

Parhaat WebFOCUS-käytännöt

Sisällysluettelo

Parhaat WebFOCUS-käytännöt	1
Yleistä	2
Sovelluskehitys	2
Yleisosien käyttö	2
Kommentointi	3
Koodin luettavuus.....	3
Nimeämiskäytännöt.....	4
Kansiointi	4
Aputaulut ja datamartit	4
Suorituskyky, testaus ja virheiden etsintä.....	5
Versionhallinta.....	6
Tietoturva	7
Syötteiden tarkistus.....	8
Virheilmoitukset.....	8
Cookie-käsittely	9
HTML- koodin käsittely	9
Suorien SQL-komentojen esto	10
Monitasoarkkitehtuuri.....	10
DBA-ominaisuuksien käyttö	11

Yleistä

Tähän dokumenttiin on kerätty WebFOCUS-sovelluskehityksessä hyödynnettäviä parhaita käytäntöjä. Kaikki dokumentissa esitetyt käytännöt eivät ole oleellisia jokaisessa tilanteessa, mutta oheisista kohdista voi saada myös silloin ideoita esim. jatkokehitystä varten.

Sovelluskehitys

Yleisosien käyttö

Miksi

WebFOCUS-ohjelmoinnissa voidaan hyödyntää tehtyjä ohjelmia muiden ohjelmien sisällä. Tämän ominaisuuden ansiosta voidaan luoda nk. yleisosia, eli ohjelmia, jotka voivat olla usealle raporttiohjelmalle yhteisiä. Yleisosien ansiosta asioita tarvitsee tehdä vain yhden kerran, jonka jälkeen raporttien kehittäminen on nopeampaa ja yleisosan hoitama käsittely yhdenmukaista kaikissa ohjelmissa. Myös ylläpidettävyys ja useampaan paikkaan vaikuttavien muutosten tekeminen helpottuu.

Miten

Yleisosia voidaan kutsua ohjelmissa -INCLUDE tai -EX-komennoilla. Tyypillisesti yleisosissa määritellään esim. seuraavia asioita:

Yleiset määrittelyt

- Ympäristömääritykset (hakemistot, osoitteet, tiedostojen nimet, käyttöoikeustarkistukset)
- Yleiset muuttujat (yrityksen nimi, pvm, kelloaika, kausi)
- Oletusarvot (tulostusmuoto, fontit, CSS)
- Testauksen ja debuggauksen hallinta (ECHO, TRACE)

Lomakesivujen valintalistat

- Valintalistan sisällön luonti ja valittujen merkintä
- Valintalistan html-osuuden luonti
- Valintalistan arvojen purkaminen where-lauseeksi ja valintojen tallennus tulostusta varten

Ulkoasumäärittelyt

- Tulostusasetusten määrittely (format, page-num, emptyreport, bydisplay)
- Käyttäjän valintojen tulostus raportille
- Tyyli tietojen määrittelyt (omat määrittelyt eri tulostustavoille)

Kommentointi

Miksi

Kun koodi on kommentoitu kattavasti, on helpompi hahmottaa mitä ohjelma tekee, millaisia parametrejä sen kanssa voidaan käyttää ja millainen sen sisäinen logiikka on.

Miten

WebFOCUS-ohjelmien alkuun on hyvä sijoittaa kommenttiblokki, jossa on kuvaus ohjelman käyttötarkoituksesta ja sisään tulevista muuttujista. Sen lisäksi aloitusblokissa on hyvä dokumentoida ohjelman muutoshistoria; mitä muutoksia on tehty, milloin, ja kuka on ollut muokkaaja.

Ohjelmassa on myös hyvä sijoittaa eri osioiden tai vaiheiden eteen kommenttiosio, esim. yksittäinen seliterivi, jossa kuvataan seuraava vaihe.

Jos tehdään sovelluskehitystä käyttäen tekstieditoria, ei kommentteja tulisi sijoittaa koodin vaiheiden, esim. TABLE-vaiheen keskelle, sillä tämä vaikeuttaa koodin avaamista jatkossa graafisessa editorissa.

Poista vanhat koodikokeilut jonkin ajan kuluttua pois. Jos jotain koodia pitää kommentoida pois, laita kommentteihin myös päivämäärä ja nimi, sekä selitys miksi koodipätkä on poistettu ja miksi se pitää jättää kommentteihin.

Muista päivittää kommentteja samalla kun päivität koodia.

WebFOCUS-koodissa kommenttinotaatio on -* rivin alussa, HTML-koodissa <!-- /--> -blokki ja javascript-koodissa /* */ -blokki tai // rivin alussa.

Koodin luettavuus

Miksi

Kun koodi on selkeän näköistä, sisennetty ja eri osiot selkeästi erotettu toisistaan, on kehittäjien, jotka eivät ole alun perin ohjelmaa tehneet, helpompi hahmottaa koodin osien välisiä suhteita ja käytettyä logiikkaa.

Miten

Sisennyksiä kannattaa käyttää mm. tasaamaan kenttiä ja arvoja, esim. muuttujien määrittämisosioissa.

```
-SET &YOUTPUT = 'HTML';  
-SET &YWIDTH = 500;  
-SET &YHEIGHT = 300;  
-SET &YOTSIKKO = 'Raportin otsikko';  
jne.
```

Eri osiot kannattaa erottaa toisistaan ainakin tyhjällä rivillä. Osioiden sisäisiä asioita kannattaa sisentää sovelluslogiikan tasojen mukaan.

Nimeämiskäytännöt

Miksi

Kun ohjelmat, tietolähteet, väliaikaistiedostot ja koodissa esiintyvät muuttujat ovat selkeästi nimettyjä, on kehittäjien helpompi hahmottaa koodin logiikkaa ja löytää oikeat ohjelmat.

Miten

Nimeämiskäytäntöjä kannattaa käyttää säännönmukaisesti ja ainakin seuraavissa osioissa:

- Ohjelmat esim. ohjelman sisällön mukaisesti: sales_by_region, kauttajatilasto, jne., tai jonkin tunnisteiden mukaisesti: WIT002, WIT003 jne.
- Yleisohjelmat esim. y-alkuisesti: y_style_set, y_output, jne
- Muuttujat esim. muuttujan sisällön mukaisesti: sales, latitude, jne.
- Yleismuuttujat, väliaikaiset muuttujat, tietolähteestä luettavat muuttujat esim. alkutunnisteella: youtput, xwidth, tsales jne.
- HOLD-tiedostot esim. alkutunnisteella: HLD_SALES, HLD_USERS, jne.

Nimeämiskäytännöt voivat olla aivan erilaisiakin kuin yllämainitut esimerkit, tärkeintä on se, että sovitaan käytännöistä ja noudatetaan niitä johdonmukaisesti.

Kansiointi

Miksi

Looginen kansiointi helpottaa oikeiden ohjelmien ja kuvausten löytämistä.

Miten

Esimerkiksi yleisöille ja synonyymeille kannattaa tehdä omat kansionsa. Kansiot kannattaa laittaa myös profiiliin APP PATHiin, jolloin ohjelmat löytyvät ne automaattisesti.

Aputaulut ja datamartit

Miksi

Hyödynnä aputauluja esim. valintalistojen sisällön hakemisen nopeuttamiseen. Jos valintalistojen sisältöjä haetaan aina operatiivisesta faktataulusta, voi suorituskyvyn kanssa nousta ongelmia datamäärän kasvaessa. Harkitse luetaanko mikä osa raportoinnin tarvitsemasta tiedosta suoraan operatiivisista tietolähteistä vaiko aputauluista, raportointikannoista / datamarteista tai tietovarastosta.

Datamarteista ja tietovarastosta on suoraan operatiivisista tietolähteistä lukemiseen verrattuna seuraavia hyötyjä:

- tietojen manuaalinen ja raporteilla tapahtuva yhdistely vähenee,
- raporttien logiikkaa voidaan siirtää osin tietovarastokerrokseen,
- tiedot voidaan puhdistaa epäloogisuuksista ja muokata liiketoimintasääntöjen mukaiseksi,
- tietolähde voidaan optimoida raportoinnin näkökulmasta sekä
- eri tyyppisistä raportointikannoista tehtäviin koosteraportteihin saadaan tehokkuutta lukemalla kaikki tieto jo kertaalleen prosessoidusta tietovarastosta.

Miten

WebFOCUSin ETL (Extract, Transform and Load) -komponentti DataMigratoria voidaan käyttää aputaulujen, datamarttien ja tietovarastojen rakentamiseen ja päivittämiseen. Jos ETL-komponenttiin on liitetty nk. Change Data Capture (CDC)-ominaisuus, päivittäminen voi olla lähes reaaliaikaista. Lokipohjaisen CDC-ominaisuuden avulla voidaan havaita lähdejärjestelmissä (esim. DB2, Microsoft SQL Server tai Oracle) tapahtuvat tietojen muutokset reaaliaikaisesti ja toimittaa muuttuneet tiedot ETL-prosessin kautta raportointikantaan tai tietovarastoon. Jos lisäksi on käytössä tiedon laadun komponentteja, voidaan tietovaraston sisältämän tiedon laatua seurata ja valvoa reaaliaikaisesti.

Suorituskyky, testaus ja virheiden etsintä

Miksi

Ohjelmien suorituskyky koostuu useista eri osa-alueista. Näitä kaikkia kannattaa testata, arvioida ja tutkia, jotta saadaan esim. käyttäjien kokemat vasteajat mahdollisimman pieniksi.

Miten

Ohjelmien suorituskykyä voidaan tutkia useilla eri tavoilla ja eri osa-alueilta:

- Tietolähteen suorituskyky
 - tarkista WebFOCUS Reporting Serverin konsolilta, miten kauan ohjelmakoodi on tietolähteessä suoritettavana
 - tarkista, onko tietokanta optimoitu oikein kyselyitä varten
 - tarkista, onko tietolähteissä oikeat indeksit raportointia varten
 - tarkista, onko WebFOCUS-adapterin luoma SQL-koodi optimaalista. Ota SQL Trace ja aja SQL-kysely suoraan tietolähteessä. Kestääkö saman ajan kuin WebFOCUSin kanssa?
 - tarkista, ovatko Joinit ja lajittelut optimaalisia (esim. pienempi lähde joinataan isompaan)
 - arvioi, auttaisivatko erilliset aputaulut, datamartit tai tietovarastointi
 - arvioi, auttaisivatko FOCCACHE-välimuistin käyttö
 - arvioi, auttaisiko tietokanta-alustan skaalaus, ja onko se ylipäättään mahdollista esim. taloudellisesti.
- Suorituskyky tietolähteestä WebFOCUSiin
 - tarkista WebFOCUS Reporting Serverin konsolilta, miten paljon IO:ta käytetään tietojen siirtämiseen
 - tarkista Traceista siirtääkö tietolähde kaiken aineiston vai vain tuloksen WebFOCUSille (onko aggregointi tietolähteessä kunnossa)
- WebFOCUS-alustan suorituskyky
 - arvioi, auttaisiko WebFOCUS-alustan skaalaus, ja onko se ylipäättään mahdollista esim. taloudellisesti.
 - lisää vinkkejä esityksissä:
 - <http://www.informationbuilders.com/php/summit/presentations.php?id=19784905943548797362967804739491>
 - <http://www.informationbuilders.com/php/summit/presentations.php?id=66736364206939593102967808081471>

- WebFOCUS-sovellusten suorituskyky
 - tarkista, onko koodi optimaalista, haetaanko turhaa tietoa lähteistä, tehdäänkö asiat järkevässä järjestyksessä ja mahdollisimman yksinkertaisesti.
 - tarkista WebFOCUS-traceistä tapahtuuko suorituksessa jotain poikkeavaa, esim. autentikoinnin ja autorisoinnin suhteen
 - tarkista WebFOCUSin ECHO-, TYPE ja ?FILE -komentojen avulla mitä tapahtuu tapahtuuko suorituksessa jotain poikkeavaa, esim. välitiedostojen luonnin tai muuttujien suhteen
 - lisää vinkkejä esityksessä:
 - <http://www.informationbuilders.com/php/summit/presentations.php?id=-2038931119332827167297089796950>
- Suorituskyky WebFOCUSilta selaimelle
 - tarkista verkkoliikenteen koko, esim. siirrettävien pakettien koko selainlaajennusten avulla (esim. Firebug / Mozilla tai F12 Developer tools / IE)
 - tarkista muut yleiset nettisivujen suorituskykyohjeet esim. <https://developer.yahoo.com/yslow/> -sivustolta
 - aseta gzip-pakkaus päälle web-palvelimelta. Se voi pienentää siirrettäviä tiedostoja jopa 60-70%.
- Selaimen suorituskyky
 - tarkista javascript-suorituskyky selainlaajennusten avulla (esim. Firebug / Mozilla tai F12 Developer tools / IE)
 - tarkista muut yleiset nettisivujen suorituskykyohjeet esim. <https://developer.yahoo.com/yslow/> -sivustolta
 - arvioi, auttaisiko asteittainen (ja asynkrooninen) lataaminen, eli ladataan sivun elementtejä tarpeen mukaan, ladaten vain osia, jotka muuttuvat käyttäjien tehdessä valintoja. Eräät javascript-kirjastot, kuten jQuery ovat hyvänä apuna tässä.

Versionhallinta

Miksi

Versionhallinta on tarpeen, kun ohjelmiin ja dokumentteihin tehdään muutoksia tai tarkistetaan onko tarpeelliset muutokset jo tehty. Vähimmillään täytyy tietää kuka ohjelman on luonut, jotta tiedetään keneltä kysyä muutoksista. Käytännössä pitää jo muutoksia tehdessä varautua siihen että kenen tahansa samassa asemassa olevan henkilön täytyy kohtuullisessa ajassa voida todeta onko tarpeelliset muutokset tehty ja tarvittaessa peruuttaa johonkin aikaisempaan tilanteeseen tai jatkaa muutosten tekemistä.

Versionhallinnan avulla jokainen ohjelmiin tehty muutos tulee hallituksi ja jäljitettäväksi. Versionhallintajärjestelmien avulla voidaan seurata esim. kuka teki millaisia muutoksia sekä milloin ja miksi ne tehtiin.

Versionhallintajärjestelmien hyödyntäminen auttaa myös tiimityöskentelyn hallinnassa, jolloin esim. samaan ohjelmaan eri tekijöiden toimesta tehtävät muutokset voidaan yhdistää helposti lopulliseksi versioksi.

Miten

Jos käytössä ei ole versionhallintajärjestelmää, kannattaa vähintään tehdä seuraavat tiedostokohtaiset toimenpiteet.

Pidä tuotantoympäristö aina erillään kehitysympäristöstä vähintään kansiotasolla. Tuotantoversiosta täytyy aina olla varmuuskopio sellaisella tasolla että ei synny kohtuutonta työtä vaikka kaikki tuotantoympäristön ohjelmat häviäisivät.

Pidä työversioista päiväkohtaiset varmuuskopiot alikansioissa. Näin pääset palaamaan edellisiin kehitysversioihin, jos muutokset lähtivät etenemään väärään suuntaan tai joku muu tallensi oman työversionsa tekemiesi muutosten päälle.

Kirjaa kaikki muutokset suoraan muokattavaan tiedostoon esim. proseduurin alkuun kommentteihin tai dokumentin viimeiselle erilliselle sivulle. Joskus muutokset kannattaa kirjata myös erilliseen muutosdokumenttiin. Vähintään kirjoitetaan päiväys, nimikirjaimet ja muutoksen kuvaus esim.

-* 20110924 IBU/XYZ Lisätty uusi yleisosa Y_TARKISTA_TUNNUS.FEX

Lisäksi kannattaa kirjoittaa esim. muutetun koodirivin edellinen ja uusi muoto ja mitä lisämuutoksia täytyy tehdä.

Versionhallintajärjestelmällä nämä toimenpiteet hoituvat keskitetysti ja osittain automaattisesti. Järjestelmän tietovarasto (Repository) toimii varmuuskopiona ja siinä säilytetään kaikki muutokset. Tyypillisesti ohjelmatiedostoja ei talleteta sellaisenaan vaan ainoastaan muutokset säilytetään ja koko Repository pakataan, joten levytilaa säästyy tiedostojen kopiointiin verrattuna. Järjestelmä osaa automaattisesti pakata, purkaa ja rekonstruoida minkä tahansa version mistä tahansa tiedostosta.

Tiedostojen työversiot viedään ja haetaan Repository:stä Commit- ja Update-toiminnoilla. Jokaisen Commit:n yhteydessä tallentuu tekijän nimi ja muutoskommentit. Myöhemmin näistä voidaan tarkasti nähdä kaikki muutokset. Järjestelmä ohjaa hyvään sovelluskehitystapaan ja helpottaa tiimityöskentelyä.

WebFOCUS integroituu yleisimpiin versionhallintajärjestelmiin, esim. Concurrent Versions System (CVS), Subversion, Microsoft Visual SourceSafe, Rational ClearCase, PVCS ja muut SCC API standardia noudattavat järjestelmät.

Ohje uusimman WebFOCUS versio 7.7.03:n integroimisesta eri versionhallintajärjestelmiin löytyy osoitteesta

http://documentation.informationbuilders.com/masterindex/html/html_wf_7703/wf77crgt/source/source_control.htm

Versionhallinnan voi toteuttaa myös ilman DevStudio integraatiota. Esim. suosittuun Subversion:iin on saatavilla ilmaisia client-ohjelmia. Windows-ympäristössä yksi parhaista on TortoiseSVN (<http://tortoisesvn.net/>).

Tietoturva

Erityisesti ulospäin suunnatuissa palveluissa, mutta myös sisäisissä palveluissa tietoturvasta huolehtiminen on tärkeää. Tietojen väärinkäyttö voi onnistuessaan aiheuttaa vakavia ongelmia, joten erilaiset mahdollisuudet päästä väärin tietoihin tulisi tukkia jo etukäteen.

Ohessa muutamia dokumentteja ja linkkejä, josta löytyy tarkempia tietoja ja muitakin tietoturvaan liittyviä asioita:

http://documentation.infobuilders.com/masterindex/html/pdf_wf_bp/bp_SecurityMethod.pdf

http://documentation.infobuilders.com/masterindex/html/html_wf_77/wf77sec/index.htm?url=opener.htm

http://documentation.infobuilders.com/masterindex/html/html_wf_77/wf77ddlang/index.htm?url=opener.htm

Syötteiden tarkistus

Miksi

Vaikka käyttöliittymässä olisi vakioitu syötteiden muodot, voi osaava käyttäjä esim. selaimen lisäosien avulla yrittää muuttaa lähetettäviä lomakkeita, ja tehdä esim. muutoksia, joilla yritetään katkaista normaali koodin suoritus ja suorittaa asiaan kuulumattomia komentoja. WebFOCUS-ohjelmien hyvällä suunnittelulla erikoismerkittävät syötteet saadaan estettyä.

Miten

Hyväksy vain erikoismerkittömät syötteet lomakkeilta:

- sovelluskohtaisesti,
- sovelluskohtaisesti mutta yleisosan avulla tai
- keskitetysti WebFOCUS Clientin asetuksissa (site.wfs): <SET> MUUTTUJANNIMI(alpha)

Miinuksena sovelluskohtaisessa vaihtoehdossa on se, että kaikkiin ohjelmiin pitää tehdä alkuun kaikille parametreille tarkistukset tai yleisosakutsu.

Miinuksena keskitetyssä vaihtoehdossa on se, että kun tehdään muutoksia tai lisätään muuttujia, täytyy ne muistaa ylläpitää asetuksiin tai erilliseen tiedostoon. Käsittely voidaan hajauttaa myös useisiin erillisiin tiedostoihin, joita asetuksissa kutsutaan <if>-haaroissa <INCLUDE>-komentoilla.

Virheilmoitukset

Miksi

Käyttäjälle palautettavista html- sivuista (normaalit tai virhesivut) voi saada tietoja WebFOCUS-ympäristöstä ja käytetystä koodista. Tuotantoympäristössä tämä tulisi estää.

Miten

Laadi selväkieliset virheilmoitukset kaikkia ”oikeita” (oikean käyttäjän käyttötapausten mukaisia) käyttötilanteita varten.

Poista kaikista sivuista (lähdekoodin kommentteista) ja virheilmoitussivuista WebFOCUS koodit ja mahdolliset virhekoodit:

- Aseta kaikki ohjelmakoodista löytyvät ECHO:t pois päältä. Tämä lisäksi WebFOCUS versiosta 7.7.01:sta lähtien on mahdollista asettaa yleisasetukseksi SET DEFECHEO=NONE Tämä asetus poistaa ECHOT koko ympäristöstä.
- Aseta FOCUS-virheilmoitukset pois päältä asetuksella SET EMGSRV=OFF.

Cookie-käsittely

Miksi

Selainsovelluksissa käytettävät evästeet (cookieet) voivat olla tietoturvariski, mikäli niitä ei käsitellä ja suojata oikein.

Miten

Tarkista cookie-käsittelyn tietoturva-asetuksia Clientin konsolilta:

- Salaa kaikki WebFOCUS-evästeet ja lisää salauksen vahvuutta. Oletuksena vain WF_COOKIE ja MR_COOKIE -evästeet on salattu. Aseta WF_ENCRYPT_USER = YES ja REDIRECT_COOKIE = ON, jotta kaikki evästeet salataan. Aseta WFENCR :n arvoksi jokin korkeimmista salaustasoista..
- REDIRECT_COOKIE = ON estää selainpyynnön välittämisen toiselta koneelta toiselle. Salattu tunniste valvoo, että alkuperäisen kyselyn lähettäjä on sama kuin pyynnön uudelleen lähettänyt. WebFOCUS versiosta 7.6.10 lähtien oletusarvona on REDIRECT_COOKIE = ON.
- IBI_COOKIE_SECURE=Y estää evästeiden käytön muissa kuin HTTPS-sessioissa. Oletusarvona valinta ei ole päällä.
- Aseta ibi_apps.xml-tiedostossa Cookieen HTTPOnly-lippu, jolloin vain palvelin saa käsitellä evästä, ei esim. mikään client-ohjelmisto, esim. selain:
<Context path="/ibi_apps" reloadable="true"
docBase="C:\ibi\webfocus77\webapps\webfocus" useHttpOnly="true" />
- Aseta WF_COOKIE_EXPIRATION ja <session-timeout> web.xml:ssä samaksi, jotta sessio ja evästeet vanhenevat samanaikaisesti.
- WebFOCUS 7.7.02:stä lähtien on mahdollista asettaa yleisasetukseksi webconfig.xml-tiedostossa: session_fidelity_check = YES. WebFOCUS Client tekee ylimääräisen vahvistuksen siitä, että MR_COOKIE ja WF_COOKIE liittyvät oikeaan sessio-ID:hen. Tällä estetään evästeiden kaappausyritykset.

HTML- koodin käsittely

Miksi

Tietolähteeseen tai syötteisiin tahallisesti syötetty HTML-koodi saattaa aiheuttaa tietoturvariskin tai erilaisia häiriöitä ohjelmiin. Ylimääräisten HTML-koodien avulla tehdyn häirinnän estäminen on mahdollista WebFOCUSin asetuksia säätämällä.

Miten

Aseta palvelimen profiiliin HTML-tagien renderöinti pois päältä. Tarvittaessa salli ohjelmakoodissa HTML-tagien renderöinti vain tietyissä, suunnitelluissa kohdissa.

- SET HTMLENCODE=ON. Asetus estää datasta tai DEFINE ja COMPUTE-lauseissa muodostuvien HTML-tägien muodostamisen selaimessa.

Suorien SQL-komentojen esto

Miksi

Tietolähteeseen tai syötteisiin tahallisesti syötetty SQL-koodi saattaa aiheuttaa tietoturvariskin tai erilaisia häiriöitä ohjelmiin. Ylimääräisten SQL-koodien avulla tehdyn häirinnän estäminen on mahdollista WebFOCUSin asetuksia säätämällä.

Miten

Aseta palvelimen profiliin SQL-lauseiden välitys pois päältä. Tarvittaessa salli ohjelmakoodissa SQL-lauseiden käyttö vain tietyissä, suunnitelluissa kohdissa.

- Asetus SET DPT=OFF estää käyttäjiä antamasta nk. "Direct SQL Passthru"-pyyntöjä, jotka ohittavat normaalisti WebFOCUS DBA -rajotukset. Jos relaatiotietokannan suojaukset eivät ole riittävällä tasolla, aseta tämä asetus OFF-tilaan.

Monitasoarkkitehtuuri

Miksi

Jos WebFOCUS Client ja Reporting Server sijaitsevat omilla palvelimillaan, saadaan mm. seuraavia etuja:

- TCP/IP-liikenne voidaan kryptata Clientin ja Reporting Serverin välillä, jotta komponenttien välisiä sanomia ei voi tulkita mikäli ne kaapataan
- Reporting Servereille voidaan asentaa palomuurit siten, että vain vastaava Client pääsee läpi, eli testiympäristön Clientilla ei voi ottaa esim. tuotannon Reporting Serveriin yhteyttä. Tällöin estetään esim. pääsy sensitiiviseen tuotantodataan.
- Saavutetaan lisää suorituskykyä kun yhteensä käytettävissä olevien ytimien määrä kasvaa.
- Jos Web-palvelin on erillinen, ovat Reporting Server ja sen kanssa samalla palvelimella olevat osat, kuten tietokanta-clientit, paremmin suojattuja.

Miten

Asenna WebFOCUS Client ja Reporting Server omille palvelimilleen.

Salaa TCP/IP-liikenne Clientin ja Reporting Serverin välillä. Erilaisten salausalgoritmien käyttö on ohjeistettu WebFOCUS Security and Administrator-ohjeissa:

<http://ecl.informationbuilders.com/wf/index.jsp?topic=/client/SecurityAdmin/source/opener.htm>

Asenna Reporting Serverien palvelimilla palomuurit siten, että vain vastaava Client pääsee läpi. Tämä tehdään käytössä olevan palvelinalustan käytäntöjen mukaisesti.

DBA-ominaisuuksien käyttö

Miksi

WebFOCUS DBA-ominaisuuksia käyttämällä voidaan rajoittaa pääsyä tietolähteisiin ja ohjelmakoodiin, esim. voidaan estää tuotannossa kehittäjienkin pääsy tiettyihin tietoihin tai muuttamaan synonyymejä tai fexejä). Ohessa kuvaus DBA-ominaisuuksista:

"The DBA facility provides a number of security options:

- *You can limit the user who have access to a given data source using the USER attribute*
- *You can restrict a user access rights to read, write, or update only using the ACCESS attribute*
- *You can restrict a user access to certain fields or segments using the RESTRICT attribute*
- *You can ensure that only records that pass a validation test are retrieved using the RESTRICT attribute*
- *You can limit the values a user can read or write to the data source or you can limit which values a user can alter using the RESTRICT attribute*
- *You can control the source of access restrictions in a multi-file structure using the SET DBASOURCE command*
- *You can point to passwords and restrictions stored in another Master File with the DBAFILE attribute*
- *You can use the WebFOCUS DBA exit routine to allow an external security system to set the WebFOCUS password.*
- *You can place security on FOCEXECs"*

Miten

WebFOCUS DBA-ominaisuuksien käyttö on ohjeistettu WebFOCUS Security and Administrator-ohjeissa:

<http://ecl.informationbuilders.com/wf/index.jsp?topic=/client/SecurityAdmin/source/opener.htm>

Kryptaa Master-kuvaukset lisäämällä ENCRYPT-attribuutti Master-kuvaukseen. Muut kuin DBA:t eivät pääse tämän jälkeen kuvauksiin käsiksi.

Anna vain DBA-käyttäjille palvelimen admin-oikeudet. Muut eivät tämän jälkeen voi luoda synonyymejä Dev Studiosta tai konsoliilta.