Hyperteksti ja versiointi

Mikko Koskenniemi

Hypermediajärjestelmät

3.5.2002

Helsingin yliopisto

Tietojenkäsittelytieteen laitos

Sisällys

Johdanto 1
Versionhallinan käsitteistöä 2
Tila- ja tehtäväperustainenversion hallinta.....4
Hypermedian versiointi.....5
Tilanteita joissa linkkien hallinnasta on hyötyä 6
Sisältöviittauslinkit 7
Dokumenttien periminen.....8
VTML 9
Yhteenveto.....10
Viitteet.....12

Johdanto

Versiohalintaa tarvitaan ohjaamaan ja hallitsemaan jatkuvasti muuttuvista osista muovautuvaa kokonaisuutta[VF99] [PA95]. Tällainen kokonaisuus on esimerkiksi hypermediajärjestelmä sen rakennus- ja ylläpitovaiheessa. Tässä työssä käsitellään hypermediajärjestelmien tarvetta versioinnille.

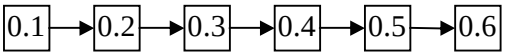
Aluksi esitellään versionhallinan käsitteistöä ja näytetään muutamia esimerkkejä tilanteista, joissa versionhallinnasta on hyötyä. Sitten esitellään versiointistrategioita, jotka soveltuvat hypermedian versiointiin sekä ongelmallisia tilanteita, joihin joudutaan jos hypermediaa kehitetään ympäristössä, joka ei tue versionhallintaa. Esimerkiksi linkkieheyden säilyttäminen dokumentteja muokattaessa.

Versionhallinan käsitteistöä

Ohjelmistokehitykselle luonteenomaista on jatkuva muutos. Muutoksen kohteena on yleensä iso kokonaisuus, joka koostuu osista joita useat ihmiset muokkaavat. Yleensä yksittäiset ihmiset tai ryhmät työstävät komponentteja, joiden tarkoitus on lopuksi toimia yhdessä muiden komponenttien kanssa. Versionhallinnan tarkoituksena on tarjota luotettava säilytyspaikka komponenteille kehityksen ajaksi. Myös lopullista versiota kokonaisuudesta säilytetään usein version hallinnassa.

Versionhallinta tuo kiistatta jonkin verran ylimääräistä työtä, mutta antaa vastineeksi hallittavuutta, ja sen tarpeellisuudesta on useita näyttöjä[VF99]. Yksi merkittävimmäksi tekijäksi digitaalisia dokumenttikokoelmia ylläpidettäessä on osoittautunut versionhallinta[PA95]. Ei pelkästään jotta voidaan tarjota eri versioita asiakkaille ja kehittäjille, vaan myös siksi että jää arvokasta historiaa kokoelman kehityksestä [PA95].

Versionhallinan tärkein tehtävä on hallita sinne talletettujen dokumenttien historia: kaikki versiot ensimmäisestä viimeiseen ovat palautettavissa. Kuvassa 1 on esitetty kuvitteellisen ohjelman kehitys versiosta 0.1 versioon 0.6. Jokaisella askelleella on ohjelmaan tehty joitain muutoksia. Mikä tahansa talletetuista versioista voidaan palauttaa.

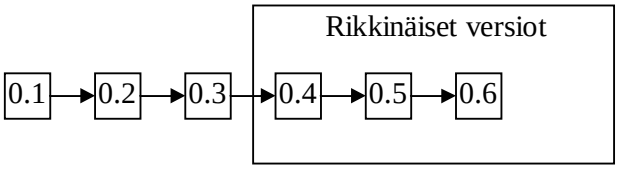


kuva 1

Versionhallinta voidaan toteuttaa esimerkiksi dokumenttien nimeämiskäytännön avulla lisäämällä nimen perään versionumero, mutta tämä ratkaisu johtaa ylitsepääsemättömiin vaikeuksiin jos hallittavana on paljon dokumentteja [PA95]. Parempi ratkaisu on käyttää tähän tarkoitukseen suunniteltuja järjestelmiä. Jos käytetään versionhallintaa, joka tallettaa ainoastaan dokumenttien väliset eroavaisuudet, säästetään huomattavat määrät levytilaa. Lisäksi nimeämiskäytäntöä ei tarvitse erikseen hallita, sillä dokumenttien nimet eivät muutu versioista toiseen siirryttäessa.

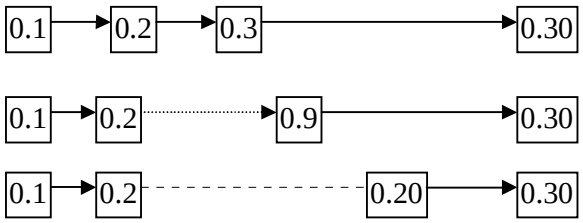
Yleensä työskentely versionhallinnan kanssa tapahtuu varaamalla jokin tietty dokumentti muokkausta varten. Tällöin muut järjestelmän kanssa työskentelevät eivät pääse muokkaamaan ko. dokumenttia, millä vältetään lomittamisessa mahdollisesti syntyvät virheet. Kun muokkaus on valmis tallennetaan dokumentti version hallintaan, joka automaattisesti selvittää eroavaisuuden vanhasta versiosta, ja tallettaa vain eroavuudet sekö antaa versionumeron.

Version hallinnan avulla voidaan selvittää esimerkiksi mitä muutoksia on tapahtunut tietystä versiosta toiseen siirryttäessä. Tätä tarvitaan esimerkiksi jos huomataan vika toiminnossa, joka on aikaisemmin toiminut. Vian korjaaminen on huomattavasti helpompaa, jos tiedetään mitä on muutettu silloin kun vika syntyi. Ilman version hallintaa on todennäköistä, ettei ehjää versiota enää löydy, tai se on niin vanha että muutoksia sen ja rikkinäisen välillä on huomattavia määriä. Vian etsintä version hallinnan avulla kannattaa aloittaa selivittämällä, missä versiossa vika esiintyi ensimäisen kerran. Kuvan 2 tapauksessa vika on havaittu versiossa 0.6 ja se on syntynyt versioon 0.4.



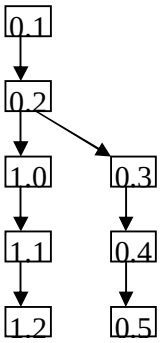
kuva 2

Yksittäisten tiedostojen versionumerot eivät välttämättä ole synkroonissa: Jos jotain tiedostoa on muokattu paljon, on sen versionumero todenäköisesti suurempi kuin sellaisen, jota on muokattu vähemmän. Jotta tiedetään, mitkä dokumentit toimivat yhdessä on mahdollista antaa tietylle versionhallinan tilalle yksiselitteinen tunniste. Esimerkiksi (kuva 3) jos järjestelmä koostuu kolmesta osasta A,B,C. A:tä ei ole muokattu juurikaan, B:tä on muokattu kymmenen kertaa ja C:tä satoja. Kun järjestelmä vihdoin toimii, voidaan kaikille kolmelle antaa yhteinen tunniste. Tunnisteen avulla voidaan kyseinen tila palauttaa vuosienkin jälkeen.



kuva 3

Ohjelmasta, joka koostuu monesta komponentista voidaan version hallinnan avulla hallita myös useampia rinnakkaisia versioita. Kun ohjelma saavuttaa riittävän kypsyyden tai toimintavarmuuden, voidaan sitä varten erityttää oma versiohaara, jossa keskitytään vain virhekorjauksiin. Alkuperäisessä haarassa voidaan toteuttaa uusia toimintoja, ilman että toinen haara häiriytyy. Kuvassa 4 esimerkki versionhallinta puusta jossa on eriytetty versio.



kuva 4

Versionhallintaan talletetaan usein muutakin kuin pelkkä lähdekoodi esim. suunnitteludokumentit, testitulokset, vaatimusmäärittelyt jne. Tällöin pystytään tunnistamaan, mitkä testaus ohjeet ja vaatimusmäärittelyt kuuluvat millekkin versiolla.

Tila- ja tehtäväperustainen versionhallinta

Perinteisessä versionhalliinassa on jokaisesta versioidusta kohteesta tallessa versiohistoria. Dokumenttiin viitattaessa on mainittava yksiselitteinen versiotunniste jonka avulla järjestelmä pystyy esittämään dokumentinsta pyydetyn version. Tarvittaessa voidaan myös

nähdä mitä muutoksia ko. dokumenttiin on tehty tiettyjen versioiden välillä. Versiotunniste voi olla esimerkiksi juokseva numero tai kirjain yhdistelmä.

Tilaperustaisessa (State based) versionhallinnassa ei ole helposti mahdollista saada selville useampaan dokumentiin tehtyjä muutoksia. Jos esimerkiksi virhekorjaus on kohdistunut useaan dokumenttiin, pitää jokaisesta dokumentista pyytää muutosluettelo tarkkojen versionumeroiden avulla. Ongelmana on, etteivät versionumerot välttämättä ole kaikilla dokumenteilla samoja eikä tiedetä mihin dokumentteihin ko. korjaus kohdistui.

Haake on kuvannut tehtäväpohjaisen versionhallinan soveltuvuutta hypermedian käsittelyyn [HH96]. Tehtävä pohjaisessa versionhallinnassa järjestelmään talletetaan dokumentit tehtävä ohjatusti eli versiointi tapahtuu järjestelmään kohdistuneiden tehtävien mukaan[MSV93]. Kun jokin toimenpide on valmis, esimerkiksi virhekorjaus, viedään muuttuneet dokumentit versionhallintaan jolle ilmoitetaan että tehtävä on suoritettu. Tällöin versionhallinta tietää, mitkä dokumentit ovat olleet osallisina ko. virheen korjauksessa ja mitä niille on tehty.

Hypermedian versiointi

Hypermedian alistaminen versionhallintaan on aivan yhtä tarpeellista kuin ohjelmistokehityksessä ohjelmien lähdekoodien. Pääsy vanhoihin versioihin, pitää tarjota jatkokehityksen ja virhekorjauksen takia. [PA95]

Tarjolla olevat, yleisesti käytössä olevat, versionhallinta järjestelmät: CVS, VSS (Visual Source Safe, Microsoft) pystyvät tallettamaan lähes minkälaisia tiedostoja ja dokumentteja tahansa, esimerkiksi kuvia, ohjelmia, testi tuloksia jne.

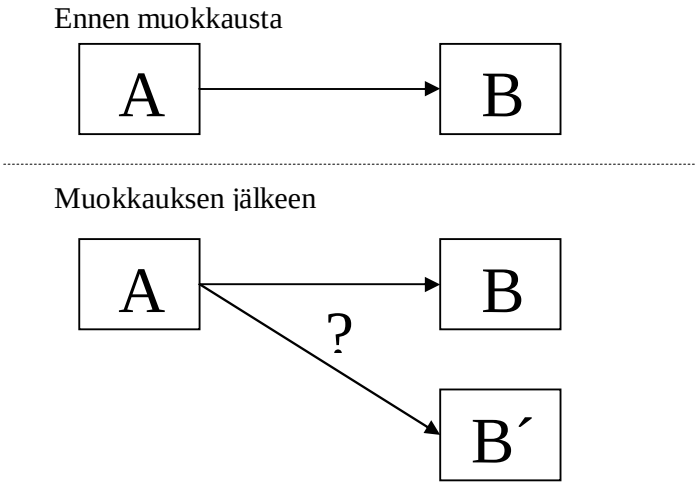
Normaalisti ohjelmistokehityksen alkuvaiheessa päätetään, mitä alistetaan version hallintaan. Tyypillisesti sinne laitetaan kaikki, joka voi vähänkin muuttua. Esim lähdekoodit, kuvat, testityökalut jne. Hypermediaa tuottavissa projekteissa talletettavana on lisäksi dokumenttien välinen linkitys. Koska linkkirakenne sijaitsee joko dokumenttien sisällä tai ulkoisissa rakenteissa, jotka pohjimmiltaan ovat tiedostoja, ei niiden tallettamisessa melkein mihinkä tahansa versionhallintajärjestelmään ole mahdotonta.

On kuitenkin tilanteita, joissa versionhallinta työkaluilla olisi mahdollisuus auttaa mm linkityksen eheyden hallinnassa. Seuraavaksi esitellään, millaisia etuja voidaan saavuttaa, jos versionhallinnan yhteydessä linkkejä pystytään käsittelemään ”älykkäästi”.

Tilanteita joissa linkkien hallinnasta on hyötyä

Kun versionhallinnan alaisuudessa olevaa dokumenttia muokataan, syntyy siitä uusi versio. Muokkauksen syynä voi olla esimerkiksi korjaus, lisäys tai muutos. Jos kyseessä on korjaus, voidaan olettaa, että tähän dokumenttiin viittaavat linkit eivät vaadi päivitystä, mutta jos kyseessä onkin muutos ei tilanne ole aivan selvä.

Esimerkiksi kuvassa 6 on esitetty dokumentit A ja B joiden välillä on linkki. Jos dokumenttia B muokataan, pitäisikö linkin osoittaa uuteen vai vanhaan versioon?



kuva 6

Kun dokumentista luodaan uusi versio, jotain tietoa esimerkiksi päivitetään, voidaan versionhallinnan avulla päivittää kaikki tähän dokumenttiin viittaavat linkit kerralla. Linkkeihin voidaan lisäksi merkitä päivitystarve: Saako päivittää? Pitääkö päivittää, vai esitetäänkö tilanne ihmiselle, joka tämän valinnan tekee, vai jätetäänko valinta loppukäyttäjälle, esimerkiksi sivustoa selattaessa?

Joissain tapauksissa linkkien päivittäminen ei edes ole mahdollista: esimerkiksi ulkopuolelta tulevien linkkien päivittäminen on mahdotonta, koska niiden löytäminen on työlästä ja niihin kirjoittamista ei yleensä ole sallittu.

Linkki eheyttä on käsitelty tarkemmin Hugh C. Davis jossa linkkien rikkoutuminen on eritelty kahteen eroavaan tilanteeseen: rikkinäiset linkit (Broken links) ja sisältöviittaus ongelmat (Content refernce problems) [HCD99]. Hypermedian versioinnin yhteydessä kyseessä on ylensä jälkimmäinen. Ratkaisuksi Hugh esittää esimerkiksi aikaleimaa, linkin yhteyteen, jota voidaan verrata dokumentin aikaleimaan kun dokumenttia ladataan. Korjaus tehdään niin että koitetaan ”arvata” minne linkin pitäisi osoittaa. Etuna on, ettei korjausta lasketa turhaan, vaan vasta kun on tarve [HCD99]. Toimiva, tällainen järjestelmä, vaatii kuitenkin alleen versionhallinan.

Sisältöviittauslinkit

Hypermediadokumenteissa on usein tarve viitata dokumenttin sisään, johonkin määriteltyyn kohtaan. HTML:ssä on tätä tarkoitusta varten olemassa dokumenttiin upotetut ohjausmerkit, joiden avulla identifioidaan tällaisten viittausten kohdepisteet. HTML:n menetelmässä on sekä vahvat että heikot puolet: Vahvana ominaisuutena on linkkieheyden säilyminen, jos kohdedokumenttia muokataan. Heikkona ominaisuutena on, että kohdedokumentti tarvitsee muokkausta jos sopivaa kohde ankkuria ei ole valmiiksi olemassa. Jos muokkaus on sallittua ei ongelmaa ole, mutta useissa tilanteissa dokumenttia pystytään ainoastaan lukemaan.

Jos linkkirakenne talletetaan varsinaisten dokumenttien ulkopuolella, kuten XML:ssä, ei kirjoitusoikeutta varsinaiseen kohdedokumenttin tarvita, mutta linkkirakenteeseen tarvitaan. Tämällöin huononapuolena on viite-eheyden rikkoutuminen jos kohde dokumenttia muokataan.

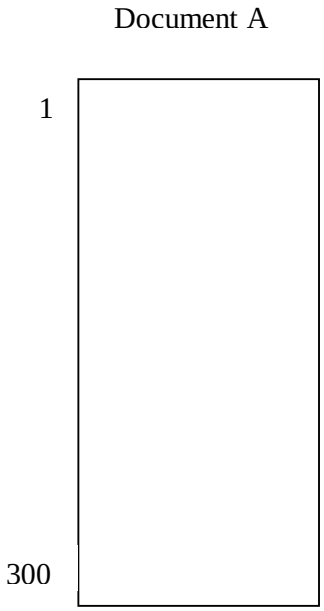
Version hallinnalla voidaan kuitenkin saavuttaa molempien hyvät puolet. Jos linkkirakenne säilytetään erillisenä, mutta alistetaan versionhallintaan joka osaa hallita hypermedia linkkejä, säilyy viite-eheys vaikka kohde dokumenttia muokattaisiinkin [MSV93].

Dokumenttien periminen

Dokumenttien periminen, joka voidaan mieltää versioinnin yhdeksi muodoksi on esitelty jo 60-luvulla [NTH99]. Xanalogical stucrure, syntyi Xanadau nimisen projektin tuloksena. Projektin tarkoituksena oli luoda järjestelmä jossa toisiinsa runsaasti viittaavien dokumenttien lukeminen ja tuottaminen olisi helppoa. Projektin tuloksena syntyi mm. Dokumenttien kuvaus kieli, jossa dokumentit rakentuvat perimällä aikaisempien dokumenttien sisältöä. Ainoastaan uusi, dokumentin varsinainen lisäarvo tarvitsi kirjoittaa uudelleen.

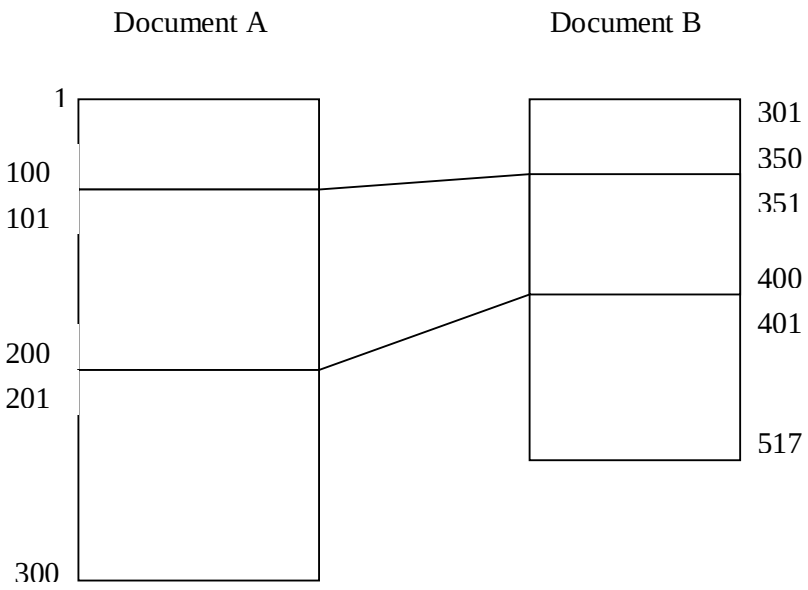
Seuraava esimerkki kuvaa ideaa:

Henkilö 1 kirjoittaa dokumentin A, jossa on 300 merkkiä. Merkeille varataan peräkkäiset, yksiselitteiset osoitteet 1-300 (Kuva 7).



kuva 7

Henkilö 2 luo dokumentin B, jossa hän kommentoi dokumenttia A väliltä 102 – 200. Itse hän tuottaa tekstiä ensin 50 merkkiä jonkä jälkeen tulee dokumentin A lainaus ja lopuksi vielä 116 merkkiä. Dokumentti B kuvassa 8.



kuva 8

Koska Xanalogical strukuuri ei tue versiointia, ei alkuperäisiä dokumentteja voi editoida ilman että uudet dokumentit muokkautuvat.

Versionhallinta ja ulkoinen linkkirakenne luovat hyvät puitteet myös dokumenttien perimismekanismille, jossa alkuperäisiä dokumentteja voidaan muokata luomalla niistä ensin uusiversio. Kun muokattaessa luodaan uusiversio, ei dokumentin perineille dokumenteille tarvitse tehdä päivityksiä.

Toisinaan WWW:n hakuautomaateillakin päätyy sivulle, jolta ei enää löydy annettua hakusanaa. Tällöin olisi vanhojen versioiden saatavuudesta hyötyä. [VD95] on esittänyt kuvaus kielen tälle VTML.

VTML

VTML (Versioned Text Markup Langueage) on kehitetty mahdollistamaan WWW:n avulla tapahtuvaa yhteistoimintaa. Se on tarkoitus saavuttaa antamalla globaali kirjoitusoikeus julkaistuihin dokumentteihin. Eli kuka tahansa pystyy tekemään lisäyksiä ja muutoksia olemassa oleviin hypertekstidokumentteihin, ilman että alkuperäinen rikoutuu. Tämä on mahdollista, sillä VTML:n ja sen alapuolella toimivan versiointijärjestelmän avulla voidaan tarjota dokumenteista useita versioita, jolloin alkuperäisiä dokumenttejä voidaan

tarkistaa ja korjata, niin ettei korjaaja tarvitse kirjoitusoikeutta alkuperäiseen vaan muutokset talletetaan tätä varten luotuun uuteen versioon.

Käytännössä sivun muokkaaminen tapahtuu paikallisesti, lataamalla alkuperäinen sivu omalle työasemalle ja muokkaamalla sitä halutulla tavalla. Kun uusi versio on valmis se lähetetään palvelimelle, joka tallettaa sen erilleen alkuperäisestä dokumentista. Uutta varten luodaan uusi versionumero ja ainoastaan eroavaisuudet säilytetään. Jatkossa tämä sivu on luettavissa verkon kautta, antamalla viittaus dokumentin nimen lisäksi oikeaan versioon.

Esitetty menettely ei vaadi jatkuvaa yhteyttä työasemien ja sivuja hallitsevan palvelimen välillä. Dokumenttia voidaan muokata samanaikaisesti eri tahoilla, jolloin saattaa syntyä lomitus tarvetta.

Uuden dokumentin luominen perimällä jo olemassa oleva, voidaan tehdä tietyn version pohjalta. Tällöin alkuperäiseen tehtävät päivitykset eivät vaikuta ”jatko” versioihin.

Sisältölinkkien, dokumentin tietyyn kohtaan viittavien linkkien, lisääminen onnistuu VTML:n avulla myös HTML dokumentteihin. HTML:ssä sisäviitteet on talletettu dokumentin sisälle, koska dokumentteihin on harvoin kijroitusoikeutta ei tällaisi linkkejä voidan ilman omistajan apua lisätä. VTML:n avulla voidaan ko. dokumentista luoda sellainen versio, jossa haluttu linkin kohde on olemassa.

Yhteenveto

Hypermedian tuotantojärjestelmissä on paljon samoja ongelmia kuin muissakin luovaa työtä sisältävissä ympäristöissä[VF99]. Suurin osa versioinnin tarpeesta on tuotantovaiheessa ja tulee projektien hallinnasta ja toteutuskurista. Hypermedia eroaa kuitenkin muista ohjelmistoprojekteista tuotettavien osien välisten suhteiden hallinnan takia. Ongelmat syntyvät yleensä kun muokkauksen kohteena on tai pitäisi olla useampia dokumentteja. Jos versionhallinnan avulla saadaan selville, mitkä dokumentit olivat ko. muokkauksessa mukana ja mitä niille tehtiin on virheiden korjaaminen helpompaa [HH96].

Jos versionhallinta tukee hypermediassa tarvittavia linkkejä, voi se huolehtia useimmista sellaisista tilanteista automaattisesti, joissa muuten tarvittaisiin käsityötä.

Hypermedia hyötyy myös versioinnista julkaisun yhteydessä. Yleensä versionhallinta on vain tuotantovaiheen työkalu, mutta hypermedian julkaiseminen sellaisessa ympäristössä, joka tarjoaa versionhallintaa, ratkaisee joitakin ongelmia. Esimerkiksi HTML:n sisältöviittaukset, dokumenttien periminen ja kommentointi.

Viitteet

HH96 Haake, A., Hicks, D., VerSE: Towards Hypertext Versioning Styles. ACM Hypertext ’96, Washington DC, 224-234, 1996

VF99 Vitali, F., Versioning hypermedia. ACM computing surveys, Vol. 31 December 1999

PA95 Pettengil, R., Arango, P., Four lessons learned from managing World Wide Web digital libraries. http://www.csdl.tamu.edu/DL95/papers/pettengill/pettengill.html

HCD99 Hugh, C., D., Hypertext Link Integrity. ACM computing Surveys 31(4), Devember 1999.

VD95 Vitali, F., Durand D. G., Using Versioing to provide Collaboration on the WWW. World Wide Web Journal 1, O’Reilly, 37-50, 1995

NTH99 Nelson, T. H., Xanalogical Media: Needed Now More Than Ever, ACM Computing Surveys Symposium on Hypertext and Hypermedia, 1999

MSV93 Maioli, C., Sola, S., Vitali, F., Versioning issues in a Collaborative Distributed Hypertext System, Technical report UBLCS-93-6, Bologna, 1993