suunnittelun. Tarkemmin menetelmä koostuu taulukon 6 mukaisesti kolmesta askeleesta. Menetelmän ensimmäinen askel nimeltään käsitteellinen mallintaminen vastaa perinteisistä ohjelmistonsuunnittelun vaiheista analyysivaihetta. Toinen askel on nimeltään navigoinnin mallintaminen ja kolmas askel esitysmallin mallintaminen. Yhdessä nämä askeleet vastaavat suunnitteluvaihetta. Kuten edellä kerrotusta voi päätellä, UWE-menetelmässä otetaan erityisesti huomioon navigoinnin ja käyttöliittymän suunnitteluun liittyvät seikat.

TAULUKKO 6. UWE-menetelmän askeleet ja muodostettavat mallit.

Menetelmän askeleet	**Muodostettavat mallit**
Käsitteellinen mallintaminen	Käsitteellinen malli.
Navigoinnin mallintaminen	Navigoinnin tilamalli, navigoinnin rakennemalli.
Esitysmallin mallintaminen	Staattinen esitysmalli, dynaaminen esitysmalli.

Taulukosta 6 käyvät ilmi myös eri vaiheissa muodostettavat mallit. Käsitteellisen mallintamisen aikana muodostettava malli on nimeltään käsitteellinen malli (conceptual model). Seuraavaan askeleeseen eli navigoinnin mallintamiseen kuuluvat navigoinnin tilamallin (navigation space model) ja navigoinnin rakennemallin muodostaminen. (navigational structure model). Menetelmän kolmannessa vaiheessa muodostettavat mallit ovat nimeltään staattinen esitysmalli (static presentation model) ja dynaaminen esitysmalli (dynamic presentation model). Seuraavassa käsitellään kunkin mallin osalta niissä käytettäviä käsitteitä sekä notaatiota.

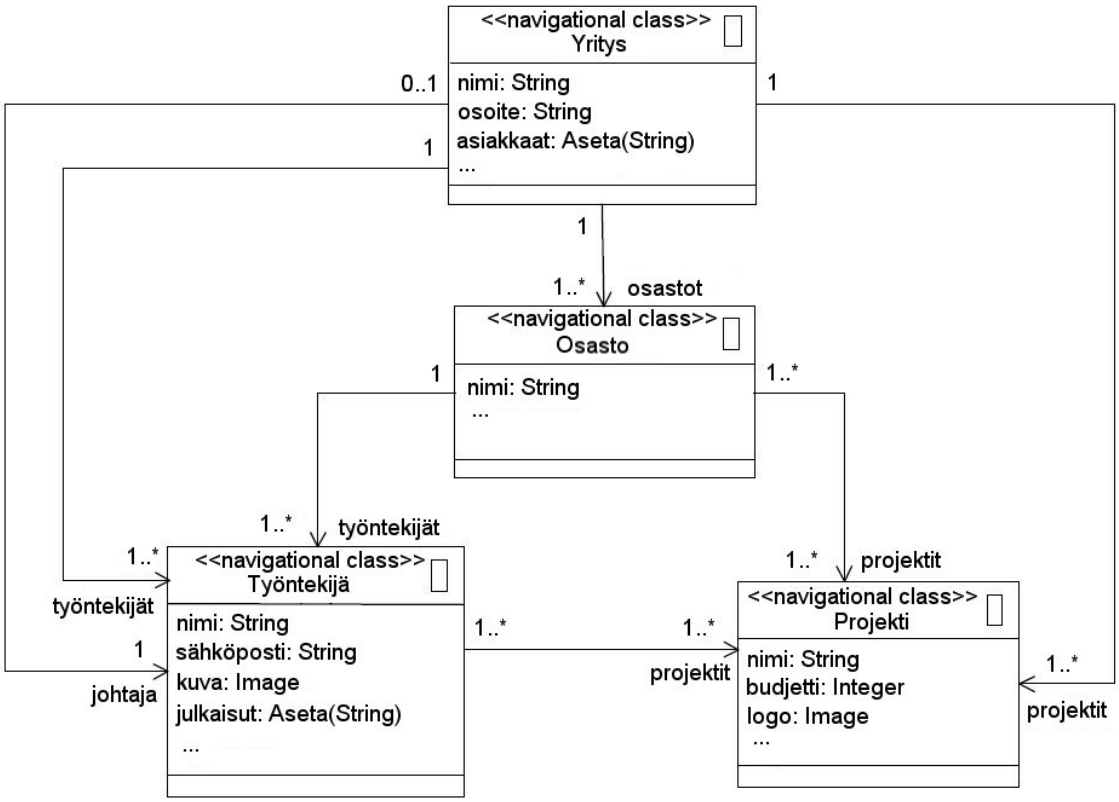
Käsitteellinen malli muodostetaan täysin UML:n mukaisesti käyttäen lähtökohtana vaatimusmäärittelyn aikana tehtyjä käyttötapauskaavioita. Tämän mallin avulla kuvataan koko järjestelmän tietosisältö. Käsitteellinen malli esitetään UML:n luokkakaavion avulla.

Navigoinnin tilamalli muodostetaan käsitteellisen mallin pohjalta ja esitetään UML:n luokkakaaviona. Navigoinnin tilamalli määrittää, minkä olioiden luona voidaan navigoida WWW-sovelluksen avulla. Esimerkki navigoinnin tilamallista esitetään kuviossa 6. Esimerkki mallintaa palveluyrityksen WWW-sivuja, joilla tarjotaan tietoa itse yrityksestä, sen asiakkaista, työntekijöistä sekä heidän suhteistaan projekteihin ja osastoihin. Yleistäen voidaan sanoa, että navigoinnin tilamalli sisältää vähemmän luokkia kuin käsitteellinen malli, sillä kaaviossa esitetään vain navigoitavissa olevat luokat. Jotkin käsitteellisessä

mallissa esiintyvät luokat voidaan esittää attribuutteina navigoinnin tilamallin luokissa. Malli perustuu kahteen käsitteeseen:

- *Navigointiluokka* (navigational class) tarkoittaan luokkaa, jonka ilmentymien luona voidaan käydä navigoinnin aikana. Luokka kuvataan stereotyypillä <<navigational class>>. Vaihtoehtoisesti voidaan käyttää myös graafista kuvaketta kuvion 6 mukaisesti.

- *Navigointisuhde* (direct navigability) kuvataan suunnatulla nuolella navigointiluokasta toiseen. Nuolen kärki osoittaa navigoinnin suunnan. Navigoidun luokan puoleiseen päähän merkitään roolinimi. Navigointisuhteelle käytetään stereotyyppiä <<direct navigability>>. Esimerkkikaavioon ei ole merkitty navigointisuhteisiin stereotyyppejä, sillä kaavioissa oletetaan implisiittisesti, että kaikki assosiaatiot ovat navigointisuhteita.



KUVIO 6. Navigoinnin tilamalli (Hennicker & Koch 2001a, 7).

Navigoinnin rakennemalli muodostetaan navigoinnin tilamallin pohjalta ja esitetään UML:n oliokaaviona. Tämä malli määrittää, millaisin keinoin navigointi oliosta toiseen tapahtuu. Erilaisia vaihtoehtoja ovat indeksit (indexes), johdetut läpikäynnit (guided tours), kyselyt (queries) sekä valikot (menus). Näitä navigointikeinoja nimitetään *hakuelementeiksi* (access elements). Esimerkki navigoinnin rakennemallista esitetään kuviossa 7. Esimerkkikaavio pohjautuu kuviossa 6 esitettyyn navigoinnin tilamalliin. Seuraavassa tarkastellaan navigoinnin rakennemallissa käytettäviä hakuelementtejä:

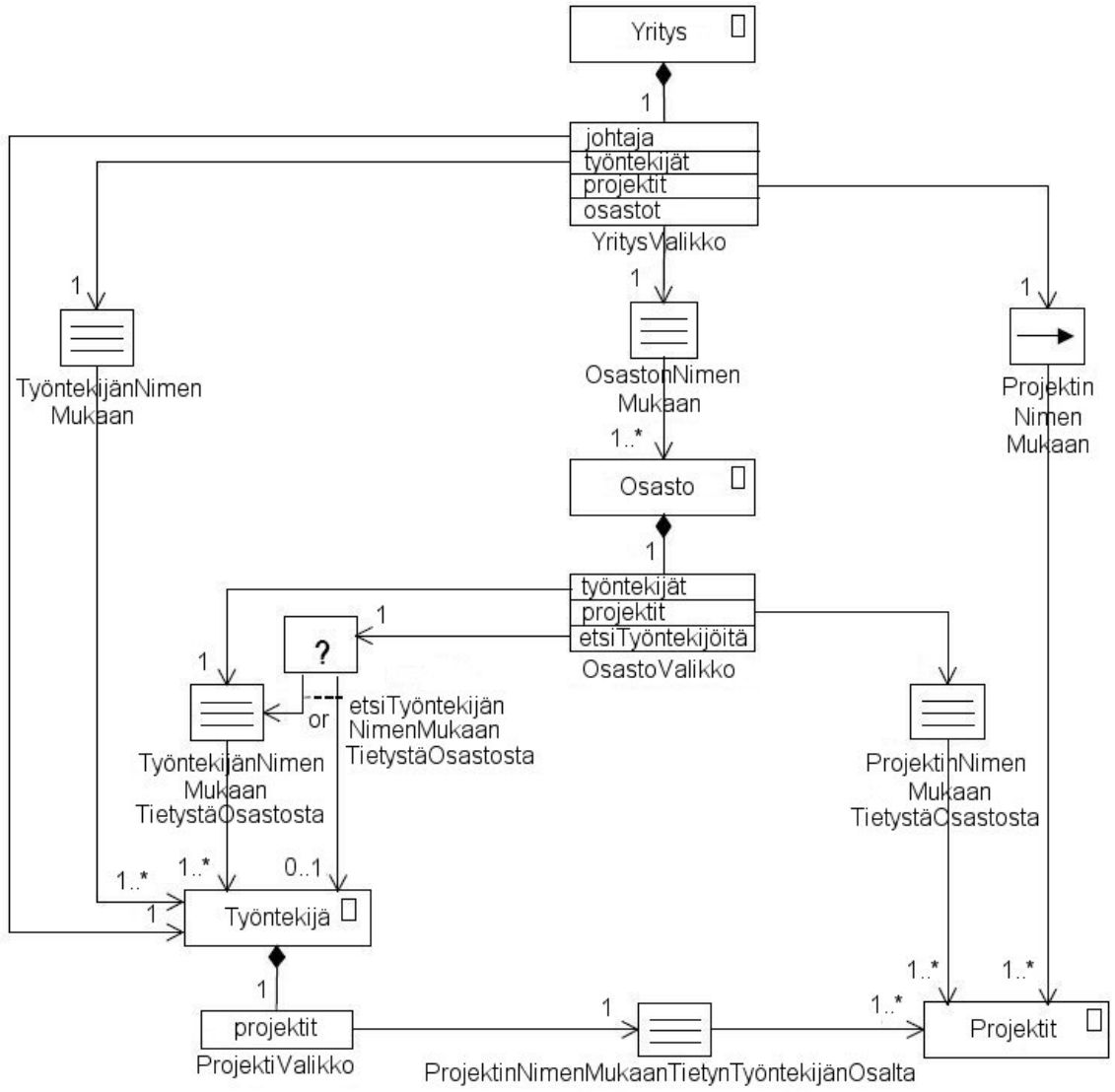
- *Indeksi* sisältää mielivaltaisen joukon *indeksielementtejä*, joilla on nimi sekä linkki jonkin navigointiluokan ilmentymään. Indeksistä käytetään stereotyyppiä <<index>>. Kaavioissa voidaan käyttää myös graafista kuvaketta. Kuviossa 7 näkyy indeksistä käytettävä kuvake; indeksi esitetään laatikkona, jonka sisällä on kolme vaakasuoraa viivaa.

- *Johdetulla läpikäynnillä* tarkoitetaan tietyn navigointiluokan ilmentymien läpikäyntiä tietyssä järjestyksessä. Jokainen johdettu läpikäynti täytyy yhdistyä suunnatulla assosiaatiolla johonkin navigointiluokkaan. Johdettu läpikäynti kuvataan stereotyypillä <<guidedTour>>. Vaihtoehtoisesti voidaan käyttää graafista kuvaketta, joka esitetään kuviossa 7. Johdetun läpikäynnin kuvakkeessa laatikon sisällä on nuoli, joka osoittaa oikealle.

- *Kyselyllä* tarkoitetaan hakuehtojen mukaista navigointia. Kysely voi johtaa suoraan tietyn navigointiluokan ilmentymään tai indeksiin, mikäli hakuehdon täyttäviä ilmentymiä löytyy useampia. Kyselystä käytetään stereotyyppiä <<query>> tai graafista kuvaketta, joka esitetään kuviossa 7. Indeksin kuvake esitetään laatikkona, jonka sisällä on kysymysmerkki.

- *Valikko* sisältää määrätyn määrän valikkoelementtejä, joilla on sekä nimi että linkki jonkin navigointiluokan ilmentymään tai toiseen hakuelementtiin. Valikosta käytetään stereotyyppiä <<menu>> tai

graafista kuvaketta kuvion 7 mukaisesti. Valikon kuvake esitetään riveihin jaettuna laatikkona. Riveillä esitetään valikkoelementtien nimet.



KUVIO 7. Navigoinnin rakennemalli (Hennicker & Koch 2001a, 12).

Staattinen esitysmalli muodostetaan navigoinnin rakennemallin pohjalta ja siinä määritellään, kuinka navigoinnin rakennemallin osat esitetään käyttäjälle. Malli kuvaa WWW-sovelluksen abstraktin käyttöliittymän, jossa keskitytään esityksen rakenteelliseen organisointiin. Mallista ei käy ilmi käyttöliittymän fyysinen ulkomuoto, kuten fontit tai värit. Nämä asiat päätetään vasta toteutusvaiheessa. Seuraavassa kuvataan staattisen esitysmallin käsitteet:

- *Kehysjoukko* (frameset) tarkoittaa säiliötä, joka voi sisältää esityksellisiä olioita tai toisia kehysjoukkoja. Esimerkki kehysjoukosta esitetään kuviossa 8. Esimerkki liittyy aiemmin esitettyihin kaavioihin. Kaaviossa vasemmalla esitetään yritysvalikko ja oikealla esityksellisen luokan yritys sisältö. Kehysjoukosta käytetään stereotyyppiä <<frameset>> tai graafista symbolia, joka käy ilmi kuviosta 8.

- *Esityksellisellä luokalla* (presentational class) kuvataan navigointiluokkaa tai hakuelementtejä. Esityksellinen luokka voi sisältää esimerkiksi tekstiä, kuvia, videoita, ääntä, tekstilinkkejä sekä kokoelmia. Esimerkki esityksellisestä luokasta esitetään kuviossa 9. Kuvaustapana käytetään stereotyyppiä <<presentational class>> tai graafista kuvaketta, joka esitetään kuviossa 9.

- *Teksti* (text) koostuu merkeistä.

- *Tekstilinkki* (anchor) on klikattava osa tekstiä, joka toimii navigoinnin aloituskohtana.

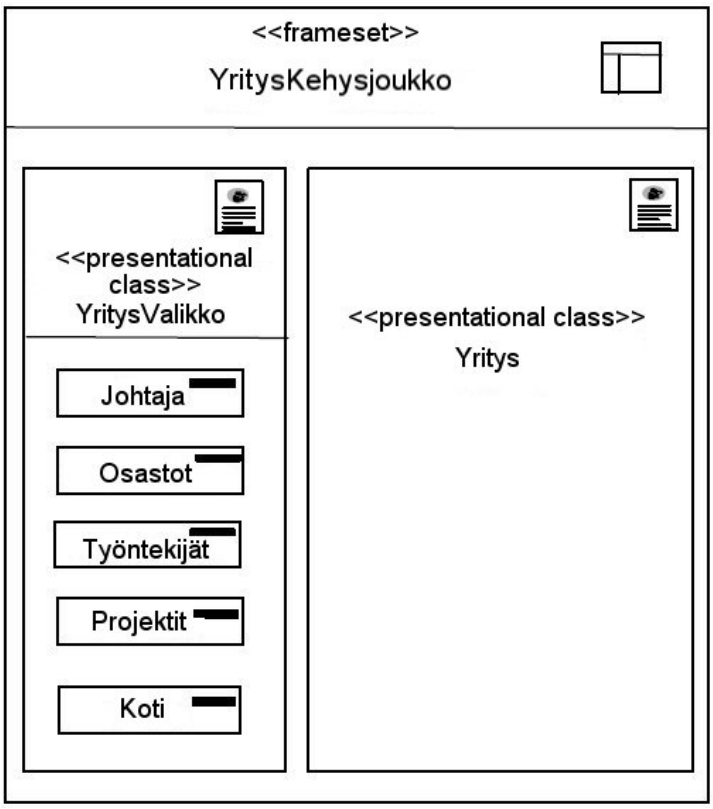
- *Painike* (button) on klikattava alue, johon yhdistetään jokin toiminto.

- *Kuva* (picture), *ääni* (audio) sekä *video* (video) ovat multimedia-komponentteja.

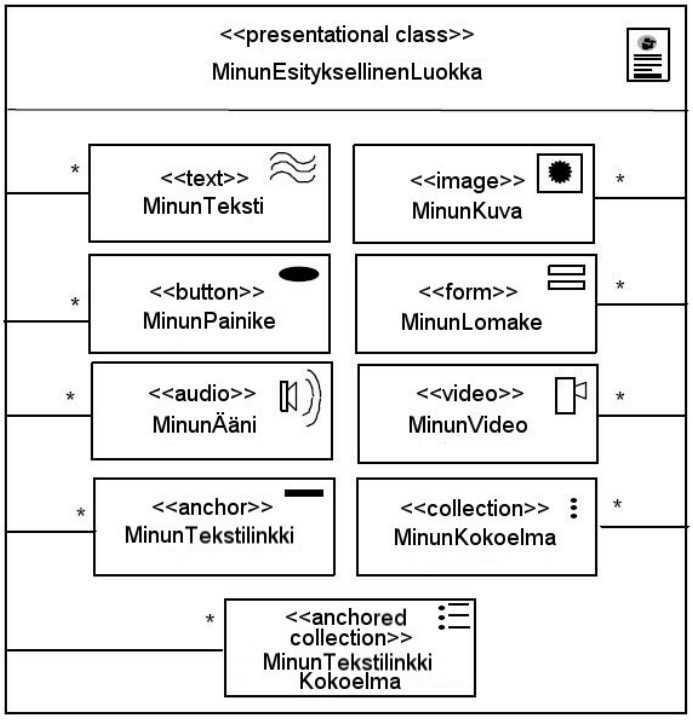
- *Lomaketta* (form) käytetään tiedon hakemiseksi järjestelmästä tai syöttämiseksi järjestelmään.

- *Kokoelma* (collection) ja *tekstilinkkikokoelma* (anchored collection) koostuvat mallielementeistä. Esimerkki kokoelmasta on lista tekstielementtejä.

Tekstin, tekstilinkin, painikkeen, kuvan, äänen, videon, lomakkeen, kokoelman sekä tekstilinkkikokoelman stereotyypit käyvät ilmi kuviosta 9.

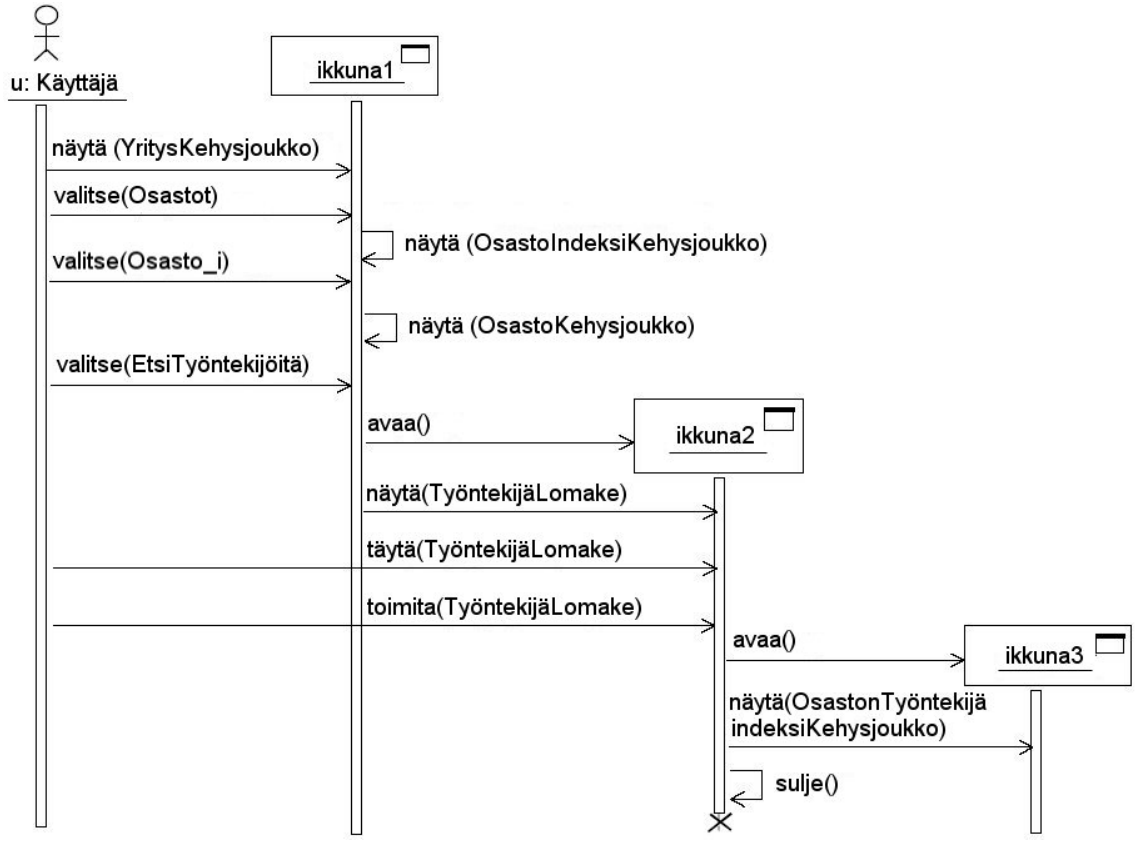


KUVIO 8. Kehysjoukko (Hennicker & Koch 2001a, 16).



KUVIO 9. Esityksellinen luokka (Hennicker & Koch 2001a, 15).

Dynaaminen esitysmalli kuvataan ikkunavuomallina (window flow model), joka esitetään UML:n tapahtumakaavion avulla. Tämän kaavion avulla kuvataan käyttöliittymän muutokset käyttäjän ollessa vuorovaikutuksessa järjestelmän kanssa. Esimerkkikaavio, joka pohjautuu aiemmin tässä kohdassa esitettyihin kaavioihin, esitetään kuviossa 10. Esimerkkikaaviossa mallinnetaan tietyn osaston työntekijöiden haku. Haun suorittamiseen tarvitaan kolme erillistä ikkunaa. Dynaamiseen esitysmalliin kuuluu vain yksi uusi käsite ikkuna (window). Tällä tarkoitetaan käyttöliittymän rajattua aluetta, jolla kehysjoukot ja esitykselliset oliot esitetään. Ikkunan kuvaustapana käytetään stereotyyppiä <<window>> tai kuviosta 10 ilmi käyvää graafista kuvaketta.



KUVIO 10. Ikkunavuomalli (Hennicker & Koch 2001a, 23).

5.3.2 W2000

W2000 on viitekehys WWW-sovellusten suunnitteluun (Baresi, Garzotto & Paolini 2001). Keskeisenä osana viitekehystä on WWW-sovellusalueelle suunniteltu UML-profiili. W2000-viitekehyksessä esitetään joitakin toimintoja WWW-sovellusten suunnitteluun, mutta kehittäjät eivät pidä viitekehystään menetelmänä. W2000-viitekehykseen on otettu paljon vaikutteita HDM-menetelmästä (Garzotto, Paolini & Schwabe 1993). Käytetyt käsitteet on lainattu pääosin kyseisestä menetelmästä. W2000 esiteltiin alun perin artikkelissa Baresi, Garzotto ja Paolini (2001). Tämän jälkeen viitekehyksestä on esitetty myös metamalli artikkelissa Baresi, Garzotto ja Maritati (2002). Kyseisessä artikkelissa ei valitettavasti esitetä metamallia kokonaisuudessaan, mutta artikkelista käy ilmi, että alkuperäiseen viitekehykseen on tehty samalla pieniä muutoksia. Tehdyistä muutoksista ei ole kuitenkaan saatavilla tarpeeksi tietoa, jotta niitä pystyttäsiin tässä yhteydessä tarkastelemaan. Tämän takia tässä kohdassa tarkastellaan W2000-viitekehystä Baresin ym. (2001) mukaan. Tarkoitus on syventyä käsittelemään erityisesti sekä viitekehyksen käsitteitä että notaatiota. Seuraavaksi esitellään kuitenkin ensin lyhyesti toiminnot, joiden ohessa W2000-viitekehyksen mallit tuotetaan, jotta saadaan yleiskuva viitekehyksestä.

W2000 kattaa ohjelmistotuotannon vaiheista vaatimusmäärittelyn, analyysin ja suunnittelun. Tarkemmin viitekehys koostuu taulukon 7 mukaisesti viidestä toiminnosta, joista osa koostuu edelleen osatoiminnoista. Vaatimusmäärittely ja analyysivaihe käsitetään yhtenä toimintona, joka jakaantuu kahteen osatoimintoon. Tästä toiminnosta käytetään nimeä vaatimusanalyysi (requirement analysis) ja sen osatoiminnoista toiminnallinen vaatimusanalyysi (functional requirements analysis) ja navigoinnin vaatimusanalyysi (navigational requirements analysis). Suunnitteluvaihe käsittää neljä erillistä toimintoa, tilakehityssuunnittelun (state evolution design), hypermedian suunnittelun (hypermedia design), toiminnallisen suunnittelun (functional design) sekä näkymäsuunnittelun (visibility design). Hypermedian suunnittelu

jakaantuu edelleen kahteen osatoimintoon, informaatiosuunnitteluun (information design) ja navigoinnin suunnitteluun (navigation design). Viitekehyksessä esitettyjä toimintoja voidaan suorittaa myös rinnakkain. Kuten edellä kerrotusta voi päätellä, W2000-viitekehyksessä otetaan erityisesti huomioon navigointiin ja tietosisältöön liittyvät seikat. W2000-viitekehyksen kehittäjät ymmärtävät myös käyttöliittymän suunnittelun tärkeänä osana WWW-sovelluksen suunnittelua, mutta he eivät esitä artikkelissaan ratkaisua tähän ongelmaan.

TAULUKKO 7. W2000-viitekehyksen toiminnot ja muodostettavat mallit.

Toiminnot	**Osatoiminnot**	**Muodostettavat mallit/kaaviot**
Vaatimusanalyysi	Toiminnallinen vaatimusanalyysi	Toiminnallinen käyttötapausmalli
	Navigoinnin vaatimusanalyysi	Navigoinnin käyttötapausmalli
Tilakehityssuunnittelu		Tilakehityskaavio
Hypermedian suunnittelu	Informaatiosuunnittelu	Informaatiomalli
	Navigoinnin suunnittelu	Navigointimalli
Toiminnallinen suunnittelu		Skenaariokaavio
Näkymäsuunnittelu		

Eri toiminnoille on ominaista, että niissä määritellään tilanteen mukaan eri määrä kaavioita. Tässä yhteydessä mallilla tarkoitetaan kaaviojoukkoa. Eri toiminnoissa muodostettavat mallit ja kaaviot käyvät ilmi taulukosta 7. Vaatimusanalyysin aikana muodostettavat mallit ovat toiminnallinen käyttötapausmalli (functional use-case model) ja navigoinnin käyttötapausmalli (navigational use-case model). Tilakehityssuunnittelun yhteydessä muodostettavat kaaviot ovat nimeltään tilakehityskaavioita (state evolution diagram). Tämän toiminnon mukaisia kaavioita tarvitaan yleensä vain

monimutkaisia järjestelmiä mallinnettaessa. Hypermedian suunnittelu kattaa informaatiomallin (information model) ja navigointimallin (navigation model) muodostamisen. Näitä malleja voidaan pitää W2000-viitekehyksen keskeisimpinä malleina. Toiminnallisen suunnittelun aikana muodostettavat kaaviot ovat nimeltään skenaariokaavioita (scenario diagram). Vaikka viitekehyksessä määritellään näkymien suunnittelu erillisenä toimintona, tämän toiminnon mukaisia kaavioita ei kuitenkaan esitetä. Seuraavassa tarkastellaan edellä mainittuja malleja ja malleissa käytettäviä käsitteitä tarkemmin.

Toiminnallinen käyttötapausmalli mallinnetaan täysin UML:n mukaisesti, kuten normaali käyttötapauskaavio. Ainoastaan kaavion nimi on muutettu. *Navigoinnin käyttötapausmallista* käyvät ilmi kunkin käyttäjäryhmän navigointimahdollisuudet. Tässä mallissa voi olla mukana myös käyttötapauksia, jotka eivät sisällä mitään toiminnallisuutta. Kummassakaan käyttötapausmallissa ei esitetä uusia käsitteitä.

Tilakehityskaavion avulla kuvataan, kuinka tietyn objektin tila voi muuttua järjestelmän käytön aikana. Tilakehityskaavio ei eroa normaalista UML:n tilakaaviosta. Uusia käsitteitä ei esitetä, ainoastaan nimi on eri.

Informaatiomallin avulla määritellään ja organisoidaan sovelluksen informaatiosisältö. Informaatiomalli jakaantuu kahteen tasoon, perustasoon (hyperbase layer) ja saantitasoon (access layer). Näillä tasoilla muodostettavia malleja kutsutaan perustasomalleiksi (hyperbase model) ja saantirakennemalleiksi (access structure model). Edellä mainituilla tasoilla voidaan muodostaa tilanteen mukaan yksityiskohtaisia kaavioita (in-the-small) sekä kaavioita, joissa ei esitetä kaikkia yksityiskohtia (in-the-large). Informaatiomallin kaikki kaaviot muodostetaan UML:n luokkakaavioina, joissa hyödynnetään stereotyyppejä.

Informaatiomallin perustasolla kuvataan sovelluksen entiteetit ja niiden väliset assosiaatiot. Esimerkki informaatiomallin perustasosta esitetään kuviossa 11. Esimerkki on osa WWW-pohjaisen konferenssinhallintajärjestelmän informaatiomallia, jossa näkyvät artikkelin, tekijän, arvostelun sekä konferenssin jäsenen väliset suhteet. Tässä kuviossa ei tuoda esille luokkien yksityiskohtia. Kuviossa 12 esitetään artikkelityypin yksityiskohtainen kuvaus informaatiomallin perustasolla. Kuviosta käy ilmi, että artikkelityyppi koostuu kolmesta osasta, tiivistelmästä, hyväksymisehdotuksesta ja julkaisuvalmiutta osoittavasta osasta. Artikkeliin kuuluu aina tiivistelmä, mutta muut komponentit saattavat puuttua. Seuraavassa kuvataan informaatiomallin perustasolla käytettävät käsitteet:

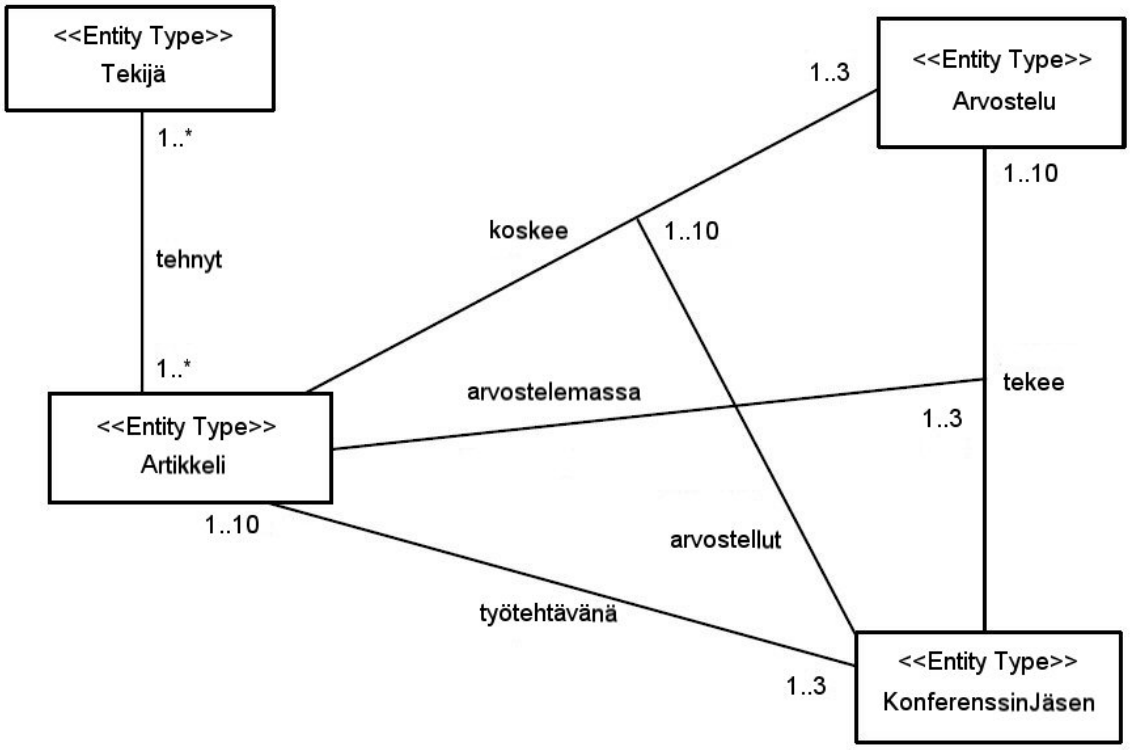
- *Entiteettiyypillä* (entity type) tarkoitetaan käyttäjälle näkyvää luokkaa. Entiteettityyppi kuvataan stereotyypillä <<Entity Type>>.

- *Komponenteilla* (component) kuvataan entiteettityyppien rakenne. Komponentit voidaan jakaa edelleen osakomponenteiksi. Komponentista käytetään stereotyyppiä <<Component Type>>.

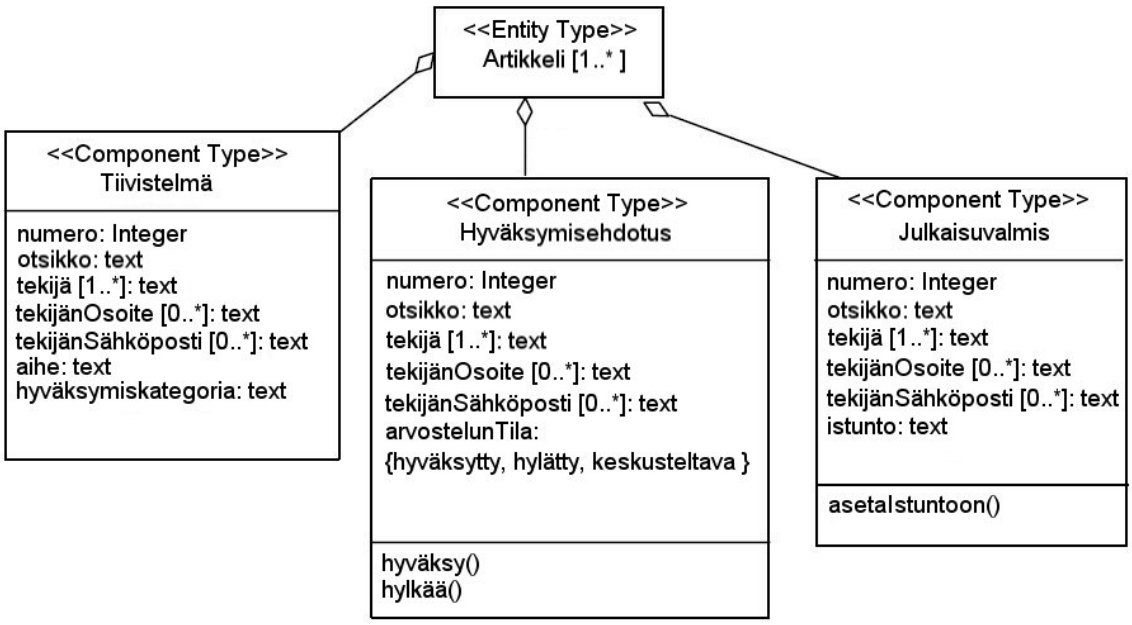
- *Paikalla* (slot) tarkoitetaan tietoarvoa. Käytännössä paikat ovat tyypitettyjä attribuutteja, jotka kuvaavat komponenttien sisällön. Esimerkkejä paikoista ovat muun muassa merkkijono, kokonaisluku ja musiikkikappale. Huomioitavaa on, että paikkoja käytetään vain yksityiskohtaisissa kaavioissa.

- *Semanttisella assosiaatiolla* (semantic association) voidaan yhdistää entitettityyppejä, komponentteja ja toisia assosiaatioita.

- *Assosiaatiokeskus* (association center) sisältää tietoa assosiaatiosta ja voi toimia navigoinnin lähtökohtana. Assosiaatiokeskuksesta käytetään stereotyyppiä <<Center Type>>



KUVIO 11. Informaatiomallin perustaso ilman yksityiskohtia (vrt. Baresi ym. 2001, 6).

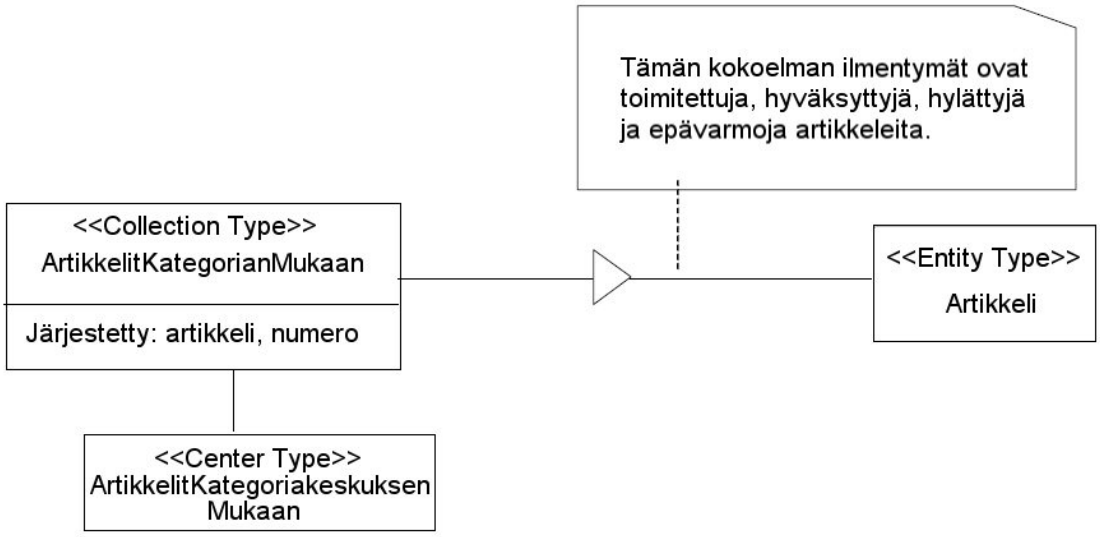


KUVIO 12. Informaatiomallin yksityiskohtainen perustaso (Baresi ym. 2001, 6).

Informaatiomallin saantitasolla kuvataan, kuinka informaatio-entiteettejä voidaan ryhmitellä ja organisoida. Kuviossa 13 esitetään yksinkertainen esimerkki informaatiomallin saantitasosta. Esimerkissä ArtikkelitKategorianMukaan kokoelmaan on koottu eri kategorioiden mukaisia artikkeleita. Seuraavaksi kuvataan informaatiomallin saantitasolla käytettävät käsitteet:

- *Kokoelmalla* (collection) tarkoitetaan säiliötä, joka voi sisältää entiteettityyppejä, komponentteja, assosiaatioita sekä toisia kokoelmia. Esimerkki kokoelmasta voisi olla lista tietyn kirjailijan kirjoista. Kokoelma kuvataan stereotyypillä <<Collection Type>>.

- *Kokoelmakeskus* (collection center) sisältää tietoa kokoelmasta. Keskus kokoaa yleensä tietoa kokoelman jäsenistä. Tieto esitetään keskuksen attribuutteina. Kokoelmakeskus voi toimia navigoinnin lähtökohtana. Kokoelmakeskuksesta käytetään stereotyyppiä <<Center Type>>.



KUVIO 13. Informaatiomallin saantitaso (Baresi ym. 2001, 7).

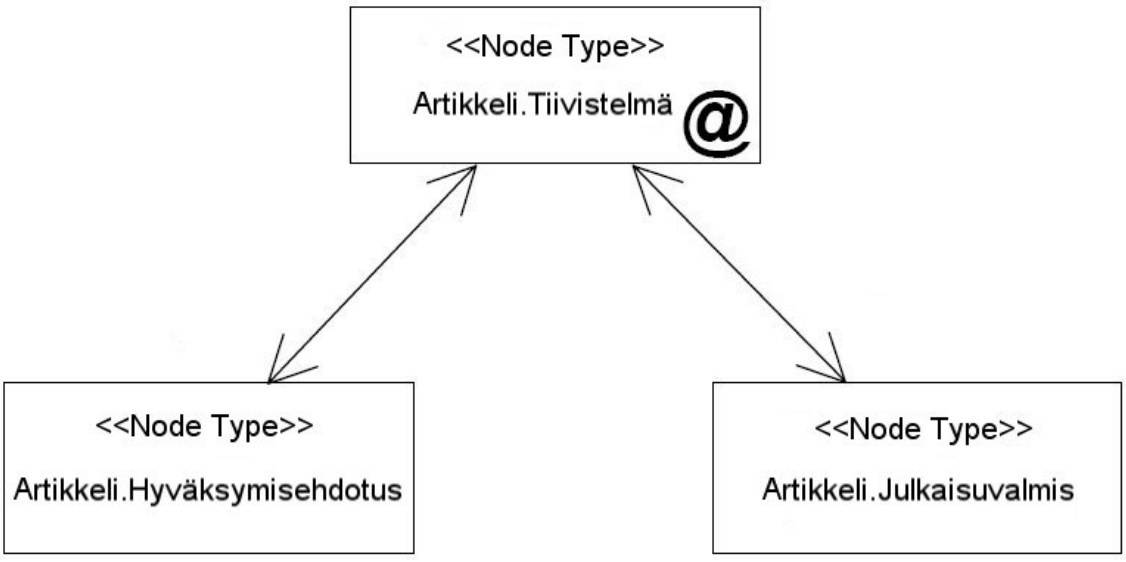
Navigointimallin avulla määritellään, kuinka informaatio-entiteettien ja saantirakenteiden välillä voidaan navigoida. Navigointimalli jakaantuu kahteen tasoon, perustasoon ja saantitasoon. Näillä tasoilla voidaan muodostaa

yksityiskohtaisia kaavioita ja kaavioita, joissa ei puututa yksityiskohtiin. Käytäntö on sama kuin informaatiomallin kohdalla. Navigointimallin kaikki kaaviot muodostetaan UML:n luokkakaavioina.

Navigointimallin perustasolla kuvataan navigointipolut informaatio-entiteettien välillä. Yksinkertainen esimerkki navigoinnin perustason mallista, jossa ei ole mukana yksityiskohtia, esitetään kuviossa 14. @-merkki kuvaa navigoinnin aloituspaikan. Esimerkin mukaan navigointi aloitetaan tiivistelmästä, josta voidaan navigoida hyväksymisehdotukseen tai julkaisuvalmiisiin tietoihin. Edellä mainituista solmuista voidaan navigoida myös takaisin tiivistelmään. Seuraavassa kuvataan navigoinnin perustasolla käytettävät käsitteet:

- *Solmulla* (node) tarkoitetaan informaatiomallin elementeistä johdettua kokonaisuutta. Yksinkertaisessa tapauksessa solmu voi vastata komponenttia tai kokoelmakeskusta. Solmu kuvataan stereotyypillä <<Node Type>>.

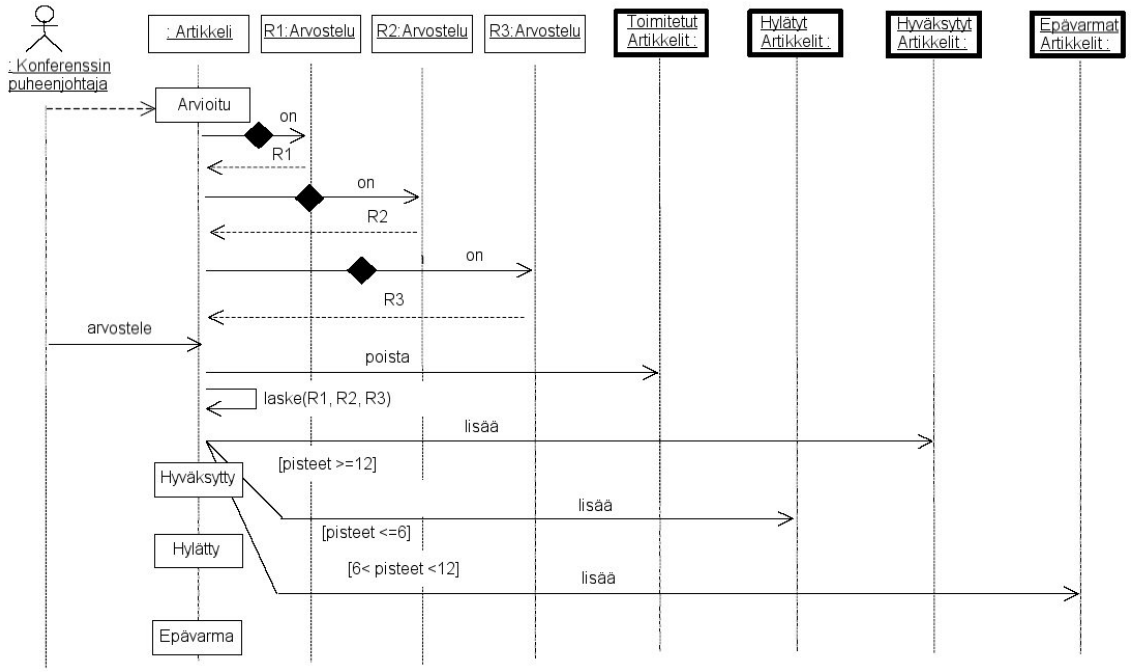
- *Linkki* (link) kuvataan nuolella kahden solmun välillä. Nuolensuunta osoittaa mahdollisen navigoinnin suunnan.



KUVIO 14. Navigointimallin perustaso ilman yksityiskohtia (Baresi ym. 2001, 7)

Navigointimallin saantitasolla määritellään navigointilinkit kokoelmiin. Linkkien avulla määritellään, kuinka käyttäjät voivat navigoida keskuksien ja kokoelmien jäsenien välillä. Tässä kaaviossa voidaan käyttää navigointikaavoja, kuten indeksejä ja ohjattuja läpikäyntejä. Näitä käsitteitä ei kuitenkaan esitellä laajennuksen yhteydessä. Navigointimallin saantitasolla ei esitetä uusia käsitteitä.

Skenaariokaavion avulla kuvataan, kuinka WWW-sovelluksen elementit toimivat vuorovaikutuksessa saadakseen aikaan toiminnot, jotka esitetään toiminnallisessa käyttötapausmallissa. Tavoitteena on, että toiminnallisen käyttötapausmallin jokaisesta käyttötapauksesta tehtäisiin ainakin yksi skenaariokaavio. Skenaariokaavioiden kuvaamiseen voidaan käyttää tapahtumakaavioita ja yhteistyökaavioita. Näissä kaavioissa käytettävät objektit ovat komponentteja, solmuja, semanttisia assosiaatioita sekä kokoelmia. Laajennuksena tavallisiin tapahtumakaavioihin ja yhteistyökaavioihin esitetään viestinvälityksen lisäksi mahdollisuus vapaaseen ja rajoitettuun navigointiin objektien välillä. Vapaa navigointi objektien välillä esitetään katkoviivalla ja rajoitettu navigointi viivalla, jonka keskellä esitetään timanttikuvio. Vapaalla navigoinnilla tarkoitetaan käyttäjän mahdollisuutta navigoida vapaasti käyttämättä tiettyjä navigointipolkuja. Rajoitettu navigointi pakottaa käyttäjän käyttämään tiettyjä ennalta määrättyjä navigointipolkuja. Esimerkki skenaariokaaviosta esitetään kuviossa 15. Esimerkissä konferenssin puheenjohtaja valitsee vapaasti jonkin artikkelin. Tämän jälkeen hän tutkii artikkelista tehtyjä arviointeja, joihin johtavat rajoitetut navigointipolut. Lopuksi sovellus poistaa artikkelin toimitettujen papereiden kokoelmasta ja laskee artikkelin saaman pistemäärän. Mikäli artikkeli saa vähintään 12 pistettä, se siirretään kokoelmaan hyväksytyt artikkelit. Mikäli pistemäärä jää kuuteen tai sen alle, artikkeli siirretään kokoelmaan hylätyt artikkelit. Mikäli pistemäärä on kuuden ja 12 välillä, siirretään artikkeli kokoelmaan epävarmat artikkelit.



KUVIO 15. Skenaariokaavio (Baresi ym. 2001, 9).

5.3.3 WAE

WAE (Web Application Extension for UML) on UML-profiili WWW-sovellusten suunnitteluun. Profiili on esitetty ensimmäisen kerran Conallenin (1999a) artikkelissa. Tämän jälkeen laajennuksesta on julkaistu myös toinen artikkeli (Conallen 1999b) sekä kirja (Conallen 2000), jossa laajennus esitetään yksityiskohtaisesti. WAE on Rationalin hyväksymä profiili, ja Rational Rose-väline tarjoaa vastaavan mallinnustuen. Vaikka WAE on Rationalin hyväksymä, se ei ole kuitenkaan virallinen standardi. Tässä kohdassa WAE-profiilia tarkastellaan Conallenin (2000) mukaan.

WAE kattaa ohjelmistotuotannon vaiheista ainoastaan suunnitteluvaiheen. Conallenin (2000, 3-7) mukaan muut WWW-sovelluksen kehitysvaiheet voidaan suorittaa esimerkiksi RUP-prosessia (Rational Unified Process) (Kruchten 2000) mukaillen, ja hyödyntäen UML:n peruskaavioita. WAE-profiililla on kolme keskeistä tavoitetta. Ensimmäinen tavoite on mahdollistaa

WWW-sovelluksille ominaisten käsitteiden, kuten WWW-sivujen, sivujen suhteiden, navigointipolkujen, asiakaspään skriptien sekä sivujen generoinnin mallintaminen. Toisena tavoitteena on asioiden mallintaminen eri abstraktiotasoilla. Kolmantena tavoitteena on mahdollistaa WWW-spesifien käsitteiden käyttö yhdessä yleisten käsitteiden kanssa. (Conallen 2000, 5) Tiivistetysti voidaan sanoa, että WAE-profiilissa keskitytään erityisesti arkkitehtuurin mallintamiseen käyttäen hyväksi WWW-sovellusten toteutustekniikoihin läheisesti liittyviä stereotyyppejä. Huomioitavaa on, että laajennuksessa ei huomioida ollenkaan käyttöliittymän mallinnusta.

Conallenin (2000, 147–151) mukaan suunnitteluvaiheessa tarkennetaan ja täydennetään analyysivaiheessa muodostettuja luokkakaavioita ja tapahtumakaavioita. Lisäksi voidaan muodostaa komponenttikaavioita. Tarkennettujen mallien avulla tulisi voida suoraan toteuttaa järjestelmä. Tärkeänä tehtävänä suunnitteluvaiheessa pidetään sivujen mallintamista, sillä sivut toimivat yhdistävänä tekijänä selaimen ja muun järjestelmän välillä. Sivut jaetaan selainsivuihin (client page) ja palvelinsivuihin (server page) niiden erilaisen luonteen takia. Conallen (2000, 221–230) määrittelee lukuisia WWW-sovelluksille ominaisia käsitteitä, joita voidaan käyttää tarkennettaessa suunnitteluvaiheen malleja. Uudet käsitteet pohjautuvat luokan, assosiaation, attribuutin ja komponentin käsitteisiin. Luokan käsitteeseen pohjautuvia käsitteitä käytetään pääasiallisesti luokka- ja tapahtumakaavioissa. Palvelinsivut ja selainsivut voidaan esittää myös komponenttikaaviossa. Seuraavassa esitellään luokan käsitteeseen pohjautuvia uusia käsitteitä.

- *Palvelinsivulla* (server page) tarkoitetaan sivua, joka sisältää palvelimella toteutettavia skriptejä. Luokan operaatiot esittävät skriptien funktioita ja attribuutit muuttujia, jotka ovat funktioiden käytettävissä. Palvelinsivuilla voi olla suhteita vain palvelimella oleviin olioihin. Palvelinsivu kuvataan stereotyypillä <<ServerPage>>.

- *Selainsivulla* (client page) tarkoitetaan HTML-muotoista sivua. Selainsivut voivat sisältää myös selaimessa toteutettavia skriptejä. Luokan operaatiot vastaavat skriptien sisältämiä funktioita ja attribuutit skripteissä esiteltyjä muuttujia, jotka ovat funktioiden käytettävissä. Selainsivusta käytetään stereotyyppiä <<ClientPage>>.

- *Lomake* (form) koostuu kokoelmasta syöttökenttiä, jotka ovat osa selainsivua. Lomake vastaa HTML:n <form> merkkausta. Kuvaustapana käytetään stereotyyppiä <<Form>>.

- *Kehysjoukko* (frameset) käsittää useita sivuja. Näyttö on jaettu pienempiin osiin, joita kutsutaan kehyksiksi. Kehyksen sisältö voi olla WWW-sivu tai toinen kehysjoukko. Kehysjoukko vastaa HTML:n <frame> merkkausta, ja se kuvataan stereotyypillä <<Frameset>>.

- *Kohde* (target) on selainikkunan nimetty osa, johon sivu ladataan. Stereotyypitetyn luokan nimi on kohteen nimi. Tyypillisesti kohde on kehysjoukon kehys, mutta kohde voi olla myös kokonaan uusi selaimen ilmentymä tai ikkuna. Kohteiden nimien tulee olla yksilöllisiä. Kohteesta käytetään stereotyyppiä <<Target>>.

- *Java-skriptiolio* (Java script object) on selainsivuilla käytettävä skripti. Selaimessa täytyy olla JavaScript tuki, jotta näitä skriptejä voidaan käyttää. Kuvaustapana käytetään stereotyyppiä <<JavaScript Object>>.

- *Selainskriptiolio* (client script object) on erillinen kokoelma selainskriptejä, jotka sijaitsevat erillisessä tiedostossa. Näiden sisällyttämiseen sivulle tarvitaan myös erillinen pyyntö selaimelta. Kuvaustapana käytetään stereotyyppiä <<ClientScript Object>>.

Assosiaatiota metaluokkana käyttävien käsitteiden luonnollisimmat käyttökohteet ovat luokka- ja komponenttikaavio. Seuraavassa esitellään kyseisiä käsitteitä:

- *Linkki* (link) on osoitin sivulta toiselle. Linkki voi osoittaa selainsivulta toiselle selainsivulle tai palvelinsivulle. Linkki vastaa HTML:n <anchor> merkkausta. Linkistä käytetään stereotyyppiä <<Link>>.

- *Kohdistettu linkki* (targeted link) on assosiaatio, jonka mukaan sivu ohjataan avautumaan tiettyyn kohteeseen. Kohdistettu linkki vastaa HTML:n <anchor> merkkausta, jossa kohde osoitetaan target attribuutin avulla. Kuvaustapana käytetään stereotyyppiä <<TargetedLink>>.

- *Kehyksen sisältö* (frame content) on kehyksen sisällön tietyltä sivulta tai tietyltä kohteelta osoittava koosteassosiaatio. Kehyksen sisältö saattaa osoittaa myös toiseen kehysjoukkoon, jolloin kyseessä ovat sisäkkäiset kehykset. Kuvaustapana käytetään stereotyyppiä <<FrameContent>>.

- *Toimittaa* (submit) assosiaatio voi vallita vain lomakkeen ja palvelinsivun välillä. Lomakkeiden avulla toimitetaan kenttien arvot palvelimelle käsiteltäviksi. Kuvaustapana käytetään stereotyyppiä <<Submit>>.

- *Rakentaa* (builds) assosiaatiota käytetään palvelinsivun ja selainsivun välillä. Palvelinsivut sijaitsevat palvelimella, ja niitä käytetään selainsivujen rakentamiseen. Palvelinsivu voi rakentaa useita selainsivuja, mutta selainsivu voidaan rakentaa vain yhdellä palvelinsivulla. Tästä käsitteestä käytetään stereotyyppiä <<Builds>>.

- *Ohjaa* (redirect) assosiaatiota voidaan käyttää selainsivujen, palvelinsivujen tai selainsivun ja palvelinsivun välillä. Ohjaa-assosiaatiolla voidaan esimerkiksi välittää jonkin palvelinsivun saama palvelupyyntö toiselle palvelinsivulle. Kuvaustapana käytetään stereotyyppiä <<Redirect>>.

- *IIOP* (Internet inter-ORB protocol) on erityislaatuinen suhde selainpään olioiden ja palvelinpään olioiden välillä. IIOP:llä tarkoitetaan jotain

muuta mekanismia kuin http:tä asiakkaan ja palvelimen väliseen kommunikointiin. Kuvaustapana käytetään stereotyyppiä <<IIOP>>.

- *RMI* (remote method invocation) on mekanismi, jonka mukaan Java appletit ja JavaBeanit lähettävät viestejä toisille JavaBeaneille eri koneilla. Kuvaustapana käytetään stereotyyppiä <<RMI>>.

Attribuutin käsitteeseen pohjautuvia uusia käsitteitä voidaan käyttää luokkakaavioissa. Näitä käsitteitä käytetään, mikäli halutaan tuoda lomakkeiden yksityiskohdat esille. Seuraavassa esitellään kolme attribuutin käsitteeseen pohjautuvaa käsitettä:

- *Syöttöelementtiä* (input element) käytetään lomakkeissa yhden sanan tai rivin syöttämiseen. Syöttöelementti vastaa HTML:n <input> merkkausta. Syöttöelementti kuvataan stereotyypillä <<InputElement>>.

- *Valintaelementtiä* (select element) käytetään lomakkeissa valittaessa yksi tai useampi vaihtoehto listatuista vaihtoehdoista. Valintaelementistä käytetään stereotyyppiä <<SelectElement>>.

- *Tekstialue-elementtiä* (text area element) käytetään lomakkeissa, mikäli halutaan mahdollistaa useampien rivien tekstin syöttö. Kuvaustapana käytetään stereotyyppiä <<TextAreaElement>>.

Komponenttia metaluokkanaan käyttäviä käsitteitä käytetään vain komponenttikaavioissa. Seuraavassa esitellään lyhyesti kyseisiä käsitteitä:

- *WWW-sivu* (web page) on komponentti, jota voidaan kutsua nimeltä selaimella. Tämä komponentti voi sisältää selainpään tai palvelinpään skriptejä. WWW-sivusta käytetään stereotyyppiä <<WebPage>>.

- *ASP-sivu* (ASP page) on komponentti, joka sisältää ASP-koodia. Kuvaustapana käytetään stereotyyppiä <<ASPPage>>.

- *JSP-sivu* (JSP page) on komponentti, joka sisältää JSP-koodia. Kuvaustapana käytetään stereotyyppiä <<JSPPage>>.

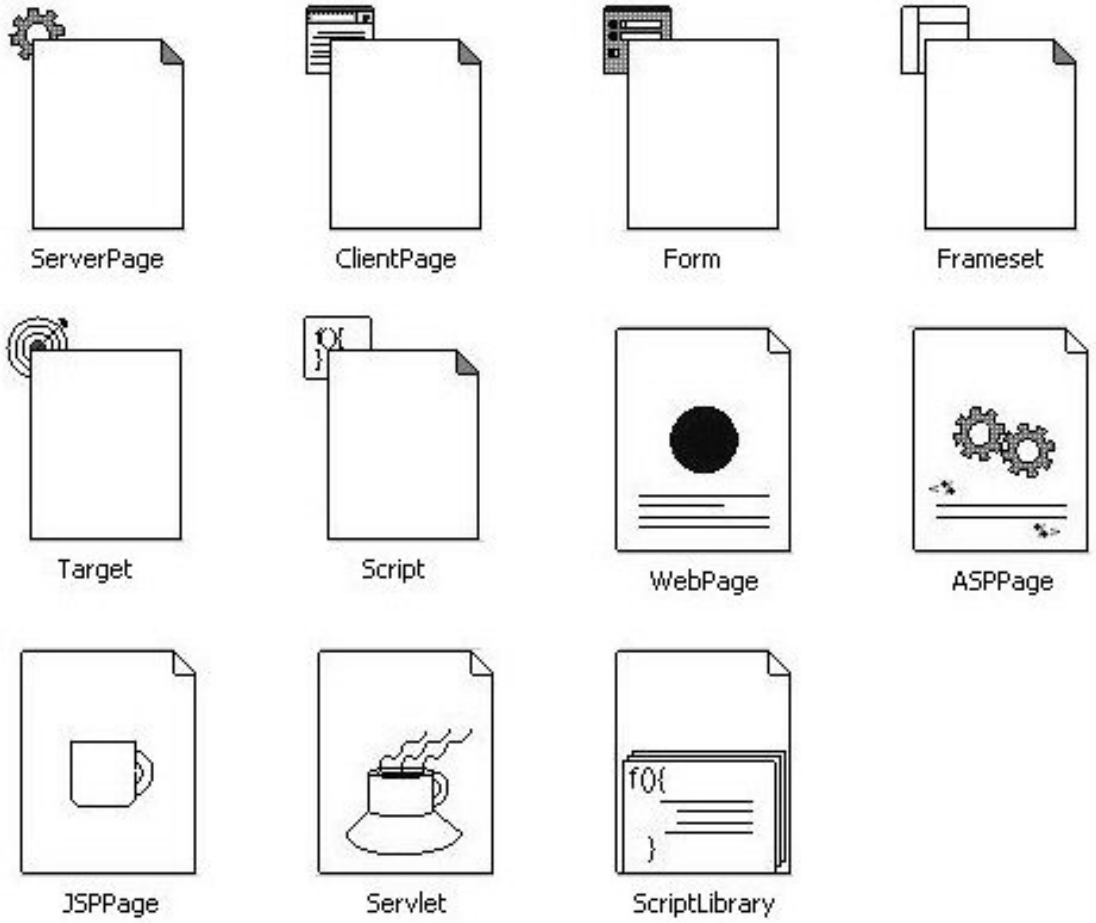
- *Servletti* (servlet) on Java-servlettikomponentti. Servletistä käytetään stereotyyppiä <<Servlet>>.

- *Skriptikirjasto* (script libary) on komponentti, joka sisältää alirutiineja sekä funktioita, jotka voidaan sisällyttää toisiin komponentteihin. Tästä käsitteestä käytetään stereotyyppiä <<ScriptLibary>>.

Useiden stereotyyppien yhteydessä voidaan käyttää myös profiilissa määriteltyjä nimettyjä arvoja. Näiden nimettyjen arvojen esittely tässä yhteydessä ei ole kuitenkaan tarpeellista. Tärkeimpien stereotyyppien validit assosiaatiosuhteet käyvät ilmi taulukosta 8. Taulukkoa luetaan niin, että assosiaation suunta on rivin käsitteestä sarakkeen käsitteeseen päin. Osalle stereotyypeistä on määritelty myös graafiset kuvakkeet, jotka käyvät ilmi kuviosta 16. Huomattavaa on, että sivua tarkoittavilla kuvakkeilla on pieni ”koirankorva” oikeassa ylänurkassa.

TAULUKKO 8. WAE-stereotyyppien väliset validit assosiaatiosuhteet (Conallen 2000, 231).

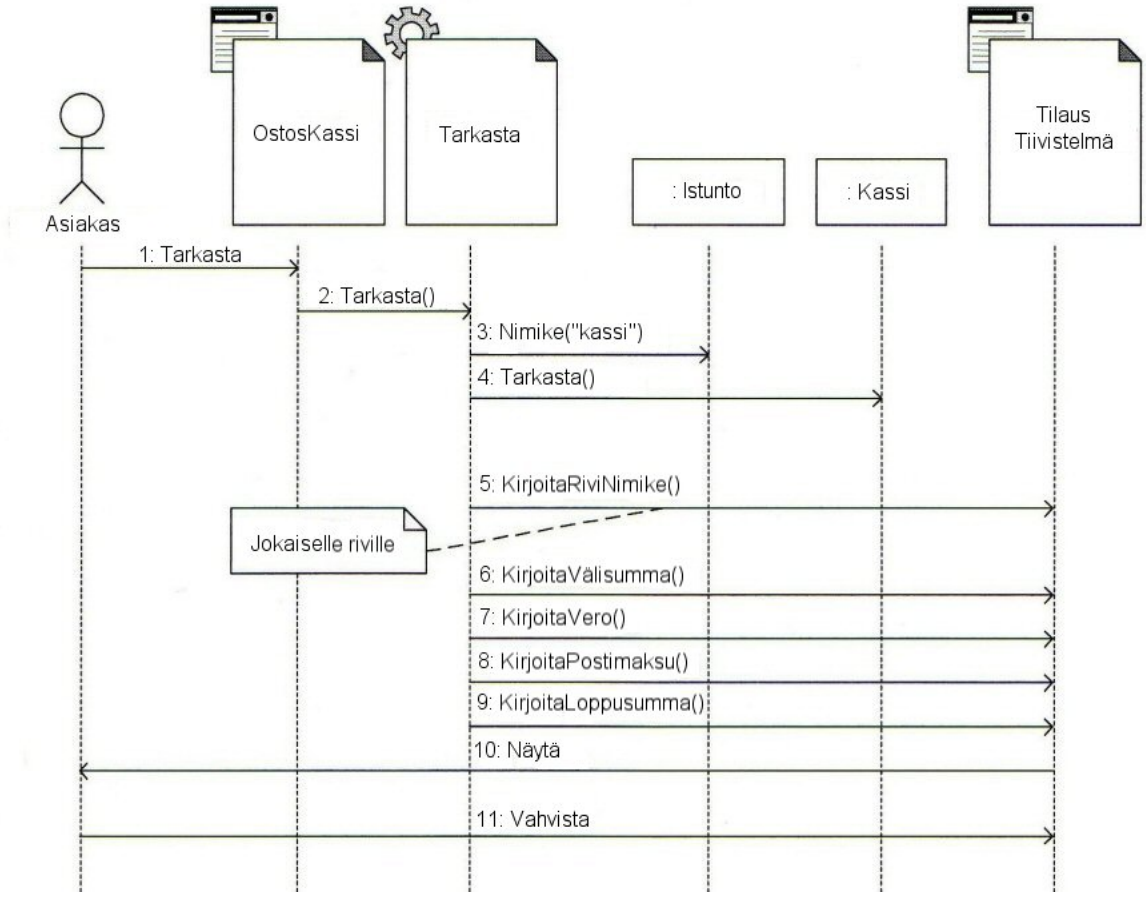
	Selainsivu	Palvelinsivu	Kehysjoukko	Kohde	Lomake
Selainsivu	Linkki, kohdistettu linkki, ohjaa.	Linkki, kohdistettu linkki, ohjaa.	Linkki, kohdistettu linkki, ohjaa.	Riippuvuus	Kooste
Palvelinsivu	Rakentaa, ohjaa.	Ohjaa	Ohjaa, rakentaa.		
Kehysjoukko	Kehyksen sisältö.		Kehyksen sisältö.	Kehyksen sisältö.	
Kohde					
Lomake	Kooste	Toimittaa			



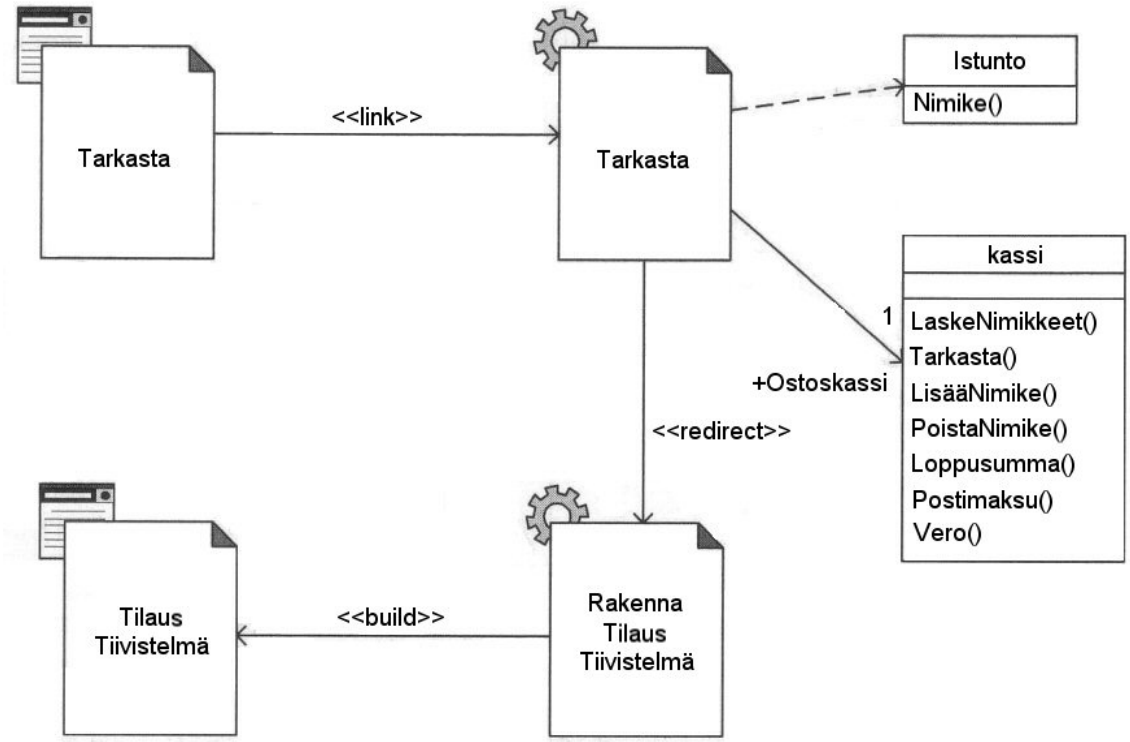
KUVIO 16. WAE-stereotyyppien graafiset kuvakkeet (vrt. Conallen 2000, 222–230).

Kuviossa 17 on esimerkki tapahtumakaaviosta, jonka mukaan asiakas tarkistaa ostoskassin sisällön. Esimerkistä käy ilmi selainsivun ja palvelinsivun käyttö sekä sivujen vuorovaikutus järjestelmän muiden olioiden kanssa. Esimerkin mukaan asiakas klikkaa OstosKassi nimisellä selainsivulla linkkiä tarkasta. Selainsivu lähettää kyseisen pyynnön Tarkasta nimiselle palvelinsivulle. Tämä palvelinsivu hakee tarvittavat tiedot istunto ja kassi olioilta sekä kirjoittaa tiivistelmän tilaustiedoista TilausTiivistelmä nimiselle selainsivulle, joka näytetään asiakkaalle. Kuviossa 18 esitetään edelliseen esimerkkiin liittyvä luokkakaavio. Samassa kaaviossa näkyvät sekä WWW-sivut että liiketoimintaluokat. Erona edelliseen esimerkkiin on, että tässä kaaviossa palvelinsivu Tarkasta ei rakenna itse selainsivua TilausTiivistelmä vaan ohjaa

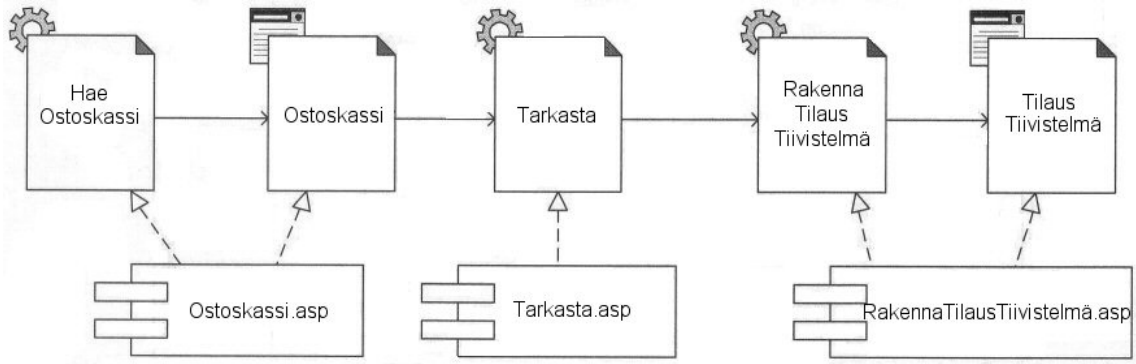
kyseisen sivun rakentamisen erilliselle palvelinsivulle RakennaTilausTiivistelmä. Esimerkin luokkakaaviossa ei esitetä sivujen sisältämiä attribuutteja eikä operaatioita. Yleensä ainakin palvelinsivut sisältävät useita operaatioita ja attribuutteja. Hyvä tapa on muodostaa ensin luokkakaavio ilman yksityiskohtia ja keskittyä tämän jälkeen miettimään sivujen tarkempaa sisältöä. Kuviossa 19 esitetään esimerkki komponenttikaaviosta. Kuviossa on kolme komponenttia, joista jokainen esittää WWW-sivua, jotka toteutetaan lopulliseen järjestelmään. Esimerkiksi komponentti Ostoskassi.asp käsittää palvelinsivun HaeOstoskassi ja selainsivun OstosKassi.



KUVIO 17. WAE-tapahtumakaavio (vrt. Conallen 2000, 153).



KUVIO 18. WAE-luokkakaavio (vrt. Conallen 2000, 156).



KUVIO 19. WAE-komponenttikaavio (vrt. Conallen 2000, 156).

5.4 Yhteenveto

Tässä luvussa tarkasteltiin WWW-sovelluksia yleisesti ja WWW-sovellusalueelle kehitettyjä UML-laajennuksia. Ensin luotiin yleiskatsaus WWW-sovelluksiin sekä määriteltiin WWW-sovelluksen käsite. Seuraavaksi tuotiin esille WWW-sovellusten kehittämistyön erityispiirteitä. Havaittiin, että WWW-sovellusten kehittämistyölle on ominaista navigoinnin, käyttöliittymän, arkkitehtuurin, tietosisällön sekä käyttäytymisen mallintaminen. Näitä ominaispiirteitä voidaan käyttää hyödyksi vertailtaessa UML:n laajennuksien sopivuutta WWW-sovellusalueeseen. Tämän jälkeen esiteltiin yksityiskohtaisesti kolme WWW-sovellusalueelle kehitettyä UML:n laajennusta. Nämä laajennukset olivat nimeltään UWE, W2000 sekä WAE. Laajennuksista kuvattiin sekä käsitteellistä perustaa että notaatiota. Erityisesti keskityttiin esittelemään laajennuksissa esiteltyjä stereotyyppejä.

Luvun keskeinen tarkoitus oli tutkia WWW-sovellusten erityispiirteitä ja esitellä sovellusalueelle esitetyt laajennukset yksityiskohtaisesti. Seuraavassa luvussa vertaillaan ja analysoidaan tässä luvussa esitettyjä laajennuksia. Vertailu pohjautuu luvussa 4 esitettyyn viitekehykseen ja tässä luvussa määriteltyihin WWW-sovellusten kehittämisen ominaispiirteisiin.

6 LAAJENNUSTEN ARVIOINTI JA VERTAILU

Tässä luvussa arvioidaan ja vertaillaan kolmea edellisessä luvussa esiteltyä UML-laajennusta. Nämä laajennukset ovat nimeltään UWE (esim. Hennicker & Koch 2001a), W2000 (esim. Baresi, Garzotto & Paolini 2001) sekä WAE (esim. Conallen 2000). Laajennusten vertailu pohjautuu luvussa 4 muodostettuun viitekehykseen. Tämän luvun rakenne vastaa viitekehyksessä esitettyä luokkajakoa: jokaista laatuluokkaa vastaa tässä luvussa erillinen kohta. Luku päättyy yhteenvetoon.

6.1 Laajennusten sopivuus sovellusalueeseen

Tässä kohdassa vertaillaan laajennusten sopivuutta sovellusalueeseen. Toisin sanoin selvitetään, kuinka ilmaisuvoimaisia laajennukset ovat. Ensimmäisen viitekehyksen sisältämän kriteerin mukaan käsitteellisen perustan tulisi olla niin laaja, että kaikki sovellusalueelle kuuluva voidaan mallintaa. Toisen käsitteelliseen perustaan liittyvän kriteerin mukaan sovellusalueeseen kuulumatonta ei tulisi voida mallintaa. Notaation osalta viitekehyksessä on yksi kriteeri. Sen mukaan ulkoinen ilmaisu ei saisi sekoittaa käsitteellistä perustaa. Seuraavassa tarkastellaan aluksi ensimmäiseen kriteeriin liittyviä seikkoja. Tämän jälkeen esitetään lyhyesti huomioita jälkimmäisten kriteerien osalta.

Muodostetussa viitekehyksessä ei eritellä sovellusaluekohtaisesti sitä, mikä on ominaista kullekin sovellusalueelle. Täytyy siis selvittää erikseen, mitä erityisiä mallinnustarpeita kullakin sovellusalueella on. Tämän kohdan arviointi ja vertailu pohjautuu kohdassa 5.2 esitettyihin WWW-sovellusten kehittämistyön ominaispiirteisiin. Sellaisia erityispiirteitä, jotka kuuluvat WWW-sovellusten mallintamiseen, ovat käyttöliittymän, navigoinnin ja sovelluksen arkkitehtuurin mallintaminen. Lisäksi mallinnuskielen avulla täytyy olla mahdollista mallintaa luonnollisesti myös sovelluksen tietosisältö ja sen rakenteet sekä toiminnallisuus samaan tapaan kuin tavallisten järjestelmien yhteydessä.

Vertailussa mukana olevien laajennusten kattavuutta edellä mainittujen ominaispiirteiden suhteen voidaan arvioida ja vertailla vertailemalla laajennuksissa esitettyjä stereotyyppejä, sillä lähes kaikki laajennusten uudet ominaisuudet on toteutettu stereotyyppien avulla. Laajennuksissa esitellään myös joitakin ominaisuuksia ja nimettyjä arvoja sekä rajoituksia. Nämä laajennusmekanismit ovat kuitenkin yleensä alisteisia stereotyypeille, niitä voidaan käyttää siis stereotyyppien yhteydessä.

Laajennusten kattavuutta WWW-sovellusten ominaispiirteiden suhteen pyritään havainnollistamaan taulukon 9 avulla. Taulukossa stereotyypit on luokiteltu WWW-sovellusten kehittämisen ominaispiirteiden mukaan. Taulukon riveillä pyritään esittämään laajennuksissa esitettyjen stereotyyppien vastaavuudet. Huomioitavaa taulukossa on se, että se kattaa WWW-sovellusten kehittämisen ominaispiirteistä vain tietosisällön, navigoinnin ja käyttöliittymän mallintamisen. Toiminnallisuuden ja arkkitehtuurin mallintamisen osalta vastaavaa taulukkomuotoista vertailua ei ole järkevää tehdä, koska vastaavuuksia stereotyyppien väliltä ei löydy. Toinen huomioitava seikka taulukossa on se, että siinä ei esiinny ainuttakaan WAE-profiilin stereotyyppiä. WAE-profiili on lähtökohdiltaan hyvin erilainen verrattuna UWE-menetelmään ja W2000-viitekehykseen. UWE ja W2000 pohjautuvat hypermedian mallintamisessa käytettäviin menetelmiin. Sen sijaan WAE-profiilissa esitellään paljon WWW-sovellusten toteutustekniikoihin läheisesti liittyviä stereotyyppejä. Tämän takia suoria vastaavuuksia stereotyyppien välillä ei juuri ole. UWE-menetelmän ja W2000-viitekehyksen osalta taulukko kattaa kaikki niissä esiintyvät stereotyypit. Näidenkin laajennusten osalta vain harvojen stereotyyppien väliltä löytyy suoria vastaavuuksia. Mikäli stereotyyppien välillä ei löydy suoria vastaavuuksia, taulukossa kerrotaan kyseisessä kohdassa, kuinka kyseinen seikka on huomioitu laajennuksessa, vai onko asiaa huomioitu ollenkaan. Seuraavassa kerrotaan yksityiskohtaisesti laajennusten välisistä eroista WWW-sovellusten erityispiirteiden mallinnuksessa.

TAULUKKO 9. Laajennusten stereotyyppien vastaavuudet.

Mallintamisen erityispiirre	**UWE**	**W2000**	**WAE**
Tietosisältö	Mallinnetaan perus UML:n mukaisesti	Entiteettityyppi	Mallinnetaan perus UML:n mukaisesti
		Komponentti	
		Paikka	
		Semanttinen assosiaatio	
		Assosiaatiokeskus	
		Kokoelma	
		Kokoelmakeskus	
Navigointi	Navigointiluokka	Solmu	Ei ole olemassa suoria vastaavuuksia. WAE-profiilin mukaan navigointia voidaan mallintaa vain sivujen välillä.
	Navigointisuhde	Linkki	
	Indeksi	Navigointimallin saantitasolla voidaan käyttää vastaavia hakuelementtejä kuin UWE-laajennuksessa. Näitä stereotyyppejä ei kuitenkaan esitellä.	
	Johdettu läpikäynti		
	Kysely		
	Valikko		
Käyttöliittymä	Kehysjoukko	W2000 ei mahdollista käyttöliittymän mallintamista.	WAE ei mahdollista käyttöliittymän mallintamista.
	Esityksellinen luokka		
	Teksti		
	Tekstilinkki		
	Painike		
	Kuva, ääni, video		
	Lomake		
	Kokoelma		
	Tekstilinkkikokoelma		
	Ikkuna		

Kaikki vertailussa mukana olevat laajennukset mahdollistavat WWW-sovellusten tietosisällön mallintamisen. Tietosisältö mallinnetaan laajennuksesta riippumatta UML:n luokkakaavion avulla. UWE-menetelmässä mallia, jonka avulla tietosisältö mallinnetaan, kutsutaan käsitteelliseksi malliksi. W2000-viitekehyksessä kyseistä mallia kutsutaan informaatiomalliksi. WAE-profiilissa ei varsinaisesti esitellä uutta mallia tietosisällön mallintamiseen, vaan käytetään tavallista UML:n luokkakaaviota. W2000-viitekehyksessä esitetään joukko uusia stereotyyppejä, joita voidaan käyttää mallinnettaessa WWW-sovelluksen tietosisältöä.

Navigoinnin mallintamiseen esitetään kattavin joukko stereotyyppejä UWE-menetelmässä. Tässä menetelmässä käytetään navigoitavista olioista stereotyyppiä navigointiluokka. Navigointipolkuja navigointiluokkien välillä kuvataan navigointisuhteilla. W2000-viitekehyksessä navigointiluokkaa vastaa solmu ja navigointisuhdetta linkki. Edellä mainittuja käsitteitä käytetään UWE-menetelmässä navigoinnin tilamallissa ja W2000-viitekehyksessä navigointimallin perustasolla. UWE-menetelmässä esitellään myös stereotyypit indeksi, johdettu läpikäynti, kysely sekä valikko hakuelementeille. Näitä stereotyyppejä voidaan käyttää navigoinnin rakennemallissa määriteltäessä, millaisin keinoin navigointi oliosta toiseen tapahtuu. Vastaavia hakuelementtejä voidaan käyttää W2000-viitekehyksen navigointimallin saantitasolla. Kyseisiä stereotyyppejä ei kuitenkaan esitellä viitekehyksessä. WAE-profiilissa navigointiin liittyviä seikkoja ei ole otettu yhtä hyvin huomioon kuin UWE-menetelmässä ja W2000-viitekehyksessä. Suoria vastaavuuksia WAE-profiilissa esitetyistä stereotyypeistä on vaikea löytää suhteessa W2000-viitekehykseen ja UWE-menetelmään. WAE-profiilin avulla voidaan kuitenkin mallintaa navigointia sivujen välillä käyttämällä selainsivuja, palvelinsivuja ja linkkejä. Tapa on siis hyvin erilainen verrattuna UWE-menetelmän ja W2000-viitekehyksen tapaan mallintaa navigointia.

Käyttöliittymän mallintaminen huomioidaan vain UWE-menetelmässä, jossa esitellään 12 stereotyyppiä, joiden avulla voidaan mallintaa WWW-sovelluksen käyttöliittymä. UWE-menetelmässä huomioidaan muun muassa erilaisten medioiden kuten tekstin, kuvan, videon ja äänen mallintaminen käyttöliittymässä.

WWW-sovelluksen toiminnallisuuden mallintaminen huomioidaan parhaiten WAE-profiilissa. Profiilissa esitellään 23 läheisesti WWW-sovellusten toteutustekniikoihin liittyviä stereotyyppiä. Suurinta osaa stereotyypeistä voidaan käyttää luokkakaavioissa mallinnettaessa sovelluksen arkkitehtuuria ja tapahtumakaavioissa mallinnettaessa tarkemmin sovelluksen toiminnallisuutta. Malleja tehtäessä joudutaan ottamaan kantaa läheisesti käyttäviin toteutustekniikoihin. Tätä voidaan pitää myös huonona puolena. Myös W2000-viitekehyksen avulla voidaan mallintaa WWW-sovelluksen toiminnallisuutta. Tämä tehdään skenaariokaavioiden avulla. WWW-sovelluksen toiminnallisuutta ei pystytä kuitenkaan kuvaamaan yhtä hyvin kuin WAE-profiilin avulla, koska laajennuksessa ei esitellä tarvittavia stereotyyppejä, kuten selainsivuja ja palvelinsivuja. UWE-menetelmässä toiminnallisuuden mallintaminen on otettu huonoiten huomioon. Menetelmän avulla voidaan mallintaa vain käyttöliittymän muutoksia dynaamisen esitysmallin avulla.

WWW-sovelluksen arkkitehtuurin mallintamisen osalta WAE-profiili on ylivoimaisesti paras. WAE-profiilissa esitetään kattava joukko stereotyyppejä, joita voidaan käyttää luokkakaavioissa mallinnettaessa WWW-sovelluksen arkkitehtuuria. Osaa stereotyypeistä voidaan käyttää myös komponenttikaavioissa. Lähtökohtana arkkitehtuurin mallintamisessa käytetään sivujen mallintamista. Sivut toimivat yhdistävänä tekijänä selaimen ja muun järjestelmän välillä. UWE-menetelmä ja W2000-viitekehys eivät mahdollista sivujen mallintamista.

Toisen käsitteellistä perustaa koskevan kriteerin mukaan sovellusalueeseen kuulumatonta ei tulisi voida mallintaa. Kaikki kolme laajennusta täyttävät tämän kriteerin. Laajennuksissa ei esitetä stereotyyppejä, jotka eivät kuulu sovellusalueelle.

Notaatiota koskien tämä laatuluokka sisältää vain yhden kriteerin. Sen mukaan ulkoinen ilmaisu ei saisi sekoittaa käsitteellistä perustaa. Jokaiselle käsitteellisessä perustassa esitetylle ilmaukselle tulisi olla yksikäsitteinen esitystapa. W2000-viitekehyksessä assosiaatiokeskuksesta ja kokoelmakeskuksesta käytetään samaa stereotyyppimerkintää Center Type, joten tässä kohdalla sääntö rikkoutuu. Muilta osin laajennukset noudattavat tätä kriteeriä.

Yhteenvetona voidaan todeta, että yksikään vertailussa mukana olleista laajennuksista ei kata kaikkia WWW-sovellusten kehittämiselle ominaisia piirteitä. Tätä havaintoa voidaan pitää yllätyksenä, sillä laajennusten olettaisi sopivan hyvin sovellusalueelleen. Laajennuksillahan nimenomaan pyritään vastaamaan yksittäisen sovellusalueen vaatimuksiin. Laajennuksilla on selkeästi omat vahvuusalueensa. UWE on ainoa laajennus, joka kattaa käyttöliittymän suunnittelun. Lisäksi UWE sisältää kattavimman joukon navigoinnin mallintamiseen käytettäviä stereotyyppejä. W2000 puolestaan on hyvin tietokeskeinen, jossa huomioidaan myös navigoinnin mallintaminen. WAE on selkeästi arkkitehtuurikeskeinen, jossa huomioidaan hyvin myös WWW-sovelluksen toiminnallisuuden mallinnus.

6.2 Osallisten tietämys kielestä

Tässä kohdassa vertaillaan osallisten tietämystä kielestä. Tämän laatuluokan ensimmäisen kriteerin mukaan kielen käsitteellisen perustan tulisi vastata mahdollisimman paljon sitä, miten yksilöt kokevat todellisuuden. Toisen kriteerien mukaan laajennusten käytön opettelun kynnys pysyy matalana, mikäli laajennuksessa esitetään mahdollisimman vähän uutta verrattuna perus

UML:ään. Käsitteiden esityksellisten vastineiden tulisi myös olla ymmärrettäviä ja kohdettaan kuvaavia.

Vertailussa mukana olevissa laajennuksissa uusien käsitteiden esittely pohjautuu lähes poikkeuksetta stereotyyppien käyttöön. Laajennuksissa esitellyt stereotyypit vastaavat pääsääntöisesti hyvin todellisuutta ja ovat helposti opittavissa. Varsinkin UWE-menetelmän ja WAE-profiilin kohdalla stereotyyppien nimet ovat pääosin hyvin informatiivisia. Esimerkiksi UWE-menetelmässä esitettyjen käyttöliittymän mallintamiseen tarkoitettujen stereotyyppien, kuten teksti, tekstilinkki, painike, lomake jne. käyttötarkoitus on hyvin selkeä. WAE-profiilin stereotyypeistä selainsivu ja palvelinsivu ovat hyviä esimerkkejä informatiivisista stereotyypeistä. Eniten stereotyyppien nimissä olisi hiottavaa W2000-viitekehykessä. Esimerkiksi stereotyypit entiteettityyppi, paikka sekä kokoelmakeskus eivät kuvaa parhaalla mahdollisella tavalla todellisuuden vastaavia ilmiöitä. Osuvampia nimiä entiteettityypille ja paikalle olisivat olleet esimerkiksi UML:ssä yleisesti käytettävät luokka ja attribuutti.

UWE-menetelmässä esitetään kaikille ja WAE-profiilissa tärkeimmille stereotyypeille myös graafiset vastineet. Sen sijaan W2000-viitekehyksessä ei esitetä stereotyypeille graafisia vastineita. Kaikista esitetyistä kuvakkeista ei pysty suoraan päättelemään, mitä stereotyyppiä niillä tarkoitetaan, mutta ne lisäävät kuitenkin kaavioiden havainnollisuutta. Muun muassa UWE-menetelmän stereotyyppien tekstilinkki, teksti, kuva, lomake ja navigointiluokka graafiset vastineet voisivat olla enemmän kohdettaan kuvaavia. Esimerkiksi navigointiluokasta käytetään kuvakkeena suorakaidetta, jota on hankala yhdistää navigointiluokan käsitteeseen. WAE-profiilin kuvakkeista ainakin WWW-sivun ja servletin kuvakkeita voisi kehittää enemmän kohdettaan kuvaaviksi. WWW-sivun kuvakkeessa on keskellä ympyrä ja tämän alla neljä poikkiviiva. Tätä kuvaketta on vaikea yhdistää WWW-sivun käsitteeseen.

Kaikki vertailussa mukana olevat laajennukset perustuvat UML:n sisäisten laajennusmekanismien käyttöön, joten niitä voidaan pitää UML-profiileina. Tämä on hyvä seikka osallisten tietämyksen kannalta. Laajennusten käytön opettelu on melko vaivatonta, sillä uudet käsitteet ovat UML:n mukaisia.

Yhteenvetona voidaan todeta, että osallisten tietämyksen osalta laajennusten välillä ei ole suuria eroja. Kaikkia laajennuksia voidaan pitää kohtalaisen helposti opittavina. UWE-menetelmän ja WAE-profiilin stereotyyppejä voidaan pitää kuitenkin hieman enemmän todellisuuden ilmiöitä vastaavina, kuin W2000-viitekehyksen stereotyyppejä. UWE ja WAE mahdollistavat myös stereotyyppien graafisten vastineiden käytön. Joidenkin kuvakkeiden ulkoasussa on kuitenkin parantamisen varaa.

6.3 Käsitettävyys

Käsitettävyydellä tarkoitetaan sitä, että mallinnuskielen käyttäjän tulisi voida ymmärtää mallinnuskielen mukaiset ilmaisut. Tässä kohdassa käsitettävyyttä arvioidaan useiden viitekehyksessä esitettyjen kriteerien pohjalta. Ensin laajennuksia vertaillaan käsitteellisen perustan osalta ja tämän jälkeen notaation osalta.

Ensimmäisen käsitettävyyden laatuluokkaan kuuluvan kriteerin mukaan kielen käsitteiden tulisi olla helposti eroteltavissa toisistaan. Vertailussa mukana olevissa laajennuksissa uudet käsitteet on nimetty yksilöllisillä nimillä, joten niiden erottaminen toisistaan on helppoa.

Toisen kriteerin mukaan käsitteiden lukumäärän tulisi olla kohtuullinen. Mikäli käsitteiden lukumäärä on iso, niiden pitäisi olla organisoitu hierarkkisesti. Laajennuksissa lähes kaikki uudet käsitteet esitetään stereotyyppien muodossa, joten on loogista vertailla stereotyyppien lukumäärää. Eniten uusia stereotyyppejä esitetään WAE-profiilissa yhteensä 23 kappaletta. Vähiten stereotyyppejä esitetään W2000-viitekehyksessä, vain 9 kappaletta. UWE-

menetelmässä esitellään 18 uutta stereotyyppiä. Käsitteiden määrä vaihtelee siis laajennusten välillä kohtalaisen paljon. Tähän on syynä lähinnä se, että laajennukset kattavat eri alueita WWW-sovellusten ominaispiirteiden mallintamisesta, kuten kohdassa 6.1 havaittiin. UWE-menetelmän suuri käsitteiden määrä selittyy sillä, että laajennus on ainoa, jossa huomioidaan myös käyttöliittymän suunnittelu. UWE-menetelmän stereotyypeistä jopa 12:ta käytetään käyttöliittymän mallintamiseen. WAE-profiilin suuri käsitteiden määrä selittyy osittain sillä, että suuri osa niistä on toteutusteknologiariippuvaisia. Lisäksi arkkitehtuurin mallintamiseen tarvitaan useita erilaisia stereotyyppejä. W2000-viitekehyksen stereotyypit liittyvät vain WWW-sovelluksen tietosisällön ja navigoinnin mallintamiseen. Tämän takia stereotyyppien määrä on joukon pienin. Huomattavaa on se, että harvassa mallinnettavassa järjestelmässä löytyy kaikille esitetyille stereotyypeille käyttöä. Kaikkien stereotyyppien opettelu ei ole siis yleensä tarpeellista. Missään vertailussa mukana olleessa laajennuksessa stereotyyppejä ei ole organisoitu hierarkkisesti. Kuitenkin se, missä kaavioissa stereotyyppejä voidaan hyödyntää, kerrotaan pääsääntöisesti.

Kolmantena kriteerinä käsitettävyyden laatuluokassa esitetään, että käsitteiden käytön tulisi olla yhdenmukaista. Samojen käsitteiden käyttö usean ilmiön yhteydessä tai eri käsitteiden käyttö saman ilmiön yhteydessä tekee kielestä sekavan. UWE-menetelmässä ja WAE-profiilissa käsitteiden käyttöä voidaan pitää yhdenmukaisena. Sen sijaan W2000-viitekehyksessä käsitteiden käyttöä ei voida pitää täysin yhdenmukaisena. W2000-viitekehyksessä assosiaatiokeskukselle ja kokoelmakeskukselle esitellään sama stereotyyppi nimeltä Center Type. Assosiaatiokeskus sisältää tietoa assosiaatiosta ja kokoelmakeskus sisältää tietoa kokoelmasta. Käsitteiden käyttötapa on siis hyvin samankaltainen. Tästä huolimatta olisi hyvän tavan mukaista käyttää eri käsitteistä eri stereotyyppejä mahdollisten sekaannusten välttämiseksi. Lisäksi W2000-viitekehyksessä esitellään joukko stereotyyppejä, jotka on tarkoitettu

sovelluksen tietosisällön mallintamiseen. Nämä stereotyypit ovat suurimmalta osaltaan turhia. Sovelluksen tietosisältö voitaisiin mallintaa perus UML:ää käyttämällä. Tämä käytäntö onkin käytössä muissa laajennuksissa. Esimerkiksi entitettityyppi voitaisiin mallintaa luokan avulla, komponentit voitaisiin mallintaa koostesuhteiden avulla ja paikat voitaisiin mallintaa attribuutteina.

Neljännen kriteerin mukaan kielen tulisi olla joustava yksityiskohtien määrän esittämisen suhteen. Ilmauksien tulisi olla helposti laajennettavissa ilmauksilla, jotka tuovat esille enemmän yksityiskohtia. Toisaalta yksityiskohtien tulisi olla helposti piilotettavissa. Tämä on huomioitu parhaiten W2000-viitekehyksessä. Viitekehyksen mukaan informaatiomallista ja navigointimallista voidaan muodostaa yksityiskohtaisia kaavioita ja kaavioita, joissa ei puututa yksityiskohtiin. Käytännössä tämä tarkoittaa sitä, että kaavioissa, joissa ei haluta puuttua yksityiskohtiin, luokkiin ei sisällytetä attribuutteja eikä operaatioita. Vastaavanlaisia kaavioita voitaisiin tehdä aivan hyvin myös UWE-menetelmän ja WAE-profiilin mukaisesti. Näiden laajennusten osalta tätä mahdollisuutta ei kuitenkaan erityisesti painoteta. Jokaisessa laajennuksessa monien kaavioiden tekeminen perustuu aikaisemmassa vaiheissa tuotettuihin kaavioihin. Tällä tavoin kaavioita täydennetään ja niihin tuodaan lisää tarvittavia yksityiskohtia. Esimerkiksi W2000-viitekehyksen informaatiomallin ja navigointimallin perustasoja voidaan täydentää saantitasojen käsitteillä. UWE-menetelmässä navigoinnin tilamalli perustuu käsitteelliseen malliin ja navigoinnin rakennemalli perustuu navigoinnin tilamalliin. WAE-profiilissa täydennetään analyysivaiheessa muodostettuja kaavioita.

Viidentenä käsitteelliseen taustaan liittyvänä kriteerinä on mallien jaettavuus luonnollisiin osiin. Tätä mahdollisuutta ei tuoda erikseen esiin laajennusten yhteydessä. Mallien jaettavuus on kuitenkin mahdollista UML:n mukaisesti. Tehdyt laajennukset eivät rajoita tätä mahdollisuutta mitenkään. Esimerkiksi luokkakaavioita voidaan tehdä useita järjestelmän eri osista.

Ensimmäisen notaatiota koskevan kriteerin mukaan symbolien erottaminen toisistaan pitäisi olla helppoa. Laajennuksissa esitellään uusia symboleita lähinnä vain stereotyyppien muodossa. Laajennuksien stereotyypeillä on yksilölliset nimet, joten symbolien erottaminen tapahtuu lähinnä nimien perusteella. UWE-menetelmässä ja WAE-profiilissa on mahdollista käyttää myös graafisia vastineita stereotyypeistä. Tämä auttaa symbolien erottamisessa. Graafisten kuvakkeiden erottaminen toisistaan on helpompaa kuin niiden tekstuaalisten vastineiden.

Toisen notaatioon liittyvän kriteerin mukaan symbolien käytön tulisi olla yhdenmukaista. UML:ssä stereotyypit voidaan ymmärtää käsitteinä ja niiden tekstuaaliset ja graafiset vastineet voidaan käsittää symboleina. Laajennuksien välillä on hieman erilainen tapa käyttää kyseisiä symboleita. W2000-viitekehyksen tapa on puhtaasti tekstuaalinen. UWE-menetelmässä voidaan käyttää stereotyyppien kuvaamiseen tekstuaalista tapaa taikka stereotyyppien graafisia vastineita. Joissain UWE-menetelmän kaavioissa graafisia kuvakkeita voidaan käyttää itsenäisinä symboleina ja joissain kaavioissa upotettuina symboleina. Esimerkiksi navigoinnin rakennemallissa voidaan käyttää hakuelementeistä itsenäisiä graafisia kuvakkeita. Kehysjoukossa ja esityksellisessä luokassa voidaan käyttää puolestaan kaavioihin upotettuja graafisia symboleita. WAE-profiilissa kaikki stereotyypit voidaan kuvata tekstuaalisesti, lisäksi joillakin stereotyypeillä on graafiset vastineet. Stereotyyppien graafiset kuvakkeet on kiinnitetty kiinteästi symbolien yhteyteen ja kuvakkeiden käyttötapa on kaaviosta riippumatta sama. Yhtenäisyyttä kuvakkeiden käytössä lisää se, että sivua tarkoittavilla kuvakkeilla on pieni ”koirankorva” oikeassa ylänurkassa. Kaikissa laajennuksissa symboleita voidaan käyttää yhdenmukaisesti. Koska laajennuksissa on erilaisia tapoja mallintaa käsitteitä, kaavioiden yhdenmukaisuus riippuu mallintajan tavasta käyttää symboleita.

Kolmannen notaatioon liittyvän kriteerin mukaan symbolien tulisi olla yksinkertaisia. Laajennuksissa käytettävät symbolit ovat perus UML:n mukaisia lukuun ottamatta stereotyyppien tekstuaalisia ja graafisia vastineita. Laajennuksissa esitettyjä symboleita voidaan pitää pääsääntöisesti melko yksinkertaisina. Jotkin symbolit ovat kuitenkin huomattavasti toisia yksinkertaisempia. Esimerkiksi UWE-menetelmän navigointiluokan graafinen symboli on pelkkä suorakaide. Sen sijaan esityksellisen luokan symboli on jo paljon monimutkaisempi. Kaikissa laajennuksissa esitetyt symbolit ovat kuitenkin niin yksinkertaisia, että niiden mallintaminen on mahdollista sekä tietokoneen avulla että käsin.

Neljäntenä kriteerinä viitekehyksessä notaatiolta vaaditaan, että korostuksien käytön tulisi olla yhteydessä ilmaisujen tärkeyteen mallissa. Vertailussa mukana olleissa laajennuksissa ei käytetä korostuksia, joten tähän seikkaan ei voida ottaa kantaa.

Viimeisenä kriteerinä notaatioon liittyen on, että symbolien sommittelu kaavioissa tulisi voida tehdä esteettisesti miellyttäväksi. Symbolien sommittelu laajennuksissa ei eroa siitä tavasta, jolla perus UML:ssä voidaan sommitella symboleita eri kaavioissa. Symbolien sommittelun esteettisyys riippuu myös paljon käytettävän CASE-välineen tavasta sommitella symboleita.

Yhteenvetona voidaan todeta, että käsitettävyyden suhteen laajennusten välillä on melko vähän eroavaisuuksia. Tämän voidaan olettaa johtuvan siitä, että laajennuksilla on yhteisenä perustana UML. Käsitteellisen taustan osalta laajennusten välillä vallitsee suurin ero käsitteiden lukumäärässä ja käsitteiden käytön yhdenmukaisuudessa. Uusien käsitteiden lukumäärän eroavaisuus johtuu hyvin pitkälti siitä, että laajennukset keskittyvät WWW-sovellusten eri ominaisuuspiirteiden mallintamiseen. Käsitteiden käytön yhdenmukaisuudessa W2000-viitekehys pärjäsi tästä joukosta heikoiten. Tämän laajennuksen käsitteiden käyttöä ei voida pitää täysin yhdenmukaisena. WAE-profiilin ja

UWE-menetelmän käsitteiden käyttö on yhdenmukaista. Notaatioon liittyen suurin ero laajennusten välillä on stereotyyppien kuvaamiseen käytetyssä tavassa. Kaikissa laajennuksissa voidaan käyttää puhtaasti tekstuaalista tapaa. UWE-menetelmässä ja WAE-profiilissa stereotyypeistä voidaan käyttää myös graafisia kuvakkeita. Laajennusten tavassa käyttää kuvakkeita on tiettyjä eroavaisuuksia.

6.4 Teknisten toimijoiden tulkittavuus

Teknisten toimijoiden tulkittavuutta voidaan pitää tärkeänä, sillä yleensä mallinnuksessa käytetään apuna jotakin välinettä. Vertailussa mukana olevat laajennukset ovat UML-profiileja. Tämän takia profiilien mukaisten kaavioiden tekeminen pitäisi olla mahdollista yleisimmillä CASE-välineillä. CASE-välineisiin täytyy kuitenkin määritellä käytettävät stereotyypit. Rational Rose-välineeseen on saatavilla valmiiksi tuki WAE-profiilin käsitteille. UWE-menetelmän käsitteille on saatavana tuki ArgoUML:ään. W2000-viitekehyksen tekijät eivät suosittele käytettäväksi mitään tiettyä CASE-välinettä.

6.5 Stereotyypeille ominaiset vaatimukset

Tässä kohdassa vertaillaan, kuinka hyvin laajennuksissa esitetyt stereotyypit vastaavat viitekehyksessä niille asetettuihin vaatimuksiin. Käsittely perustuu neljään viitekehyksessä esitettyyn kriteeriin.

Ensimmäisen kriteerin mukaan stereotyyppien tulisi olla selkeästi määriteltyjä. Parhaiten stereotyypit on määritelty WAE-profiilissa, jossa kaikista stereotyypeistä esitetään metaluokka, johon stereotyyppi pohjautuu, tarkka kuvaus, käytettävä kuvake, rajoitteet sekä mahdollisesti käytettävät nimetyt arvot. Myös UWE-menetelmän stereotyypit on määritelty selkeästi. Määrityksissä ei ole kuitenkaan yhtä selkeää rakennetta stereotyyppien eri piirteille kuin WAE-profiilissa. Stereotyyppien käyttö on kuitenkin mahdollista

annettujen kuvauksien avulla. W2000-viitekehyksen stereotyypit on määritelty huonosti. Stereotyypeistä annetaan lähes poikkeuksetta vain lyhyet kuvaukset.

Toisen kriteerin mukaan stereotyyppien tulisi olla hyödyllisiä. Ensinnäkin stereotyypin tulisi määrittää uusi käsite tai piirre, joka ei sisälly peruskieleen. Toiseksi stereotyypin tulisi helpottaa mallinnuskielen käyttöä halutulla sovellusalueella. Pääsääntöisesti laajennuksissa esitettyjä stereotyyppejä voidaan pitää hyödyllisinä. UWE-menetelmässä esitettyjä stereotyyppejä voidaan pitää kaikkia tarpeellisina. W2000-viitekehyksessä määriteltyjen tiedon mallintamiseen käyttävien stereotyyppien hyödyllisyyttä voidaan pitää melko kyseenalaisena, sillä sovelluksen tietosisällön mallintaminen onnistuisi ilman kyseisiä stereotyyppejäkin. WAE-profiilissa esitettyjen teknologialäheisten stereotyyppien, kuten IIOP:n sekä RMI:n tarpeellisuus voidaan myös kyseenalaistaa. Tällaisia stereotyyppejä tarvitaan todella harvoin. Lisäksi kyseiset stereotyypit ovat sidosteisiä tiettyihin toteutusteknologioihin.

Kolmannen kriteerin mukaan stereotyyppien tulisi muodostaa johdonmukainen joukko. Stereotyyppi ei saa olla yhteensopimaton muiden stereotyyppien kanssa, ellei sitä ole selkeästi määritelty yhteensopimattomaksi tiettyjen stereotyyppien kanssa. Kaikissa laajennuksissa stereotyypeille määritelläänkin rajoitteita, jotka estävät niiden käyttöä tiettyjen stereotyyppien kanssa. Joidenkin stereotyyppien käyttö rajataan vain tiettyihin kaavioihin. Usein stereotyypit muodostavat siis kaavioiden sisäisesti johdonmukaisen joukon. WAE-profiilissa määritellään selkeästi taulukkomuodossa validit assosiaatiot stereotyyppien välillä. UWE-menetelmässä ja W2000-viitekehyksessä määritellään, missä kaavioissa stereotyyppejä voidaan käyttää. Lisäksi kyseisissä laajennuksissa stereotyypeille määritellään joitakin rajoitteita.

Viimeisen kriteerin mukaan laajennuksissa tulisi suosia kuvailevia ja rajoittavia stereotyyppejä ja välttää koristeellisia ja uudelleen määritteleviä stereotyyppejä. Jotta tätä kriteeriä voidaan arvioida, täytyy tehdä laajennuksissa käytettyjen

stereotyyppien luokitus. Luokittelussa käytetään Bernerin, Glinzin ja Joosin (1999) muodostamaa stereotyyppien luokitusta, joka on kuvattu yksityiskohtaisesti tämän tutkielman kohdassa 2.3.3. Seuraavaksi kerrataan kuitenkin lyhyesti, mitä erityyppisillä stereotyypeillä tarkoitetaan.

Koristeelliset stereotyypit muuntavat mallinnuskielen elementin syntaksia, eivätkä tee mitään muuta. Kuvailevat stereotyypit laajentavat tai muuntavat mallinnuskielen elementin abstraktia syntaksia ja määrittelevät uuden elementin pragmatiikan. Peruskielen semantiikka pysyy muuttumattomana. Rajoittavat stereotyypit ovat muuten samanlaisia kuin kuvailevat stereotyypit, mutta niiden avulla voidaan laajentaa uuden elementin semantiikka. Tyypillisesti semantiikan avulla määritetään rajoituksia elementeille. Uudelleen määrittelevät stereotyypit määrittelevät mallinnuselementin muuttaen sen alkuperäistä semantiikkaa. Näin voidaan luoda mallinnuselementti, joka ei koske mitään peruskielen elementtiä. Taulukosta 10 käy ilmi UWE-stereotyyppien luokittelu. Taulukossa 11 näkyy W2000-stereotyyppien luokitus, ja taulukossa 12 esitetään WAE-stereotyyppien luokitus.

TAULUKKO 10. UWE-stereotyyppien luokittelu.

Stereotyypin nimi	**Koristeellinen**	**Kuvaileva**	**Rajoittava**	**Uudelleen määrittelevä**
Navigointiluokka		x		
Navigointisuhde			x	
Indeksi			x	
Johdettu läpikäynti			x	
Kysely			x	
Valikko			x	
Kehysjoukko			x	
Esityksellinen luokka			x	

(jatkuu)

TAULUKKO 10. UWE-stereotyyppien luokittelu. (jatkuu)

Stereotyypin nimi	**Koristeellinen**	**Kuvaileva**	**Rajoittava**	**Uudelleen määrittelevä**
Teksti		x		
Tekstilinkki		x		
Painike		x		
Kuva		x		
Ääni		x		
Video		x		
Lomake		x		
Kokoelma			x	
Tekstilinkkikokoelma			x	
Ikkuna			x	

TAULUKKO 11. W2000-stereotyyppien luokittelu.

Stereotyypin nimi	**Koristeellinen**	**Kuvaileva**	**Rajoittava**	**Uudelleen määrittelevä**
Entiteettityyppi	x			
Komponentti	x			
Paikka	x			
Semanttinen assosiaatio			x	
Assosiaatiokeskus		x		
Kokoelma			x	
Kokoelmakeskus		x		
Solmu			x	
Linkki			x	

TAULUKKO 12. WAE-stereotyyppien luokittelu.

Stereotyypin nimi	**Koristeellinen**	**Kuvaileva**	**Rajoittava**	**Uudelleen määrittelevä**
Palvelinsivu			x	
Selainsivu			x	
Lomake			x	
Kehysjoukko			x	
Kohde			x	
Java-skriptiolio			x	
Selainskriptiolio			x	
Linkki			x	
Kohdistettu linkki			x	
Kehyksen sisältö			x	
Toimittaa			x	
Rakentaa			x	
Ohjaa			x	
IIOP			x	
RMI			x	
Syöttöelementti			x	
Valintaelementti			x	
Tekstialue-elementti			x	
WWW-sivu		x		
ASP-sivu		x		
JSP-sivu		x		
Servletti		x		
Skriptikirjasto		x		

Laajennuksissa esiteltyjen stereotyyppien luokittelu ei ollut kovin suoraviivainen tehtävä. Eniten luokittelua hankaloittivat stereotyyppien puutteelliset kuvaukset. Koristeellisten ja uudelleen määrittelevien stereotyyppien tunnistaminen oli pääsääntöisesti helppoa. Sen sijaan rajanveto sen suhteen kuuluuko stereotyyppi kategoriaan kuvailevat vai rajoittavat stereotyypit, on välillä hankalaa. Mikäli stereotyypille on määritelty selkeät rajoitteet, se on helppo kategorioida rajoittavaksi. Mutta jos stereotyypin määrityksessä ei selkeästi esitetä rajoitteita, on vaikeaa sijoittaa kyseistä stereotyyppiä oikeaan luokkaan. UWE-menetelmän osalta stereotyyppien luokittelua helpotti se, että joidenkin stereotyyppien osalta kerrottiin suoraan, mihin luokkaan ne kuuluvat. WAE-profiilin osalta puolestaan oli saatavilla taulukkomuodossa validit assosiaatiosuhteet tärkeimpien stereotyyppien välillä. Tämän taulukon perusteella suurin osa stereotyypeistä määriteltiin rajoittaviksi.

Edellä esitetyn stereotyyppiluokituksen mukaan UWE-menetelmän stereotyypeistä kahdeksan on luokaltaan kuvailevia ja kymmenen rajoittavia. WAE-profiilin stereotyypeistä suurin osa eli 18 on rajoittavia, kuvailevia on vain viisi. W2000-viitekehyksen stereotyyppijoukko koostuu kolmesta koristeellista, kahdesta kuvailevasta ja neljästä rajoittavasta. Yhdessäkään laajennuksessa ei esitellä uudelleenmääritteleviä stereotyyppejä. Tätä seikkaa voidaan pitää hyvänä puolena. UWE-menetelmän ja WAE-profiilin stereotyyppejä voidaan pitää luokittelun mukaan hyvin suunniteltuina, sillä kaikki stereotyypit sijoittuvat luokkaan kuvailevat tai rajoittavat stereotyypit. W2000-viitekehyksessä stereotyypit entiteettityyppi, komponentti ja paikka kuuluvat luokkaan koristeelliset stereotyypit. Kyseisiä stereotyyppejä voidaan pitää melko turhina, sillä ne eivät lisää kieleen mitään uutta. Myös semanttista assosiaatiota voidaan pitää melko turhana stereotyyppinä. Kyseinen stereotyyppi on kuitenkin luokaltaan rajoittava, koska sillä voidaan yhdistää vain entiteettityyppejä, komponentteja ja toisia assosiaatioita. Semanttinen

assosiaatio voitaisiin korvata tavallisella assosiaatiolla, mikäli entiteettityypit korvattaisiin luokilla, paikat korvattaisiin attribuuteilla ja komponentit koostesuhteisilla luokilla.

Yhteenvetona voidaan todeta, että UWE-menetelmän ja WAE-profiilin stereotyyppejä voidaan pitää valtaosin hyvin muodostettuina. Stereotyypit ovat selkeästi määriteltyjä, vaikuttaisivat hyödyllisiä ja ne muodostavat johdonmukaisen joukon. Huomattavaa on, että stereotyyppien hyödyllisyys ratkeaa lopulta kuitenkin vasta käyttökokemusten myötä. Kaikki kyseisissä laajennuksissa esitetyt stereotyypit kuuluvat myös luokkiin kuvailevat ja rajoittavat. W2000-viitekehyksen stereotyypit eivät ole kaikki hyvin muodostettuja. Osa stereotyypeistä on huonosti määritelty. Lisäksi osa stereotyypeistä on koristeellisia, toisin sanoin ne eivät vaikuta kovin hyödyllisiltä.

6.6 Yhteenveto

Tässä luvussa tehtiin WWW-sovellusalueen UML-laajennusten arviointi ja vertailu. Vertailussa olivat mukana: UWE-menetelmä (esim. Hennicker & Koch 2001a), W2000-viitekehys (esim. Baresi, Garzotto & Paolini 2001) sekä WAE-profiili (esim. Conallen 2000). Laajennusten vertailu jaettiin viiteen erilliseen kohtaan. Kyseinen kohtajako perustuu käytetyn viitekehyksen luokkajakoon.

Vertailussa havaittiin, että laajennuksista yksikään ei ole täysin sopiva sovellusalueeseen. Mikään laajennuksista ei kata kaikkia WWW-sovellusten kehittämiselle ominaisia piirteitä. Jokainen laajennus on kuitenkin ilmaisuvoimainen jonkin tietyn erityispiirteen osalta. UWE on ainoa laajennus, joka mahdollistaa WWW-sovelluksen käyttöliittymän suunnittelun. Lisäksi UWE-menetelmä sisältää kattavimman joukon navigoinnin mallintamiseen käytettäviä stereotyyppejä. W2000-viitekehystä voidaan pitää tietokeskeisenä, jossa huomioidaan myös navigoinnin mallintaminen. WAE-profiili on

arkkitehtuurikeskeinen, jossa huomioidaan hyvin myös WWW-sovelluksen toiminnallisuuden mallintaminen.

Osallisten tietämyksen osalta laajennusten välillä ei ole suuria eroja. Kaikkia laajennuksia voidaan pitää suhteellisen helposti opittavina. UWE-menetelmän ja WAE-profiilin etuna voidaan pitää mahdollisuutta käyttää stereotyypeistä myös graafisia vastineita.

Myös laajennusten välisessä käsitettävyydessä on vain vähän eroavaisuuksia. Tämän voidaan olettaa johtuvan siitä, että laajennuksilla on yhteisenä perustana UML. Eroja löytyi kuitenkin muun muassa käsitteiden lukumäärässä ja käsitteiden käytön yhdenmukaisuudessa. Esiteltyjen käsitteiden lukumäärien erot johtuvat siitä, että laajennukset kattavat mallintamisen eri alueita. Käsitteiden käytön yhdenmukaisuudessa W2000-viitekehyksessä olisi parantamisen varaa. WAE-profiilissa ja UWE-menetelmässä käsitteiden käyttöä voidaan pitää yhdenmukaisena.

Kaikkien laajennusten teknisten toimijoiden tulkittavuutta voidaan pitää hyvänä, sillä vertailussa mukana olleet laajennukset ovat kaikki UML-profiileja. Kaavioiden tekeminen pitäisi olla mahdollista yleisimmillä CASE-välineillä.

Stereotyyppejä koskien havaittiin, että UWE-menetelmän ja WAE-profiilin stereotyyppejä voidaan pitää pääosin hyvin muodostettuina. Sen sijaan W2000-viitekehyksen kaikkia stereotyyppejä ei voida pitää hyvin muodostettuina.

7 JOHTOPÄÄTÖKSET

Tässä luvussa esitetään tutkielman johtopäätökset jaoteltuina kolmeen osaan. Ensiksi analysoidaan tutkielman alussa tehdyn UML-laajennusten kartoituksen tuloksia ja merkitystä. Toiseksi tehdään johtopäätöksiä WWW-sovellusalueen UML-laajennusten vertailusta. Kolmanneksi arvioidaan muodostetun viitekehyksen soveltuvuutta käyttötarkoitukseensa. Tämän jälkeen esitetään jatkotutkimusaiheita.

7.1 Laajennusten kartoituksesta

Tutkielman yhtenä keskeisenä tarkoituksena oli selvittää, mille sovellusalueille UML-laajennuksia on esitetty. Kartoituksen tekeminen oli välttämätöntä, koska vastaavanlaista kartoitusta ei ole aikaisemmin tehty kirjallisuudessa. Kartoituksen perusteella pystyttiin valitsemaan sovellusalue, joka soveltuu yksityiskohtaisempaan tarkasteluun.

Kartoituksen tekeminen oli haastava tehtävä, sillä kirjallisuudessa on esitetty todella suuri joukko UML-laajennuksia. Relevantin kirjallisuuden etsiminen ja läpikäyminen oli tämän tutkielman suurin yksittäinen työ. Tutkimukseen etsittiin laajennuksia kaikista mahdollisista tieteellisistä julkaisuista, jotka olivat tutkijan saatavilla. Kartoitukseen liittyviä artikkeleita löydettiin yhteensä lähes sata kappaletta. Kaikki artikkelit luettiin tai vähintään silmäiltiin. Huomioitavaa kartoituksessa on, että siinä kartoitettavien laajennusten joukkoa jouduttiin rajaamaan erilaisten kriteerien perusteella. Kartoitus ei pyri olemaan siis täysin kattava. Kartoituksen perusteella saadaan kuitenkin selkeä käsitys siitä, mille sovellusalueille UML-laajennuksia on esitetty.

Kartoituksen perusteella nousi esiin kolme sovellusaluetta, joille on ollut erityisesti tarvetta kehittää laajennuksia. Näitä sovellusalueita ovat WWW-sovellukset, agentti-perusteiset sovellukset sekä tosiaikajärjestelmät. Kuitenkin myös useille muille sovellusalueille on esitetty yksittäisiä UML-laajennuksia.

Kartoituksessa laajennuksista selvitettiin joitakin ominaispiirteitä, kuten laajennusten esittämisperusteet. Tätä seikkaa selvitettäessä havaittiin, että UML:ssä on lukuisia puutteita, joita täydentämään laajennuksia on esitetty. Toisaalta tämä on ymmärrettävää, sillä UML:n eräs keskeinen tavoite on olla yleiskäyttöinen. Kartoituksen perusteella UML:n sisäiset laajennusmekanismit tuntuisivat olevan UML:n vahvuus, sillä suurin osa laajennuksista on pystytty toteuttamaan niiden avulla. Kuitenkaan aivan kaikkiin tilanteisiin UML:n sisäiset laajennusmekanismit eivät ole riittävät. Erityisesti agentti-paradigmaan perustuvissa laajennuksissa on jouduttu turvautumaan myös metamallin laajentamiseen.

Kartoitusta tehtäessä kävi ilmi, että laajennukset on määritelty hyvin kirjavalla tavalla. Tähän on syynä lähinnä se, että UML-dokumentaatiossa ei määritellä riittävän tarkasti, kuinka laajennukset tulisi dokumentoida. Jotta laajennuksista saataisiin ymmärrettävämpiä, tulisi niiden dokumentointiin kiinnittää enemmän huomioita. Tähän seikkaan onkin luvassa parannusta UML:n versiossa 2.0.

Laajennusten kartoitusta voidaan pitää yhtenä tämän tutkimuksen tärkeimpänä tuloksena, sillä vastaavan laajuista UML-laajennusten kartoitusta ei ole aikaisemmin kirjallisuudessa esitetty. Kartoitusta voidaan käyttää hyväksi monilla eri tavoilla. Ensinnäkin tutkijat voivat nähdä kartoituksen mukaisesti, millaisia laajennuksia on jo tehty ja millä perusteilla. Lisäksi tutkijat voivat kartoituksen avulla valita jonkin laajennuksen jatkokehityksen kohteeksi tai kokeilla laajennusten toimivuutta empiirisesti. Useimpien laajennusten soveltuvuudesta käytäntöön ei ole tietoa, joten empiiristen tutkimusten tekeminen olisi hyödyllistä. UML:n kehittäjät voivat puolestaan havaita, minkä mallin osalta sovellusaluekohtaista eriyttämistarvetta on esiintynyt ja mahdollisesti ottaa tämän huomioon UML:n jatkokehityksessä. Käytännön työlle kartoitus toimii hakemistona sopivan mallinnustavan löytämiseksi.

7.2 WWW-sovellusalueen laajennusten arvioinnista ja vertailusta

Tässä tutkielmassa keskityttiin arvioimaan ja vertailemaan yksityiskohtaisesti WWW-sovellusalueen laajennuksia. Kyseinen sovellusalue valittiin, koska alueen laajennuksista oli saatavilla riittävästi tietoa, jotta yksityiskohtainen vertailu pystyttiin tekemään. Lisäksi tutkijan mielenkiinto tätä ajankohtaista ja nopeasti kehittyvää sovellusaluetta kohtaan vaikutti valintaan.

Vertailuun otettiin mukaan kaikki kolme kartoituksen avulla löydettyä laajennusta. Vertailussa olivat mukana: UWE-menetelmä (esim. Hennicker & Koch 2001a), W2000-viitekehys (esim. Baresi, Garzotto & Paolini 2001) sekä WAE-profiili (esim. Conallen 2000).

Vertailussa havaittiin, että UWE-menetelmän vahvuudet ovat WWW-sovelluksen käyttöliittymän suunnittelu ja navigoinnin mallintaminen. UWE-menetelmässä käyttöliittymä mallinnetaan navigointimallin pohjalta. UWE on ainoa laajennus, joka tarjoaa stereotyyppejä käyttöliittymän suunnitteluun. Lisäksi UWE-menetelmä sisältää kattavimman joukon navigoinnin mallintamiseen käytettäviä stereotyyppejä. UWE-menetelmää voidaan pitää myös hyvin dokumentoituna. Siinä esitetyt stereotyypit ovat hyvin määriteltyjä ja muodostettuja.

W2000-viitekehystä voidaan pitää tietokeskeisenä, jossa huomioidaan myös navigoinnin mallintaminen. W2000 on ainoa vertailussa mukana ollut laajennus, jossa esitetään stereotyyppejä WWW-sovelluksen tietosisällön mallintamiseen. Tämän ominaisuuden tarpeellisuutta voidaan kuitenkin kritisoida, sillä tietosisällön mallintaminen on mahdollista myös UML:n perus kaavioiden avulla. Navigoinnin mallintamiseen ei esitellä yhtä kattavaa joukkoa stereotyyppejä kuin UWE-menetelmässä. W2000-viitekehys on vertailussa mukana olleista laajennuksista huonoiten dokumentoitu. Siinä esitetyt stereotyypit ovat epämääräisesti määriteltyjä.

WAE-profiili on arkkitehtuurikeskeinen, jossa huomioidaan hyvin myös WWW-sovelluksen toiminnallisuuden mallintaminen. WAE-profiili on lähtökohdiltaan hyvin erilainen kuin UWE ja W2000, joissa perustana ovat hypermedian mallintamiseen käytettävät menetelmät. WAE-profiilia voidaan pitää hyvin toteutusteknologiakeskeisenä. WAE-profiili on hyvin dokumentoitu. Siinä esitellyt stereotyypit ovat hyvin määriteltyjä ja muodostettuja.

Vertailussa mukana olevia laajennuksia yhdistää se, että ne ovat kaikki UML-profiileja. Muun muassa tämän takia niiden opettelua voidaan pitää suhteellisen helppona UML:ää tunteville henkilöille. Kaikkien laajennusten käyttäminen yleisimmissä CASE-välineissä on myös mahdollista tämän seikan takia.

Edellä kuvatut laajennukset siis kattavat WWW-sovelluksen mallintamisen eri ominaispiirteitä. Tästä seuraa ajatus laajennusten mahdollisesta yhdistämisestä: miksi ei yhdistettäisi kunkin laajennuksen vahvuuksia? Hennicker ja Koch (2001b) esittävätkin tavan yhdistää UWE-menetelmän käyttöliittymästereotyyppejä ja WAE-profiilin stereotyyppejä. Heidän esittämänsä tapa vaikuttaa aivan toimivalta. Lähestymistavassaan he käyttävät WAE-stereotyypeistä niiden tekstuaalisia vastineita graafisten kuvakkeiden sijaan. Mikäli eri laajennusten stereotyyppejä lähdettäisiin laajemmin yhdistelemään, stereotyyppien graafiset kuvakkeet saattaisivat muodostaa ongelman, sillä laajennuksissa käytetään graafisia kuvakkeita hieman eri tavoin. UWE-menetelmässä käytetään itsenäisiä ja kaavioihin upotettuja kuvakkeita. WAE-profiilissa kuvakkeet ovat kiinteästi symbolien yhteydessä. Eri laajennusten stereotyyppien suora yhdistely kaavioissa johtaisi epäyhdenmukaiseen esitykseen.

Tässä tutkielmassa tehdyn UML-laajennusten vertailun tuloksia arvioitaessa täytyy muistaa, että tutkimustulokset ovat täysin käsitteellisteoreettisia ja

perustuvat ennalta määriteltyyn kriteeristöön. Laajennusten todellinen hyöty ja toimivuus voidaan todeta vain empiirisesti kokeilemalla ja tutkimalla.

Vertailun tuloksia voidaan pitää kuitenkin hyödyllisinä, sillä vertailussa mukana olleita laajennuksia ei ole aikaisemmin arvioitu eikä vertailtu kirjallisuudessa. WWW-sovellusalueen laajennusten valintaan vaikutti erityisesti se, että kyseinen sovellusalue on nuori, nopeasti kehittyvä ja melko tutkimaton.

Tehdyn arvioinnin avulla tutkijat saavat arvokasta tietoa laajennusten luonteesta ja käytöstä kussakin tapauksessa. Lisäksi tutkijat saavat arvioita laajennusten ”hyvyydestä” ennalta määritellyn kriteeristön nojalla. Vertailu antaa hyviä lähtökohtia monenlaiselle laajennuksia täydentävän tutkimuksen tekemiselle. Vertailun perusteella laajennusten kehitystyötä tarvitaan, sillä arvioiduissa UML-laajennuksissa on kussakin omat puutteensa. Minkään vertailussa mukana olleen laajennuksen avulla ei voida mallintaa kaikkia WWW-sovelluksen ominaispiirteitä. Laajennusten arvioinnista on hyötyä myös käytännön työlle, sille se antaa yksityiskohtaisia kuvauksia ja arvioita yksittäisten laajennusten kuvausalasta ja ominaisuuksista.

7.3 Viitekehyksen soveltuvuus tutkimukseen

Yksi tutkielman keskeisiä tavoitteita oli muodostaa viitekehys, jonka avulla UML-laajennuksia voidaan arvioida ja vertailla. Kirjallisuudessa ei ole esitetty aiemmin tähän käyttötarkoitukseen suoraan soveltuvaa kriteeristöä.

Yleisesti ottaen voidaan todeta, että luvussa 4 muodostetun viitekehyksen soveltaminen WWW-sovellusalueen laajennusten vertailuun onnistui hyvin. Viitekehyksen sisältämien kriteerin avulla saatiin tuotua esiin eroja laajennusten välillä, kuten edellisestä kohdasta ilmenee. Viitekehyksen luokkarakenne toimi myös hyvin vertailun jäsentäjänä.

Viitekehyksen heikkoutena voidaan pitää sitä, että sitä täytyy täydentää sovellusaluekohtaisesti laatuluokan laajennusten sopivuus sovellusalueeseen osalta. Jokainen sovellusalue sisältää kuitenkin omat erityispiirteensä, eikä niiden sisällyttäminen ole mahdollista viitekehykseen, joka mahdollistaa laajennusten tarkastelun sovellusalueesta riippumatta. Tämän luokan kriteerien tunnistaminen sovellusaluekohtaisesti on olennaista, muuten vertailusta tulee turhan yleinen. Tässä tutkimuksessa tunnistettiin erikseen tähän luokkaan kuuluvat kriteerit WWW-sovellusalueen osalta. Näiden kriteerien avulla löydettiinkin olennaisia eroavaisuuksia WWW-sovellusalueen laajennusten välillä.

Laatuluokat osallisten tietämys kielestä, käsitettävyys sekä teknisten toimijoiden tulkittavuus sisältävät enimmäkseen yleisiä kriteereitä, joita voidaan soveltaa vertailtaessa mitä tahansa mallinnuskieliä. Kriteerit eivät ole siis sidosteisiä mihinkään sovellusalueeseen, eikä suurin osa edes UML:ään. Tämä johtuu siitä, että kriteerit ovat lähtöisin suurimmaksi osaksi Krogstien ja Solvbergin (2000) viitekehyksestä. Näiden laatuluokkien sisältämien kriteerien avulla ei löydetty suuria eroja WWW-sovellusalueen laajennusten välillä. Mahdollisena selityksenä tälle havainnolle on, että kaikilla laajennuksilla on yhteisenä perustana UML. Näiden kolmen laatuluokan mukaisten kriteerien käsitteleminen oli melko suoraviivainen tehtävä.

Laatuluokan stereotyypeille ominaiset vaatimukset sisältävät kriteerit kohdistuvat vain UML-laajennuksiin. Tämän luokan kriteerien käsitteleminen laajennusten vertailussa oli suoraviivaista lukuun ottamatta stereotyyppien luokittelua koristeellisiin, kuvaileviin, rajoittaviin sekä uudelleen määritteleviin stereotyyppeihin. Stereotyyppien luokittelu oli vertailun haastavin tehtävä johtuen siitä, että kaikkia stereotyyppejä ei ollut määritelty luokittelun vaatimalla tarkkuustasolla.

Viitekehystä on hankala vertailla konkreettisesti mihinkään muuhun viitekehykseen, koska täysin vastaavia viitekehyksiä ei ole olemassa. Ylipäätään mallinnuskielten vertailuunkin on olemassa vain harvoja viitekehyksiä. Mielestäni viitekehys täytti tehtävänsä hyvin ainakin WWW-sovellusalueen laajennusten vertailussa. Yksi huomion arvoinen seikka viitekehyksessä on kuitenkin se, että vertailtaessa laajennuksia jotkin yksittäiset havainnot nousivat esiin useamman kuin yhden kriteerin tarkastelun yhteydessä. Esimerkiksi W2000-viitekehyksen tapa käyttää samaa stereotyyppiä Center Type assosiaatiokeskuksesta ja kokoelmakeskuksesta nousi esiin tarkasteltaessa kahta eri kriteeriä. Tämä asia tuli ilmi tarkasteltaessa kriteeriä, jonka mukaan ulkoinen ilmaisu ei saisi sekoittaa käsitteellistä perustaa, ja käsiteltäessä kriteeriä, jonka mukaan käsitteiden käytön tulisi olla yhdenmukaista. Myös W2000-viitekehyksen arveluttava tapa käyttää tietosisällön mallintamiseen erityisiä stereotyyppejä nousi esiin kolmessa yhteydessä, tarkasteltaessa osallisten tietämystä kielestä, käsitettävyyttä ja stereotyypeille ominaisia vaatimuksia. Tehtäessä laajennusten arviointia muodostetun viitekehyksen mukaisesti saatetaan joidenkin kriteerien osalta saada siis päällekkäisiä havaintoja.

Muodostettua viitekehystä voidaan pitää tärkeänä kontribuutiona, sillä UML-laajennusten arvioitiin ja vertailuun soveltuvaa kriteeristöä ei ole aiemmin esitetty kirjallisuudessa. Koska UML:ään on esitetty lukuisa joukko laajennuksia, on tärkeää, että niitä pystytään myös arvioimaan ja vertailemaan. Viitekehystä voidaan hyödyntää tutkimustyössä ainakin kahdella tavalla. Luonnollisesti tutkijat voivat käyttää viitekehystä jonkun toisen sovellusalueen laajennusten arvioimiseen ja vertailuun. Myös viitekehyksen edelleen kehittäminen on mahdollista. Toiseksi viitekehystä voi hyödyntää kehitettäessä UML-laajennuksia. Kriteeristö voi toimia eräänlaisena muistilistana sille, mitä seikkoja kannattaa huomioida laajennusta suunniteltaessa.

7.4 Jatkotutkimusaiheita

Tutkielmaa tehdessä nousi esiin erilaisia jatkotutkimusaiheita. Ensinnäkin laajennusten kartoituksesta ilmenee, että olisi mahdollista tehdä myös joidenkin muiden kuin WWW-sovellusalueen laajennusten yksityiskohtainen arviointi ja vertailu. Tällaisia sovellusalueita olisivat ainakin tosiaikajärjestelmät ja agenttiperusteiset sovellukset. Kyseisten sovellusalueiden laajennusten vertailussa voitaisiin käyttää tässä tutkielmassa muodostettua viitekehystä.

Toisena jatkotutkimusaiheena voisi olla WWW-sovellusalueelle suunnattujen laajennusten mahdollinen yhdistäminen. Kuten tutkielmassa havaittiin, esitellyillä laajennuksilla on omat vahvuutensa. Näiden vahvuuksien kokoaminen yhteen voisi olla hyödyllistä.

Kolmantena jatkotutkimusaiheena voisi tutkia, kuinka UML:n sisäiset laajennusmekanismit ovat kehittyneet UML:n 2.0 versiossa. Lisäksi voisi tutkia onko UML:n versiossa 2.0 huomioitu paremmin WWW-sovellusten kehittämisen erityispiirteet. Oletettavasti näin ei kuitenkaan ole, sillä UML:n keskeinen periaate on olla mahdollisimman yleiskäyttöinen. Tämän tutkielman valmistuessa UML 2.0 on julkaistu lähes kokonaan.

8 YHTEENVETO

UML:ää voidaan pitää nykyään oliokeskeisten sovellusten mallintamista merkittävästi yhtenäistävänä standardina. UML:stä on muodostunut hyvin monenlaisilla sovellusalueilla yleisesti käytettävä standardi. Erityisalojen käsitteet eivät kuitenkaan sisälly suoraan UML:ään. Tämän takia kirjallisuudessa on esitetty kymmeniä, ellei satoja UML-laajennuksia. UML:n suosiosta huolimatta UML-laajennuksiin kohdistuvaa arviointia ja vertailua ei ole juurikaan tehty.

Tämän tutkielman ensimmäisenä tavoitteena oli tarkastella UML:ää sen laajentamisen näkökulmasta. Tähän tavoitteeseen liittyvänä tärkeänä tehtävänä oli selvittää UML:n laajennustavat. OMG:n (2003a 131–133) mukaan on olemassa kaksi pääasiallista tapaa laajentaa UML:ää. UML:n kehittäjien suosittelema tapa on käyttää pelkästään sisäänrakennettuja laajennusmekanismeja. Tällaista UML:n laajennusta kutsutaan profiiliksi. (OMG 2003a, 131–133) Mikäli sisäänrakennetut laajennusmekanismit eivät ole riittävät, joudutaan tekemään lisäyksiä UML:n metamalliin. Tällaisesta laajennustavasta käytetään nimeä raskaansarjan (heavyweight) laajennusmekanismi. (OMG 2003a, 131–133)

Toisena tavoitteena oli tehdä kartoitus UML:n laajennuksista sovellusalueittain. Kartoituksen perusteella laajennusten kehittely on keskittynyt seuraaville alueille: WWW-sovellukset, agentti-perusteiset sovellukset ja tosiaikajärjestelmät. Myös lukuisille muille sovellusalueille on esitetty yksittäisiä laajennuksia. Kartoituksen perusteella UML:ssä tuntuu olevan yllättävän paljon heikkouksia, joita täydentämään laajennuksia on esitetty. Kartoitusta tehtäessä havaittiin myös, että yleensä tietyn sovellusalueen sisällä esitetyt laajennukset kohdistuvat pääosin tiettyihin kaaviotyyppeihin, vaikka vaihteluakin ilmenee aika paljon. Ylipäätään luokkakaavioon on esitetty eniten laajennuksia riippumatta sovellusalueesta. Useimmat laajennukset on toteutettu

UML:n sisäisten laajennusmekanismien avulla. Selkeästi käytetyin yksittäinen laajennusmekanismi on stereotyyppi.

Tutkielman kolmantena tavoitteena oli muodostaa viitekehys, jonka avulla UML-laajennuksia voidaan vertailla sovellusaluekohtaisesti. Keskeisenä tarkoituksena oli laatia viitekehyksestä yleinen, jotta sitä voidaan käyttää useilla sovellusalueilla. Tästä johtuen muodostettua viitekehystä joudutaan täydentämään sovellusaluekohtaisesti sovellusalueelle ominaisilla kriteereillä. Viitekehys on muodostettu olemassa olevan kirjallisuuden pohjalta ja se koostuu viidestä laatuluokasta, jotka sisältävät lukuisia kriteereitä. Keskeisenä lähtökohtana kehitystyössä on käytetty Krogstien ja Solvbergin (2000) muodostamaa viitekehystä.

Tutkielman neljäntenä tavoitteena oli testata viitekehyksen toimivuutta arvioimalla ja vertailemalla WWW-sovellusalueen UML-laajennuksia. Vertailussa olivat mukana: UWE-menetelmä (esim. Hennicker & Koch 2001a), W2000-viitekehys (esim. Baresi, Garzotto & Paolini 2001) ja WAE-profiili (esim. Conallen 2000). Vertailusta saatujen kokemusten perusteella viitekehys vastasi hyvin tarkoitustaan. Viitekehyksen avulla laajennusten välillä tunnistettiin selkeitä eroja. Vertailussa muun muassa havaittiin, että laajennuksista yksikään ei ole täysin sopiva sovellusalueeseen. Mikään laajennuksista ei kata kaikkia WWW-sovellusten kehittämiselle ominaisia piirteitä. Jokaiselle laajennuksella tuntuisi olevan kuitenkin omat vahvuutensa. UWE on ainoa laajennus, joka mahdollistaa WWW-sovelluksen käyttöliittymän suunnittelun. Lisäksi UWE-menetelmä sisältää kattavimman joukon navigoinnin mallintamiseen käytettäviä stereotyyppejä. W2000-viitekehys on tietokeskeinen, jossa huomioidaan myös navigoinnin mallintaminen. WAE-profiili on arkkitehtuurikeskeinen, jossa huomioidaan hyvin myös WWW-sovelluksen toiminnallisuuden mallintaminen.

LÄHDELUETTELO

Alhir S. 1998. Unified Modeling Language Extension Mechanisms. Distributed Computing Magazine, joulukuu [online]. [viitattu 27.10.2003]. Saatavilla www-muodossa <http://home.earthlink.net/~salhir/UMLExtensionMechanisms.PDF>.

Ambler S. 2002. A UML Profile for Data Modeling [online]. [viitattu 26.1.2004]. Saatavilla www-muodossa <http://www.agiledata.org/essays/umlDataModelingProfile.html>.

Apvrille L., de Saqui-Sannes P., Lohr C., S´enac P. & Courtiat J.-P. 2001. A New UML Profile for Real-Time System Formal Design and Validation. Teoksessa: Gogolla M. & Kobryn C. (toim.) Proceedings of the 4th International Conference on the Unified Modeling Language. Modeling Languages, Concepts, and Tools, Toronto, Canada, lokakuu 1–5. LNCS 2185. Berlin: Springer-Verlag, 287–301.

AUML web site [online]. [viitattu 27.10.2003]. <http://www.auml.org/auml/main.shtml>.

Axelson J. 2001. Unified Modeling of Real-Time Control Systems and Their Physical Environments Using UML. Teoksessa: Proceedings of the Eighth Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems, Washington, D.C., USA, huhtikuu 17–20. Los Alamitos: IEEE Computer Society, 18–25.

Baresi L., Garzotto F. & Paolini P. 2001. Extending UML for Modeling Web Applications. Teoksessa: Proceedings of the 34th Hawaii Annual International Conference on System Sciences, Decision Technologies for Management Track, Unified Modeling Language: A Critical Review and Suggested Future Minitrack, Maui, USA, tammikuu 3–6. 1285–1294.

Baresi L., Garzotto F. & Maritati M. 2002. W2000 as a MOF Metamodel. Teoksessa: Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics - Web Engineering Track, Orlando, USA, heinäkuu 14–16.

Baskerville R. & Pries-Heje J. 2001. Racing the E-bomb: How the Internet Is Redefining Information Systems Development. Teoksessa: Russo L., Fitzgerald B. & DeGross J. (toim.) Proceedings of the Working Conference on Realigning Research and Practise in Information Systems Development: The Social and Organisational Perspective, Boise, Idaho, heinäkuu 27–28. New York: Kluwer, 49–68.

Bastos R., Dubugras D. & Ruiz A. 2002. Extending UML Activity Diagram for Workflow Modeling in Production Systems. Teoksessa: Ralph H. & Sprague Jr. (toim.) Proceedings of the 35th Hawaii Annual International Conference on System Sciences, Big Island, Hawaii, tammikuu 7–10. Los Alamitos: IEEE Computer Society, 3786–3795.

Bauer B., Müller J. & Odell J. 2001. Agent UML: A Formalism for Specifying Multiagent Software Systems. International Journal of Software Engineering and Knowledge Engineering 11(3), 1–24.

Baumeister H., Koch M. & Mandel L. 1999. Towards a UML Extension for Hypermedia Design. Teoksessa: France R. & Rumpe B. (toim.) Proceedings of the Second International Conference on the Unified Modeling Language: Beyond the Standard, Fort Collins, CO, USA, lokakuu 28–30. LNCS 1723. Berlin: Springer-Verlag, 614–629.

Bergenti F. & Poggi A. 2000. Exploiting UML in the Design of Multi-Agent Systems. Teoksessa: Omicini A., Tolksdorf R. & Zambonelli F. (toim) Proceedings of the First International Workshop on Engineering Societies

in the Agents World, Berlin, Germany, elokuu 21. LNCS 1972. Heidelberg: Springer-Verlag, 106–113.

Berner S., Glinz M. & Joos S. 1999. A Classification of Stereotypes for Object-Oriented Modeling Languages. Teoksessa: France R. & Rumpe B. (toim.) Proceedings of Second International Conference on The Unified Modeling Language: Beyond the Standard, Fort Collins, CO, USA, lokakuu 28–30. LNCS 1723. Berlin: Springer-Verlag, 249–264.

Bichler L., Radermacher A., & Schurr A. 2002. Evaluating UML Extensions for Modeling Real-Time Systems. Teoksessa: Proceedings of the Seventh International Workshop on Object-Oriented Real-Time Dependable Systems, Sandiago, California, tammikuu 7–9. Los Alamitos: IEEE Computer Society, 271–278.

Björkander M. 2000. Graphical Programming Using UML and SDL. IEEE Computer 33(12), 30–35.

Booch G. 1991. Object-Oriented Analysis and Design with Applications, 1st edition. Redwood City, Calif: Benjamin Cummings.

Booch G., Rumbaugh J. & Jacobson I. 1998. The Unified Modelling Language User Guide. Reading, MA: Addison Wesley Longman.

Caire G., Leal F., Chainho P., Evans R., Garijo F., Gomez J., Pavon J., Kearney P., Stark J. & Massonet P. 2002. Agent Oriented Analysis Using MESSAGE/UML. Teoksessa: Wooldridge M., Ciancarini P. & Weiss G. (toim.) Second International Workshop on Agent-Oriented Software Engineering, Montreal, Canada, toukokuu. 101–108.

Castro J., Kolp M. & Mylopoulos J. 2002. Towards Requirements-Driven Information Systems Engineering: the Tropos Project. Information Systems 27(6), 365–389.

Ceri S., Fraternali P., Bongio A., Brambilla M., Comai S. & Matera M. 2003. Designing Data-Intensive Web Applications. Amsterdam: Morgan Kaufmann.

Conallen J. 1999a. Modeling Web Application Design with UML [online]. [viitattu 9.4.2004]. Saatavilla www-muodossa <http://www.conallen.org/whitepapers/webapps/ModelingWebApplications.htm>.

Conallen J. 1999b. Modeling Web Application Architectures with UML. Communications of the ACM 42(10), 63–70.

Conallen J. 2000. Building Web Applications with UML. Reading, MA: Addison-Wesley.

Cook S., Kleppe A., Mitchell R., Rumpe B., Warmer J. & Wills A. 1999. Defining UML Family Members Using Prefaces. Teoksessa: Mingins C. & Meyer B. (toim.) Proceedings of the Technology of Object-Oriented Languages and Systems, Melbourne, Australia, marraskuu 22–25. Los Alamitos: IEEE Computer Society, 102–114.

D'Souza D., Sane A. & Birchenough A. 1999. First-Class Extensibility for UML-Profiles, Stereotypes, Patterns. Teoksessa: Franceand R. & Rumpe B. (toim.) Proceedings of the Second International Conference on the Unified Modeling Language. Beyond the Standard, FortCollins, CO, USA, lokakuu 28–30. LNCS 1723. Berlin: Springer-Verlag, 265–277.

Eriksson H. & Penker M. 2000. UML. Helsinki: IT Press.

Flake S., Geiger C. & Küster J. 2001. Towards UML-based Analysis and Design of Multi-Agent Systems. Teoksessa: Proceedings of the International NAISO Symposium on Information Science Innovations in Engineering of Natural and Artificial Intelligent Systeme, Dubai, maaliskuu.

Flake S. 2002. Real-Time Constraints with the OCL. Teoksessa: Bacellar L., Puschner P. & Hong S. (toim.) Proceedings of the Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, Washington, D.C., huhtikuu 29–toukokuu 1. Los Alamitos: IEEE Computer Society, 425–426.

Flake S. & Mueller W. 2002. Specification of Real-Time Properties for UML Models. Teoksessa: Ralph H. & Sprague Jr. (toim.) Proceedings of the 35th Annual Hawaii International Conference on System Science, Big Island, Hawaii, tammikuu 7–10. Los Alamitos: IEEE Computer Society, 3977–3986.

Garzotto F., Paolini P. & Schwabe D. 1993. HDM—a Model-Based Approach to Hypertext Application Design. ACM Transactions on Information Systems 11(1), 1–26.

Gemino A. & Wand Y. 2002. Common Dimensions in Empirical Comparisons of Conceptual Modeling Techniques. Teoksessa: Halpin T., Siau K. & Krogstie J. (toim.) Proceedings of the 7th International Workshop on Evaluating of Modeling Methods in Systems Analysis and Desing, Toronto, Canada, toukokuu 27–28. 144–151.

Ginige A. & Murugesan S. 2001. Web Engineering: an Introduction. IEEE Multimedia 8(1), 14–18.

Glass R. & Vessey I. 1995. Contemporary Application-Domain Taxonomies. IEEE Software 12(4), 63–76.

Glass R & Vessey I. 1998. Focusing on the Application Domain: Everyone Agrees It's Vital, But Who's Doing Anything about It? Teoksessa: Ralph H. & Sprague Jr. (toim.) Proceedings of the Thirty-First Hawaii International on System Sciences, Maui, USA, tammikuu 6–9. Los Alamitos: IEEE Computer Society, 187–196 vol.3.

Gomez J., Cachero C. & Pastor O. 2001. Conceptual Modeling of Device-Independent Web Applications. IEEE Multimedia 8(2), 26–39.

Gornik D. 2002. UML Data Modeling Profile. [online]. [viitattu 26.1.2004]. Saatavilla www-muodossa <http://www.rational.com/media/whitepapers/tp162.pdf>.

Henderson-Sellers B. & Firesmith D. 1999. Comparing OPEN and UML: the Two Third-Generation OO Development Approaches. Information and Software Technology 41(3), 139–156.

Hennicker R. & Koch N. 2001a. Systematic Design of Web Applications with UML. Teoksessa: Siau K. & Halpin T. (toim.) Unified Modeling Language: Systems Analysis, Design and Development Issues. IDEA Group Publishing, 1–20.

Hennicker R. & Koch N. 2001b. Modeling the User Interface of Web Applications with UML. Teoksessa: Evans A., France R. & Moreira A. (toim.) Practical UML-Based Rigorous Development Methods - Countering or Integrating the eXtremists, Workshop of the pUML-Group at the UML 2001, Gesellschaft für Informatik, Germany, lokakuu. Köllen Druck+Verlag, 158–173.

ISO web site [online]. [viitattu 21.7.2004]. < http://www.iso.org>.

ITU-T. 1999. Z.100: The Specification and Description Language [online]. [viitattu 21.7.2004]. Saatavilla www-muodossa < http://www.itu.int/ITU-T/studygroups/com10/languages/Z.100_1199. pdf>.

Jacobson I., Christerson M., Jonsson P. & Övergaard G. 1992. Object-Oriented Software Engineering, A Use Case Driven Approach. Workingham, England: Addison-Wesley.

Jürjens J. 2002. UMLsec: Extending UML for Secure Systems Development. Teoksessa: Jézéquel J.-M., Hussmann H. & Cook S. (toim.) Proceedings of the 5th International Conference on the Unified Modeling Language, Dresden, Germany, syyskuu 30 - lokakuu 4. LNCS 2460. Heidelberg: Springer-Verlag, 412–425.

Kavi K., Aborizka M. & Kung D. 2002. A Framework for Designing, Modeling and Analyzing Agent Based Software Systems. Teoksessa: Zhou W., Chi X., Goscinski A. & Li G. (toim.) Proceedings of the Fifth International Conference on Algorithms and Architectures for Parallel Processing, Beijing, China, lokakuu 23–25. Los Alamitos: IEEE Computer Society, 196–200.

Kavi K., Kung D., Bharnbhani H., Pancholi G., Kanikarla M. & Shah R. 2003. Extending UML to Modeling and Design of Multi-Agent Systems. Teoksessa: Proceedings of the International Conference on Software Engineering, the 2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems, Portland, Oregon, toukokuu 3-10.

Kim J., Hahn J. & Hahn H. 2000. How Do We Understand a System with (So) Many Diagrams? Cognitive Integration Processes in Diagrammatic Reasoning. Information Systems Research 11(3), 284–303.

Koch N., Baumeister H., Hennicker R. & Mandel L. 2000. Extending UML for Modeling Navigation and Presentation in Web Applications. Teoksessa: Winters G. & Winters J. (toim.) Proceedings of Modeling Web Applications in the UML Workshop, York, England, lokakuu.

Koch N. & Kraus A. 2002. The Expressive Power of UML-based Web Engineering. Teoksessa: Schwabe D., Pastor O, Rossi G. & Olsina L. (toim.) Proceedings of the Second International Workshop on Web-Oriented Software Technology, Málaga, Spain, kesäkuu 10–14.

Korbyn C. 1999. UML 2001: A Standardization Odyssey. Communications of the ACM 42(10), 29–37.

Kraus A. & Koch N. 2003. A Metamodel for UWE. Ludwig-Maximilians-Universität München, Technical Report 0301.

Krogstie J. (1995). Conceptual Modeling for Computerized Information Systems Support in Organizations. University of Trondheim, Norway, väitöskirja.

Krogstie J. & Solvberg A. 2000. Information Systems Engineering: Conceptual Modeling in a Quality Perspective. Trondheim, Norway: Information Systems Groups, NTNU.

Kruchten P. 2000. The Rational Unified Process: An Introduction. Reading, MA: Addison-Wesley.

Lindland O., Sindre G. & Solvberg A. 1994. Understanding Quality in Conceptual Modeling. IEEE Software 11(2), 42–49.

Lohr C., Apvrille L., Saqui-Sannes P. & Courtiat J.-P. 2003. New Operators for the TURTLE Real-Time UML Profile. Teoksessa: Formal Methods for Open Object-Based Distributed Systems. LNCS 2884. Heidelberg: Springer-Verlag, 214–228.

Marcos E., Vela B. & Cavero J. 2001. Extending UML for Object-Relational Database Design. Teoksessa: Gogolla & Kobryn C. (toim.) Proceedings of the 4th International Conference on the Unified Modeling Language. Modeling Languages, Concepts, and Tools, Toronto, Canada, lokakuu 1–5. LNCS 2185. Berlin: Springer-Verlag, 225–239.

Marshall C. 2000. Enterprise Modeling with UML : Designing Successful Software Through Business Analysis. Reading, MA: Addison Wesley Longman.

Medvidovic N., Rosenblum D., Redmiles D. & Robbins J. 2002. Modeling Software Architectures in the Unified Modeling Language. ACM Transactions on Software Engineering and Methodology 11(1), 2–57.

MESSAGE web site [online]. [viitattu 21.1.2004]. <http://www.eurescom.de/public/projects/P900-series/p907/>.

Moody D. 2003. Theoretical and Practical Issues in Evaluating the Quality of Conceptual Models. Teoksessa: Proceedings of the Advanced Conceptual Modeling Techniques, Tampere, Finland, lokakuu 7–11. LNCS 2784. Heidelberg: Springer-Verlag, 241–242.

Moody D., Sindre G., Brasethvik T. & Solvberg A. 2003. Evaluating the Quality of Information Models: Empirical Testing of a Conceptual Model Quality Framework. Teoksessa: Proceedings of the 25th International Conference on Software Engineering, Portland, Oregon USA, toukokuu 3–10. Los Alamitos: IEEE Computer Society, 295–305.

Morris S. & Spanoudakis G. 2001. UML: an Evaluation of the Visual Syntax of the Language. Teoksessa: Proceedings of the 34th Annual Hawaii International Conference on System Sciences, Maui, Hawaii, tammikuu 3–6. Los Alamitos: IEEE Computer Society, 1223–1232.

Mäkinen V. 2003. Analysis of Use Case Appproaches to Requirements Engineering. Jyväskylän yliopisto, Tietojärjestelmätieteen pro gradu -tutkielma.

Odell J., Parunak V. & Bauer B. 2000. Extending UML for Agents. Teoksessa: Wagner G., Lesperance Y. & Yu. E. (toim) Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National conference on Artificial Intelligence, Austin, Texas, heinäkuu, 30. ICue Publishing, 3–17.

Offutt J. 2002. Quality Attributes of Web Software Applications. IEEE Software 19(2), 25–32.

OMG. 1999. Requirements for UML Profiles [online]. Version 1.0 [viitattu 17.12.2003]. Saatavilla www-muodossa <http://www.omg.org/docs/ad/99-12-32.pdf>.

OMG. 2002. Meta Object Facility (MOF) Specification [online]. Version 1.4 [viitattu 15.10.2003]. Saatavilla www-muodossa <http://www.omg.org/docs/formal/02-04-03.pdf>.

OMG. 2003a. OMG Unified Modeling Language Specification [online]. Version 1.5 [viitattu 15.10.2003]. Saatavilla www-muodossa <http://www.omg.org/technology/documents/formal/uml.htm>.

OMG. 2003b. UML Profile for Schedulability, Performance and Time [online]. Version 1.0 [viitattu 29.1.2004]. Saatavilla www-muodossa <http://www.omg.org/docs/formal/03-09-01.pdf>.

Opdahl A. & Henderson-Sellers B. 2002. Ontological Evaluation of the UML Using the Bunge–Wand–Weber Model. Software and Systems Modeling 1(1), 43–67.

Paige R., Ostroff J. & Brooke P. 2000. Principles for Modeling Language Design. Information and Software Technology 42(10), 665–675.

Pastor O., Schwabe D., Rossi G. & Olsina L. 2002. Web-Oriented Software Technology. Teoksessa: Hernández J. & Moreira A. (toim.) Proceedings of the Object-Oriented Technology. ECOOP 2002 Workshop Reader : ECOOP 2002 Workshops and Posters, Málaga, Spain, kesäkuu 10–14. LNCS 2548. Heidelberg: Springer-Verlag, 55–69.

Pinheiro da Silva P. & Patton N. 2001. A UML-based Design Environment for Interactive Applications. Teoksessa: Proceedings of the Second

International Workshop on User Interfaces to Data Intensive Systems, Zuric, Switzerland, toukokuu 31–kesäkuu 1. Los Alamitos: IEEE Computer Society, 60–71.

Pinheiro da Silva P. & Patton N. 2003. User Interface Modeling in UMLi. IEEE Software 20(4), 62–69.

Poels G., Nelson J., Genero M & Piattini M. 2003. Quality in Conceptual Modeling - New Research Directions. Teoksessa: Proceedings of the Advanced Conceptual Modeling Techniques, Tampere, Finland, lokakuu 7–11. LNCS 2784. Heidelberg: Springer-Verlag, 243–250.

Price R., Ramamohanarao K. & Srinivasan B. 1999a. Spatiotemporal Extensions to Unified Modeling Language. Teoksessa: Cammelli A., Tjoa A. & Wagner R. (toim.) Proceedings of the Tenth International Workshop on Database and Expert Systems Applications, Florence, Italy, syyskuu 1–3. Los Alamitos: IEEE Computer Society, 460–461.

Price R., Srinivasan B. & Ramamohanarao K. 1999b. Extending the Unified Modeling Language to Support Spatiotemporal Applications. Teoksessa: Mingins C. & Meyer B. (toim.) Proceedings of the Technology of Object-Oriented Languages and Systems, Melbourne, Australia, marraskuu 22–25. Los Alamitos: IEEE Computer Society 163–174.

Rammig F. 2002. OCL Goes Real-Time. Teoksessa: Bacellar L., Puschner P. & Hong S. (toim.) Proceedings of the Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, Washington, D.C., huhtikuu 29–toukokuu 1. Los Alamitos: IEEE Computer Society, 423–424.

Rossi G., Schwabe D. & Lyardet F. 1999. Web Application Models Are More Than Conceptual Models. Teoksessa: Chen P., Embley D., Kouloumdjian J., Liddle S. & Roddick J. (toim.) Proceedings of the Advances in Conceptual Modeling: Workshops on Evolution and Change in Data

Management, Reverse Engineering in Information Systems, and the World Wide Web and Conceptual Modelingm, Paris, France, marraskuu 15–18 LNCS 1727. London: Springer-Verlag, 239–253.

Rubart J. & Dawabi P. 2002. Towards UML-G: A UML Profile for Modeling Groupware. Teoksessa: Haake J. & Pino J. (toim.) Proceedings of the 8th International Workshop on Groupware: Design, Implementation and Use:, La Serena, Chile, syyskuu 1-4. LNCS 2440. Heidelberg: Springer-Verlag, 93–113.

Rumbaugh J. 1991. Object-Oriented Modelling and Design. Englewood Cliffs: Prentice-Hall.

Rumbaugh J., Jacobson I. & Booch G. 1999. The Unified Modelling Language Reference Manual. Reading, MA: Addison Wesley Longman.

Sauer S. & Engels G. 1999. Extending UML for Modeling of Multimedia Applications. Teoksessa: Proceedings of the IEEE Symposium on Visual Languages, Tokyo, Japan, syyskuu 13–16. Los Alamitos: IEEE Computer Society, 80–87.

Sauer S. & Engels G. 2001. UML-Based Behavior Specification of Interactive Multimedia Applications. Teoksessa: Proceedings of the IEEE Symposia on Human-Centric Computing Languages and Environments, Stresa, Italy, syyskuu 5–7. Italy: IEEE Computer Society, 248–255.

Schleicher A. & Westfechtel B. 2001. Beyond Stereotyping: Metamodeling Approaches for the UML. Teoksessa: Proceedings of the 34th Annual Hawaii International Conference on System Sciences, Maui, USA, tammikuu 3–6. 1243–1252.

Schwabe D., Rossi G. & Barbosa S. 1996. Systematic Hypermedia Application Design with OOHDM. Teoksessa: Proceedings of the the Seventh ACM

Conference on Hypertext, Bethesda, Maryland, United States, maaliskuu 16–20. New York, USA: ACM Press, 116–128.

Schwabe D., Esmeraldo L., Rossi G. & Lyardet F. 2001. Engineering Web Applications for Reuse. IEEE Multimedia 8(1), 20–31.

Selic B., Gullekson G. & Ward P. 1994. Real-Time Object-Oriented Modeling. New York, USA: John Wiley & Sons.

Selic B. & Rumbaugh J. 1998. Using UML for Modeling Complex Real-Time Systems [online]. [viitattu 13.1.2004]. Saatavilla www-muodossa <http://www.rational.com/products/whitepapers/UML-rt.pdf>.

Selonen P. & Xu J. 2003. Validating UML Models Against Architectural Profiles. Teoksessa: Proceedings of the 9th European Software Engineering Conference, Helsinki, Finland. New York, USA: ACM Press, 58–67.

Tyndale-Biscoe S., Sims O., Wood B. & Sluman C. 2002. Business Modelling for Component Systems with UML. Teoksessa: Proceedings of the Sixth International Enterprise Distributed Object Computing Conference, Lausanne, Switzerland, syyskuu 17–20. Los Alamitos: IEEE Computer Society, 120–131.

Vidgren R., Avison D., Wood B. & Wood-Harper T. 2002. Developing Web Information Systems. Great Britain: MPG Books Ltd.

Yim H., Cho K., Kim J. & Park S. 2000. Architecture-Centric Object-Oriented Design Method for Multi-Agent Systems. Teoksessa: Proceedings of the Fourth International Conference on MultiAgent Systems.

Zendler A., Pfeiffer T., Eicks M. & Lehner F. 2001. Experimental Comparison of Coarse-Grained Concepts in UML, OML, and TOS. Journal of Systems and Software 57(1), 21–30.