hyväksymispäivä arvosana

arvostelija

Hypermedia ja versiointi

Arttu Valo

Helsinki 12.11.2004

Seminaaritutkielma

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Matemaattis-luonnontieteellinen | Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Arttu Valo

Työn nimi — Arbetets titel — Title

Hypermedia ja versiointi

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Seminaaritutkielma	12.11.2004	13 sivua

Tiivistelmä — Referat — Abstract

Hypermedian versionhallinta koetaan erityisen tärkeäksi sähköisen julkaisemisen kasvattaessa suosiotaan ja semanttisen webin tehdessä tuloaan. Hypermedian versionhallinnassa on kuitenkin ongelmia verrattuna yksinkertaiseen tiedostopohjaiseen versionhallintaan. Suurin ongelmista ovat linkit, jotka yhdistävät hypermediadokumentteja toisiinsa ja siten aiheuttavat riippuvuussuhteita. Pahimmillaan versiot hypermediaverkosta olisivat aina koko verkon laajuinen ja versioinnin tilavaatimus vastaava.

Vastauksena ongelmaan esitellään muutamia hypermedian versiointia tukevia järjestelmiä. Niiden kevyempi versiontimallinnus pohjautuu pääasiassa ER-mallinnukseen ja niihin pohjaaviin sisällysmalleihin. Sisällysmallien sovelluksissa hypermediasta erotetaan hypermediarakenne linkkeineen ja koosteineen varsinaisesta sisällöstä ja versioidaan molempia erikseen. Esitettävistä malleista vain WebDAV on laajemmassa kuin tutkimuskäytössä, sekin lähinnä rajapintamaisuutensa ja historiallisen yhteensopivuutensa vuoksi.

Tulevaisuudessa versiointiin kykenevien hypermediajärjestelmien odotetaan kuitenkin yleistyvän.

ACM Computing Classification System (CCS):
H.5.4 [Information Interfaces and Presentation]: Hypertext / Hypermedia

Avainsanat — Nyckelord — Keywords
Hypermedia, versiointi

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Sisältö

1 Johdanto

Hypermedia on www:n käytön yleistyessä muuttunut visionäärisestä tulevaisuuden tiedon verkostoideasta laajalle levinneeksi käytännön työkaluksi. Dokumenttien käsittelyn siirtyessä entistä laajemmassa mittakaavassa elektroniseksi ja tyypillisesti joko www:hen tai muuhun hypermediajärjestelmään, tarvitaan mekanismeja dokumenttien muutosten hallintaan ja seurantaan. Www:n perustoiminnot eivät tarjoa suoraan ratkaisuja tähän tarpeeseen, mutta myös monet muut hypermediajärjestelmät jättävät dokumenttien muutosten hallinnan ja seurannan vähälle huomiolle. Muutostenhallinta on erittäin tärkeä osa virallisia dokumentteja ja monia muita julkaisuja.

Tässä seminaarikirjoituksessa käsitellään hypermedian muutoksien hallintaan ja seurantaan soveltuvia versiointia ja versioinninhallintaa tarjoavia järjestelmiä. Käsittelyssä keskitytään nimenomaan järjestelmien muutosten hallintaan liittyviin toimintoihin ja ominaisuuksiin ja jätetään muut vähemmälle huomiolle.

Kirjoituksen rakenne on seuraava. Kappaleessa 2 kerrataan perustason versionhallinnan toteuttamista. Kappaleessa 3 tarkastellaan hypermedian lisävaatimuksia versionhallinnalle ja näiden vaatimusten täyttämistä. Kappaleessa 4 esitellään lyhyesti muutamia oleellisia hypermedian versiointia tukevia järjestelmiä. Lopuksi kappaleessa 5 tehdään lyhyt yhteenveto ja esitetään arvioita järjestelmien jatkokehityksestä.

2 Versionhallinnasta

Dokumentteja muutettaessa ja muutokset talletettaessa muodostuu alkuperäisestä dokumentista uusia *versioita* (*version*). Vanhaan dokumenttiin palaamisen mahdollisuus on kuitenkin monien dokumenttien kanssa tarpeellista. *Versionhallinta* on järjestelmien tarjoamaa tukea ja toimintoja tällaiselle dokumenttien kehityskaaren seuraamiseen. Tyypillisesti versionhallinta pitää kirjaa muutoksesta, sen tekijästä, ajankohdasta ja mahdollisesti muista mielenkiintoisista metatiedoista. Oleellisesti versionhallintajärjestelmän tulee tarjota mahdollisuus tarkastella tai käsitellä tallennettujen tietojen jokaista versiota eheänä kokonaisuutena.

Versiointijärjestelmien tulisi myös erottaa *versiointimekanismi versiointipolitiikasta.* Versiointimekanismi pitää huolta versioiden eheydestä ja järjestyksestä. Versiointipolitiikka taas määrää, milloin muutokset aiheuttavat uuden version luomisen. Esimerkiksi CVS toteuttaa yksinkertaisen muutostenhallinnan, jossa muutoksia voi

Koostetaso	Kokonaisuuksien jakaminen atomisiin palasiin
Versiointitaso	Versiot ylempää annetuista palasista
Fyysinen tallennus	Atomien tallennus massamuistiin tai tietokantaan

Kuva 1: Versioinnin kerrostetun verkkomallin perustasot

tehdä paikallisen tietovarastonsa dokumenteille rajatta. Uusi versio versiointijärjestelmään luodaan vasta kun käyttäjä erityisesti kutsuu `cvs ci`:llä muutosten sisällyttämistä keskustietovarastoon.

Ontologioiden yhteydessä voidaan versiointina pitää myös faktojen vaihtumista ajan yli, esimerkiksi Saksan muuttumista toisen maailmansodan jälkeen kahdeksi, Itä- ja Länsi-Saksaksi ja jälleen Kylmän sodan jälkeen yhdistyessä yhdeksi [Kau04]. Tulkitsen tällaisen kuitenkin tässä tekstissä versionhallintajärjestelmälle tavallisena tiedon muutoksena, enkä erityistoimenpiteitä vaativana poikkeustapauksena.

2.1 Tietomalleja

Versionhallinnassa syntyvien dokumenttien versioiden säilyttämiseen on ehdotettu useita erilaisia tietomalleja, mukaan lukien lista-, puu- ja kerrostetut verkkomallit (*layered network models*) [HLN98]. Listamalli on hyvin yksinkertainen; uusin versio lisätään listan päähän ja muutoshistoriaa voidaan tutkia siirtymällä listassa eteen- ja taaksepäin. Puumalli mahdollistaa lineaarisuuden lisäksi haarautumisen.

Nykyaikaiselle versionhallinnalle mainituista mielenkiintoisin on kerrostettu verkkomalli. Verkkomaisuus mahdollistaa dokumenttien versioiden vapaan haarautumisen (*branching*) ja liittymisen (*merging*). Kerrostus taas jakaa järjestelmän palvelut omiksi palvelukerroksikseen, joiden toimintoja ja palveluita voidaan kutsua toisilta tasoilta – aivan kuten kerrostetut verkkoprotokollat ovat jakaneet tietoliikenteen toimintoja esimerkiksi datayhteys- (*datalink layer*), verkko- (*network layer*) ja muihin kerroksiin.

Versionhallinnassa kerrokset vaihtelevat hieman toteutusjärjestelmien mukaan, mutta yhteistä toteutuksille on suurinpiirtein kuvan 2.1 kaltainen jako. Kerroksista jokainen vastaa yhteen versionhallinnan ydinkysymykseen.

Ylimpänä oleva koostetaso vastaa tiedon jakamisesta atomisiksi kokonaisuuksiksi. Vanhemmissa versiointijärjestelmissä taso on ollut käytännössä manuaalinen ja pohjannut levyjärjestelmän tiedostoon pienimpänä atomisena yksikkönä. Tiedosto pie-

nimpänä yksikkönä soveltuu kuitenkin huonosti moniin tarkoituksiin. Kappaleessa 4 esiteltävistä järjestelmistä useissa on tällä tasolla kohtuullisen kehittyneitä palveluita.

Keskimmäisen tason tehtävänä on tarjota versiointipalvelut kaikelle sille, mitä yläpuolelta tarjotaan, oli kyseessä kokonaisia tiedostoja, atomisia palasia tai rakenteita kuten koosteita.

Alin taso toteuttaa jokaisen versiointitason sille toimittaman tallennus- tai hakupyynnön aina tallennusjärjestelmään asti. Tällöin versiointitaso ei joudu ymmärtämään yksittäisten atomisten palasten tallentamista, vaan tallennustaso vastaa siitä, että kaikkien palasten kaikki versiot ovat jossain tallessa ja löydettävissä.

Käyttömukavuuden ja järjestelmän tietojen eheyden säilyttämiseksi mainittujen versionnin kerrosten palveluiden tulisi myös olla mahdollisimman automaattisia ja läpinäkyviä [Haa92].

2.2 Versioitavan tiedon muoto

Versionhallinnassa myös talletettavien tietojen muodolla on käytännön merkitystä. Tallennustilan säästämiseksi monet vanhemman polven versiointijärjestelmät tallentavat vain muutokset edelliseen versioon verrattuna. Tämä sopii erinomaisesti esimerkiksi tekstipohjaisille dokumenteille ja helpottaa jossain määrin muutosten raportointia ja seurantaa, mutta esimerkiksi binääridatalla kuten kuvatiedostoilla tilansäästöä ei useinkaan synny ja jokaisen version voidaan vain havaita eroavan bittitasolla edellisestä.

Tallennettavan tiedon luonne aiheuttaa myös lisävaatimuksia versioinnin toteuttamiselle. Hypermedian aiheuttamia vaatimuksia käsitellään seuraavassa kappaleessa.

3 Hypermedian versionhallinta

Hypermediassa dokumenteista toisiin osoittavat *hyperviitteet* luovat dokumenttien välille riippuvuusverkoston, joka pitää ottaa huomioon versioinnissa. Muutoksien seuranta on ongelmallista, koska joko viittaavaa tai viitattua dokumenttia voidaan muuttaa tai itse viittaus voidaan lisätä, poistaa tai siirtää osoittamaan muualle. Siksi hypermedian versioinnissa on otettava huomioon kaikki koko hypermediaverkkoon tehdyt muutokset. Tämä rakenteen versiointi on hypermediajärjestelmien versioin-

nin ydinongelma [PWG04].

Sama ongelma koskee myös ohjelmistojen kehityksen ja konfiguraatioiden hallintaa. Myös ne sisältävät toisistaan riippuvia, muuttuvia ja usein tekstipohjaisia dokumentteja. Siksi tutkimus on jossain määrin painottunut kyseiselle erikoisalalle ja tutkimusten yhteydessä kehitettävät järjestelmät, kuten esimerkiksi Molhado [NMB04], soveltuvat yhteisten abstrahoitujen käsitteidensä puolesta melko hyvin käytettäväksi sekä hypermedian että ohjelmistokehitysprojektien versioinnissa.

Hypermediajärjestelmiä voidaan myös luokitella sen mukaan, kykenevätkö ne toimimaan avoimien hypermediamallien, kuten www:n kanssa. Toinen tapa luokitella on järjestelmissä käytettyjen linkkirakenteiden mukaan.

Molhado-järjestelmän yhteydessä on listattu yleisen hypermedian versiointijärjestelmän vaatimuksia [NMB04]. Näitä ovat kyky käsitellä useamman kuin yhden kohteen tai lähteen linkkejä (*variable arity hyperlinks*), kyky viitata kokonaisiin dokumentteihin tai vastaaviin rakenteisin (*coarse-grained linking*), kyky viitata pienimpiikin rakenteisiin ja atomeihin (*fine-grained linking*), hypertekstirakenteen ja dokumenttien sisällön erottaminen sekä versioinnin läpinäkyvyys käyttäjän näkökulmasta.

3.1 Dokumenttikeskeinen versionhallinta

Dokumenttikeskeinen versionhallintaa käytetään monissa yhteyksissä. Dokumenttikeskeisyyden parhaita puolia ovat yksinkertaisuus ja olemassaolevat erinomaiset toteutukset, kuten esimerkiksi CVS[1]. Lisäksi yksinkertaisuuden vuoksi dokumenttikohtaiseen versionhallintaan voidaan periaatteessa rakentaa kehittyneemmätkin toiminnot.

Dokumenttikeskeisen versionhallinnan huonona puolena on, ettei se yleensä tue riippuvuuksia dokumenttien välillä – esimerkiksi hyperlinkkejä. Tällöin hypermedian versioiden eheydenhallinta joudutaan suorittamaan käsin järjestelmän ulkopuolella, mikä vaikeuttaa toimintaa ja lisää virhemahdollisuuksia.

3.2 Verkon tilannekuvat

Koko hypermediajärjestelmästä tilannekuvia (*snapshots*) ottamalla saadaan kierrettyä dokumenttikeskeisen versionhallinnan dokumenttien välisten riippuvuuksien

[1] `http://www.cvshome.org`

puutteita. Rajaamalla otoksen koko yhteen dokumenttiin on tilannekuvien ottaminen sama asia kuin dokumenttikohtainen versionhallinta. Verkon tilannekuvia otettaessa yksittäisten atomisten resurssien merkitys vähenee, kun pienin versioitava yksikkö on koko verkko tai sen osa.

Ottamalla tilannekuva koko verkosta saadaan yhtenäinen, eheä versio koko hypermediarakenteesta tai sen osasta. Tilannekuvien huono puoli on huomattava tilavaatimus jokaiselle pienellekin muutokselle. Koko verkon versiointi myös olettaa jatkuvan yhtenäisen pääsyn koko verkon kaikkiin resursseihin – uusi versio täytyy voida luoda snapshot-tyyliin heti yhdenkin muutoksen tapahtuessa. Tämä voi olla liian hidasta ja käytännössä ajankulun vuoksi mahdotonta laajassa mittakaavassa.

Verkon tilannekuvia tekee esimerkiksi omaan versioivaan tietovarastoonsa tiukasti sidottu java-kehitysympäristö VisualAge for Java. Laajimmassa mittakaavassa osittaisia tilannekuvia jättimäisestä ja jatkuvasti muuttuvasta hypermediaverkosta tarjoaa www:n sivustoista versioita tallentava Internet Archive[2].

3.3 Sisältyvyysmallit

Sisältyvyysmallit (*containment modeling frameworks*) ovat kevyt kehys sisällönhallintajärjestelmien kuten esimerkiksi hypermediajärjestelmien tietovarastojen kuvaukseen. Ne ovat toistaiseksi paras ratkaisu hypermedian versioinnin ongelmiin. Niiden avulla voidaan esittää monien olemassaolevien hypermediajärjestelmien suhteet [EJW02].

Sisältyvyysmalleissa mekaanisen bittitason versiotallennuksen sijaan luodaan malli ja kehys tavalle, jolla järjestelmän tieto ja sen suhteet rakentuvat. Sisältyvyysmalleissa on käytössä vain kaksi mallinnuspalaa: *entiteetti* (*entity*) ja *suhde* (*relationship*) [WGP04]. Hypertekstijärjestelmiä mallinnettaessa entiteettivaihtoehtoja on kaksi: *kooste* (*container*) ja *atomi* (*atom*). Atomit ovat järjestelmän pienimpiä, jakamattomia tietopalasia ja ne sisältävät esimerkiksi yksittäisiä metatietoarvoja tai osia dokumenttien sisällöstä. Koosteisiin voi liittyä kahdenlaisia suhteita: *viitteitä* (*referential relationship*) ja *osa-suhteita* (*inclusive relationship*).

Lisäksi koosteille voidaan yleensä antaa ominaisuuksia (esimerkiksi rooleja), jolloin näiden koosteiden voidaan ajatella olevan semanttisesti paremmin määriteltyjä ja kykenevän siksi erilaisiin toimintoihin.

[2] `http://www.archive.org`

Sisältyvyysmalleilla voidaan esimerkiksi erottaa hypermediasta hyperviitteet ja sisältö tekemällä hyperviittauksista ja niiden kokoelmista omat koosteensa ja antamalla kyseisille koosteille semanttinen erityismerkitys “linkityksestä”.

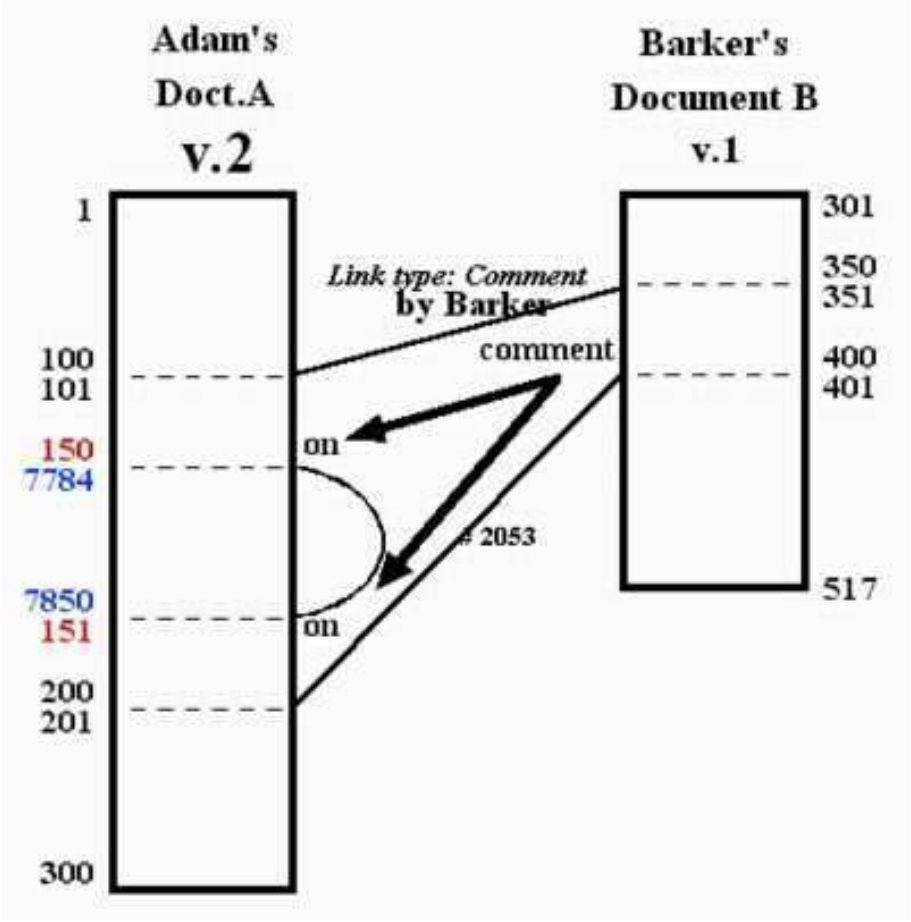
4 Esimerkkijärjestelmiä

Tässä kappaleessa esitellään muutama hypermedian versiointia ja sen hallintaa jollain tavalla toteuttava järjestelmä. Näistä Xanadu edustaa klassisempaa ja yksinkertaisinta dokumenttitason versionhallintaa ja WebDAV on HTTP:n päälle rakennettu laajennus, joka määrittää hajautettuun dokumenttien julkaisuun, muokkaamiseen ja versionhallintaan perustoiminnot. HURL tarjoaa yleistä hypermediakehystä (*framework*), jonka osana on versiointi. Bamboo, Chrysant ja Molhado vievät versionhallinnan pitimmälle sisältyvyysmalleja eri tavoin soveltamalla.

4.1 Xanadu

Xanadua ja sen *xanalogista* hypermediarakennetta on kehitetty 1960-luvulta lähtien [Nel99]. Xanaloginen hypermediarakenne tarjoaa hyperviittauksina kaksisuuntaiset sisältölinkit (*content links* tai *deep links*) ja sisällytyksen (*transclution*). Xanadussa dokumentit, versiot ja linkit rakentuvat viitelistoista, joissa viitataan toisiin rakenteisiin. Kaikki rakenteet sijaitsevat loppumattomassa Suuressa Osoiteavaruudessa (*Grand Address Space*). Tässä mielessä Xanadun rakenne muistuttaa sisällytysmalleja käyttäviä versiointijärjestelmiä: hankalahko “tiedoston” käsite hukataan ja kokonaisuudet muodostetaan atomisista palasista ja toisista kokonaisuuksista.

Kuvassa 2 esitellään esimerkki Xanadun hyperviittauksista ja versioinnista. Kuvan esittämässä tilanteessa Adam on kirjoittanut dokumentin, jota Barker on intoutunut kommentoimaan. Barker tai hänen Xanadu-kykenevä järjestelmänsä on merkinnyt dokumenttiin viittauksen väliltä 350-400 Adamin alkuperäiseen dokumenttiin välille 100-200. Tämän viittauksen teon jälkeen Adam on käynyt muokkaamassa päivitetyn alueen tekstiä.



Kuva 2: Hyperviittauksen osoittaman tekstin versiointi Xanadussa

4.2 WebDAV

WebDAV on (*Web Distributed Authoring and Versioning*) IETF:n[3] työryhmä, jonka työn tulos tunnetaan samalla nimellä. WebDAV ei varsinaisesti ole erityinen hypermedian versiointijärjestelmä, vaan se laajentaa www:n siirtoprotokolla HTTP:aa[4] lisäämällä siihen mekanismit synkronoimattomalle hajautetulle dokumenttien kirjoittamiselle [WhW98]. Käytännössä WebDAV määrittelee rajapinnan ja kutsut URIen osoittamien resurssien muokkaamiseen versioivan verkkolevyjärjestelmän tavoin. WebDAV:n toteuttava palvelin voi sisäisesti toteuttaa dokumentinhallinnan parhaalla mahdollisella tavalla, esimerkiksi tietokannassa, levyjärjestelmässä tai toisen versionhallintajärjestelmän kääreenä toimimalla.

WebDAV lisää kuusi laajennusmääritystä HTTP:aan: kirjoituslukot, metatieto-omi-

[3]Internet Engineering Task Force, `http://www.ietf.org`

[4]Hypertext Transfer Protocol, `http://www.w3.org/Protocols/`

naisuudet, nimiavaruuden hallinnan, versionhallinnan, kehittyneet kokoelmat sekä pääsynvalvonnan (*access control*). Www:lle ominaisesti WebDAV tarjoaa vain kirjoituslukot – lukemiseen ei ole lukkoja ja viimeisin tallennettu versio resurssista on suoraan HTTP:lläkin saatavilla, vanhemmat versiot suoraan vain WebDAV:lla. WebDAV:n laajennuksista kirjoituslukot, nimiavaruuden hallinta (käytännössä resurssien siirto paikasta toiseen, uudelleennimeäminen, poistaminen ym.) ja pääsynvalvonta ovat nykyaikaisen usean käyttäjän levyjärjestelmän perustoimintoja, kuten myös osa metatieto-ominaisuuksista – esimerkiksi resurssin koko tavuissa ja viimeinen muutospäivämäärä.

Yhteensopivuuden ja päällekkäisyyksien minimoimiseksi WebDAV:n versionhallinta on määritelty vielä omana, IETF:n DeltaV-työryhmänsä pohjalta nimen saaneena laajennuksena WebDAV:lle [CA+02]. WebDAV:ssa versiointipolitiikan saa halutessaan erotettua tallennuksista.

WebDAV-yhteensopivia ohjelmistoja kehitetään innokkaasti. Ainakin Microsoft, Apple ja KDE-projekti ovat sisällyttäneet WebDAV:n suoraan käyttöliittymäänsä. Apache-organisaation[5] Slide[6] sisältää sekä WebDAV-palvelimen java webapp-sovelluksena että yksinkertaisen komentorivipohjaisen asiakasohjelman. Slide toteuttaa perusmallillaan yksinkertaisen levyjärjestelmän kansioihin perustuvan versiointimallin, mutta myös tietokantavaihtoehto on käytettävissä.

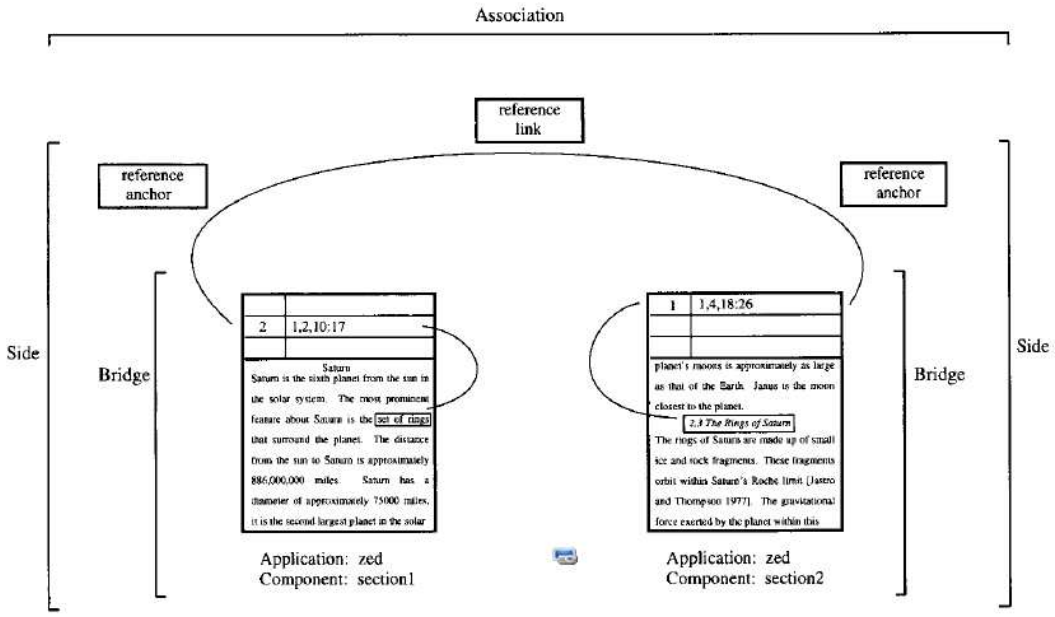
Jonkinlaisina WebDAV:n vaihtoehtoina hajautettuun julkaisu- ja kehitystyöhön on tarjolla hyvin levinneessä käytössä esimerkiksi Twiki-järjestelmät[7].

4.3 HURL

HURL on ehdotus mahdollisimman avoimesta hypermediakehyksestä kaikkien sovellusten käytettäväksi [HLN98]. HURL pyrkii erottamaan kolme tasoa hypermediajärjestelmän perustoiminnoissa: fyysisen tallennuksen, hypermedian hallintajärjestelmän ja dokumenttien sovelluskohtaiset toiminnot.

Kuvassa 3 on esimerkki HURL:n hypermediamallin hyperviittauksesta dokumentista toiseen. HURL:n hyperviittaus rakentuu kahdesta tai useammasta sillasta (*bridge*) – vähintään yhdestä kummassakin hyperviittauksen päässä (*side*) – ja itse viitelinkistä (*reference link*). Sillan määrittää kolmikko, jonka muodostavat pysyvän valinnan

[5] `http://www.apache.org`
[6] `http://jakarta.apache.org/slide/`
[7] `http://www.twiki.org`



Kuva 3: Esimerkki HURL:n hypermediaviittauksesta

tunniste (*persistent selection identifier*), dokumentin tunniste (*component identifier*) sekä käytetty sovellus (*application*). Esimerkki lienisi www:n toimintaa ajatteleville helpompi ymmärtää, jos toinen ohjelma olisi www-selain, jossa olisi jonkin tietyn URLin www-sivu auki ja sivulla olisi kuva. Kuvasta olisi silta ja viitelinkki piirto-ohjelmaan, jossa sama kuva olisi vain osa isommasta kuvasta.

HURLissa versiointi on hypermedian hallintajärjestelmän tasolla, joten kaikki kehystä käyttävät sovellukset ja niiden data pääsevät siitä hyötymään. Monimutkaisemmat logiikat, rakenteiden ja atomisten palasten erottelun ja politiikat joutuu itse rakentamaan sovellustasolle. Samaten sovellusten täytyy yleensäkin olla tietoisia ja kykeneviä kehyksen palveluista, jotta ne voisivat niitä itse hyödyntää. HURLin täysipainoiseksi käyttöönottamiseksi tämä tarkoittaisi huomattavaa uudelleenohjelmointia sovelluksissa, vaikkakin kehyksen versiointiominaisuudet ovat joustavat.

4.4 Bamboo

Bamboo on työkalu automatisoidun hypertekstivaraston luomiseksi [WGP04]. Sen toiminta perustuu sisältyvyysmallien hyödyntämiselle. Bamboossa erilaisille hypermediajärjestelmille voidaan luoda kullekin oma mallinsa, jonka pohjalta järjestelmä luo automaattisesti tarvittavan java-lähdekoodin järjestelmälle, joka toteuttaa mal-

lin MySQL-tietokannan avulla. Jokaiseen malliin voidaan myös lisätä rooleja (*roles*), joilla voidaan toteuttaa esimerkiksi versiointi. Jokaisen Bamboolla luodun hypertekstivaraston siis saa automaattisesti järjestelmän tarjoamien kehystoimintojen avulla myös versioituviksi.

Bamboon automaattisesti luomien rajapintakuvausten avulla voi myös yhdistää muutoin erillisiä hypermediajärjestelmiä. Tämä vaikuttaa poikkeukselliselta perinteisesti yhteensopimattomien hypermediajärjestelmien keskellä.

4.5 Chrysant

Chrysant tarjoaa versiointitoiminnot dokumenttimalleille, joissa dokumentit sisältävät myös hyperviittaukset – erityisesti html-tiedostoille [PWG04]. Chrysantkin soveltaa sisältyvyysmallia itse tiedostojen sisällön ja niihin osoittavat hyperviittausten versiointiin erikseen.

Chrysant myös määrittelee *rooleja* (*roles*) eli antaa tietyille koosteiden suhteille semantiikkoja. Tällaisia rooli on esimerkiksi linkki (*link*).

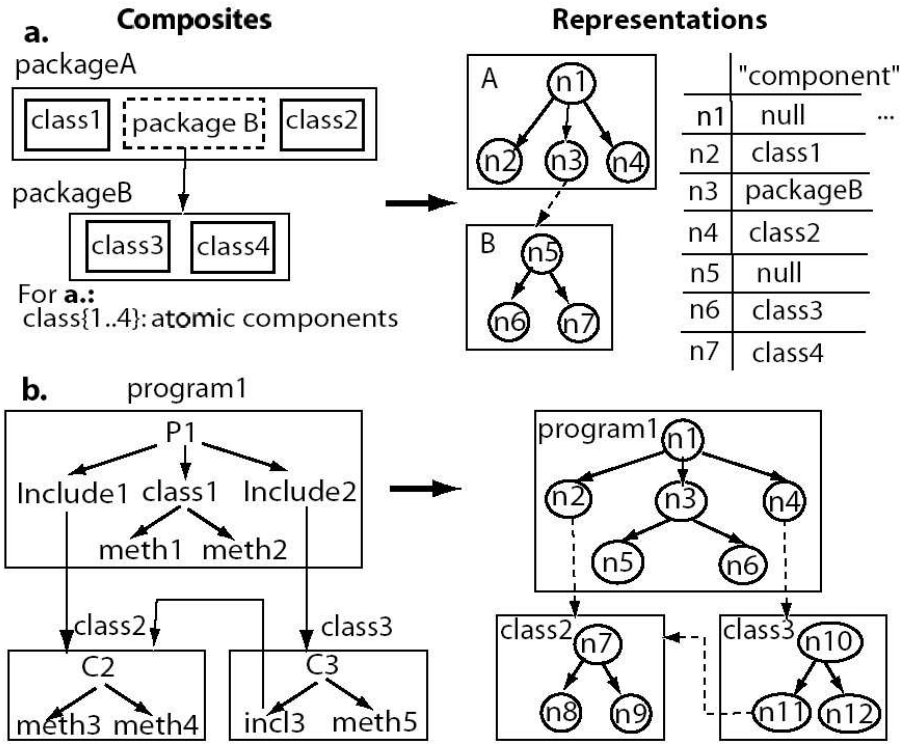
Chrysant soveltuu vain yksittäisten sivustojen versionhallintaan, koska se tulkitsee vain suhteelliset hyperlinkit versioitavaksi rakenteeksi. Tämä on pieni virhe, jonka korjaamisesta tesktissä oli ehdotus ja lisäksi vain suhteellisten linkkien olettaminen voi olla hyvin lähellä reaalimaailman sivustojen toteutustapaa.

4.6 Molhado

Molhado on ohjelmistokehitykseen ja konfiguraationhallintaan suunniteltu hypermedian versiointijärjestelmä, joka soveltuu myös muuhun versioituun hypermedian käsittelyyn [NMB04]. Bamboohon ja Chrysantiin verrattuna Molhado on suunniteltu tietylle sovellusalueelle: ohjelmistokehitykseen.

Molhadon kantavina ideoina ovat ensinnäkin varsinaisten dokumenttien viipalointi atomisiksi kokonaisuuksiksi, joista jokaiselle pidetään yllä versiohistoriaa. Esimerkiksi lähdekooditiedostojen yksittäiset funktiot tai metodit tai ohjelmistodokumentaation yksittäiset kappaleet voivat olla atomisia palasia. Toisekseen, järjestelmä tarjoaa monikäyttöisen versioidun yhdisterakenteen (*composite*), jolla voidaan esittää monenlaisia versioituja asioita, kuten atomisista palasista kootut kokonaisuudet tai eri dokumenttien palasten suhteet.

Molhadon on tarkoitus tunkeutua sisältämiensä dokumenttien sisäiseen rakenteeseen



Kuva 4: Esimerkki Molhadon koosteista

asti ja hyödyntää valmiita rakenteita yhdisterakenteiden löytämiseksi ja atomien erottamiseksi. Kuvassa 4 on esimerkki Molhadossa olevista dokumenteista, jotka ovat pilkkoutuneet osiin.

5 Yhteenveto

Hypermedian versiointi on oleellinen kysymys rakennettaessa toisiinsa liittyviä, pitkäikäisiksi suunniteltuja tietovarastoja. Valitettavasti ratkaisut ovat usein sovellusalakohtaisia, mikä rajoittaa niiden yleistymistä. Lisäksi tutkimuksessa ei välttämättä oteta huomioon yhteensopivuutta muiden hypermediajärjestelmien kanssa. Vain WebDAV:lla ja Bamboolla voi olettaa voivansa kytkeä erilaisia hypermediajärjestelmiä yhteen. Tärkeää tällainen yhdistäminen olisi siksi, että se mahdollistaisi järjestelmien kasvattamisen vähitellen nykyisen www:n rinnalle siten, että tieto kulkisi järjestelmien välillä.

Versioinnissa lupaavin ratkaisu on sisällytysmallien soveltaminen. Sisällytysmalleissa hypermediadokumenteista erotetaan rakenne ja linkit varsinaisista atomisista sisäl-

töosista ja versioitiin molempia erikseen.

Esitellyistä järjestelmistä luottaisin WebDAV:n leviämiseen. Sillä on tietotekniikan huippuyrityksien tuki, se on IETF:n RFC-standardi, se tukeutuu olemassaolevaan www-infrastruktuuriin ja laajentaa sitä, se on verrattaen helppo toteuttaa – ja mikä ehkä kaikkein tärkeintä, se skaalautunee ylivertaisesti, kun sitä verrataan mihin tahansa muuhun tässä esiteltyyn järjestelmään – tai itse asiassa sillä voidaan peittää muita versionhallintajärjestelmiä. Skaalautuminen ja yksinkertaisuus – käytännössä hyvin pitkälti HTTP:n 404-virhekoodi – saivat puutteellisen nyky-www:n yleistymään, ja en näe syytä olettaa tilanteen muuttuneen.

Mielenkiintoista on myös seurata markkinoivalla dominoivassa asemassa olevan käyttöjärjestelmäntoimittajan ratkaisuja tulevan levyjärjestelmän kanssa. Microsoft Windows sisältää jo nykyisellään tuen WebDAV:lle, mutta seuraavan Windows-version mukana lupailtu uudistettu levyjärjestelmä saattaa tarjota joitain näissäkin järjestelmissä esitettyjä piirteitä yleisempään käyttöön.

Lähteet

CA+02 Clemm, G., Amsden, J., Ellison, T., Kaler, C. ja Whitehead, J., *Versioning Extensions to WebDAV*. IETF, 2002. `http://www.ietf.org/rfc/rfc3253.txt`. [5.11.2004]

EJW02 E. James Whitehead, J., Uniform comparison of data models using containment modeling. *Proceedings of the thirteenth ACM conference on Hypertext and hypermedia*. ACM Press, 2002, sivut 182–191.

WGP04 E. James Whitehead, J., Ge, G. ja Pan, K., Automatic generation of hypertext system repositories: a model driven approach. *Proceedings of the fifteenth ACM conference on Hypertext & hypermedia*. ACM Press, 2004, sivut 205–214.

WhW98 E. James Whitehead, J. ja Wiggins, M., Webdav: IETF standard for collaborative authoring on the web. *IEEE Internet Computing*, 2,5(1998).

Haa92 Haake, A., Cover: a contextual version server for hypertext applications. *Proceedings of the ACM conference on Hypertext*. ACM Press, 1992, sivut 43–52.

HLN98 Hicks, D. L., Leggett, J. J., Nürnberg, P. J. ja Schnase, J. L., A hypermedia version control framework. *ACM Trans. Inf. Syst.*, 16,2(1998), sivut 127–160.

Kau04 Kauppinen, T., The ontology versioning framework. *C-Sarja 2004, Tietojenkäsittelytieteen laitos, Helsingin Yliopisto.*

Nel99 Nelson, T. H., Xanalogical structure, needed now more than ever: parallel documents, deep links to content, deep versioning, and deep re-use. *ACM Comput. Surv.*, 31,4es(1999), sivu 33.

NMB04 Nguyen, T. N., Munson, E. V. ja Boyland, J. T., The molhado hypertext versioning system. *Proceedings of the fifteenth ACM conference on Hypertext & hypermedia.* ACM Press, 2004, sivut 185–194.

PWG04 Pan, K., E. James Whitehead, J. ja Ge, G., Hypertext versioning for embedded link models. *Proceedings of the fifteenth ACM conference on Hypertext & hypermedia.* ACM Press, 2004, sivut 195–204.