hyväksymispäivä arvosana

arvostelija

Linkkirakenteen eheys

Sami Männistö

Helsinki 19. huhtikuuta 2002
Hypermediajärjestelmät -seminaari
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Linkkirakenteen eheys

Sami Männistö

Hypermediajärjestelmät –seminaari
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos
19.04.2002, 16 sivua

Kun hypertekstin määrä avoimissa hypermediajärjestelmissä kasvaa ja sitä ylläpidetään ja uudelleen järjestellään, saattaa käydä, että linkkirakenne ei pysy eheänä. Tätä kutsutaan rikkinäiseksi linkiksi. Rikkinäisiä linkkejä ovat linkit, jotka eivät enää osoita sen alkuperäisesti tarkoitettuun tietoon.
Jos linkki ei onnistu yhdistämään sisältöviittauksen osoittamaan paikkaan, sen sanotaan olevan rikki. Jos yhdistäminen ei onnistu, koska tunniste ei enää vastaa yhtään validia dokumenttia eli dokumenttia ei löydy, niin sanotaan, että linkki on irrallinen. Tämä ongelma ilmenee, jos dokumentti on poistettu tai siirretty ilman linkin päivittämistä. Jos yhdistäminen epäonnistuu, koska sisältö ei enää vastaa oikeellista sisältöä tarkoitetussa dokumentissa eli linkki viittaa väärään paikkaan, niin kysessä on sisältöviittausongelma.
Näiden kummankin ongelman esiintymistä voidaan joko yrittää välttää järjestelmän suunnitteluvaiheessa tai niihin voidaan kehittää korjauskeinoja.

Aiheluokat(Computing Reviews 1998): H.5.4, H.5.1, I.7.4

Avainsanat: Linkkirakenteen eheys, rikkinäinen linkki

SISÄLTÖ

1. JOHDANTO 1

2. LINKKIEN TALLENNUSMALLIT 2

2.1 UPOTETTU LINKKI 2
2.2 ULKOINEN LINKKI NIMETYLLÄ PÄÄTEPISTEELLÄ 3
2.3 ULKOINEN LINKKI VIITTAUKSELLA 4

3. RATKAISUJA IRRALLISEN LINKIN ONGELMAAN 6

3.1 W3OBJEKTIN- VIITTAUSMALLI 6
3.1.1 W3Objekti-mallin edelleenviittaus 8
3.2 YKSILÖLLINEN NIMEÄMINEN 9
3.3 HYPERKANNAT 10

4. RATKAISUJA SISÄLTÖVIITTAUSONGELMAAN 11

4.1 JULKAISUMALLI 11
4.2 MANUAALINEN LINKKIEN EDITOINTI JA LINKIT HUOMIOON OTTAVAT EDITOINTITYÖKALUT 11
4.3 LINKKIEN KORJAAMINEN KÄYTÖN AIKANA 12
4.4 HAKUJEN KÄYTTÖ LINKKIEN MÄÄRITTELYSSÄ 12

5. YHTEENVETO 13

LÄHTEET 15

1. Johdanto

Hypermedian pääperiaatteita on mahdollisuus navigoida laajojen multimediatiedostojen välillä. Hypermedian käytetyin navigointikeino on linkkien seuraaminen. Linkkimallit voivat vaihdella tiukasti yhdistetyistä linkeistä, joita käytetään WWW:ssä HTML-kielen linkeissä aina löysästi kytkettyihin linkkeihin, joita esiintyy esimerkiksi joidenkin linkkipalveluiden yhteydessä. Kun hypertekstin määrä avoimissa hypermediajärjestelmissä kasvaa ja sitä ylläpidetään ja uudelleen järjestellään, saattaa käydä, että linkki, joka aikanaan osoitti sen tekijän tarkoittamaan paikkaan lakkaa tekemästä näin. Tätä kutsutaan rikkinäiseksi linkiksi. Rikkinäisiä linkkejä ovat linkit, jotka eivät enää osoita sen alkuperäisesti tarkoitettuun tietoon.

Yleisesti linkki voi yhdistää useita ankkureita. Linkki voi olla joko yksi- tai kaksisuuntainen. Ankkurit voivat olla suunnattuja, joko lähde- tai kohdeankkureita (yksisuuntainen linkki) tai kaksisuuntaisia, jotka käyttäytyvät sekä lähde- että kohdeankkureina (kaksisuuntainen linkki). Ankkuri koostuu dokumentin tunnisteesta ja sisältöviittauksesta. Tunnisteesta systeemi päättelee komponentin tai joukon komponentteja, joista koostuu käyttäjälle esitettävä dokumentti. Sisältöviittaus taas on joku tietty kohta tai isompi alue, jota voidaan klikata lähdeankkurissa tai johon viitataan kohdeankkurissa.

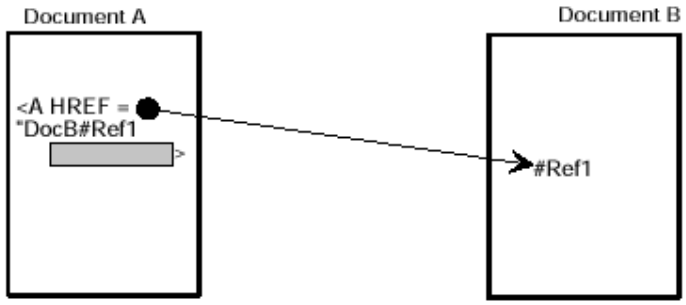
Jos ankkuri ei onnistu yhdistämään sisältöviittauksen osoittamaan paikkaan, sen sanotaan olevan rikki. Jos yhdistäminen ei onnistu, koska tunniste ei enää vastaa yhtään validia dokumenttia eli dokumenttia ei löydy, niin sanotaan, että linkki on irrallinen tai roikkuva (dangling). Tämä ongelma ilmenee, jos dokumentti on poistettu tai siirretty ilman linkin päivittämistä. Jos yhdistäminen epäonnistuu, koska sisältö ei enää vastaa oikeellista sisältöä tarkoitetussa dokumentissa eli linkki viittaa väärään paikkaan, niin kysessä on sisältöviittausongelma (content reference). Ongelma ilmenee, jos dokumenttia vaihdetaan päivittämättä siihen viittaavia linkkejä. [Dav98],[Dav99]

2. Linkkien tallennusmallit

Käsittelen tässä dokumentissa kolmea erilaista linkkien tallennusmallia ja sen jälkeen sekä irrallisen linkin ongelman ja sisältöviittausongelman esiintymistä niissä ja mahdollisia ratkaisuongelmia linkkirakenteen eheyden ylläpitämiseen. Kolme linkkien tallennuksen mallia ovat *upotettu linkki*, *ulkoinen linkki nimetyllä päätepisteellä* sekä *ulkoinen linkki viittauksella*. Kaikilla kolmella tavalla on omat etunsa ja haittansa linkkirakenteen eheyden kannalta.

2.1 Upotettu linkki

Upotetussa linkissä lähdeankkuri on upotettu sisältöön ja myös kohdeankkuri on tässä kohdassa (kaksisuuntaisessa linkissä). Upotettu linkki on yleisesti WWW:ssä käytetty metodi. Tälläiset linkit ovat sidottuja tiettyyn objektiin lähdedokumentissa ja kuvaavat kohdeankkurin esim. U R L:llä. Kuvassa 1. esitetään upotetutun linkin periaate.



Kuva 1. Upotettu linkki

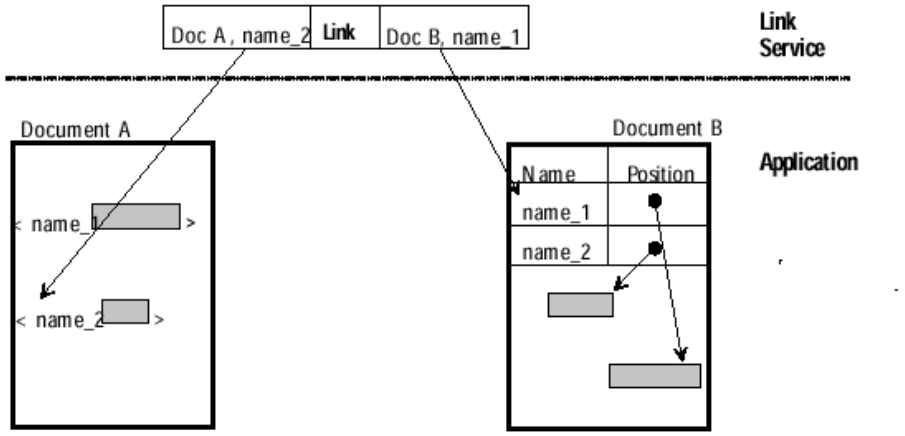
Tälläisen tallennusmuodon etuja ovat yksinkertaisuus ja skaalattavuus. Haittana taas on se, että vain linkin sisältävä solmu on tietoinen siitä. Näin ollen

linkit voivat olla vain yksisuuntaisia. Linkkirakenteen tutkiminen on myös erittäin työlästä. Joitakin työkaluja on kehitetty tälläisen linkkirakenteen tutkimiseen. Niiden toiminta perustuu siihen, että annetusta alkupisteestä lähtien kuljetaan kaikkien linkkien läpi ja mallinnetaan linkkirakennetta. Linkkirakenteen eheyttä on kuitenkin erittäin vaikea ylläpitää, kun solmu poistetaan. WWW:n tapauksessa se tarkoittaisi jokaisen HTML-dokumentin läpikäyntiä mahdollisten linkkien etsimiseksi ja muuttamiseksi. Koska tämä ei onnistu, niin seurauksena on irrallisia linkkejä. Vaikka onnistuisimmekin paikantamaan linkit, niin niiden muuttaminen olisi mahdotonta, koska ne on upotettu sisältöön, jota meillä ei ole oikeutta ylläpitää. [Dav98]

2.2 Ulkoinen linkki nimetyllä päätepisteellä

Toisessa mallissa linkkejä ylläpidetään keskitetyssä linkkikannassa (linkbase). Linkkikannassa olevat linkit kytkevät toisiinsa ankkurit, joilla on muuttumattomat nimet. Käytettävä sovellus ymmärtää nämä nimet ja osaa etsiä ja näyttää niitä vastaavat dokumentit. Tällaisia nimettyjä ankkureita voivat olla mm. merkkaus SGML dokumentissa, objekti CAD-piirroksessa, alue taulukossa, kirjanmerkki word-dokumentissa tai nimetty viittaus dokumentin osaksi tallennetussa viittaustaulukossa (Kuten kuvan 2. dokumentissa B).

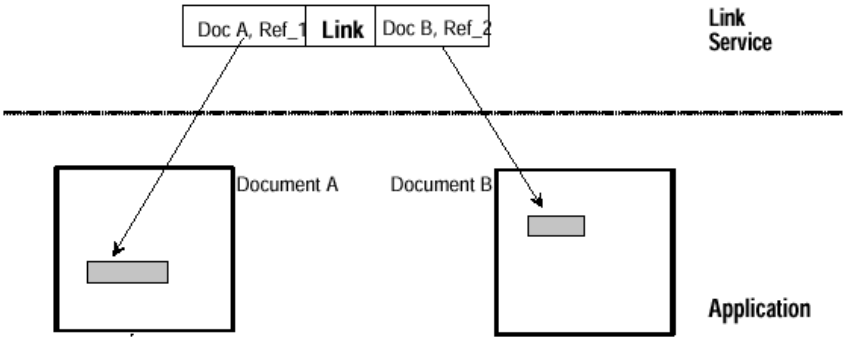
Kun linkit on erotettu sisällöstä, niin linkkirakenteen analysointi on helpompaa. Jos linkkipalvelua ylläpidetään, niin on mahdollista siirtää dokumentteja aiheuttamatta irrallisia linkkejä. Dokumentteja on myös mahdollista editoida kunhan upotetut ankkurinimet pysyvät muuttumattomina. Tämä linkkirakenne tuo parannuksia upotettuun linkkimalliin, mutta on myös monimutkaisempi eikä niin skaalattava, koska jokaisessa linkin seuraamisessa on käytävä linkkipalvelussa.[Dav98]



Kuva 2. Esimerkki ulkoisesta linkistä, jossa ankkurinimet on tallennettu link-kisolmuun.

2.3 Ulkoinen linkki viittauksella

Kolmannessa tallennustavassa ulkoiseen linkkikantaan tallennetaan viittaus sisältöön linkin mukana. Tästä esimerkki on kuvassa 3. Tämän viittauksen tyyppi riippuu dokumentin datatyypistä. [Dav98]



Kuva 3. Linkki ja sisältöviittaus linkkipalvelussa

On olemassa myös muita keinoja määritellä viittaus objektiin datan sisällä kuin nimeäminen. Tässä on esitetty kaksi muuta tapaa.

Laskeminen. Dataa käsitellään aina jossain tietyssä järjestyksessä. Tähän voidaan tukeutua myös viittausta määriteltäessä. Esimerkkejä: Tietty merkkijono, joka on on 1293-1314 merkki tekstitiedostossa. 23. kuva videoleikkeessä. Ääniraita, joka esiintyy 5123-6824 ms CD-levyn alusta. Laskentaan perustuva malli toimii lähes jokaisessa dataformaatissa ja on siksi käytetyin. Tätä metodia käytetään myös usein määrittelemään linkin päätepistettä. Tällainen linkki kuitenkin herkästi rikkoontuu, jos tiedostoa editoidaan ilman linkkipalvelun päivitystä.

Hakeminen. Hakukonetta tai scriptiä voidaan käyttää paikallistamaan linkin päätepiste. Ankkuri voidaan ilmaista merkkijonona tekstitiedostosta tai SQL kyselynä tietokantaan. Jokin muuttumaton hakuehto, jonka avulla linkin päätepiste pystytään paikallistamaan.

Näitä kahta ehtoa voidaan myös yhdistää nimen perusteella tapahtuvaan loppupisteen paikallistamiseen. Etuna viittausmallin käytössä on se, että nyt linkkikantaan ei tarvitse tallentaa mitään tietyn merkkauskielen merkkiä eikä datan mukana tarvitse olla viittaustaulukkoa, johon viittaukset on tallennettu. Tästä on hyötyä esim.

1. Kolmannen osapuolen sovellukset voivat käyttää linkkikantaa.
2. Mahdollisuus luoda geneerisiä linkkejä. Jos linkin ankkuri on upotettu dataan, niin linkin seuraaminen on mahdollista vain sieltä, missä lähdeankkuri on. Kun taas ankkuri pidetään erillään datasta, niin mahdollistetaan linkin seuraaminen useammasta paikasta. Esimerkiksi Microcosm [Car94] mahdollistaa tällaisten geneeristen linkkien seuraamisen ja luomisen käytön aikana.
3. Linkkejä on mahdollista luoda myös tiedostoihin, joihin käyttäjällä on vain lukuoikeus.
4. Mahdollisuus ylläpitää vaihtoehtoisia käyttäjäkohtaisia linkkivalikoimia tai linkkiverkkoja.

Koska tässä metodissa sisältöviittaus ei ole enää niin tiukasti kytketty dataan, niin dokumentin editoinnista voi seurata se, että ankkuri ei enää viittaa oikeaan kohteeseeen eli esiintyy sisältöviittausongelma.

3. Ratkaisuja irrallisen linkin ongelmaan

Irrallisen linkin ongelma ilmenee, kun ankkuri ei enää viittaa validiin dokumenttiin. Irrallisia linkkejä voi esiintyä kaikissa edellä esitetyissä linkkien tallennusmalleissa. Järjestelmissä, joissa käytetään ulkoisia linkkejä voivat sekä lähde- että kohdeankkurit olla irrallisia. Lähdeankkurin irrallisuus ei tietenkään haittaa käyttäjää, koska hän ei ikinä pääse käyttämään linkkiä. Sen sijaan se aiheuttaa linkkikantaan turhaa dataa, joka pitää poistaa roskienkeruun avulla. [Dav98]

Tällä hetkellä yleisin tapa hoitaa irrallisen linkin ongelma on jättää se hoitamatta. Vastuu viittauksien eheydestä jätetään linkkien tekijän vastuulle. Jokaisen on hoidettava itse omien linkkiensä toimivuus. Tähän on kuitenkin kehitetty myös työkaluja, joilla linkkien eheyttä voidaan tutkia. Tällaiset agenttiohjelmistot käyvät läpi kaikki linkit ja raportoivat niiden toimivuudesta ja joko poistavat linkkejä tai haluttaessa jäljittävät kohdedokumentin. Kun linkit on tallennettu ulkoisen linkkikantaan, niin niiden toimivuuden tutkiminen on helpompaa. Tällöin voidaan tutkia myös lähdeankkuristaan roikkuvat linkit ja poistaa ne linkkikannasta.

3.1 W3Objektin- viittausmalli

W3Objekti-malli on kehitetty tutkimuksessa, jossa kehitettiin laajennettavaa verkkoinfrastruktuuri- mallia, joka pystyisi käsittelemään laajoja resurssi- ja palvelujoukkoja eikä vain dokumenttipohjaisia olioita kuten nykyinen WWW. Mallissa verkon resurssit esitetään W3Objekteina, jotka ovat yksilöllisesti nimettyjä, kapseloituja resursseja, joilla on määritelty sisäinen tila. W3Objektit voivat tukea myös useita rajapintoja. [ICL96]

W3Objekti mallissa on haluttu kehittää toimiva ja yhtenäinen viittausmalli, joka määrittelee objektin luonnin, poiston ja siirron aiheuttamat toimenpiteet. Yleisin syy linkkien irrallisuuteen on sellaisten dokumenttien poistaminen tai siirtäminen, joihin yhä viitataan muista dokumenteista. Kun dokumenttia siirretään on olemassa neljä tapaa käsitellä viittauksia ja niiden eheyttä. [ICL96]

1. Älä tee mitään. Kun viittausta seuraavan kerran kutsutaan, niin virhe huomataan ja objekti pitää hakea uudelleen.
2. Korvaa viittattava objekti edelleenviittauksella (forward reference). Kutsut voidaan näin ohjata uuteen paikkaan.
3. Päivitä jotakin nimipalvelinta tai -palvelua. Kun virhe kutsun aikana aiheutuu, niin nimipalvelun kautta saadaan uusi sijaintitieto.
4. Lähetä ilmoitus kaikille viittaaville objekteille, saattaksesi heidän tietoonsa uusi sijaintipaikka.

Taulukossa 1. on vertailtu näiden neljän vaihtoehdon edullisuutta W3Objekti-ympäristössä. Edelleenviittaus on todettu edullisimmaksi vaihtoehdoksi, koska siinä viestien välitystä ei tapahdu lainkaan siirtovaiheessa (Migration costs) ja kutsuvaiheessa (Invocation costs) tarvitaan vain yksi ylimääräinen kutsu jokaisesta viittauksesta ensimmäisellä kerralla.

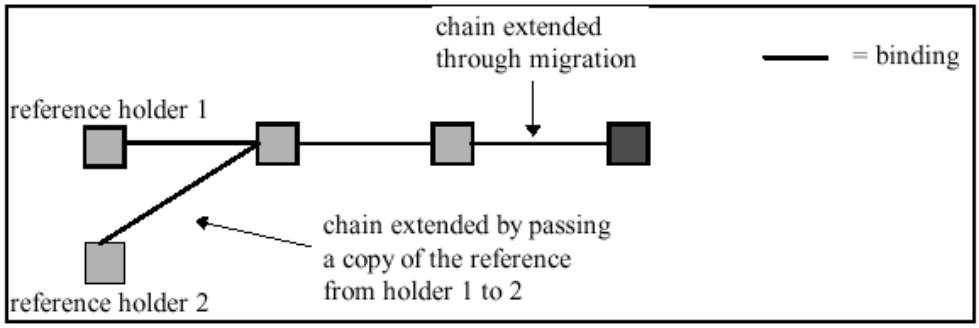
	Migration costs (message overhead)		**Invocation costs (message overhead)**
	Fault-free	With faults	
Forward Referencing	0		R - 1
Search	0		(1 ... (S - 2)) * (R - 1)
Callback	R - 1	[R]	0
Name service	N	[N]	(R - 1) * 2

Taulukko 1. Kustannukset W3Objekti- viittausmallissa.

Edelleenviittaus on myös nykyisessä WWW:ssä erittäin käytetty metodi. Normaalisti WWW ympäristössä siirrettävän dokumentin tilalle jätetään solmu, jolla on tieto uudesta sijainnista. Tämä toimii niin kauan kun palvelin ja sen hakemistorakenne pysyvät muuttumattomina. Tällaiset edelleenohjaukset voivat kuitenkin alkaa ärsyttää, jos niitä ei ajan mittaan korjata.

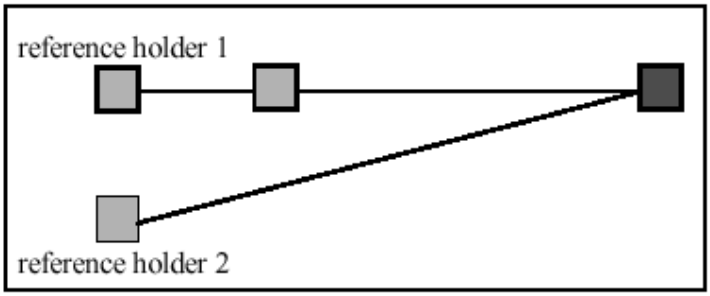
3.1.1 W3Objekti-mallin edelleenviittaus

Kun objekti siirretään uuteen kohteeseen, kaikki viittaukset siihen jäävät roikkuviksi, eli ne eivät viittaa oikeaan kohteeseen. Kun taas siirrettävästä objektista jätetään edelleenviittaus, niin viittauksesta on välillinen polku viitattavaan objektiin. Uudet siirrot voivat aiheuttaa ketjun tällaisia välillisiä viitauksia. Ketju voi myös laajentua toiseen suuntaan, jos viittava puoli välittää viittauksen toiselle objektille. Nämä molemmat mallit on esitetty Kuvassa 4.



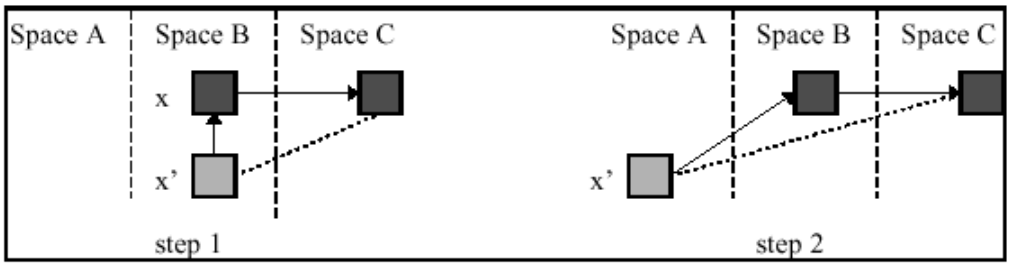
Kuva 4. Laajentuneet viittausketjut

W3Objektien viittausmallissa ensimmäinen kutsu välillisen viittauksen kautta aiheuttaa oikopolun luomisen. Tämän jälkeen roskienkeruu huolehtii kaikkien turhien edelleenviittauksien poistamisesta. Kuvassa 2. toisen viittauksen haltijan (Reference holder 2.) kutsu käynnistää operaation ja oikopolku luodaan käyttäen saman viestin sisään pakattua tietoa, jotan ylimääräistä viestiä ei tarvita.



Kuva 5. Tilanne oikopolun luomisen jälkeeen

Kun viittaus välitetään toiselle osapuolelle, viittauksen haltija luo ensin uuden viittauksen omaan osoiteavaruuteensa, jolloin uusia viestejä ei tarvita, ja sitoo sen olemassaolevaan viittaukseen. Tämän jälkeen hän välittää viestin uudesta viittauksesta toiselle osapuolelle, ja ketju laajenee taaksepäin. Viestiin kuitenkin sisällytetään vihje, joka mahdollistaa uuden viittauksen ohittamisen, jotta vältyttäisiin turhilta edelleenviittauksilta. Uusi viittaus varmistaa alkuperäisen viittauksen eheyden, kunnes vihje on saatu perille ja optimoitu polku käyttöön. Tilanne on kuvattu kuvassa 5.



Kuva 6. Viittauksen välitys

W3Objekti-mallin edelleenviittaus voi aiheuttaa joitakin ongelmiakin. Jos viittausketju sisältää edelleenviittauksia, niin mahdolliset virhepaikat lisääntyvät. Myös järjestelmän suorituskyky viittauksen kutsuaikana saattaa heiketä. [ICL96]

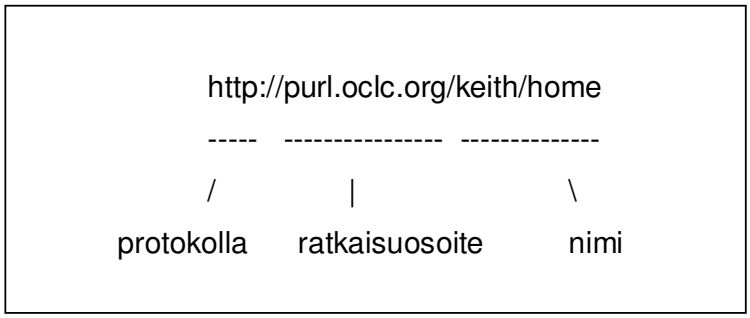
3.2 Yksilöllinen nimeäminen

Jos jokainen dokumentti on nimetty globaalisti muuttumattomalla nimellä, niin linkit voivat viitata näihin nimiin absoluuttisen dokumentin sijaan. Nimipalvelin on silloin vastuussa absoluuttisen osoitteen paikantamisesta. Tällaisena nimipalveluna toimii OCLC:n PURL. [Sha96]

PURL on oikeastaan välimuoto, joka on kehitetty odoteltaessa URN syntaksin valmistumista ja käyttöönottoa. PURL (Persistent Uniform Resource Locator) on toiminnallisesti kaikille tuttu URL. Kuitenkin sen sijaan, että se osoittaisi suoraan johonkin Internetin resurssiin, se osoittaa välitason ratkaisupalveluun.

Palvelu yhdistää PURL:n varsinaiseen U R L:n ja palauttaa sen asiakas-sovellukselle. Asiakas voi sen jälkeen jatkaa siirtymäänsä U R L : n takana olevaan resurssiin normaalisti. Toiminto on http:n perus redirect.

PURL osoitteessa on kolme osiota , U R L:sta tuttu protokolla, ratkaisuosoite ja varsinainen nimi.



Kuva 7. RURL- nimen rakenne.

Kun dokumentin ylläpitäjä siirtää dokumenttia, niin hän välittää muuttuneen tiedon PURL-palvelulle.

3.3 Hyperkannat

Hyperkannat ovat hypermediasysteemejä, jotka ovat usein rakennettu perinteisen tietokannanhallintasysteemin päälle. Tällaisessa systeemissä kaikista muutoksista, kuten poistoista ja siirroista, pitää ilmoittaa systeemille jollakin viestillä. Tämän seurauksena systeemi automaattisesti päivittää linkkien kohde- ja lähdeankkuritietoja systeemiinsä. Linkkipalvelu tai linkkitietokanta on osa tällaista systeemiä. Hyperkannassa myös kaikki muut objektit on kytketty kantaan. Tämä aiheuttaa sen, että tällaista järjestelmää käytettäessä ei voida työskennellä enää tavallisten lukuohjelmien kanssa, vaan ohjelman on kyettävä kommunikoimaan järjestelmän kanssa. Tällaisia hyperkantajärjestelmiä ovat mm. Hyper-G [AKM95], HyperDisco [WiL96] ja HB1 [Sch93].

4. Ratkaisuja sisältöviittausongelmaan

Sisältöviittausongelma esiintyy vain, jos ankkurin sisältöviittaus esitetään ulkoisena viittauksena sisältödataan. Jos dataa editoidaan, niin viittaus ei enää ole oikea. Esiintyessään tämä tilanne on aika vakava. Siitä syystä useat hypermediajärjestelmien suunnittelijoista ovat hylänneet tällaisen linkkirakenteen, vaikka se muuten toisikin useita etuja suunnitteluun. [Dav98]

Linkkipalvelut kuitenkin voivat tutkia rakenteen eheyttä ja ainakin varoittaa käyttäjää tällaisessa tilanteessa. Lisäksi hypermedia-järjestelmien dokumenteista vain harvoja editoidaan jatkuvasti ja ne joita editoidaan ovat yleensä aina tekstidokumentteja. Tästä voimme päätellä että tekstipohjaiset sisältöviittaukset ovat kaikkein kriittisimpiä.

Ongelman ratkaisuun on kaksi lähestymistapaa. Ensimmäisessä käyttäjä tietää editoidessaan dokumenttia, että siitä voi aiheutua ongelmia ja valmistautuu estämään niitä. Toisessa käyttäjä luottaa siihen että systeemi estää ongelman synnyn.

4.1 Julkaisumalli

Yksinkertaisin ratkaisu on datan oikeuksien rajoittaminen. Jos dataan annetaan vain lukuoikeudet, kun se on kerran lisätty järjestelmään, niin ongelmaa ei pääse syntymään. Monet hypertekstijärjestelmät ovatkin itseasiassa tällaisia. Opetusmateriaalit ja tekniset dokumentaatiot usein julkaistaan CD-rom levyllä, jota ei käyttäjä pääse editoimaan. Uuden version luominen taas ei tällaisessa järjestelmässä onnistu. [Dav98]

4.2 Manuaalinen linkkien editointi ja linkit huomioon ottavat editointityökalut

Yksi keino välttää sisältöviittausongelmaa on jättää vastuu linkkien eheydestä käyttäjälle, joka editoi dokumentteja. Tämä on myös kaikkein hankalin mene-

telmä. Tässä keinossa järjestelmässä pitää olla keino etsiä kaikki viittaukset kyseiseen dokumenttiin, jotta käyttäjä voisi muuttaa ne dokumentin editoinnin jälkeen manuaalisesti. [Dav98]

Käyttäjälle voidaan myös tarjota editointivälineitä, jotka ymmärtävät järjestelmän dataformaattia ja ovat yhteydessä järjestelmän linkkikantaan. Näin käyttäjä editoidessaan näkee dokumentin viittaukset ja pystyy ottamaan ne huomioon editointia tehdessään. Tämän keinon huonona puolena on se, että avoimissa hypermediajärjestelmissä pitäisi pystyä työskentelemään millä dataformaatilla tahansa. Tämä taas tarkottaisi, että jokaiselle dataformaatille pitäisi tehdä omat editointityökalunsa, eikä standardityökaluilla pystyisi työskentelemään. [Dav98]

4.3 Linkkien korjaaminen käytön aikana

Linkkikantaan voidaan laittaa päiväys linkin yhteyteen, koska linkki on luotu. Kun käytämme linkkiä, niin käyttämämme lukuohjelma voi tarkistaa päiväyksen myös tiedostosta ja verrata niitä. Jos tiedostoa on editoitu myöhemmin, niin voimme ainakin varoittaa käyttäjää tai korjata linkin. [Dav98]

Voimme myös tallettaa linkin yhteyteen kopion kohdedatasta. Kun linkki aktivoidaan, niin lukuohjelma voi tarkistaa onko data sitä mitä haluttiin. Jos se ei ole, niin ohjelma voi yrittää etsiä oikea dataa järjestelmästä linkin yhteyteen tallennetun datan avulla. Kun oikea data löytyy voimme korjata myös linkin. Tällaisen järjestelmän etuja on, että se ei tee oletuksia linkkirakenteen eheydestä ennenkuin linkkiä käytetään. Haittana on, ettemme pysty käyttämään standardilukuohjelmia, vaan lukuohjelmien pitää olla tietoisia linkkikannasta.

4.4 Hakujen käyttö linkkien määrittelyssä

Joissakin järjestelmissä linkkejä ei määritellä tietystä ankkurista toiseen ankkuriin, vaan ne määritellään erilaisten kyselyiden tai hakujen perusteella.

Tähän on kaksi vaihtoehtoa. Voimme määritellä linkkikantaan valmiiksi hakuehdon, jonka perusteella vastaava dokumentti etsitään järjestelmästä. Tässä tavassa linkki pysyy eheänä niin kauan kun kyseinen kohta löytyy jostakin dokumentista. Mitkään muut editoinnit tai siirrot eivät vaikuta siihen.

Yleensä hypertekstijärjestelmät on tehty siten, että käyttäjälle tarjotaan valmiita polkuja kuljettavaksi hypertekstissä. Järjestelmä voidaan kuitenkin rakentaa myös siten, että käyttäjälle annetaan mahdollisuus personalisoida omaa käyttöään, luomalla dynaamisia linkkejä käytön aikana. Tällaisissa järjestelmissä linkit voidaan luoda käytön aikana käyttäjän mielenkiinnon ja tarkoitusperien mukaisesti. Käyttäjä voi määritellä linkkejä käytön aikana valitsemalla esimerkiksi tekstiä dokumentista ja kysymällä järjestelmältä onko tästä aiheesta lisätietoa. Haun perusteella relevanteimmasta dokumentista tulee linkin päätepiste. Tällaiset linkit vaihtelevat käyttäjä- ja sessiokohtaisesti. Etuina on, että tällaiset dynaamiset linkit eivät mene rikki ja myös uusi järjestelmään lisätty materiaali on välittömästi linkkien käytössä. Haittana on, että suorituskyky kärsii, koska haku jokaisen linkin kohdalla vie resursseja. [BoC99]

5. Yhteenveto

Kaikissa järjestelmissä, joissa kytketään toisiinsa rakenteellista tietoa on mahdollisuus epäeheyteen. Hypermediajärjestelmissä on mahdollisuus irrallisiin linkkeihin ja sisältöviittausongelmaan. Ratkaisumahdollisuuksia näiden välttämiseen tai korjaamiseen on useita. Niistä on erotettavissa viisi erilaista tapaa lähestyä ongelmaa.[Dav98]

1. *Älä välitä*. Muutosten jälkeen käyttäjän linkit järjestelmässä saattavat lakata toimimasta. Ongelma ei ole vakava, kunhan käyttäjää informoidaan asiasta.
2. *Vältä ongelmaa*. Älä käytä tarkasti määriteltyjä ankkureita, vaan selittäviä linkkejä tai kyselyjä linkkien määrittelyssä.
3. *Löyhästi sidotut linkit.* Systeemi ei ylläpidä linkkirakennetta, vaan tarjoaa käyttäjälle mahdollisuuden siihen. Jos käyttäjä ei tätä tee, niin rikkinäisiä linkkejä voi esiintyä.

4. *Automaattinen linkkien korjaus*. Systeemi hyväksyy että rikkinäisiä linkkejä voi esiintyä, mutta tarjoaa työkaluja niiden korjaamiseen käytön aikana.
5. *Tiukasti sidotut linkit*. Kaikki hypertekstin objektit ovat hyperkannan kontrollissa ja kaikki muutokset systeemissä kulkevat sen kautta. Näin myös rakenteen eheys on taattu.

Lähteet

AKM95 Andrews, K., Kappe, F., Maurer, H., The Hyper-G Network Information System. J.UCS, 1, 4(1995).

BoC99 Bodner, R., Chignell, M., Dynamic Hypertext: Querying and Linking. ACM Computing Surveys, 31, 4 (1999).

Car94 Carr, L. et al., The Microcosm Link Service and its Application to the World Wide Web. Proceedings of the First WWW Conference, Geneva, May 1994.

Dav98 Davis, H.C., Referential Integrity of Links in Open Hypermedia Systems. The Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia, Hypertext '98, Pittsburgh, Pennsylvania , 207-216.

Dav99 Davis, H.C., Hypertext Link Integrity. ACM Computing Surveys, 31, 4 (1999).

ICL96 Ingham, D.B., Caughey, S.J., Little M.C., Fixing the “Broken-link” Problem: The W3Objects Approach. Computing Networks & ISDN Systems, 28, 7-11 (1996), 1255-1268: Proceedings of the Fifth International World Wide Web Conference, Paris, France, 6-10 May 1996.

Sch93 Schnase, J.L. et al., Design and Implementation of the HB1 Hyperbase Management System. Electronic Publishing: Origination, Dissemination and Design, 6, 1 (1993), 35-63,.

Sha96 Shafer, K. et.al, J., Introduction to Persistent Uniform Resource Locators., http://purl.oclc.org/OCLC/PURL/INET96 [10.4.2002]

WiL96 Wiil, U.K., Legget, J.L., The HyperDisco Approach to Open Hypermedia Systems. Proceedings of Hypertext '96, Washington, DC, March 16-20, 1996, 140-148.