Kulttuuritietokantajärjestelmän suunnittelu matkapuhelinten lokalisoinnin apuvälineeksi

Pekka Mäkiaho

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos
Tietojenkäsittelyoppi
Pro gradu -tutkielma
Ohjaaja: Hannu Kangassalo
Huhtikuu 2006

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos

Tietojenkäsittelyoppi
Tekijän Nimi: Pekka Mäkiaho
Pro gradu -tutkielma, 42 sivua + 137 liitesivua

Maaliskuu 2006

Tässä tutkielmassa selvitän mahdollisuuksia rakentaa tietokantajärjestelmä, jota käytettäisiin hyödyksi matkapuhelinten toteuttamisessa eri kielten ja kulttuurien vaatimuksia vastaavaksi.
Teoriaosassa määrittelen kirjallisuuslähteiden avulla lokalisoinnin ja internationalisoinnin peruskäsitteitä sekä kerron internationalisointi-lähestymistavan eduista.
Tietojärjestelmän rakentamisen mahdollisuuksia tutkin keräämällä eri käyttäjäryhmien tarpeita tällaiselle järjestelmälle. Näiden tarpeiden perusteella kokosin vaatimusmäärittelyn, jonka pohjalta toteutettiin prototyyppiohjelmisto harjoitustyönä Tampereen yliopiston Tietojenkäsittelytieteiden laitoksella.
Valmistunutta prototyyppiä testattiin teknisesti asentamalla se Nokian intranetiin ja testaamalla sitä alkuperäistä vaatimusmäärittelyä vasten.
Järjestelmän hyödyllisyyttä tutkittiin käyttäjäpalautteen perusteella.
Tulokseksi sain, että vaatimusmäärittelyn mukainen järjestelmä on teknisesti toteutettavissa. Käytettävyyden kannalta suurin ongelma oli tiedon jatkuvasta muutoksesta sekä käyttäjäryhmien erilaisista tarpeista aiheutuvat ylläpito-ongelmat.
Järjestelmä kannattaa toteuttaa, mutta järjestelmään tulisi kerätä ennemminkin pysyvää kuin muuttuvaa tietoa sekä rajata käyttäjäryhmiä. Samoin teknistä ratkaisua kannattaa hieman muuttaa. Liitteinä on esitetty uuden järjestelmän tekninen määrittely [Liite 9, Pandora – Technical Specification] sekä tietokantakuvaus [Liite 8, Pandora – Database Specification].

Avainsanat ja -sanonnat: internationalisointi, lokalisointi, kulttuuri, tietokanta, matkapuhelin.

Sisällys

1. Johdanto

Tässä tutkimuksessa selvitän mahdollisuuksia toteuttaa Nokia Mobile Phones -yritykselle tietokantajärjestelmä, jota voisi käyttää hyödyksi matkapuhelinten sopeuttamisessa erilaisille markkinoille. Käytän tässä tutkielmassa termiä tietokantajärjestelmä (engl. database system) tai lyhyemmin järjestelmä kuvaamaan tietokantaa, siihen liittyvää tietokantaohjelmistoa sekä tietokannan sisältöä. [Atk-sanakirja, 1997]

Tähän järjestelmään on tarkoitus tallentaa alue- ja kielikohtaisesti jaoteltuna matkapuhelinten lokalisoinnin kannalta hyödyllistä tietoa. Olen jaotellut tiedon karkeasti kahteen osaan tekniseen ja kulttuurilliseen. Teknistä tietoa on esimerkiksi tietyllä alueella, tietyssä kielessä käytettävä tietokonenäppäimistön järjestely. Kulttuurillista taas voisi olla tieto eri värien ja symboleiden merkityksestä. Edellä mainittua tietoa tarvitaan ensisijaisesti NMP:n lokalisointiyksikössä, mutta vastaavaa alue- ja kielikohtaisesti jaoteltua tietoa käytetään myös muissa yksiköissä esimerkiksi markkinointi- tai lakiosastoilla. Tarkoituksena on, että ohjelmistoa voitaisiin käyttää koko Nokian laajuisesti.

Teknisen tiedon tarpeellisuus on ilmeistä: puhelimia on pystyttävä käyttämään maasta ja kielestä riippumatta. Kulttuurillisen tiedon merkitys on kuitenkin vähintään yhtä tärkeä: esimerkiksi väärillä väreillä, symboleilla tai mainoslauseilla voidaan, paitsi tehdä käyttöliittymästä ja käyttöohjeista epämiellyttäviä käyttää, myös syvästi loukata jonkin kulttuurin edustajia tai jopa rikkoa paikallisia lakeja.

Kun kulttuuritietokantaa alettiin suunnitella v 1999, en löytänyt aikaisempia tieteellisiä tutkimuksia, joissa käsiteltäisiin juuri vastaavaa aihetta. Internetistä löytyi lukuisia sivustoja, jotka kattoivat jonkin osa-alueen suunnitellusta järjestelmästä; esimerkiksi CIA:n World Factbook [CIA, 2003] sisälsi hyvin paljon valtiokohtaista tietoa, mutta ei esimerkiksi kielikohtaista tietoa. Samoin yhteen tai kahteen kulttuurialueeseen keskittyviä www-sivuja oli useita, mutta ei sellaista sivua tai tietokantaa, joka olisi suoraan vastannut yrityksen tarpeita

Tätä kirjoitettaessa (Huhtikuu, 2006) tilanne on edelleen sama..

Oletan – ja pidän jopa todennäköisetnä, että tällaisia järjestelmiä on yrityksillä käytössä, mutta ne eivät ole julkisia.

On selvää, että kustannustehokkuutta ajatellen ei ole järkevää tuottaa eri kieliversioita erillisinä ohjelmistoprojekteina vaan käyttää uudelleen olemassa olevaa lähdekoodia eri versioissa mahdollisimman paljon hyväksi. Optimaalisessa tapauksessa kaikki kieliversiot käyttävät samaa lähdekoodia (single code base) ja kieli- sekä kulttuurisidonnaiset osat ohjelmistosta on eristetty erillisiin tiedostoihin – esimerkiksi Windows-käyttöjärjestelmässä resurssitiedostoihin. Tätä lokalisointia edeltävää työtä kutsutaan internationalisoinniksi (internationalisation, i18n).

Käytän tässä alkuperäisistä termeistä internationalisation ja localisation suomennoksia internationalisointi ja lokalisointi. Internationalisointia kutsutaan myös kansainvälistämiseksi ja vastaava käännös sanalle localisation olisi paikallistaminen. Pidän kuitenkin valitsemiani käännöksiä selkeämpinä.

Internationalisointi siis edeltää lokalisointia; Käpyahon [2001, 6] mukaan nämä vaiheet menevät osittain päällekkäin, eikä voida aina täsmällisesti määritellä, missä internationalisointi loppuu ja lokalisointi alkaa. Globalisointia (engl. globalisation) näkee myös käytettävän yleisnimikkeenä koko ohjelmistotuotantoprosessille alkaen suunnittelusta ja toteutuksesta päätyen lokalisointiin ja myyntiin kohdemarkkinoille [Deitch and Czarnecki, 2001, 9]. Itse käytän tässä kuitenkin sanasta globalisation pelkästään käännöstä globaalistuminen tarkoittaen Kasvion ja Niemisen [1998] mukaan "niitä tapahtumasarjoja ja prosesseja, joiden seurauksena maailma integroituu yhdeksi kattavaksi sosiaaliseksi järjestelmäksi". Tämä on perinteinen globaalistumisen määritelmä, mutta toisaalta voi miettiä, onko tällainen yksi kattava sosiaalinen järjestelmä koskaan mahdollista. Voi ajatella, että aina tulee olemaan useita erilaisia alakulttuureja ja jo erilaisten uskontojen olemassaolo aiheuttaa sen, että aina tulee olemaan erilaisia sosiaalisia järjestelmiä – ei koskaan yhtä ainoata ”kattavaa” järjestelmää.

Uskonto on tärkeä osa kulttuuria, mutta tästä tutkielmasta uskontojen vaikutus kulttuuriin on jätetty aihealueen rajaamiseksi kokonaan pois.

Käsitteen kulttuuri tutkiminen voisi olla kokonaisen väitöskirjan aihe; Yeo [1996] määrittelee kulttuurin "tietylle ihmisryhmälle tai luokalle tyypillisenä käyttäytymisenä." Tässä tutkielmassa käytän tätä määritelmää siten, että ihmisryhmät tunnistetaan kielen ja asuinalueen perusteella. Näin lähestytään paikanteen (engl. locale) määritelmää, josta tarkemmin luvussa2.2. Luvussa 2.1 esittelen räätäälöinnin - perinteisen menetelmän ohjelmiston kieliversioiden tuottamiseen. Kerron räätälöinnin ongelmista - etenkin erilaisista kirjoitusjärjestelmistä aiheutuvista, käyn läpi yleisimpiä kirjoitusjärjestelmiä sekä määrittelen lokalisointiin liittyviä peruskäsitteitä.

Luvussa 3 määrittelen tarkemmin lokalisoinnin ja internationalisoinnin sekä käyn läpi käytännön perussääntöjä internationalisointityössä.
Luvussa 4 esitän syitä, miksi kulttuuritietokanta voisi olla hyödyllinen ja mihin sitä käytettäisiin.
Luvussa 5 käsittelen varsinaista tutkimusongelmaa, eli *onko tällaisen järjestelmän rakentaminen liiketoiminnallisesti* järkevää? Liiketoimminnallisella järkevyydellä en tarkoita pelkästään, saadaanko tällaista järjestelmää käyttämällä välitöntä voittoa – vaan myös, miten järjestelmän käyttäminen – tai käyttämättä jättäminen – voi vaikuttaa yrityksen imagoon. Ongelman selvittämiseksi toteutettiin järjestelmän prototyyppi Tampereen yliopiston Tietojenkäsittelytieteiden laitoksella. Kuvaan tässä luvussa myös prototyypin vaatimuksia, sekä sen toteuttamista.
Luvussa 6 arvioin ohjelmiston käyttökelpoisuutta sen jälkeen kun se oli asennettu Nokian intranetiin.
Prototyyppiprojektin läpiviennistä saatujen kokemusten ja järjestelmän arvioinnin tulosten perusteella osoitan luvussa 7, että kyseisen järjestelmä kannattaa toteuttaa tietyin rajoituksin.

2. Kansainväliset ohjelmistot

Tietokoneohjelmistoja suunnitellaan yhä vähemmän pelkästään yksikielisiksi. Ohjelmistojen markkina-alue on globaalistumisen myötä usein monikielinen ja monikulttuurinen. Käyttäjät haluavat paitsi käyttää ohjelmistoja omalla äidinkielellään myösohjelmiston olevan kulttuurinsa mukainen.
Ranskankielinen käyttäjä Ranskassa kirjoittaa esimerkiksi vuoden 2002 toisen päivän päivämäärän muodossa 02.01.2002. kun taas kanadanranskalainen haluaisi päivämääräesityksen olevan 2002-01-02. Molemmissa tapauksissa käyttöliittymäkieli on kuitenkin (pääpiirteissään) sama. Esselinkin [1998] mukaan ohjelmiston sovittamista loppukäyttäjän kielen ja kulttuurin tarpeisiin kutsutaan *lokalisoinniksi*. Lokalisoinnista käytetään usein myös lyhennettä *l10n* sen englanninkielisen vastineen 'localisation' ensimmäisen ja viimeisen kirjaimen sekä näiden väliin jäävien kirjainten lukumäärän mukaan.

2.1. Räätälöinti

Räätälöinnillä (engl. customisation) tarkoitetaan ohjelmiston muokkaamista kirjoittamalla lähdekoodi uudelleen siten, että ohjelmisto soveltuu uuden kohdekulttuurin tarpeisiin. Tähän tilanteeseen joudutaan, jos ohjelmiston suunnittelussa ei ole alun perin otettu huomioon mahdollisuutta, että tarvitaan useampi kuin yksi kieliversio - ohjelmistoa ei siis ole internationalisoitu. Kun alkuperäiskielisestä ohjelmistosta tehdään uusi päivitysversio, niin myös kieliversioiden lähdekoodia joudutaan muuttamaan.

Vaikka joissain tapauksissa räätälöinti saattaa olla nopeampi tapa tuottaa uusi kieliversio, niin yleensä pitkällä tähtäimellä internationalisointi-lähestymistapa on kustannustehokkaampi. Tuthillin ja Smallbergin [1997, 4-5] mukaan räätälöinti edellyttää kohdekielen tuntevien ohjelmistokehittäjien palkkaamista ja lähdekoodin muokkaamista sekä uudelleen kääntämistä (compilation). He esittävät myös seuraavat räätälöintilähestymistavan heikkoudet:

- räätälöity versio tulee markkinoille paljon myöhemmin kuin alkuperäinen ja tämä vähentää myyntiä.
- tarvitaan kaksinkertainen määrä resursseja ohjelmistokehitykseen.
- ohjelmistojen ylläpito vaikeutuu, koska lähdekoodista on useita versioita.
- tuen lisääminen uudelle kielelle on vaikeaa.

Käpyaho [2001, 4-5, 10] on samoilla linjoilla ja lisää vielä, että ohjelmistosuunnittelijat ovat nykypäivänä rajallinen resurssi ja on tuhlausta käyttää heitä pelkästään olemassa olevan koodin muokkaamiseen uusia kieliversioita varten.

2.2. Paikanne

Pelkästään käyttöliittymän kielen kääntäminen ohjelmiston muokkaamiseksi eri kulttuurien tarpeita varten ei riitä, vaikka käyttöliittymä onkin hyvin näkyvä osa tietokoneohjelmistoa. Kahdella eri alueella voivat esimerkiksi valuuttasymbolit ja kellonaika- tai päivämääräformaatit erota toisistaan, vaikka kieli olisi sama. Saksankielinen loppukäyttäjä Sveitsissä haluaa käyttää desimaalierottimena pistettä, kun taas saksankielinen itävaltalainen käyttäisi erottimena pilkkua.

Kohdekulttuurin identifioimiseksi on luotu käsite *paikanne*, (engl. locale) jonka yksikäsitteisenä tunnisteena on yhdessä kielen kanssa valtio tai alue. Myös tässä tutkielmassa käytetään edellistä, tietokonemaailmassa yleisesti vallalla olevan käytännön mukaista määritelmää. Kano [1995, 2] puolestaan määrittelee paikanteen kokoelmana käyttöympäristön maasta, kielestä ja kulttuurista riippuvia ominaisuuksia. Tämä määritelmä olisi tarkempi, mutta vallitsevan käytännön ja yksinkertaistamisen vuoksi olen päätynyt käyttämään ensimmäistä määritelmää. On kuitenkin huomattava, että tästä valinnasta aiheutuu järjestelmälle selkeitä puutteita: ei voida sanoa, että kulttuuri on samanlaista aina saman paikanteen sisällä. Esimerkiksi Venäjän venäjänkielisten uusrikkaiden elämä eroaa huomattavasti Venäjän köyhälistön elämästä, vaikka molemmat ryhmät kuuluvat samaan paikanteeseen (Venäjä, venäjä)

Internationalisoitu ohjelmisto kysyy tietokoneen käyttöjärjestelmältä paikanteen ja hakee esimerkiksi juuri halutun desimaalierottimen paikanteelta. Näin lähdekoodi pysyy aina samanlaisena, vaikka käyttöliittymän ulkoasu muuttuukin paikanteen mukaan.

Eri käyttöjärjestelmissä sekä Java-virtuaalikoneessa määritellyt paikanteet eroavat jonkin verran toisistaan. Mitä enemmän ominaisuuksia käyttöjärjestelmän paikanteessa on, sitä paremmin sovellusohjelma voidaan internationalisoida. Käpyaho [2001, 15-16] luettelee, mitä paikanteeseen vähintään pitäisi kuulua:

- maan ja kielen alkuperäiset nimet, sekä englanninkieliset käännökset
- merkkijonojen lajittelujärjestys
- merkkitieto: tekstissä käytettävät merkit, merkkien luokittelu (teksti, numero, välimerkki) sekä tieto muunnoksesta isojen ja pienten kirjainten välillä
- merkkien koodaus (encoding): mitä koodauksia käytetään, mitä tietoa tarvitaan muunnokseen koodausten välillä
- numerojen muotoilu: kokonais- ja desimaalilukujen sekä valuutta-arvojen esitystavat
- kellonajan ja päivämäärän esitystavat
- tieto aikavyöhykkeestä sekä kesäajan käytöstä
- kalenterikäytännöt: virallinen kalenteri, päivä- ja viikkonumeroiden käyttö
- mittayksiköt: mitä mittayksikköjä käytetään yleisimmille suureille
- paperikoko: tavallisimmat paikanteessa käytettävät (tulosteiden) paperikoot
- osoitemuoto: mitä kenttiä postiosoitteessa käytetään ja missä järjestyksessä

- puhelinnumeron muoto: miten puhelinnumerot muodostetaan ja esitetään.

2.3. Kirjoitusjärjestelmät

Tärkeä huomioonotettava asia suunniteltaessa kansainvälisille markkinoille tarkoitettua ohjelmistoa on kirjoitusjärjestelmä eli skripti. Deitch ja Czarnecki [2001, 17] määrittelevät kirjoitusjärjestelmän "kirjoitetun informaation kulkuvälineenä".

Kirjoitusjärjestelmä on siis sopimus, miten informaatiota esitetään kirjoitetussa muodossa. Näitä kirjoitusjärjestelmiä on useita erilaisia ja samaakin kieltä voidaan kirjoittaa usealla kirjoitusjärjestelmällä. Esimerkiksi japania kirjoitetaan sekaisin tavukirjoitusjärjestelmillä (Hiragana ja Katagana) sekä ideografisella Han-skriptillä. Han-kirjoitusjärjestelmää käytetään myös kiinan ja korean esittämiseen. Kirjoitusjärjestelmän ja kielen suhde on siis monen suhde moneen.

Useimmat länsimaiset kielet (englanti, ranska, saksa, suomi) käyttävät latinalaista kirjoitusjärjestelmää. Latinalainen kirjoitusjärjestelmä kuuluu foneettisiin kirjoitusjärjestelmiin ja sitä kirjoitetaan vasemmalta oikealle. Englanninkielisen tekstin kirjoittamiseen riittää erikoissymbolien lisäksi 26 isoa ja pientä kirjainta, mutta esimerkiksi suomenkielessä tarvitaan lisäksi kirjaimet "Ä", "Ö" ja "Å". Arabialainen kirjoitusjärjestelmä on myös foneettinen, mutta se kuuluu kaksisuuntaisiin järjestelmiin ja sillä voidaan esittää mm. kieliä arabia ja kurdi.

Samankin kirjoitusjärjestelmän sisällä eri kielissä voidaan käyttää erilaista joukkoa merkkejä. Hyvin tärkeä osa internationalisointia on ottaa huomioon tuki erilaisille kirjoitusjärjestelmille. Kirjoitusjärjestelmät eroavat toisistaan mm. kirjoitussuunnan, rivinvaihtosääntöjen, välimerkityksen sekä isojen ja pienten kirjainten erottelun suhteen. Lisäksi järjestelmästä riippuen merkin kirjoitusasu saattaa muuttua sen mukaan, missä kohtaa sanaa se sijaitsee. Kaikki nämä asiat on otettava huomioon suunniteltaessa ohjelmiston lokalisointia jollekin uudelle kielelle. Seuraavissa luvuissa kuvaan tärkeimpiä käytössä olevia luonnollisia kirjoitusjärjestelmiä lähinnä keskittyen ominaisuuksiin, jotka on otettava huomioon esitettäessä kyseistä kirjoitusjärjestelmää tietokoneskriptinä. Tietoa eri skripteistä tarvitaan tehtäessä päätöksiä uuden kielen lisäämiseksi tuettuun kielivalikoimaan: tarvitaanko uusia fontteja, tukeeko käyttäjärjestelmä valmiiksi kirjoitussuuntia ja niin edelleen. Käyn läpi skriptien eroavaisuuksia ja osoitan, että tämän osan liittäminen järjestelmään on – osavastauksena tutkimusongelmaan - liiketoiminnallisesti järkevää.

2.3.1. Arabialainen kirjoitusjärjestelmä

Arabialaisesta kirjoitusjärjestelmästä on olemassa useita muunnelmia. Kaikissa arabiaa puhuvissa valtioissa tunnetaan kuitenkin standardi-arabia, jota yleensä käytetään virallisen tekstin tuottamiseen. Arabialainen kirjoitusjärjestelmä muodostuu 28 kirjaimesta ja sitä kirjoitetaan horisontaalisesti oikealta vasemmalle. Poikkeuksen kirjoitussuuntaan muodostavat vieraskieliset sanat, jotka kirjoitetaan – myös muun tekstin joukossa – kielen alkuperäisen suunnan mukaan. Samoin numerot kirjoitetaan vasemmalta oikealle. Arabialainen kirjoitusjärjestelmä on siis *kaksisuuntainen*.

Kaksisuuntainen kirjoitusjärjestelmä asettaa haasteita tietokoneohjelman käyttöliittymälle: jos esimerkiksi halutaan siirtää osoitinta tekstin joukossa eteenpäin, niin osoitinta ympäröivistä merkeistä riippuu siirrytäänkö oikealta vasemmalle vai vasemmalta oikealle. Usein käytetään sekä visuaalista että loogista kursoria: visuaalinen kursori siirtyy oikealle ja vasemmalle vastaavien näppäinpainikkeiden mukaisesti, looginen kursori liikkuu eteenpäin/taaksepäin -näppäimistä tekstikontekstin mukaan joko oikealle tai vasemmalle. Myös merkin ulkoasu saattaa vaihtua sen mukaan, sijaitseeko merkki sanan alussa, keskellä vai lopussa.

2.3.2. Han-kirjoitusjärjestelmä

Han-kirjoitusjärjestelmää käytetään kiinan, korean ja japanin kirjoittamiseen. Tosin siitä käytetään hieman eri nimitystä kunkin kielen yhteydessä: järjestelmä on Hanzi kiinassa, Kanji japanissa ja Hanja koreassa. Merkkien ulkonäöt poikkeavat myös hieman toisistaan.

Kiinaa voidaan kirjoittaa pelkästään Han-skriptillä, koreaa joko Han-skriptillä tai vain korean kirjoittamiseen tarkoitetulla Hanguilla. Japania kirjoitetaan yhdistämällä Han-, Hiragana- ja Katagana- skriptejä.

Han-skripti on piktograafis-ideograafinen eli yhdellä merkillä sinällään on semanttinen merkitys. Esimerkiksi kiinan kielessä näitä merkkejä on yli 70 000. Kukin merkki edustaa morfeemia, eli kielen pienintä merkityksellistä yksikköä. Yksi sana ei siis aina välttämättä koostu kuitenkaan yhdestä merkistä vaan esimerkiksi kiinan kielessä yleensä yhdestä, kahdesta tai kolmesta merkistä. Han-skriptiä kirjoitetaan joko vasemmalta oikealle ja ylhäältä alas tai ylhäältä alas ja oikealta vasemmalle. Merkkien suuri määrä aiheuttaa ongelmia käytettäessä skriptiä tietokoneissa. Ensinnäkään

merkkejä ei voida syöttää käyttäen tavallista konekirjoitusnäppäimistöä, vaan tarvitaan muita järjestelmiä merkkien syöttämiseen. Toiseksi varsinkin laitteissa, joissa käytettävän muistin määrä on pieni (PDA-laitteet, matkapuhelimet) voidaan joutua valitsemaan vain osajoukko kaikista merkeistä. Kolmanneksi merkkien esittäminen näytöllä vaatii enemmän kuva-alkioita, jotta suuri joukko merkkejä voidaan erottaa toisistaan.

2.3.3. Heprealainen kirjoitusjärjestelmä

Heprealainen kirjoitusjärjestelmä on hyvin samankaltainen arabialaisen kirjoitusjärjestelmän kanssa. Myös sitä kirjoitetaan kaksisuuntaisesti ja merkkien ulkoasu muuttuu sen mukaan, missä kohtaa sanaa ne sijaitsevat.

Heprealaista kirjoitusjärjestelmää käytetään myös juutalais-arabian, jiddishin ja ladinon kirjoittamiseen. Ongelmat esitettäessä skriptiä tietokoneella ovat pääsääntöisesti samat kuin arabialaisessa skriptissä.

2.3.4. Hiragana ja Katakana

Hiragana- ja Katakana- järjestelmiä käytetään japanin kielen kirjoittamiseen yhdessä Kanji-järjestelmän kanssa. Hiragana ja Katakana ovat tavukirjoitusjärjestelmiä – yksi merkki edustaa yhtä tavua.

Kirjoitetussa tekstissä voi esiintyä vuorotellen merkkejä kaikista kolmesta järjestelmästä. Kanji-merkkiä voidaan japanissa käyttää silloin kun sanalle sellainen on olemassa. Kuitenkin on aina mahdollista kirjoittaa sana myös käyttäen Hiragana- tai Katakana-järjestelmiä. Käytännössä Hiraganalla esitetään yleensä sellaisia japanilaisia sanoja, joille Kanjj-merkkiä ei löydy sekä liitepartikkeleiden esittämiseen. Katakanaa käytetään vierasperäisten sanojen esittämiseen.

Japania voidaan kirjoittaa joko ylhäältä alas ja oikealta vasemmalle tai vasemmalta oikealle ja ylhäältä alas. Luvussa 2.3.2, Han-kirjoitusjärjestelmä, on kerrottu ongelmista, joita syntyy käytettäessä Kanji-merkkejä tietokoneessa. Haastavimmat näistä ovat tarvittavan muistin määrä sekä Kanji-merkkien vaatima suuri pikselimäärä. Hiragana- ja Katakana- järjestelmissä merkkejä on vähemmän (46 kummassakin) ja merkit pystytään erottamaan toisistaan, vaikka kuva-alkioita olisi käytettävissä vähemmän. Toisaalta pikselimäärän merkkiä kohti on oltava kuitenkin suurempi kuin

esimerkiksi latinalaisessa skriptissä. Myös merkkien syöttöjärjestelmä asettaa vaatimuksia – käytössä on ilman Kanji-merkkejäkin jo 92 erilaista merkkiä.

2.3.5. Latinalainen, kreikkalainen ja kyrillinen kirjoitusjärjestelmä

Latinalaisella kirjoitusjärjestelmällä kirjoitetaan useimpia länsimaisia kieliä. Sen katsotaan syntyneen etruskien kirjoitusjärjestelmästä noin 700 eKr. Alkuperäinen roomalaisten käyttämä latinalainen kirjoitusjärjestelmä sisälsi pelkästään isot kirjaimet (A-Z). Nykyisessä latinalaisessa kirjoitusjärjestelmässä on isojen ja pienten kirjaimien (yhteensä 52) lisäksi aksentti- ja välimerkkejä, 10 numeromerkkiä sekä eri kielissä käytettäviä erikoismerkkejä.

Kyrillisessä ja kreikkalaisessa kirjoitusjärjestelmässä teksti kirjoitetaan kuten latinalaisessakin: yksisuuntaisesti vasemmalta oikealle ja ylhäältä alas. Tärkein ero näiden kirjoitusjärjestelmien välillä on erilainen merkistö. Käytettäessä kreikkalaista ja kyrillistä skriptiä tietokoneella suurin ongelma syntyy syöttömetodin vaihdosta: useimmiten kyrillistä tai kreikkalaista kirjoitusjärjestelmää käyttävä henkilö haluaa kirjoittaa myös latinalaisella skriptillä.

2.3.6. Thai-kirjoitusjärjestelmä

Thai-kirjoitusjärjestelmää voidaan kutsua puolittaiseksi tavukirjoitusjärjestelmäksi: sen 44 perusmerkkiä ovat konsonantteja, tavuun kuuluvat vokaalit lisätään peruskonsonantin alapuolelle, sivuille tai yläpuolelle. Lisäksi perusmerkin päällä voi olla *tooni*-merkki, joka kertoo, kuinka tavu äännetään.

Thain kieltä kirjoitetaan vasemmalta oikealla ja ylhäältä alas. Kirjoitussuunta ei ole kuitenkaan niin yksinkertainen kuin esimerkiksi latinalaisessa järjestelmässä, koska vokaalit ja tooni-merkit kirjoitetaan konsonantin ympärille.

Tietokoneella haasteita syntyy ensinnäkin syöttömetodille: tarvitaan ehkä visuaalinen ja looginen kursori, kuten kaksisuuntaisissa kielissä. Katso arabialainen kirjoitusjärjestelmä, 2.3.1. Koska vokaaleja voi olla peruskonsonantin ylä- tai alapuolella ja yläpuolella lisäksi tooni-merkki, tarvitaan pystysuunnassa enemmän tilaa.

2.3.7. Merkki, glyyfi, renderöinti ja fontti

Merkillä (engl. character) tarkoitetaan "kirjoitetun kielen pienintä osaa, jolla on semanttinen merkitys" [The Unicode Consortium, 2000]. Merkki on siis esimerkiksi kirjaimen "L" *abstrakti* esitys, joka ei ota millään tavalla kantaa, mikä on merkin ulkomuoto piirrettäessä eli *renderöitäessä* merkki vaikka tietokoneen näytölle tai tulostimen paperille. Tätä merkin piirrettyä ulkomuotoa kutsutaan *glyyfiksi* (engl. glyph) Glyyfin ja merkin suhde ei ole kuitenkaan yhden suhde yhteen: "*fontti koostuu* fonttitaulukosta ja joukosta glyyfejä" [Deitch and Czarnecki, 2001, 201] Merkkitaulukko yhdistää merkin ja tämän fontin mukaan esitettävän glyyfin. Myös samassa fontissa merkin ja glyyfin suhde on monen suhde moneen, esimerkiksi arabiassa merkin ulkoasu riippuu sen paikasta sanassa: merkki esitetään eri glyyfillä sanan alussa, keskellä tai lopussa. Lisäksi glyyfi voi edustaa pelkästään merkin osaa. *Parike* (engl. ligature*)* taas tarkoittaa glyyfiä, joka edustaa kahta tai useampaa erillistä merkkiä. Esimerkiksi sähköpostissa käytettävää sanaa "at" esitetään symbolilla *@*, ja et-merkkiä symbolilla *&*.

2.3.8. Merkkijoukot

Deitch ja Czarnecki [2001] määrittelevät termin *character set* tarkoittaen sillä "kirjoitusjärjestelmässä käytettyjen symbolien joukkoa" sekä *termin coded character set* (tai code set) tarkoittaen sillä kirjoitusjärjestelmän symbolien joukkoa, (engl. repertoire) jossa jokaiseen symboliin on yhdistetty yksikäsitteinen numeerinen arvo. Tätä merkin yhdistämistä numeroarvoon kutsutaan koodaukseksi (engl. encoding) Tässä esityksessä käytän sanaa *merkkijoukko* tarkoittaen nimenomaan käännöstä käsitteelle "coded character set".

Vaikka siis kaksi merkkijoukkoa sisältäisi täsmälleen samat symbolit, niin nämä joukot eivät välttämättä olisi keskenään yhteensopivia, koska tietokoneen muistissa merkit esitetään positiivisina kokonaislukuina - mikä merkki vastaa mitäkin lukua riippuu koodauksesta. Ensimmäisiä merkkijoukkostandardeja oli ASCII (American Standard Code for Information Interchancing). ASCII-standardi määritteli 7-bittisen merkistön koodin jokaiselle siihen kuuluvalle merkille ja tämä riitti englanninkielisen tekstin esittämiseen. Monien muiden kielten erikoismerkkien esittämistä varten 7 bittiä oli kuitenkin liian vähän. Syntyi useita erilaisia 8-bittisiä merkkijoukkoja. ISO [1999c] määrittelee ISO-8859-sarjan merkkijoukkostandardit. Kielissä kuten kiina ja japani

eivät kuitenkaan riitä 8 bitin mahdollistamat 256 erilaista symbolia. Siksi otettiin käyttöön erilaisia *monitavuisia* (engl. multibyte) merkkijoukkoja, joissa yhtä symbolia voi esittää yksi, kaksi tai kolme tavua. ISO 2022-sarja määrittelee standardin monitavuisille merkkijoukoille.

2.3.9. Kohti maailmanlaajuista merkistöä

Erilaisten merkkijärjestelmien käyttö aiheuttaa paljon yhteensopivuusongelmia. Näin on syntynyt tarve luoda yhteinen merkkijärjestelmä, jota voitaisiin käyttää kielestä riippumatta.

Lokakuussa 1991 Unicode Consortium julkaisi Unicode-standardin, (Unicode on peräisin sanoista universal codeset) joka määrittelee yksikäsitteisen 16-bittisen arvon maksimissaan 65536 merkille. Myös ISO määritteli samanaikaisesti universaalia merkkijärjestelmää. Tällä hetkellä (Huhtikuu, 2006) virallinen versio Unicode-standardista on 4.1 ja se on täysin yhteensopiva ISO/IEC 10646 – standardin kanssa.

Tärkeä etu Unicodella useisiin muihin merkkijärjestelmiin on, että se perustuu merkkeihin - ei glyyfeihin. Esimerkiksi heksadesimaaliarvo 039B tarkoittaa yksikäsitteisesti isoa lamdaa (Greek capital letter lamda) ottamatta kantaa, miten se milloinkin piirretään näytölle eli renderöidään. Unicoden käyttö sovelluksessa varmistaa sen yhteensopivuuden muiden Unicodea tukevien sovellusten kanssa.

Tässä luvussa kerroin perinteisestä tavasta lokalisoida ohjelmia: räätälöinnistä. Määrittelin myös lokalisoinnin kannalta olennaisen käsitteen paikanne. Lopuksi esittelin erilaisia merkkijärjestelmiä ja Unicode-standardia.

3. Internationalisointi-lähestymistapa

Luvussa 2 esittelin räätälöintilähestymistavan lokalisoitujen ohjelmistoversioiden tuottamiseen ja osoitin sen heikkouksia. Kustannustehokkaampi tapa kieliversioiden tuottamiseen on internationalisoida ohjelmisto ennen lokalisointia.

3.1. Lokalisointi

Ohjelmiston lokalisointi tarkoittaa sen muokkaamista kohdekulttuurin vaatimusten mukaiseksi. Näkyvimpiä osia lokalisoinnissa on käyttöliittymän tekstien kääntäminen kohdekielelle, mutta siihen kuuluu paljon muutakin - käyttöliittymässä näkyvien symbolien muuttamisesta kohdekulttuurin ymmärtämiksi ja hyväksymiksi aina myyntipakkauksiin liitettäviin tuotevastuulain mukaisiin vastuuvapautuslausekkeisiin sekä käyttöohjeisiin.

Esselink [1998, 2-3] määrittelee lokalisoinnin ohjelmiston muokkaamisena ja kääntämisenä toiselle kielelle siten, että se täyttää paikallisten markkinoiden kielelliset ja kulttuurilliset vaatimukset.

Lokalisointiprojekti koostuu hänen mukaansa kolmesta pääosasta: ohjelmiston (käyttöliittymän) kääntämisestä online-helppien ja käyttäjädokumentaation kääntämisestä sekä lokalisoitujen versioiden testauksesta. Ohjelmisto voi olla osittain tai kokonaan lokalisoitu; Luongin ja muiden [1995, 111–112] mukaan täydellisesti lokalisoidussa ohjelmistossa seuraavat komponentit on lokalisoitu:

- kaikki käyttöliittymässä näkyvät tekstit
- online-helpit
- tietokoneavusteinen opetusmateriaali (tutoriaalit)
- dokumentaatio
- esimerkkiohjelmat ja templaatit
- makrokieli

Tätä määritelmää voi tietysti hieman kritisoida: jos makrokieltä ei lokalisoida, se ei mielestäni vähennä tuotteen lokalisointiastetta vaan paremminkin varmistaa yhteensopivuuden eri kieliversioilla tuotettujen makrojen välillä. Lisäksi tässä on jätetty huomiotta kokonaan käyttöliittymän graafisten symbolien ja äänten lokalisointi. Ehkä kirjoittajat ovat ajatelleet symbolien ja äänten olevan hyvin internationalisoidussa ohjelmassa niin kulttuuri-neutraaleja, ettei näitä tarvitse erikseen lokalisoida. On myös harkittava tarvitseeko tuotenimeä lokalisoida.
Tuotteen lokalisointi on siis paljon enemmän kuin pelkästään käyttöliittymässä näkyvien tekstien kääntämistä kohdekielelle.

3.2. Internationalisointi

Internationalisointi on osa ohjelmiston suunnittelu- ja toteutusvaiheita sekä edeltää lokalisointia. Internationalisoinnin päämääränä on saada lokalisointi mahdollisimman kustannustehokkaaksi. Internationalisointilähestymistapaa käytettäessä kustannukset voivat olla aluksi suuremmat, koska alun perin joudutaan valmistautumaan lokalisointiin useille kielille. Pitkällä tähtäimellä nämä kustannukset saadaan kuitenkin takaisin.
Luvussa kaksi esittelin räätälöintilähestymistavan kieliversioiden tuottamiseen. Tässä luvussa esittelen internationalisoinnin ja sen edut räätälöintinlähestymistapaan

verrattuna sekä kerron joitakin käytännön perussääntöjä internationalisoinnista. Käyttäjädokumentaation internationalisoinnin jätän tässä kokonaan käsittelemättä.

Voidaan ajatella, että suurin osa ohjelmistoista on alun perin toteutettu yksikielisiksi ja tietyn kulttuurin käytäntöjen mukaiseksi. Jokainen ohjelmisto voidaan kuitenkin lokalisoida eli kääntää ja muokata toiseen kieleen ja kulttuuriin sopivaksi. Kuinka paljon työtä tämä muutos sitten aiheuttaa, riippuu paitsi kielten ja kulttuurien eroista myös, kuinka hyvin ohjelman suunnittelussa on otettu huomioon lokalisointitarpeet - siis kuinka hyvin ohjelma on internationalisoitu.

Tuthillin ja Smallbergin [1997,5] mukaan internationalisointi on "tapa suunnitella ja tuottaa ohjelmisto niin, että se voidaan helposti muokata paikallisille markkinoille sopivaksi". Internationalisointia voidaan ajatella ohjelmiston neutraloimisena niin, että kulttuurilliset ja kielelliset erityispiirteet poistetaan tai eristetään varsinaisesta lähdekoodista. Lokalisointi taas on menetelmä näiden erityispiirteiden lisäämiseksi ohjelmaan kohdekulttuurin tarpeiden mukaan.

3.2.1. Miksi internationalisoida?

Ohjelmistoyritysten päämääränä on pääsääntöisesti tuottaa voittoa. Nykyään ohjelmistoja tuotetaan yhä useammille kielille ja yhä suurempi osa tuotosta saadaan muista kieliversioista kuin alkuperäisestä. Tuthillin ja Smallbergin [1997] mukaan aikavälillä 1993–1995 yli 50 % Yhdysvaltalaisten ohjelmistotalojen voitoista tuli USA:n ulkopuolelta ja trendi on kasvava.

Yhä useammin ohjelmistoja siis kannattaa myynnin edistämiseksi (tai edes mahdollistamiseksi) lokalisoida. Valmiiksi internationalisoidun ohjelman lokalisoinnin vaatima työpanos on paljon pienempi kuin räätälöintilähestymistavassa.

3.2.2. Internationalisointi käytännössä

Tärkein perussääntö internationalisoinnissa on "Älä kovakoodaa mitään käyttöliittyvässä näkyvää, äläkä tee mitään oletuksia kielestä tai paikanteesta"

Tämä tarkoittaa, että kaikkien käyttöliittymään kuuluvien komponenttien: tekstin, symbolien, ikonien, ja valikoiden tulisi olla omissa erillisissä tiedostoissaan, tai muissa lähteissä, jotka sitten voidaan vaihtaa lokalisoidussa versiossa. Samoin ohjelman tulisi käyttää mahdollisimman paljon hyväkseen paikanteesta saatavaa informaatiota, eli ei esimerkiksi olettaa päivämäärämuodon olevan DD.MM.YYYY (Vuosituhantemme

toinen päivä esitetään ”02.01.2000”) vaan kysyä tämä käyttöjärjestelmäkutsujen avulla paikanteelta. Tällä tavoin päivämäärämuoto näkyy aina oikein, eikä sitä tarvitse muuttaa lokalisoitaessa ohjelmistoa. Seuraavissa kappaleissa on lueteltu joitakin muita tärkeitä perussääntöjä.

Käytä Unicodea. ASCII laajennuksineenkaan ei riitä kaikkien kielten tukemiseen. Unicodea käyttämällä varmistetaan myös yhteensopivuus kieliversioiden välillä

Käytä merkkijonojen järjestämiseen käyttöjärjestelmän palveluita, älä vertaile merkkien arvoja. Säännöt, joilla merkkijonoja järjestetään ovat hyvin paikanneriippuvaisia: esimerkiksi suomessa "ä" on aakkosten lopussa, mutta saksassa se on saman arvoinen merkin "a" kanssa.

Jätä käyttöliittymään tarpeeksi tilaa käännökselle. Yleensä englanninkielinen lähdeteksti vie vähemmän tilaa kuin käännös (Add-on vs. Programma aggiuntivo). Sama laajentuminen on otettava huomioon, jos käännettäville teksteille joudutaan ohjelmakoodissa määräämään enimmäispituus.

Käytä järjestettyjä parametreja merkkijonoissa. Sanajärjestys vaihtelee eri kielissä. Merkkijonon "Sending page %1 of fax number %2" käännös suomeksi voisi olla "lähetetään faxin %2 sivua %1"

Tässä luvussa määrittelin tarkemmin lokalisoinnin ja internationalisoinnin, perustelin, miksi ohjelmistoja kannattaa internationalisoida ja esitin joitakin käytännön perussääntöjä ohjelmistojen internationalisoinnissa.

4. Kulttuuritietokanta matkapuhelinten lokalisoinnin apuvälineenä

Edellisissä luvuissa osoitin, että onnistunut ohjelmistojen lokalisointi edellyttää paitsi hyvin toteutettua internationalisointityötä, myös tietämystä eri kulttuurien erityispiirteistä. Eli tämä tukee sitä, että pelkkä internationalisointi ei riitä vaan tutkimusongelman mukaista tietojärjestelmää tarvitaan.

Olen käsitellyt tähän asti tietokoneohjelmistojen lokalisointia, samat lainalaisuudet pätevät kuitenkin myös erikoistapauksena matkapuhelimiin ja kämmentietokoneisiin. Itse asiassa Fernandes [1995] kertoo esimerkin, miten eräs amerikkalainen autovalmistaja epäonnistui japanilaisilla markkinoilla, koska ei ollut ottanut huomioon kohdekulttuurin erityispiirteitä.

- Ohjauspyörä oli väärällä puolella autoa.
- Sivupeilejä ei pystynyt kääntämään auton sivuihin kiinni: autoa ei saanut ajettua pienemmille autoille mitoitettuihin talleihin.
- Istuinten ja hallintalaitteiden säädöt oli mitoitettu keskivertoamerikkalaisen mukaan: pienimmät japanilaiset pystyivät tuskin näkemään kojelaudan yli.
- Moottoreita ei ollut suunniteltu käymään japanilaisella matalaoktaanisella polttoaineella: moottorit kävivät huonosti.
- Vaihdetangon kirjaimilla "R", "D", ja "L" ei ollut mitään merkitystä ihmisille, jotka osasivat lukea vain japania.
- Käyttöohjeet olivat vain englanniksi.
- Autojen korjaus oli vaikeaa, koska tarvittiin vaikeasti saatavia tuumakokoisia työkaluja.

Tämä esimerkki osoittaa kuinka paljon asioita voi jäädä huomiotta siirryttäessä kulttuurista toiseen. Lisäksi ei riitä, että nämä asiat tiedostetaan, vaan on myös oltava tietoa, miten ominaisuuksia pitää muuttaa; edellisessä esimerkissä olisi pitänyt ottaa selville, mitkä japanilaiset symbolit olisivat vastanneet vaihdetangon symboleja "R", "D" ja "L".

Esimerkki liittyi auton käytettävyyteen, mutta jo esimerkiksi tuotteen ulkonäkö tai tapa mainostaa tuotetta vaativat tietoa kohderyhmästä. Nokia joutui lopettamaan Saksassa mainoskampanjansa yhdysvaltalaisen juutalaisjärjestön (AJS) pyynnöstä, koska kampanjan mainoslause ”Jokaiselle omansa” käännettiin saksaksi "Jedem das Seine", joka tarkoittaa myös “Jokaiselle se, minkä hän ansaitsee.” Tämä iskulause oli aikoinaan Buchenwaldin keskitysleirin portilla. Nokia luonnollisesti lopetti mainoslauseen käytön ja pyysi julkisesti anteeksi, mutta virhe oli jo tapahtunut.

Idea kulttuuritietokannan toteuttamisesta syntyi tarpeesta saada kerättyä Nokian matkapuhelimien lokalisoinnissa ja internationalisoinnissa tarvittavaa tietoa yhteen järjestelmään. Osittain tätä tietoa oli jo yrityksen sisällä, mutta hajanaisissa lähteissä, osittain tätä tietoa tarvitsi hankkia talon ulkopuolelta.

Kuten aiemmissa luvuissa on esitetty, lokalisointiin liittyy monia osa-alueita. Siksi käyttäjäryhmääkään ei haluttu rajata pelkästään lokalisointi- tai internationalisointityötä tekeviin käyttäjiin vaan tarkoituksena oli luoda *tietojärjestelmä, johon kerättäisiin maista, kielistä ja kulttuureista sellaista tietoa, josta on hyötyä NMP:n liiketoiminnassa*.

Tämän varsin väljän toimeksiannon perusteella alettiin esitellä ideaa potentiaalisille loppukäyttäjäryhmille ja kerätä heidän mielipiteitään ja ehdotuksiaan. Tämä toteutettiin haastattelemalla käyttäjiä henkilökohtaisesti, lähettämällä sähköpostikyselyitä eri yksiköille sekä kertomalla aiheesta NMP:n lokalisointiyksiköiden yhteisissä tapahtumissa. Ideaa ohjelmistosta esiteltiin useissa tilaisuudessa yhteensä noin 100 potentiaaliselle loppukäyttäjälle ja sen lisäksi lähetettiin noin 50 sähköpostikyselyä.

4.1. Tietosisältö

Eri käyttäjäryhmillä on luonnollisesti erilaisia tarpeita; koska alkuperäinen idea esitettiin vielä hyvin yleisellä tasolla, osassa vastauksista oli pyyntöjä ominaisuuksista, jotka olisivat olleet relevantteja vain pienelle osalle käyttäjiä.

Tarkoituksena oli, että sama kerättävä tieto hyödyttäisi mahdollisimman suurta joukkoa loppukäyttäjiä. Seuraavissa alakohdissa on lueteltu tärkeimpiä asioita, joista käyttäjät halusivat järjestelmään tietoa. Nämä tietosisältövaatimukset, sekä muut loppukäyttäjätoiveet priorisoitiin ja ne olivat perustana vaatimusmäärittelylle [Liite 1: Culture Database - Requirement Specification] ja samalla myös tietokannan perusrakenteelle [Liite 3: Culture Database – Database Specification]. Alakohdissa toiveet on jo osittain jaoteltu tietokannan pää-entiteettien mukaan.

4.1.1. Maakohtainen tieto

Vaikka matkapuhelinta ei lokalisoidakaan maan (vaan kielen tai mieluimmin paikanteen) mukaan niin silti usea vastaaja halusi nimenomaan valtiokohtaista tietoa.

Haluttiin ensinnäkin yleistä tietoa maan väkiluvusta, pääkaupungista, pinta-alasta: siis yleistä tietoa valtiosta. Lisäksi muita tarpeita olivat lakiosaston haluama tieto maan laeista, erityisesti, miten copyright- ja tuotevastuulait vaihtelevat maittain. Markkinointi halusi tietoa valtion taloudellisesta jakaumasta sekä mahdollisesti paikallisen yhdyshenkilön tai ko. valtion asiantuntijan yhteystiedot. Tämä siitä syystä, että osataan arvioida eri puhelinmallien myyntiä eri alueilla. On myös tärkeää tietää, mitä kieliä valtiossa puhutaan ja mitkä ovat virallisia kieliä. Toiveita tuli myös maassa käytettävistä mittayksiköistä ja valuuttasymboleista - nämä katsoin kuuluviksi paikanteeseen, joka muodostuu tässä mallissa maan ja kielen suhteesta. Tätä mallia tosin kritisoin luvussa 2.2, koska pelkkä kieli ja valtio eivät aina riitä paikanteen perusteeksi

Lisäksi saatiin yksittäisiä pyyntöjä seuraavista tiedoista: väestön keski-ikä, valtion valtauskonnot sekä maan bruttokansantuote ja lukutaitoprosentti.

4.1.2. Tietoliikenne

Maiden ja alueiden tietoliikennerakenteesta haluttiin myös tietoa.

Vastaajat olivat kiinnostuneita mm. kiinteiden puhelinliittymien ja matkapuhelinliittymien määrästä ja ennustetusta kasvusta samoin kuin Internet-käyttäjien lukumäärästä. Näitä tietoja käytettäisiin sekä markkinoinnissa että suunniteltaessa erilaisia tuotteita eri maihin.

4.1.3. Yhteydet tietokannan ulkopuolelle

Aika usea käyttäjä ymmärsi, että kaikkea tietoa ei voi - eikä kannata - yrittää kerätä yhteen tietokantaan. Kuitenkin moni päivittäin erilaisista lähteistä tietoa hakeva käyttäjä halusi, että kantaan kerättäisiin linkkejä muihin lähteisiin.

Tämä on myös käytettävyyskysymys: tiedonhakija haluaisi käyttää samaa tuttua työkalua, eikä vaihtaa ohjelmaa useaa kertaa päivässä.

Lähteitä, joihin yhteyksiä haluttiin, olivat paitsi Nokian sisäiset tietokannat myös intranet ja Internet-sivut. Haluttiin jopa erilaisia Internet- ja intranet-linkkikokoelmia aihealueittain järjestettynä.

4.1.4. Tabut

Tietoa haluttiin eri kulttuurien tabuista, kuten alkoholin käyttö arabimaissa, lehmät Intiassa, numero neljä Japanissa.

Samoin eleiden ja symbolien merkitykset eri kulttuureissa kiinnostivat. Tällaisella tiedolla voi olla hyvin tärkeä merkitys, ja varsinkin symbolien väärä käyttö väärissä kulttuureissa voi saada aikaan huomattavaa vahinkoa. Ajatellaan vaikka tilannetta, jossa suomalainen käyttäjä lähettää matkapuhelimellaan tekstiviestin ja haluaa jonkinlaisen palautteen kun viesti on onnistuneesti lähetetty. Lähetyksen onnistumisesta voisi kertoa vaikka kohotettu peukalo tai avokämmen, jossa peukalolla ja etusormella muodostetaan ympyrä ("ok"). Suomalaiselle nämä molemmat eleet merkitsevät onnistumista. Ikävämpi tilanne syntyy, jos ensimmäistä symbolia käytetään Saudi-Arabiaan myytävässä puhelimessa tai jälkimmäistä Brasiliassa: merkitys on nimittäin sama kuin keskisormen näyttäminen suomalaiselle.

4.1.5. Näppäimistöt

Loppukäyttäjät halusivat tietoa, esimerkiksi kuvina, puhelinmallien näppäimistöistä eri kieliversioissa. Tiedosta on hyötyä kun halutaan tietää voidaanko käyttää jo jotain olemassa olevaa näppäimistöä jos laajennetaan puhelimen kielivalikoimaa. Samoin haluttiin tietää näppäimistöjen suunnittelua varten, mitä merkkejä tarvitaan tietyn kielen kirjoittamiseen. Luonnollisesti tämä piti laajentaa koskemaan myös muita syöttömetodeita.
Toivottiin myös viittauksia esimerkiksi Microsoft Windowsin näppäinasetteluihin eri paikanteissa.

4.1.6. Kalenterit

Kaikkialla maailmassa ei käytetä meillä Suomessa käytössä olevaa Gregoriaanista kalenteria, vaan käytössä on useita muitakin kalentereja. Esimerkiksi helmikuun 1. päivä vuonna 2002 Suomessa käytettävän kalenterin mukaan esitetäänkin Kiinalaisessa kalenterissa " -2636-02-1." tarkoittaen vuoden 2636 toisen kuukauden ensimmäistä päivää.
Israelissa käytettävän heprealaisen kalenterin mukaan tuona päivänä on vuoden 3760 yhdeksännen kuukauden seitsemäs päivä, ja Islamilaisen (Hijri) kalenterin mukaan meidän helmikuumme 2002 alkoikin vuonna 622, 19. päivänä 7. kuukautta.
Maa- ja aluekohtaiseen tietoon (ks. kappale 4.1.1) kerättiin tietoa kalenterin käytöstä, sillä samaakin kalenteria käytetään eri tavalla eri maissa. Esimerkiksi viikkonumeroiden käyttö ja vuoden ensimmäisen viikon määrittely vaihtelee. Kalentereista haluttiin myös seuraavia tietoja:

- mihin laskenta perustuu: kuun vai auringon kiertoon
- juhlapäivät
- ero Gregoriaaniseen kalenteriin: mahdollisesti laskenta-algoritmi muuntamiseen
- karkausvuosien ja -päivien käyttö: milloin pidetään, miten lasketaan.
- kuukausien määrä ja nimet eri kielillä.

4.1.7. Värit

Käyttöliittymässä ja käyttöohjeissa väreillä on suuri merkitys. Sen lisäksi, että on otettava huomioon käyttöliittymän selkeys (värin erottuminen taustasta, ym.) on myös muistettava, että eri kulttuureissa eri väreillä on hyvinkin toisistaan poikkeava merkitys. Tietoa haluttiin siitä, missä yhteyksissä mitäkin värejä voi käyttää ja varsinkin haluttiin varmistaa, ettei tässä tehdä pahoja virheitä esimerkiksi loukkaamalla käyttäjän uskontoa.

Coe [1996, 149] erottaa luonnollisen ja kulttuurillisen yhdistämisen (natural mapping, cultural association) toisistaan. Luonnollisella yhdistämisellä tarkoitetaan, että kohteesta käytetään samaa väriä johon silmä on reaalimaailmassa "tottunut": tuli esitetään punaisena, aurinko keltaisena, vesi sinisenä. Kulttuurillisessa yhdistämisessä otetaan huomioon värin kulttuurillinen, sosiaalinen ja emotionaalinen merkitys: punainen edustaa länsimaissa vaaraa tai vihaa, mutta Kiinassa iloa; länsimaissa keltaista väriä käytetään varoituskylteissä, arabimaissa keltainen taas merkitsee iloa.

4.1.8. Ikonit

Hyvin valitut symbolit käyttöliittymässä voivat helpottaa ohjelman käyttöä huomattavasti. Galdon ja Nielsenin [1996, 257] mukaan "graafinen käyttöliittymä, joka sisältää ikoneita ja symboleita tekee ohjelmasta menestyksellisen". Ehkä tämä ei yksin riitä, mutta varmasti grafiikalla on tulevaisuudessa yhä suurempi merkitys myös matkapuhelinten käyttöliittymissä.

Tärkeää on, että käytettävät symbolit ymmärretään kohdekulttuurissa. Vaihtoehtoja on useita – ääripäitä kaksi: joko käytetään kaikissa lokalisoiduissa versioissa samoja, yleisesti tunnettuja, symboleja tai hankitaan tieto kulttuurin erityispiirteistä. Jälkimmäistä vaihtoehtoa helpottaa suunnitellun järjestelmä – Kulttuuritietokanta – toteuttaminen.

Symbolien merkityksestä eri kulttuureissa haluttiin tietoa, koska myös tässä on suuri mahdollisuus tehdä karkeita virheitä. Nike joutui vuonna 1997 vetämään islamilaisen yhteisön painostuksesta markkinoilta 38 000 paria koripallotossuja, koskanäissä oli logo, joka muistutti arabian kielen sanaa "Allah". Muslimeista oli erittäin loukkaavaa, että ”sana” esiintyi kengässä, koska islamilaisessa kulttuurissa jalkaa pidetään epäpyhänä ruumiinosana.

4.1.9. Markkinointitutkimukset

Markkinoista haluttiin tietoa maa- ja aluekohtaisesti jaoteltuna. Haluttiin, että olisi paikka, josta löytyisi suoraan sekä talon sisäisesti tehdyt tutkimukset että viittauksia ja linkkejä Nokian ulkopuolelle. Tietoa haluttiin muun muassa ostovoimasta, puhelinmarkkinoiden kehityksestä ja vallitsevista trendeistä.

4.1.10. Matkustusohjeet

Kyselyssä kävi ilmi, että eri maista haluttiin myös matkustusta varten tarvittavia tietoja ja nämä haluttiin saada yhdestä paikasta matkan suunnittelemista varten. Tällaisia olivat esimerkiksi tieto viisumipakosta, tarvittavista rokotuksista, suurlähetystöjen yhteystiedoista.

Lisäksi toivottiin myös käyttäytymisohjeita, kuten paljonko juomarahaa annetaan, miten tervehditään tai millainen on sopiva liikelahja. Esimerkkinä viimeisestä: Suomessa käytetään lahjana usein puukkoa; monissa Aasian maissa kuitenkin lahjaksi annettu veitsi kuvaa suhteen katkaisemista – eräissä Etelä-Euroopan maissa jopa tappouhkausta - ja on siten erittäin huono valinta liikelahjaksi.

4.1.11. Äänet

Kussakin kulttuurissa on omat äänimaailmansa, eri ihmiset ovat siis tottuneet erilaisiin ääniin. Tästä johtuen myös käyttöliittymässä käytettävien äänien (virheilmoitukset ym.) on oltava kulttuurin mukaisia. Samoin puhelimen soittoääniä suunnittelemista varten haluttiin tietoa äänimaailmasta - mahdollisesti ääninäyttein. Tietojärjestelmää haluttiin myös käyttää apuna markkinointimusiikkia suunniteltaessa.

4.2. Tekniset vaatimukset

Ennen vaatimusmäärittelyn kirjoittamista ei ollut annettu yksityiskohtaisia teknisiä rajoituksia. Tiedossa oli, että järjestelmä asennettaisiin yrityksen intranetiin, se ei saisi vaarantaa tietoturvaa, eikä häiritä muiden ohjelmien käyttöä.

4.2.1. Toimivuus Nokian intranetissä

Vaatimuksena oli järjestelmän toimiminen niin, että se olisi käytössä koko Nokian laajuisesti. Koska tietokannasta ei ylläpidon vuoksi ole mahdollista tehdä useita

paikallisia kopioita työasemille, on ainakin järjestelmän tietokannan oltava tavoitettavissa kaikkialta Nokian lähiverkosta. Järjestelmän tuli myös mahdollisuuksien mukaan toimia ilman erikseen asennettavaa client-ohjelmistoa työasemille. Tämä siksi, että suurin osa käyttäjistä tarvitsi tietoa harvoin, mutta nopeasti: silloin kun tietoa halutaan ei ole aikaa erillisen client-ohjelman asentamiseen.

Edellä mainittujen syiden takia järjestelmä haluttiin asentaa intranetiin siten, että yhteys tietokantaan voidaan muodostaa tavallisella Internet-selaimella.

4.2.2. Tietoturva

Järjestelmään voidaan tallentaa hyvinkin luottamuksellista tietoa, esimerkiksi vielä julkaisemattomien puhelimien näppäimistökarttoja ja kuvia tai sisäisiä markkinointitutkimuksia. Suljetun, talon sisäisen tietoverkon käyttö estää ulkopuolisten käytön pääsyn järjestelmään.

Järjestelmään eivät siis pääse muut kuin ne, jolla on oikeus päästä Nokian intranetiin. Koska intranetiin pääsevät kuitenkin myös esimerkiksi ulkoiset, ns. external-työntekijät ja koska kaikkea tietoa ei haluta antaa kaikkien käyttöön, niin järjestelmässä on oltava ominaisuus saatavan tiedon rajoittamiseen. Tarvitaan ainakin kolme eritasoista käyttäjäryhmää: ryhmä, jolla on lukuoikeus vain rajattuun osaan tietoa; ryhmä, jolla lukuoikeus kaikkeen kannassa olevaan tietoon sekä käyttäjäryhmä, jolla on oikeus ylläpitää järjestelmässä olevaa tietoa.

4.2.3. Häiriöttömyys ja yhteensopivuus

Lähiverkkoon asennettava, mahdollisesti laajassakin käytössä oleva järjestelmä ei saa häiritä muiden verkossa olevien ohjelmien toimintaa, esimerkiksi liian suurella verkkoliikenteellä. Myöskään client-puolella ohjelma ei saa käyttää liikaa työaseman resursseja.

Koska kaikkea tietoa ei välttämättä tallenneta järjestelmän omaan tietokantaan, vaan käytetään olemassa olevia tietovarastoja kuten Lotus Notesia, on rajapinnat tärkeimpien tietojärjestelmien kanssa määriteltävä.

4.3. Käytettävyysvaatimukset

Oletuksena oli, että loppukäyttäjä ei käytä järjestelmää päivittäin ja, että kun tietoa haetaan, sitä tarvitaan nopeasti.

Vaatimuksesta seuraa, että client-ohjelmiston asentamiseen ja käytön opettelemiseen ei saisi kulua juurikaan aikaa. Tämän takia haluttiin, että järjestelmää voidaan käyttää yleisimmillä, käyttäjille entuudestaan tutuilla, Internet-selaimilla. Lisäksi tiedonhaun tulisi muistuttaa tavanomaista navigointia www-sivuilla.

Tässä luvussa perustelin, miksi kulttuuritietokantajärjestelmää tarvitaan lokalisoinnin apuna sekä kerroin millaisia vaatimuksia käyttäjillä oli järjestelmälle.

5. Prototyyppiprojekti

Tutkimusongelmana oli, onko järkevää rakentaa edellisessä luvussa kerrottujen reunaehtojen puitteissa tällainen tietojärjestelmä, johon on tarkoitus tallentaa hyvin heterogeenista tietoa ja joka palvelisi hyvin erityyppisten käyttäjien tarpeita. Järkevyys tarkoittaa tässä saavutetun hyödyn suhdetta käytettyihin voimavaroihin. Mahdolliset vastaukset tutkimusongelmaan olivat:

- Tällainen järjestelmä kannattaa toteuttaa vaatimusmäärittelyn mukaisesti.
- Tällainen järjestelmä kannattaa toteuttaa, mutta vaatimusmäärittelyä pitää muuttaa.
- Järjestelmää ei kannata toteuttaa

Vastauksen selvittämiseksi tutkimusongelma on jaettu osaongelmiin, joita ovat: tekninen toteutettavuus, tiedon ylläpito, ohjelmiston käytettävyys ja toteutukseen tarvittavat resurssit

5.1. Prototyyppiprojektin vaatimukset

Ongelmaa lähdettiin ratkaisemaan toteuttamalla ohjelmistosta prototyyppi harjoitustyönä Tampereen yliopiston tietojenkäsittelytieteiden laitoksen kursseilla "Ohjelmistoprojektin johtaminen" ja "Projektityö" lukuvuonna 1999–2000.
Lisäksi ohjelmiston määrittelyyn, tietojen keräämiseen sekä prototyypin testaukseen osallistui kollegani Taija Björklund. Prototyyppiprojektin – työnimeltään Culture Database - läpiviemistä kuvataan yksityiskohtaisemmin projektisuunnitelmassa. [Liite 2: Culture Database – Project Plan]. Tässä kerrotaan tarkemmin, millaisilla resursseilla ja millaisella aikataululla projekti suunniteltiin toteutettavaksi.
Projektin ohjausryhmään kuuluivat kirjoittajan lisäksi Taija Björklund sekä projektipäälliköt Markku Hanhiniemi ja Kyekyeku Okopu-Pong. Hanhiniemi ja Okopu-Pong suorittivat kurssia Projektityön johtaminen. Varsinaisen toteutuksen teki viiden hengen opiskelijaryhmä Projektityö-kurssilta.

Projektisuunnitelman valmistuttua marraskuussa 1999, projekti jatkui edellisessä luvussa kuvattujen käyttäjävaatimusten keräämisellä vaatimusmäärittely-dokumenttiin [Liite 1: Culture Database - Requirements Specification]. Tässä vaiheessa dokumenttiin kerättiin mukaan kaikki vaatimukset. Tämä siksi, että kyseessä oli nimenomaan prototyyppiprojekti ja haluttiin tutkia mitä on mahdollista toteuttaa ja mitä ei. Lisäksi tarkoituksena oli, että tuotettua ohjelmakoodia olisi mahdollista käyttää varsinaisessa, myöhemmin mahdollisesti laajempana, toteutettavassa, ohjelmistossa – siksi haluttiin, että ohjelmisto olisi laajennettavissa. Käytettävien olevien resurssien takia (5 opiskelijaa suorittamassa viiden opintoviikon kurssia) kaikkea ei voinut olettaa toteutettavan kokonaan, joten vaatimusmäärittelyssä otettiin käyttöön kaksiulotteinen priorisointijärjestelmä, jossa kullekin vaatimukselle annettiin tärkeysjärjestys erikseen suhteessa tietokantaan, toiminnallisuuteen ja sisältöön (Kuva 1). Prioriteetti sai arvoja seuraavia arvoja: 1 – välttämätön järjestelmän kannalta, 2 - välttämätön, voidaan jättää pois vain, jos toteuttaminen vaarantaa prioriteetilla 1 olevien vaatimusten toteuttamisen, 3 – toteutetaan, mikäli mahdollista aikataulun puitteissa ja 4 - ei toteuteta prototyypissä.

Requirement #. <Requirement name>

Priority (database)	<Priority>
Priority (functionality)	<Priority>

Priority (content) <Priority>
Description: <Description>

Kuva 1: vaatimusten priorisointi

Kaksiulotteinen järjestelmä kehitettiin siksi, että prototyypistä haluttiin ensinnäkin tehdä laajennettava ja käyttää syntynyttä ohjelmistoa mahdollisesti pohjana jatkokehitykselle, toiseksi prototyypillä haluttiin kokeilla erilaisten teknisten ratkaisujen toimivuutta ja kolmanneksi koska haluttiin demota loppukäyttäjille ohjelmistoa. Siksi esimerkiksi vaatimus #1 "Country facts" sai tietokantaprioriteetikseen 1, koska maa- ja aluekohtainen tieto oli olennaista tietokannan rakenteen kannalta, mutta sisältöprioriteetikseen 3, koska prototyypin kannalta ei ollut olennaista saada esimerkiksi ajantasaista tietoa kaikkien maiden väkiluvusta.

5.2. Tekninen toteutettavuus

Teknisen toteutettavuuden tutkimisessa tarkasteltiin, onko ohjelmisto teknisesti mahdollista toteuttaa vaatimusmäärittelyn mukaisesti. Ja jos se on toteutettavissa, niin mitä laitteisto- ja ohjelmistoinvestointeja tarvitaan, kuinka paljon resursseja järjestelmän rakentaminen ja ylläpito vaatii ja mitä vaikutuksia tällä tietojärjestelmällä on muihin intranetin alijärjestelmiin.
Tietojärjestelmän tekninen toteutus on kuvattu Culture Database-ohjelmiston tietokantakuvauksessa, [Liite 3: Culture Database – Database Specification] sekä teknisessä määrittelyssä [Liite 4: Culture Database – Technical Specification]

5.2.1. Laitteisto

Millaisiin laiteympäristöihin ohjelmiston voi toteuttaa? Mitkä ovat laitteiston minimivaatimukset? Vaatimuksena oli ensinnäkin, ettei intranet-järjestelmää tarvitse muuttaa ja toiseksi, ettei loppukäyttäjän tarvitse päivittää työasemansa ohjelmistoja, eikä laitteistoa vaan, että pelkkä yhteensopivan selaimen käyttäminen riittää.

5.2.2. Ohjelmisto

Millaista ohjelmistoarkkitehtuuria kannattaa käyttää, mitä muita ohjelmistoja tarvitaan tueksi? Vaatimuksena oli tässäkin, että tarvitaan mahdollisimman vähän kolmannen osapuolen ohjelmistoja intranetiin asennettavaksi.

5.2.3. Yhteensopivuus olemassa olevien ohjelmistojen kanssa

Miten ohjelmiston toteuttaminen ja asentaminen yrityksen intranetiin vaikuttaa muihin käytössä oleviin ohjelmistoihin ja voiko näitä mahdollisesti käyttää hyödyksi?
Vaatimuksena oli, että käytetään mahdollisimman paljon intranetissä jo olevaa tietoa, joka löytyy useimmin joko suoraan tiedostojärjestelmästä tai tietokantaohjelmistojen takaa. Tämä redundantin tiedon välttämiseksi ja tiedon ylläpidon helpottamiseksi.

5.3. Sisältö ja tiedon ylläpito

Osa tiedosta on luonteeltaan staattista, esimerkiksi suomen kielen aakkosjärjestys; osa taas hyvin dynaamista kuten valtioiden bruttokansantuotteet. Tiedon on oltava ajan tasalla ja sitä on oltava helppo ylläpitää.
Onko tietoa mahdollista ostaa ulkopuolelta?

5.3.1. Tiedon luotettavuus

On tutkittava keinoja varmistaa tiedon ajantasaisuus. Jos kannassa on linkkejä intranetin ulkopuolelle, käyttäjän on saatava selville tiedon lähde.
Esimerkiksi YK:n sivuilta haetun tiedon voi olettaa olevan luotettavampaa kuin tuntemattoman yksityishenkilön omasta tietokannasta saadun. Kastemaa [2000] määrittelee tietosisällölle viisi kriteeriä:

- Tarkkuus
- Asiantuntevuus
- Objektiivisuus
- Voimassaolo

- Kattavuus
- Kuinka syvällistä aineisto on?

5.3.2. Tietoturva

Osa tiedosta on sellaista, että sen saantia voidaan joutua rajoittamaan myös intranetin sisällä. On myös ratkaistava kenellä on oikeus päivittää tietoa sekä voitava antaa käyttäjille eritasoisia oikeuksia tietoon.

5.4. Ohjelmiston käytettävyys

Loppukäyttäjien valmiudet käyttää tietokoneohjelmistoja vaihtelevat. Oletettiin, että kuitenkin jokaisella on kohtalaisen paljon kokemusta tietokoneiden ja Internetin käytöstä, mutta kaikki eivät ole välttämättä ns. tietotekniikan ammattilaisia.
Henkilö saattaa hakea tietoa kannasta satunnaisesti esimerkiksi 2-3 kertaa vuodessa tarkistaakseen vaikkapa matkakohteenaan olevan maan juhlapäivät. Tämän vuoksi on minimoitava ohjelmiston käytön valmisteluun ja opetteluun tarvittava aika tekemällä ohjelman käyttöliittymästä mahdollisimman intuitiivinen. Suunniteltu käyttöliittymä on kuvattu Culture Database – UI-määrittelyssä, [Liite 5: Culture Database – User Interface Specification]

5.5. Vaadittavat resurssit

Kun on päätetty ohjelmiston tietosisällön rajaus ja tutkittu erilaisia teknisiä ratkaisuja, arvioidaan tarvittava työmäärä sekä laitteisto- ja ohjelmistokustannukset. Arvioinnissa on otettava huomioon myös kustannukset, joita voi syntyä, jos ohjelmistoa ei toteuteta. Katso Nike-esimerkki luvussa 4.1.4, Tabut.

5.6. Toteutus

Prototyypin vaatimusmäärittelyä varten ajatusta esiteltiin loppukäyttäjäryhmälle ja kerättiin heidän mielipiteitään ja ehdotuksiaan. Tämä toteutettiin haastattelemalla

käyttäjiä henkilökohtaisesti, lähettämällä sähköpostikysely eri yksiköille sekä kertomalla aiheesta NMP:n lokalisointiyksiköiden yhteisissä tapahtumissa.
Prototyyppiin tallennettavaa tietoa jouduttiin rajoittamaan vielä enemmän kuin lopulliseen ohjelmistoon. Toisaalta oli mahdollisuus, että osia prototyypistä voitaisiin käyttää varsinaisessa toteutuksessa. Tämän takia yksi tärkeimmistä vaatimuksista jo prototyypille oli ohjelmiston laajennettavuus. Prototyyppiprojekti vietiin läpi projektisuunnitelman mukaisesti [Liite 2: Culture Database – Project Plan]. Ohjelmisto toteutettiin aikataulussa, mutta joitakin piirteitä jouduttiin karsimaan – kuten alun perin oli arvioitukin, tämän vuoksi vaatimusmäärittelyssä oli priorisointijärjestelmä.
Tässä luvussa kerroin, miten järjestelmää varten tehtiin prototyyppijärjestelmä, miten siihen kerättiin vaatimuksia, määrittelin kriteerit prototyypin onnistumiselle ja kerroin projektin läpiviennistä.

6. Prototyyppiprojektin evaluointi

Edellisessä luvussa kerroin järjestelmää varten suunnitellusta ja toteutetusta prototyypistä, tässä arvioin, kuinka prototyyppi onnistui ja mitä muutoksia siihen piti tehdä, että saataisiin aikaan varsinainen ohjelmisto – työnimeltään Pandora.

Projektia seurattiin tiiviisti toteutusvaiheen aikana viikkoraportein, sekä epäsäännöllisin (noin kerran kahdessa viikossa pidetyin) ohjausryhmän palaverein, ja kirjattiin ylös esiin tulleet ongelmat. Prototyypin valmistuttua arvioitiin sen toimivuutta, toteutusta ja siihen käytettyä työaikaa. Projekti valmistui aikataulussaan vaatimusmäärittelyn mukaisesti – tosin joitakin alhaisella prioriteetilla olleita vaatimuksia jätettiin pois.
Prototyyppi asennettiin NMP:n intranetiin aikataulun mukaisesti ja tutkittiin sen yhteensopivuutta olemassa olevien ohjelmistojen kanssa sekä korjattiin virheitä ja puutteita.
Ensimmäisessä evaluointivaiheessa rajattu joukko, noin 20 loppukäyttäjää, kokeili ohjelmistoa ja antoi siitä palautetta. Palautteen perusteella päätettiin, voiko ohjelmistoa jakaa suuremmalle joukolle sellaisenaan vai onko tarvetta muutoksiin.

Toisessa evaluointivaiheessa suurehko joukko (n 50) käyttäjiä käytti ohjelmistoa. Muutaman viikon pituisen käyttöjakson jälkeen heiltä kerättiin sähköpostitse palautetta ohjelmiston toimivuudesta.
Saadun palautteen sekä prototyypin kehityksestä saadun kokemuksen perusteella arvioidaan alkuperäistä tutkimusongelmaa eli *onko järkevää rakentaa tällainen tietokantajärjestelmä*? Seuraavissa luvuissa sekä evaluoidaan prototyyppiä että määritellään mitä muutoksia prototyyppiin tarvitaan. Prototyyppiohjelmistoon viitataan sen työnimellä *Culture Database* ja vastaavasti uuteen prototyypin perusteella tehtävään ohjelmistoon nimellä *Pandora*. Perusarkkitehtuuriin ei tarvinnut tehdä kovin paljon muutoksia. Tärkeimmät olivat tietokannan muuttaminen uutta tietosisältöä vastaavaksi sekä ohjelmistoarkkitehtuurin muuttaminen tehokkaammaksi ja muokattavammaksi.

6.1. Tekninen toteutettavuus

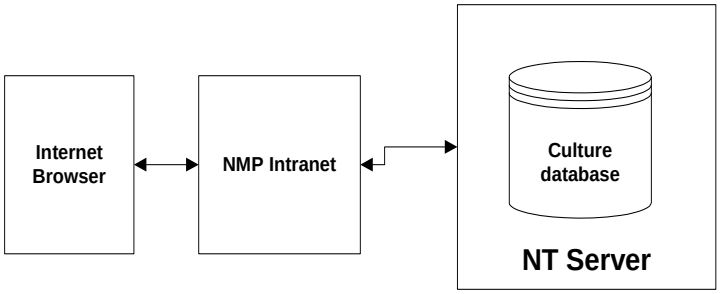
Teknisesti ohjelmisto pystyttiin toteuttamaan vaatimusmäärittelyn mukaisesti. Tosin projektin alussa jo oli selvää, että on mahdollista, ettei kaikkia vaatimusmäärittelyssä annettuja ominaisuuksia ehdi toteuttaa vaan ominaisuudet toteutetaan annettujen prioriteettien mukaan.
Teknisestä näkökulmasta tärkeimmät poisjätetyt ominaisuudet olivat eri oikeudet eri käyttäjäryhmille ja ns. Advanced Search –toiminto. Eli nämäkin olisivat olleet toteutettavissa, mutta jätettiin aikapulan vuoksi pois.

6.1.1. Laitteisto

Järjestelmä toteutettiin niin, ettei varsinaista client-ohjelmistoa tarvita vaan loppukäyttäjä voi ottaa yhteyden tietojärjestelmään tietyt kriteerit täyttävällä internet-selaimella. Laitteistoarkkitehtuuri on kuvattu yksinkertaisimmillaan kuvassa 1.

Virallisesti tuettu selain oli Netscape Navigator 4.0 tai uudempi, mutta ohjelmisto suunniteltiin niin, ettei se käytä selaimen erityispiirteitä.

Kuva 1, laitteistoarkkitehtuuri



Työasemissa oli valmiina ko. selain tai vastaava (tai ainakin ne olivat teholtaan riittäviä käyttämään sopivia selaimia), joten vaatimus siitä, ettei työasemien laitteistoa tarvitse päivittää toteutui.

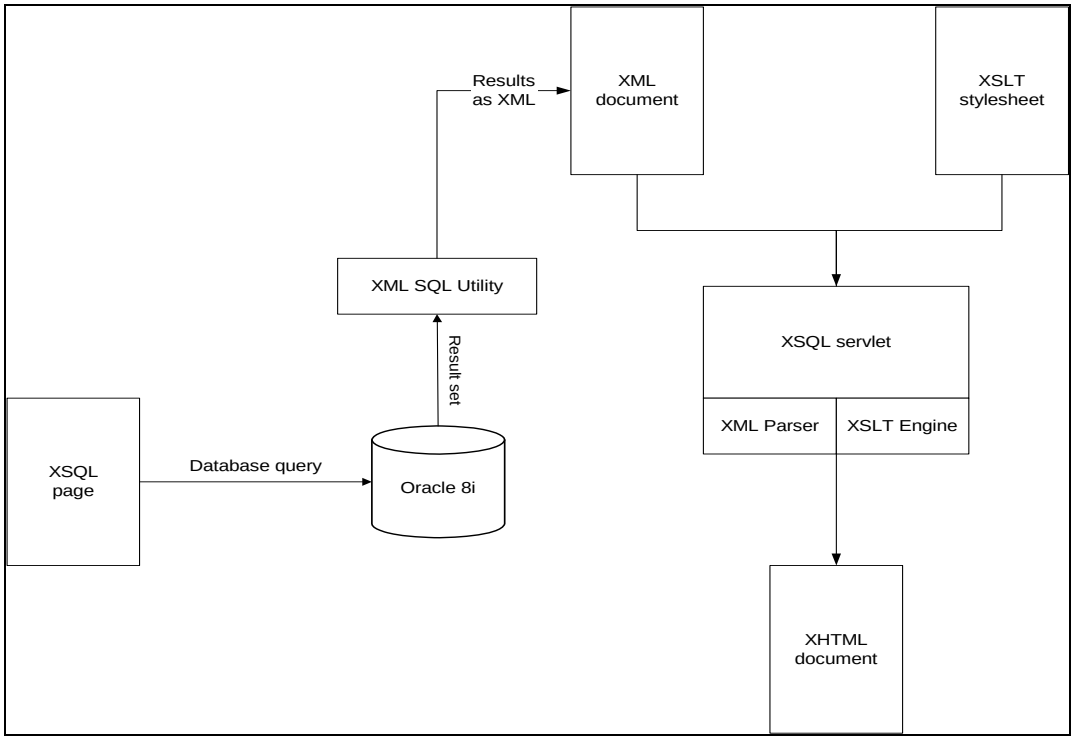
Varsinainen tietokanta pystytettiin erilliselle palvelimelle, johon oli asennettu Oracle 8i –tietokantaohjelmisto. Yhteyden tietokannan ja clientin välillä hoiti Javalla ohjelmoitu välitason ohjelmisto, middleware. Tämä ohjelmisto sijaitsi prototyyppivaiheessa käytännön syistä samalla palvelimella kuin varsinainen tietokantakin, mutta alun perin arkkitehtuuri suunniteltiin niin, että tietokanta ja middleware-ohjelmisto voidaan asentaa myös eri palvelimille, tätä myös kokeiltiin käytännössä. Intranet-järjestelmää ei siis tarvinnut muuttaa millään tavalla, pelkästään lisätä siihen yksi palvelin: tämäkin ehto siis toteutui.

6.1.2. Ohjelmisto

Ohjelmistoarkkitehtuuri koostui kolmesta erillisestä osasta: käyttöliittymästä, eli joukosta HTML-sivuja, jotka näytetään loppukäyttäjän selaimessa, välitason ohjelmistosta, joka toimii rajapintana käyttöliittymän ja tietokantajärjestelmän välillä, sekä tietokantapalvelimella sijaitsevasta tietokantaohjelmistosta.

Kun käyttäjä tekee kyselyn tietokantaan, kysely lähetetään välitason ohjelmistolle, eli erillisellä WWW-palvelimella toimivalle Java-servletille. , joka välittää kyselyn edelleen toiselle sopivalle Java-luokalle, joka taas muodostaa tietokantapalvelimelle lähetettävän SQL-kyselyn. Tietokantapalvelimelta vastaus palautuu samalle luokalle, joka muodostaa vastauksesta HTML-sivun ja palauttaa sen käyttöliittymälle. Ohjelmistoarkkitehtuuri on kuvattu yksityiskohtaisemmin Culture Databasen teknisessä määrittelyssä [Liite 4]

Ratkaisu oli toimiva, mutta sitä haluttiin lopulliseen versioon Pandoraan muuttaa tehokkuuden sekä käyttöliittymän muokattavuuden takia. Merkittävin ero arkkitehtuurissa on, että suuri osa siitä, mikä Culture Database –ohjelmistossa hoidettiin väliohjelmiston servlettien avulla hoidetaan jatkossa käyttämällä XSQL-sivuja ja XSLT-tyylimäärittelyjä.



Kuva 2, käyttöliittymän sivun muodostuminen tietokantakyselyn tuloksena (Pandora)

Kuten kuvassa 2 esitetään, käyttöliittymä muodostetaan ohjelmallisesti käyttämällä Oraclen XSQL-sivuja ja XSL-muunnoksia. XSQL-sivu on XML-dokumentti, johon on sisällytetty SQL-kysely. Oraclen XSQL-virtuaalikone muuntaa kyselyn tuloksen XML-dokumentiksi. Tämän dokumentin taas Oraclen XSLT-prosessori muokkaa esitettäväksi HTML 4.01 -sivuksi käyttäen apunaan XSL-tyylimääritystä. Yksityiskohtaisemmin tämä on esitetty Pandoran teknisessä määrittelyssä [Liite 9].

Etuna on, että se mikä aiemmin tehtiin ohjelmallisesti middlewaren java-luokissa, voidaan nyt toteuttaa XSLT-lauseilla. Tämä vähentää merkittävästi ohjelmointityön määrää ja nyt käyttöliittymää voidaan muuttaa, tarvittaessa huomattavastikin, tekemättä riviäkään uutta ohjelmakoodia tai edes ajamatta järjestelmää välillä alas.
Ohjelmistoarkkitehtuuri siis toimi sellaisenaankin, mutta sitä piti hieman muuttaa. Intranetiin ei sinänsä tarvinnut lisätä muita ohjelmistoja kuin WWW- ja Oracle-palvelimet, jotka toimivat omilla erillisillä työasemillaan.

6.1.3. Yhteensopivuus olemassa olevien ohjelmistojen kanssa

Järjestelmä asennettiin kehitysvaiheen jälkeen NMP:n intranetiin, eikä se aiheuttanut ongelmia muiden ohjelmistojen kanssa.
Edellisessä luvuissa kuvatun laitteisto- ja ohjelmistoarkkitehtuurien ansiosta ainoa mahdollinen häiriötekijä olisi voinut olla liian suuri tietoliikenteen määrä Clientin, middlewaren ja tietokantapalvelimen välillä. Tätä testattiin sekä empiirisesti pyytämällä useita käyttäjiä tekemään hakuja kannasta samanaikaisesti että mittaamalla Oraclen omilla apuohjelmilla tietoliikenteen määrää. Kuorma havaittiin lähes olemattomaksi suhteessa koko intranetissä kulkevaan tietoon. Vaatimus, ettei ohjelmisto häiritse muita intranetissä olevia ohjelmistoja siis toteutui.
Liittymiä muihin intranetin tietokantaohjelmistoihin ei toteutettu. Liittyminen esimerkiksi Lotus Notesiin huomattiin teknisesti hyvin vaativaksi. Intranetin WWW-sivuilla oleviin tietoihin voidaan tehdä käyttöliittymään suora linkki tarvittaessa, jolloin käyttöliittymä voisi toimia eräänlaisena porttaalina. Suoraan tiedostojärjestelmässä oleva tieto siirrettiin tietokantaan puolimanuaalisesti. Tästä tarkemmin dokumentissa Pandora – Content creation plan [Liite 6] Voidaan siis sanoa, että vaatimus liittymästä olemassa oleviin järjestelmiin ja olemassa olevan tiedon käyttäminen ei täysin toteutunut.

6.2. Sisältö ja tiedon ylläpito.

Tiedon ylläpito oli suurun haaste, koska vaatimuksissa haluttiin järjestelmään tietoa, joka on luonteeltaan hyvin dynaamista. Käyttäjät halusivat kuitenkin tarkistaa esimerkiksi maan väkiluvun mieluummin suoraan Internetistä. Erillistä ylläpitotyökalua ei toteutettu, joten ainoa mahdollisuus päivittää tietokannan tietoa oli sisällön keräämisen yhteydessä luotujen eräajotyökalujen käyttö [Liite 9 Pandora- Content Creation –plan]

Lisäksi huomattiin, että alkuperäisen tietokantamäärittelyn [Liite 3, Culture Database - Database Specification] mukainen tieto on osittain hyvin nopeasti muuttuvaa. Esimerkkinä lait, väkiluvut, bruttokansantuotteet. Pelkästään näiden tietojen kerääminen vaatisi usean henkilön täysipäiväisen työpanoksen. Lisäksi tämäkin tieto saadaan helpommin valmiista lähteistä.

Tietokantaa muutettiin siten, että keskityttiin enemmän staattiseen tai harvoin muuttuvaan tietoon [Liite 8, Pandora database specification] kuten esimerkiksi paikanteiden tiedot, Unicode-merkistö tai mitä merkkejä tarvitaan tietyn kielen esittämiseen. Tällaista tietoa ei tarvitse ylläpitää päivittäin vaan ajamalla ohjelmistoon päivityksiä eräajona tarvittaessa.

Vaatimus tiedon helposta ylläpidosta ei toteutunut, vaan itse järjestelmää muutettiin niin, että tietoa ei tarvitse ylläpitää jatkuvasti. Tietosisältöä muutettiin siis dynaamisesta staattisempaan.

6.2.1. Tiedon luotettavuus

Käyttäjät halusivat olla varmoja tiedon ajantasaisuudesta. Ohjelmaan haluttiin näkyviin mistä lähteestä tieto on peräisin ja milloin se on päivitetty. Järjestelmä ei tällaisen tiedon tallentamista tukenut. Tämä olikin yksi suurimmista puutteista sekä tietokantaratkaisussa että käyttöliittymässä. Tietokantaan pitää saada eräänlainen metatieto siitä, miten ja milloin tieto järjestelmään on tuotu. Tähän tietoon on oltava myös pääsy käyttöliittymästä.

6.2.2. Tietoturva

Tietoturva liittyy myös tiedon luotettavuuteen: intranetin sisällä pääsyä ohjelmistoon ei haluttu rajoittaa ja koska erilaisia käyttäjätasoja ei ollut niin käyttäjällä oli selaimen kautta vain lukuoikeus tietoihin.
Tietoturva oli ratkaistu siten, että vain muutamalla henkilöllä oli oikeus päivittää tietoa erityisten aputyökalujen kautta suoraan tietokantaan. Jatkossa järjestelmässä pitää olla erilaisia käyttäjätasoja eri oikeuksin ja samoin tietojen päivittämisen pitää tehdä automaattisesti merkintä metadataan. Kts 6.2.1 Tiedon luotettavuus.

6.3. Ohjelmiston käytettävyys

Ohjelmiston koekäyttäjät olivat Nokia Mobile Phonesin työntekijöitä: lokalisointikoordinaattoreita tai teknisiä kirjoittajia. Heillä oli keskimäärin usean vuoden käyttökokemus tietokoneista ja www-selaimista.
Saadun palautteen perusteella ohjelmisto oli erittäin helppokäyttöinen, koska se muistutti tavallista www-sivua tai hakukonetta, joita loppukäyttäjät käyttävät työssään päivittäin. Ohjelmisto sisälsi myös Online-helpin, mutta tätä ei käytetty paljoakaan. Eniten kysymyksiä aiheutti selainten asetusten muuttaminen esimerkiksi vieraiden fonttien tukemista varten.
Kokonaisuutena järjestelmää voidaan pitää helppokäyttöisenä ja käyttöliittymää intuitiivisena.

6.4. Vaaditut resurssit

Kulttuuritietokantaa oli toteuttamassa 5 Tampereen yliopiston tietojenkäsittelytieteen opiskelijaa yhden lukukauden aikana. Työmäärä oli noin [Liite 2: Culture Database – Project Plan] 125 henkilötyöpäivää.
Järjestelmän muuttaminen Pandora-ohjelmistoksi vaati [Liite 7: Pandora – Project Plan] 183 henkilötyöpäivää, joista 50 koostui tietojen keräämisestä ja tallentamisesta.

7. Yhteenveto ja loppupäätelmät

Prototyyppiprojektista Culture Database ja sen jatkoprojektista Pandora saatujen kokemusten perusteella voidaan päätellä, että tällaista kulttuuritietokantajärjestelmää tarvitaan. Järjestelmään tarvitaan kuitenkin muutoksia. Tärkeimmät muutokset ovat talletettavan tiedon muuttaminen dynaamisesta staattisempaan suuntaan päivittämistarpeen pienentämiseksi, metatiedon lisääminen tiedon alkuperän selvittämiseksi ja erilaisten käyttäjäroolien lisääminen tiedon päivittämisen helpottamiseksi.

7.1. Loppupäätelmät

Tässä tutkielmassa on kuvattu esimerkkejä siitä mitä voi tapahtua, jos kulttuurillisia eroja ei oteta huomioon. Lisäksi evaluointivaiheessa huomattiin, että ohjelmistolle on tarvetta - tosin tietyin muutoksin.

Nopeasti muuttuvaa ja ei-teknistä tietoa on vaikeampaa tallentaa ja varsinkin päivittää järjestelmään kuin staattista ja teknistä tietoa. Esimerkiksi kirjoitusjärjestelmien tiedot: suomen kielen kirjoittamiseen tarvittava merkistö ja näiden merkkien esitys Unicode-järjestelmässä on helpommin tallennettavaa ja harvemmin muuttuvaa kuin vaikka vihreän värin merkitys New Yorkissa asuville 15-25 –vuotiaille miehille.

Ohjelmiston tietosisältöä on siis entisestään rajattava. Lisäksi ns. metadata on lisättävä järjestelmään kertomaan milloin ja kenen toimesta tietoa on päivitetty. Tiedon päivitystä varten on järjestelmään luotava myös erilaisia käyttäjätasoja.

7.2. Jatkosuunnitelmat

Vuoden 2003 lopussa ohjelmisto oli tällaisenaan käytössä. Vaatimuksia järjestelmän parantamiseksi oli kerätty [Liite 10, Pandora – Phase2 Feature List], mutta konkreettista suunnitelmaa toteuttamiseksi ei ollut. Tilanne huhtikuussa 2006 on (ainakin kirjoittajan tiedon mukaan) muuttumaton.

7.3. Jatkotutkimukset ja sovellukset

Vaikka en olekaan löytänyt viittauksia tällaisen järjestelmän olemassaolosta niin uskon silti yrityksillä tällaisia olevan käytössään. Tieteellisestä näkökulmasta olisi mielenkiintoista laajentaa tätä tutkimusta mm. uskontoihin ja samalla tutkia mahdollisuuksia miten muuttuvaa dymaamista tietoa voitaisiin ylläpitää samassa järjestelmässä esimerkiksi automaattisten hakukoneiden avulla.

Toisaalta tässä voisi olla mahdollisuus toteuttaa myös (kaupallinen) sovellus, jota käyttäen yritys tai yksityinen henkilö saisi haluamansa tietopaketin esimerkiksi alueesta, jolle aikoo matkustaa tai kielestä, josta on kiinnostunut. Jokatapauksessa aiheessa on mielenkiintoisia mahdollisuuksia sekä tieteelliselle tutkimukselle että kaupallisille sovelluksille.

Viiteluettelo

[Atk-sanakirja, 1997] *Atk-sanakirja (7. uusittu painos)*. Suomen Atk-kustannus, 1994.

[Coe, 1996] Marlana Coe, *Human factors for technical communications*. John Wiley & Sons, 1996.

[Deitch and Czarnecki, 2001] Andrew Deitch, David Czarnecki, *Java internatiolization*. O'Reilly & Associates. Inc, 2001.

[del Galdo and Nielsen, 1996] Elisa M. del Galdo and Jakob Nielsen (ed.), *International User Interfaces*. John Wiley & Sons, 1996.

[del Galdo, 1990] Jakob Nielsen (ed.), *Designing user interfaces for international use*. Elsevier Science Publisher B.V., 1990.

[Esselink, 1998] Bert Esselink, *A Practical Guide to Software Localization*. John Benjamis Publishing Co, 1998.

[Fernandes, 1995] Tony Fernandes, *Global interface design: A Guide to sesigning international user interfaces.* Academic Press, 1995.

[Kano, 1995] Nadine Kano, *Developing International Software for Windows 95 and Windows NT*. Microsoft Press, 1995.

[Kasvio ja Nieminen, 1998], *Globalisaatio, työpaikkakilpailu ja Suomi – uuteen kansalliseen strategiaan?* Tietoyhteiskunnan tutkimuskeskus, työraportti 4/1998.

[ISO, 1999c] *ISO/IEC 8859-15:1999 Information Technology – 8-bit single-byte coded graphic character sets – Part 15: Latin alphabet No. 9*. International Organization for Standardization, 1999.

[Käpyaho, 2001] Jere Käpyaho, *Internationalisation in Operating Systems for Handheld Devices,* University of Tampere Department of Computer and Information Sciences, Master's thesis, December 2001.

[Luong et al., 1995] Tuoc V. Luong, James S.H.Lok, David J. Taylor, Kevin Driscoll, *Internationalisation: developing software for Global Markets*. John Wiley & Sons, Inc., 1995.

[Rudisill et al., 1996] Marianne Rudisill, Clayton Lewis, Peter B. Polson, Timothy D. McKay, *Human-computer interface design*. Morgan Kaufmaan Publishers, Inc, 1996.

[Ott, 1999] Cristopher Ott, *Global solutions for multilingual*. John Wiley & Sons, Inc., 1992.

[Russo, 1993] Patricia Russo, *How fluent is your interface? Designing for International.* In Proceedings of INTERCHI '93, Amsterdam 1993.

[Taylor, 1992] Dave Taylor, *Global Software: developing applications for the international market*. Springer-Verlag, 1992.

[The Unicode Consortium, 2000] The Unicode Consortium, *The Unicode Standard Version 3.0*. Addison-Wesley, 2000. Saatavana osoitteesta: http://www.unicode.org/unicode/standard/standard.html (tarkastettu 2006-04-18).

[Turnbull, 1999] Steve Turnbull, Alphabet soup: the internationalization of Linux, part 1. *Linux Journal* **59** (March 1999), 30-38. Saatavana osoitteesta: http://turnbull.sk.tsukuba.ac.jp/Tools/I18N/LJ-I18N.html (tarkastettu 2006-04-18).

[CIA, 2003] CIA, the World Factbook. Saatavana osoitteesta: http://www.cia.gov/cia/publications/factbook/index.html (tarkastettu 2006-04-18).

[Tuthill and Smallberg, 1997] Bill Tuthill, David Smallberg, *Creating worldwide software*. Sun Microsystems, Inc., 1997.

[Yeo, 1996] Alvin Yeo, *World-Wide CHI: Cultural User Interfaces, A Silver Lining in Cultural Diversity.* SIGCHI Bulletin Vol.28 No3, July 1996.

[Kastemaa, 2000] Heikki Kastemaa, Verkkoartikkeli: Verkkotekstien validiteetista: http://www.kaapeli.fi/sanoma-open/toimitetut/kastemaa.html (tarkastettu 2006-04-18).

Liitteet

NOKIA

LIITE 1: Culture Database Requirements Specification

Version: 1.1

State: Approved

Author: Kyekyeku Opoku-Pong, Pekka Mäkiaho

Version history:

Version	Date	Description
0.1	1999-11-25	First draft version
0.2	1999-11-30	Modified with the comments of the steering group
0.3	1999-01-11	Modified according to comments from Taija Björklund and Pekka Mäkiaho
0.4	1999-01-25	Modified according to comments from Taija Björklund and Pekka Mäkiaho
0.5	2000-02-21	Modified with the comments from the project meeting on 03.01.2000
1.0	2000-02-22	Approved by the steering group
1.1	2003-04-15	Nokia confidential information removed

INTRODUCTION

The purpose of this document is to specify the requirements for Culture Database system. Culture Database is a system for storing and retrieving information on different countries, different cultures, and different languages.

This document will serve as the requirements specification for the prototype of the Culture Database system. Students at the Department of Computer Science, University of Tampere who will implement a prototype of Culture Database will use this document as the basis of their design specifications. It can also be updated and used as the Requirements Specification of the final Culture Database system

GLOSSARY

Abbreviation	Name
CDB	Culture Database
GNP	Gross National Product
NMP	Nokia Mobile Phones

PRODUCT CONSTRAINTS

Constraints are factors that apply to the whole of the product. Culture Database like any other system must be built within the stated constraints. Constraints drive the project and shape the product

The Purpose of the Product

The project will develop a prototype of the Culture Database. Culture Database will enable users to access information on, for instance, languages, colours, taboos, character sets, religions.

Client, Customer and Other Stakeholders

The customer of the Culture Database project is Nokia Mobile Phones. Multiple departments of Nokia Mobile Phones including Documentation, Localisation, Marketing and the legal department will use Culture Database. Two employees of NMP will write their thesis on the development and the cultural aspects of Culture Database system. Markku Hanhiniemi manages this project. Kyekyeku Opoku-Pong sees to Quality issues. These two are managing the project for a course in Software Project Management at the University of Tampere where the prototype is being developed.

Users of the Product

The users of Culture Database will be employees of NMP. Users will have different access rights. Certain kinds of information should have restricted access, because Culture Database will be accessed from NMP Intranet and some subcontractors can access the NMP Intranet so

sensitive information in the database should be protected. This kind of information should be accessible only to certain users.

Requirements Constraints

Culture Database must run in the Windows NT 4.0 or later and Netscape Navigator 4.0 or later. It will be accessed from NMP Intranet and will use ORACLE database system for its database.

The prototype is being developed as a project work at the Department of Computer Science, University of Tampere. The project ends in May 2000 and that is the due date of the prototype.

The Department of Computer Science will provide the development environment; the computers needed for development and the software tools. NMP will provide any extra software or hardware needed for building the prototype.

The development environment, namely the operating system and Internet browsers are similar to that of the user environments so that porting after development should not introduce any unanticipated problems. Even though the prototype will be tested on Netscape, other browsers should be taken into account. Netscape specific code should be avoided.

Relevant Facts

This project is of importance to a number of stakeholders. The prototype will be used for feasibility study of the final product. It will provide the necessary data for two Masters thesis. The managers of the project are managing it as a course project.

All involved in the project - developers, stakeholders, managers – are expected to make a firm commitment to its success.

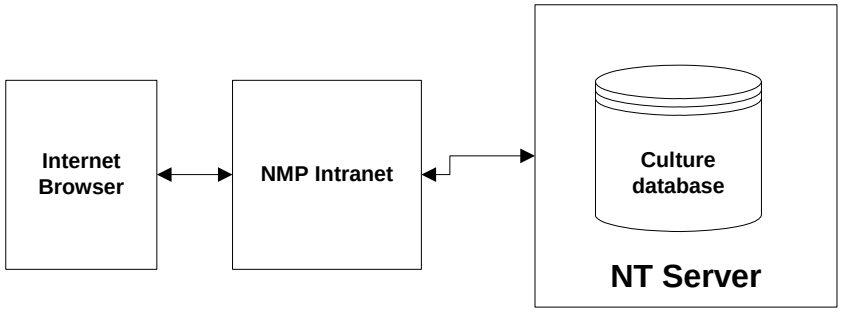
Assumptions

The assumptions in this project are that:

- all involved make a firm commitment towards its success
- nothing unforeseeable happens to go against the success of the project
- enough requirements will be gathered for a useful prototype
- the development environment and the database system chosen for the project is good enough for our purposes
- The developers have the technical know-how to build a prototype that is good enough for evaluating the feasibility of building a functional Culture Database.

The Scope of the Product

The relationship between the various parts of the systems is represented in the figure below. The database and the server are located in an NT Server. The user accesses the database through NMP Intranet.



DATABASE, FUNCTIONAL AND CONTENT REQUIREMENTS

The details of the database structure are defined in a separate document (database model, which is a part of the database specification) produced together with the client.

All the requirements will use the following notation:

Requirement #. <requirement name>
Priority (database) <priority>
Priority (functionality) <priority>
Priority (content) <priority>
Description: <description>

where:

<requirement name>	Descriptive name of the requirement
<priority>	1 – obligatory requirement, may not be left out 2 – obligatory requirement, can be left out only if causes a risk for implementing the requirements of the first priority. 3 – at least part of the requirements of the third priority need to be implemented 4 – Not implemented in the prototype.
<description>	free form textual description of the requirement

Priority (database) indicates the level of importance of the requirement in the implementation of the database, and priority (functionality) indicates the importance for the functionality of the software (e.g. searching for the information). Priority (content) states the importance of gathering the data for the database.

Culture Database will have the following database, functional and content requirements:

1: Country facts

Priority (database)	1
Priority (functionality)	1
Priority (content)	2-3
Description:	The database shall have the following information on a country: official languages, currency, capital, total population, percentage of population by age, religions, income level, GNP, infrastructure, education, literacy rate.

2: Telecommunications infrastructure

Priority (database)	1
Priority (functionality)	2
Priority (content)	3
Description:	The database shall contain the following information on the telecommunications infrastructure of a country: total amount of subscribers of fixed lines, the number or cellular subscribers, information on Internet usage, operators and their services, forecast of cellular growth, cellular technologies in use in the country.

3: Links

Priority (database)	1
Priority (functionality)	1
Priority (content)	2
Description:	The database shall provide a possibility of having links to Internet / Intranet, other (internal or external) databases, esp. Lotus Notes and to separate documents and files.

4: Language-specific information

Priority (database)	1
Priority (functionality)	1
Priority (content)	1
Description:	The database shall contain at least the following information on a language: alphabetical ordering, character sets, time, date, address,

telephone number and currency formats, separators, writing system and direction, fonts, Nokia-style used in display texts (e.g. infinitive or imperative), usage of numbers (grouping, separators, Arabic or some other numbers), name format (first name or last name first, how are last names formed - Ivanov, Ivanova, Spanish last names, de, von, van der).

5: Taboos

Priority (database)	1
Priority (functionality)	2
Priority (content)	3
Description:	The database shall contain information on taboos (e.g. numbers, words, graphics, discussion topics, and superstitious matters) in different countries and religions.

6: Keypads and keyboards

Priority (database)	1
Priority (functionality)	1
Priority (content)	2
Description:	The database shall contain information (including pictures) on relevant keyboard / keypad layouts and mapping.

7: Calendars

Priority (database)	1
Priority (functionality)	1
Priority (content)	2
Description:	The database shall contain information on relevant calendars, their country of use, week numbering, first day of the week, national holidays, name days.

8: Search engine

Priority (functionality)	1
Description:	The database shall have a powerful search engine with a possibility of conducting searches by e.g. the relevant entities of the data model and by using free text search and Boolean operators. For more information refer to the data model.

9: Colours and graphics

Priority (database)	1
Priority (functionality)	1
Priority (content)	2
Description:	The database shall contain information on the cultural associations of colours and graphics (e.g. icons, symbols of warning), and possibly on the usability of both colours and graphics.

10: Marketing and usability research

Priority (database)	2
Priority (functionality)	2
Priority (content)	3
Description:	The database shall contain either links or information in the form of documents on various trends (fashion, socio-political, consumer behaviour etc.), hobbies, popular sports, popular culture in general, sociological research on subcultures and segments. The descriptions of the documents can be searched by using free text search.

11: Input

Priority (database)	2
Priority (functionality)	3
Priority (content)	4
Description:	The database shall contain information about various input systems (at least the ones used in mobile phones, such as ITU-T, T9 etc.)

12: Knowledge network

Priority (database)	1
Priority (functionality)	2
Priority (content)	3
Description:	The database shall contain a list of NMP experts of different areas or contact persons of different regions and their contact information.

13: Restricted access

Priority (functionalit	1

y)
Description: Password protection for reading certain data and updating the information in the database (the database shall restrict access to certain data and the updating of information).

14: Behaviour

Priority (database) 1
Priority (functionality) 2
Priority (content) 3
Description: The database shall contain information about behaviour in a certain country (e.g. body language, gestures, do people shake hands, who should take the initiative, women in Arab countries, business protocols, acceptable business gifts).

15: Travelling

Priority (database) 1
Priority (functionality) 3
Priority (content) 4
Description: The database shall contain practical information for a traveler (e.g. vaccinations, visas).

16: Legal information

Priority (database) 1
Priority (functionality) 1
Priority (content) 3
Description: The database shall contain legal information on a country (such as warranty, copyrights, legal notices, trademarks, do the user guides need to be published in the official language(s) of the particular country, consumer protection, other legal matters - e.g. no calling line identification allowed in Italy)

17: Tones and sounds

Priority (database) 2
Priority (functionality) 3
Priority 4

(content)
Description: ringing tones and application tones used in Nokia phones (link to the Intranet page), sound samples, notation in music (do-re-mi, c-d-e, bitmaps etc.), scales, intervals, folk music (+attitude)

18: Brands

Priority (database) 2
Priority (functionality) 3
Priority (content) 4
Description: The database shall contain information about Nokia brand names, and accepted and rejected game and ringing tone names.

19: On-line help

Priority (functionality) **1**
Description: The database software shall have an on-line help.

UI REQUIREMENTS

This section is for features that are related to the way a user will see the product.

The UI of Culture Database shall have the look and feel and the search options of Internet search engine. The language of the interface shall be English.

USABILITY REQUIREMENTS

Ease of use

The users of the database being employees of NMP and having used different kinds of Internet programs should be able to use the program without any preliminary training.

Ease of learning

The basic use of the product should not require any pre-training, but detailed functions like Boolean search and other details may need referencing. It should not take more than one hour to learn the detailed functions of the database.

PERFORMANCE REQUIREMENTS

Speed requirements

The speed requirements of Culture Database should be comparable with typical internet programs. Designers should ensure that the design of the database does not cause any adverse side effects on speed of the user's environment.

Safety critical requirements

The product shall not allow unauthorized change and entry of data

Precision requirements

The data must be accurate and updated as they change.

Reliability and availability requirements

The product shall be available for use 24 hours per day, 365 days per year.

Access to the product shall be unavailable only when NMP Intranet is unavailable.

It should be possible to make manual and automatic backups of the data.

Culture Database should have the option of keeping a log file of its operations.

OPERATIONAL REQUIREMENTS

Expected technological environment

The product will be accessed from NMP Intranet using an Internet browser. It shall be installed on a workstation with Windows NT as the operating system. No additional software needs to be installed on the user's workstation.

MAINTAINABILITY AND PORTABILITY REQUIREMENTS

The prototype of Culture Database that will be built in this project will be enhanced for the final product for use at NMP. The prototype should take into account of the extendibility of data and the need for the prototype to be flexible. The data must be updateable at any time.

The system must have an interface for administrative purposes such as backups, turning log trace on and off, etc.

Portability requirements

The product will run in Windows NT. CDB should however be easy to port to other environments as well.

CULTURAL AND POLITICAL REQUIREMENTS

Culture Database will be used at all sites of NMP. The user interface should not contain any icons or symbols that could be considered offensive to users in any of the cultures.

LEGAL REQUIREMENTS

The database should not contain any information that can cause any legal actions from any government or organization.

OPEN ISSUES

OFF-THE –SHELF SOLUTIONS

Designers may look into products and off-the-shelf solutions that could be taken into use instead of coding from scratch.

TASKS

Documentation

The following documents will be produced in the project:

- Project Plan

  Contains the plan of the whole project. This document is written by the Project Manger, reviewed by the steering group and accepted by NMP

- Requirements Specification

  This document (Culture Database Requirements Specification) contains the requirements. The design specification will be based on this document. Reviewed by the steering group and accepted by NMP.

- Design specification

  The designers will write a design specification based on this requirements specification. The design specification shall consist of:

  - functional specification
  - database specification
  - user interface specification
  - technical specification

  The design specification should be checked against this document so that the requirements that are impossible to implement can be modified.

- Testing plan

  A designer will write a test plan based on the requirements. The Fit criterion of the requirements will be useful.

- Test report

  The results of the test of the prototype should be documented

The naming convention of these documents is cdb_xxxx

Where

- cdb stands for Culture Database.
- xxxx is the abbreviation of the actual document title.

All documents must be reviewed by the steering group, updated by the author(s) and approved by the steering group.

NOKIA

APPENDICES

REFERENCES

Document References

1. Markku Hanhiniemi, Culture Database Project Plan

Book References

1. Robertson S. and Robertson J., **Mastering the Requirements Process**, Addison-Wesley, 1999.

2. Thayer H. R. and Dorfman M., Eds, **Software Requirement Engineering, 2nd ed**., IEEE Computer Society Press, Los Alamaitos, Calif., 1997.

NMP
Sakari Hokkanen

LIITE 2: CULTURE DATABASE PROJECT PLAN

Version: 1.1

State: Approved

Author: Markku Hanhiniemi, Pekka Mäkiaho

Version history:

Version	Date	Description
0.1	1999-9-11	First draft version
0.2	1999-11-23	Comments from the steering group
0.3	1999-11-28	Corrections made according to the comments from the steering group
0.4	1999-12-02	Corrections made according to comments from Taija Björklund and Pekka Mäkiaho
1.0	1999-12-03	Approved with changes in the kick-off meeting
1.1	2003-04-15	Nokia confidential information removed

NOKIA

INTRODUCTION

Project

The goal of this project is to develop a prototype of a database software to be used for various purposes at Nokia Mobile Phones and also to collect available data to this database. The size of the project group for the project is one project manager, one contact person to Nokia Mobile Phones and five project participants. The project will also include a project steering group, which consist of the project manager, the contact person and two employees of Nokia Mobile Phones.

Product

The product of the project is a prototype of a database software, which will be used for various purposes at Nokia Mobile Phones (for example localisation and marketing). The database would contain at least computer-specific information and culture-specific information. In addition to the database, software consists also of a user interface and possible other software that uses the database.

GLOSSARY

Name	**Abbreviation**
Database specification	DB spec
Department of Computer Science of the University of Tampere	Dept. of CS
Design specification	Desg. spec
Functional specification	Func. spec
implementation phase	IP
Nokia Mobile Phones	NMP
Object Modelling Technique	OMT
Requirement specification	Req. spec
specification phase	SP
Technical specification	Tech. spec
testing phase	TP
Unified Modelling Language	UML
User interface specification	UI spec

PROJECT ORGANISATION

Structure of the organisation

The project organisation consists of the project group and the project steering group. The project group has five members and project manager Markku Hanhiniemi and contact person to NMP Kyekyeku Opoku-Pong. In addition, there are two persons (Pekka Mäkiaho and Taija Björklund) from NMP who will act as co-workers and project supervisors from the customer interest group (NMP). The steering group consists of Pekka Mäkiaho, Markku Hanhiniemi, Kyekyeku Opoku-Pong and Taija Björklund. The project group members are:

Sakari Hokkanen

NMP

Sakari Hokkanen

Kaarlo Kanerva
Jere Käpyaho
Tuukka Lahtela
Niko Saastamoinen

Persons in charge

Persons in charge for the project are the project manager and the contact person to NMP. They will be responsible mainly for NMP (as customer for the project). Also NMP has its own responsible persons for the project (who belong to the project steering group).

Interest groups

Interest groups of the project are NMP as the customer and purchaser of the project and Dept. of CS of the University of Tampere as the provider of the project group.

Project phases

The project has three main phases: specification (making the specification documents), implementation (implementation of what has been specified) and testing (testing of what has been implemented).

Every participant of the project group will also make regular reports of his progress during each phase so that the project manager can regularly review the current status of the project. The project manager will then report to the steering group.

Specification phase (SP)

The SP consists of planning and writing all the specification documents and specifying the methods for gathering the information for the database. The Design specification will be based on the Requirement specification. The Requirement specification is based on te needs and requirements of NMP that will be gathered during the implementation of Req. spec. The draft version of Req. spec should be ready before the writing of the Desq. spec begins. The Desg. Spec will be written in five parts: Func. spec, DB spec, Tech. spec, UI spec and the testing plan. The Functional specification is the first one of all these sub-specifications. Other four specifications will all be implemented more or less based on this specification.

Implementation phase (IP)

The IP consists mainly of implementation of the program prototype according to specifications made during the SP. Also gathering the information for the database according to the methods specified in the SP will be carried out during this phase. At the end of the SP and at the beginning of the IP, these two phases will overlap each other. This will happen also when there is a need to make changes to the specifications, for example, if some particular feature turns out to be impossible to implement in the way it was specified in the specification documents. The main parts of the IP are program prototype implementation, information gathering and making the specification documents more accurate, if necessary. In addition, this phase consists of module testing. Each programmer participating in the IP carries out module testing.

Testing phase (TP)

The main parts of the TP are testing the prototype implementation and modifying the implementation and related documents according to test results. This is called system testing. Also the IP and the TP phases will overlap each other. If malfunctions found during testing process contradict with specification documents made, the specification documents will also be modified.

System testing will be done in an iterative manner. Every feature of the program will be tested and, whenever any bugs or malfunctions are found, they will be reported using the reporting system defined later on to the programmer who has implemented this feature. He will then make the corrections needed according to the testing report. After that the corrected features will be retested. If corrections are still needed for the corrected features, the same procedure will be repeated.

When errors are found, they will instantly be documented in the error report. The programmers are responsible for checking the reports and making the corrections according to them.

There will be a certain model for the error reports, which could be for example an Excel-sheet. The following categories will be used for reporting errors:

1. Errors that will prevent the use of the software.
2. Errors that will cause inconveniences for using the software.
3. Suggestions for improvements.

The version from which the error was found and the version in which it was corrected must also be documented in the error report. The programmer who has made the corrections must update the error report. He must also report the errors which were not corrected and the suggested improvements which were not implemented.

Schedule of the project

After the kick-off meeting the schedule of the project consist of three main parts making specifications, implementation and testing. The kick-off meeting will start the actual project.

The following should be very preliminary schedule for the project. Assuming that the project starts at December 1999 and it should be completed in May 2000, there is approximately 20 weeks of total time.

Preliminary weekly project schedule:

Week	**Goal**	**Result document(s)**
49 – 50	Tools evaluation	
48 – 51	Training period: becoming familiar with the project and the tools that will be used with it	

49 – 03	Database design and functional designing of the software	DB spec, Func. spec
04	Planning the TP	Testing plan
01 – 05	Designing the technical specification of the software	Tech. spec
03 – 05	Designing the user interface of the software	UI spec
05	Inspection meeting: reviewing the plans and specifications	Minutes of the meeting possible changes to documents
05 – 15	Implementation, module testing, information gathering, updating Desg. Spec	Source code, more accurate versions of the Desg. Spec
12 – 17	System testing and error correction	Error reports, test report
17 – 18	Final documentation, demo, delivery of the prototype to the customer	Project end report

PROJECT STEERING

Goals and priorities

The primary goal for the project is to develop working database software that will meet the requirements of NMP. A working version of prototype will be delivered to the customer (NMP) in May 2000. This goal has the highest priority. The requirements and features will be prioritised so that the ones with the lowest priority can be left out if they cause a risk to reaching the primary goal.

Assumptions, dependencies and sideconditions

This project has the following starting assumptions:

It is possible to implement some kind of a prototype of the software

It is possible to implement a prototype that will satisfy the most important requirements of the customer.

It is possible to implement a web-based user interface to the database

The success of the project depends on these things:

Tools, personnel resources and hardware are available throughout the project.

The tools and methods that will be used in the project are familiar enough to the project group members after a training period.

Experience of the project group members is sufficient.

The schedule of the project is realistic in respect to the requirements of the customer.

Management of risks

There are following risks during the project:

1. Having unexpected difficulties in getting and keeping required personnel resources, for example project and/or steering group members

2. Having unexpected difficulties in getting the tools and the software that have been chosen for the project

3. Choosing wrong tools or wrong software for the project

4. Having an unrealistic schedule for the project

5. Having inadequate financing for the project

6. Choosing wrong methods for the design

(1) Because both the members of the project group and the project manager are students of the University of Tampere and not working for NMP, it is not easy to ensure that they will in fact stay in the project. One possible way is to seek the commitment of all involved in the project at the kick-off meeting.

(2,3) If neither NMP nor Dept. of CS at University of Tampere has required tools and/or software, they will be purchased. The software implementation should be as platform and tool independent as possible. This makes it possible to change the tools or software for the project if that is absolutely necessary.

(4) If it turns out to be impossible to plan, implement and test the software according the schedule that has been constructed, the features that are not absolutely necessary should be cut down from the final software. The key point here is that these kind of deficiencies should be noted early enough so that it is still possible to drop out some non-essential features and finish the project in time.

(6) The methods chosen for the design should be familiar to as many project members as possible.

Steering and controlling the project

The project steering group is responsible for supervising the project. Project manager is responsible for keeping the actual project going and will report to the steering group regularly. The steering group should meet at least once a month at a meeting where project manager will tell about the progress and the current situation of the project. Below the steering group is the project group itself. Project manager is also responsible of gathering the project group, getting the project on the way and

controlling the project group. Like the steering group, the project group will also have its own meetings in order to control the project progress. Project group meetings will occur more often than the steering group meetings. The project group members will write weekly reports to the project manager. Project manager is responsible for reporting to the steering group weekly by e-mail.

Planning the use of personnel resources

Actual personnel resources of the project are the five-member project group, whose job is to implement the database software that will be produced during the project, and project manager, whose job is to control the project group. In addition, there is a contact person to NMP, who will gather the information needed from NMP and who is also in charge of the quality assurance of the project. There are also two employees of NMP (Pekka Mäkiaho and Taija Björklund) who will participate at least in designing the database specification.

During the SP (consisting mainly of writing and planning documents and defining the database) it should not be difficult to plan the timing of various jobs so that there will not be any lack of personnel resources at any stage. It could be, for example, agreed that certain people would work with the specification of the part of the program, which they are going to implement.

In the IP (which will overlap with specification phase), those parts of the software that need to be implemented before the implementation of others can start should be implemented first. One example is the database. There should be some version of the database available before a user interface to it can be implemented. That is why implementation of certain parts of software will start before specification phase is complete.

The TP means testing of the entire software (system testing). This is not so-called module testing where each programmer is testing his own part of the software. The actual testing is done by persons who have been chosen from the project group to do this job. In this phase it has to be ensured those people have enough time to concentrate on testing the software, which means everything they have been doing at IP must be completed at this stage. Because not every member of the project group will get his implementation work done at the same time, those working with parts of the software that need to be implemented before others, should start the testing phase.

Because the TP is iterative the use of personnel resources should also be reasonable during the TP. This means that the system testers should have for example sequential testing tasks, whenever that is possible. When they have finished one testing task, they can take the next one without having to wait reported errors of the previous one to be corrected. The other way is to ensure that after the system tester has finished his testing and reported errors he can start correcting errors found in his own module during the system testing.

TECHNIQUES

Methods and tools

The software products that will be used in the project:

- MS-Office, mainly MS-Word, MS-Excel and Power Point for creating and maintaining the project documents

- ORACLE for planning, creating and maintaining the database

- Java-programming environment, MS-Visual Studio J++, perhaps also C++ and Perl, for creating and maintaining the user interface

- MS-Visual Source Safe for version control

- Internet browsers that support at least Java 1.1 will be used for the testing

- Database software (ORACLE) will have to support at least SQL PL/1

Possible methods that could be used in the project:

Conceptual modelling for designing the database

Object oriented programming techniques for creating user interface and other needed software (for example, OMT or UML could be used for modelling)

Version control and building

There will be a version control system for the source code and documents of the project with regular backups. There should also be a build system which should allow to build any version of the software at any given time (i.e. that allows us to go back to the previous versions).

Quality assurance

The quality targets for the project are:

1. Software, particularly the database, will meet the requirements stated by the customer

2. Software is as error-free as possible. All the errors are documented in the test report.

3. Software can be easily expanded and modified in the future

A style guide for the source code of the software should be chosen. In addition, the essential parts of the source code will be reviewed.

The results of the project (for example the usefulness of the prototype) will be published inside NMP.

NMP owns the copyright of the product of the project.

No confidential information of the product will be published in any demo-occasions. Also all the demo-occasions will be planned under the surveillance of NMP.

NMP

Sakari Hokkanen

DOCUMENTATION

Principles of documentation

The following principles of documenting will be obeyed during the project:

1. Every phase of the project should be documented closely enough.

2. Every document has an author who is responsible for maintaining it.

3. Every document has a supervisor who is responsible for accepting the changes made in the document.

4. Every change or corrective action should be updated in the corresponding document.

Produced documents

The project will produce the following documents:

1. Project plan (this document)

2. Weekly reports from the project group members to the project manager and from the project manager to the project steering group.

3. Requirement specification consisting of all the requirements that should be met by the implemented prototype software.

4. Design specification which consist of

    - functional specification (1)
    - technical specification (2)
    - database specification (3)
    - user interface specification (4)

5. Testing plan

6. Test report

7. The project end report.

Document naming practice

Documents produced by this project will be named like this:

cdb_xxxx

The cdb is an abbreviation which means Culture Database, xxxx is the abbreviation of the actual document title.

NMP

Sakari Hokkanen

Document reviews and approval

Project manager and the contact person will be the primary reviewers and approvers of the documents (or parts of the documents) made by project group members. Which documents will be written by project members will be decided later.

All changes made to the requirements specification and the project plan will need to be reviewed and accepted by the steering group. The first versions and major changes to the other documents should also be reviewed by the steering group.

Document production

Every document will have its version history printed out like this in the beginning of the document:

Version history:	Version	Date	Description
	X.X	XXXX-XX-XX	First Draft version

REFERENCES

Document references

1. Culture Database Database Specification
2. Culture Database Functional Specification
3. Culture Database Technical Specification
4. Culture Database User Interface Specification
5. Culture Database Requirements Specification
6. Culture Database Testing plan

NMP
Sakari Hokkanen

LIITE 3: Culture Database - Database Specification

Version: 1.1

State: Public

Author: Pekka Mäkiaho, Kaarlo Kanerva

Version history:

Version	Date	Description
0.1	2000-03-06	First draft version
1.0	2000-05-24	Approved by the project steering group
1.1	2003-12-03	Nokia confidential information removed.

NOKIA

GLOSSARY

Abbreviation	Name
ANSI	American National Standards Institute
DDL	Data Definition Language
ERM	Entity Relationship Modelling
FK	Foreign key
IEC	International Electrotechnical Commission
ISO	International Standardisation Organisation
NMP	Nokia Mobile Phones
PK	Primary key
PL/SQL	Procedural language extension to SQL
SQL	Structured Query Language
TBD	To be determined
UI	User Interface
URL	Universal Resource Locator

INTRODUCTION

Purpose

This document is one of a set of documents describing the Culture Database software. Conceptual schema of the object domain is represented by using entity-relationship modeling technique (ERM). The objective of this document is to describe the structure and information content of Culture Database software as the tables and their associated columns and relationships as they appear in the database.

The primary audience of this document is the project group that is involved in the design, implementation and testing of Culture Database application. This document serves as a basis for possible further development, enhancement and extension of the database information contained in the Culture Database application prototype for NMP personnel who will participate in the maintenance of the product.

NMP as the customer will be interested in the description and definition of the information contained in the database.

Scope

The scope of this document is the description of the database tables and columns and their relationships. The end users of the Culture Database application have access to information outside the database table columns via links provided in the columns of the tables. The information structure and content beyond the link columns is not included in this document.

NOKIA

Major design constraints and limitations

The Culture Database uses Oracle8i Server version 8.1.5 as the database server. Portability requirements in the Culture Database Requirements Specification set limitations to the database design. In order to make the Culture Database portable to other environments Oracle specific extensions to ANSI/ISO SQL/92 standard are avoided. The formal names of the standards are:

- ANSI X3.135-1992, "Database Language SQL"
- ISO/IEC 9075:1992, "Database Language SQL"

The following Oracle specific extensions are not used:

- Oracle specific datatypes
- PL/SQL
- Triggers

An exception to not using Oracle specific extensions is the use of CREATE SEQUENCE – statement, which creates an object to the database. It can be used to create primary key values to columns by which an entity is uniquely identified. Also VARCHAR2 datatype is used. VARCHAR datatype is currently synonymous with the VARCHAR2 datatype. Oracle recommends that you use VARCHAR2 rather than VARCHAR. In the future, VARCHAR might be defined as a separate datatype used for variable-length character strings compared with different comparison semantics.

DESIGN DESCRIPTION

TABLES AND COLUMNS

Column names are followed by column datatype, null option, primary key, foreign key and unique constraints and a description of the meaning of the column.

CALENDAR

calendar_id	NUMBER(38) NOT NULL PK Calendar identifier created internally by the CDB-software.	
calendar_name	VARCHAR2(100) NOT NULL UNIQUE Name of the calendar identified by calendar_id.	
basis	VARCHAR2(1000) NULL Description of what the calendar is based on. For example lunar, solar or imperial.	

NOKIA

chronology NUMBER(5) NULL
Year 0 (zero) compared to the Gregorian calendar

y2k_date VARCHAR2(20) NULL
Which date is the date 1.1. 2000 of the Gregorian calendar in this calendar?

month VARCHAR2(1000) NULL
Free text for the description of month. (i.e. months/year, length of one month)

year VARCHAR2(1000) NULL
Description of the definition the year is defined. (i.e. length in days)

year_first_day VARCHAR2(1000) NULL
First day of the year

errors_fixed VARCHAR2(1000) NULL
Desciption of how errors are fixed (i.e. leap days etc.)

free_text VARCHAR2(2048) NULL
Provided for more information about the calendar

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

CALENDAR_ID_ID

calendar_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

CHARACTER_SET

character_set_id NUMBER(38) NOT NULL PK
Character set identifier created internally by the CDB-software.

character_set_name VARCHAR2(50) NOT NULL UNIQUE
Name of the character set

alternative_names VARCHAR2(100) NULL
Alternative names for the character set

encoding	VARCHAR2(100)	NULL	
	Encoding system		
character_set_desc	VARCHAR2(2000)	NULL	
	Description of the character set		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information about the character set		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

CHARACTER_SET_ID_ID

character_set_id	NUMBER(38)	NOT NULL	
	Counter to create unique primary keys.		

COLOUR

colour_id	NUMBER(38)	NOT NULL	PK
	Colour identifier created internally by the CDB-software		
colour_name	VARCHAR2(50)	NOT NULL	UNIQUE
	Name of colour		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information about the colour		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COLOUR_ID_ID

colour_id	NUMBER(38)	NOT NULL	
	Counter to create unique primary keys.		

NOKIA

COUNTRY

country_id	CHAR(3)	NOT NULL PK	
	A unique country code.		
country_name	VARCHAR2(100)	NOT NULL	UNIQUE
	Name of the country		
native_name	VARCHAR2(100)	NULL	
	Native name of the country		
currency	CHAR(3)	NULL	
	Abbreviation of currency		
currency_name	VARCHAR2(100)	NULL	
	Currency name		
capital	VARCHAR2(100)	NULL	
	The capital city of the country		
population	NUMBER(11)	NULL	
	Population of country		
gnp_capita	NUMBER(9)	NULL	
	GNP per capita		
gnp	NUMBER(15)	NULL	
	GNP in dollars per year		
infrastructure	VARCHAR2(1000)	NULL	
	General information on infrastructure of the country		
education	VARCHAR2(1000)	NULL	
	Information on education of people in the country		
literacy_rate	NUMBER(4,1)	NULL	
	Literacy rate		
fixed_line_subscrs	NUMBER(11)	NULL	
	Total amount of subscribers of fixed lines		
cellular_subscrs	NUMBER(11)	NULL	
	Number of cellular subscribers		

internet_usage	VARCHAR2(1000)	NULL	
	Information on Internet usage		
cellular_growth_frc	VARCHAR2(1000)	NULL	
	Forecast on cellular growth in the country		
cellular_techs_used	VARCHAR2(1000)	NULL	
	Cellular technologies used in the country (GSM, TDMA etc.)		
traveller_advice	VARCHAR2(1000)	NULL	
	Advice for travellers in the country		
behavior	VARCHAR2(1000)	NULL	
	Information on behavior in the country		
flag_pic_loc	VARCHAR2(1000)	NULL	
	Access path and file name to locate the picture of the country flag		
flag_ftype	VARCHAR2(100)	NULL	
	Type of the file containing the flag picture		
national_holidays	VARCHAR2(4000)	NULL	
	Description and a list of national holidays of the country		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information		
url	VARCHAR2(512)	NULL	
URL to refer to where a document or file can be found.			
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		
region_id	NUMBER(38)	NULL	FK
	Region identifier created internally by the CDB-software		

COUNTRY_CALENDAR

country_id	CHAR(3)	NOT NULL	PK	FK
	A unique country code.			
calendar_id	NUMBER(38)	NOT NULL	PK	FK
	Calendar identifier created internally by the CDB-software.			
week_number_use	CHAR(1)	NULL		

Are week numbers used. '0' = no, '1' = yes

first_day_of_week VARCHAR2(50) NULL
First day of the week

first_week_of_year VARCHAR2(1000) NULL
Description of how the first week of the year is defined in the calendar used in this country

purpose VARCHAR2(1000) NULL
The purpose of the use of the calendar in the country

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

COUNTRY_COLOUR

country_id CHAR(3) NOT NULL PK FK
A unique country code.

colour_id NUMBER(38) NOT NULL PK FK
Colour identifier created internally by the CDB-software

colour_meaning VARCHAR2(1000) NOT NULL
Meaning of the colour in the country

free_text VARCHAR2(2048) NULL
Provided for more information about the colour in the country

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

1.1.1 COUNTRY_GRAPHIC

country_id CHAR(3) NOT NULL PK FK
A unique country code.

graphic_id	NUMBER(38)	NOT NULL PK FK
	Graphic identifier created internally by the CDB-software.	
graphic_meaning	VARCHAR2(1000)	NOT NULL
	Meaning of the graphic in the country	
free_text	VARCHAR2(2048)	NULL
	Provided for more information on the graphic in the country	
url	VARCHAR2(512)	NULL
URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL
	Title of the document or file referenced by URL	

COUNTRY_LANGUAGE

country_id	CHAR(3)	NOT NULL PK FK
	A unique country code.	
language_id	CHAR(3)	NOT NULL PK FK
	Code for the representation of the name of language.	
language_speakers	NUMBER(4,1)	NULL
	Percentage of the amount of people speaking the language in the country	
official_language	CHAR(1)	NULL
	Is this the official language of the country. Column value ’0’ denotes no and value ’1’ denotes yes	
free_text	VARCHAR2(2048)	NULL
	Provided for more information.	
url	VARCHAR2(512)	NULL
URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL
	Title of the document or file referenced by URL	

COUNTRY_OPERATOR

country_id	CHAR(3)	NOT NULL PK FK
	A unique country code.	

operator_id	NUMBER(38)	NOT NULL	PK FK
	Operatior identifier created internally by the CDB-software		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COUNTRY_ RELIGION

country_id	CHAR(3)	NOT NULL	PK FK
	A unique country code.		
religion_id	NUMBER(38)	NOT NULL	PK FK
	Religion identifier created internally by the CDB-software		
special_information	VARCHAR2(1000)	NULL	
	Special information on the religion in the country		
population_percentage	NUMBER(4,1)	NULL	
	Percentage of the country's population following this religion		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COUNTRY_ TABOO

country_id	CHAR(3)	NOT NULL	PK FK
	A unique country code.		
taboo_id	NUMBER(38)	NOT NULL	PK FK
	Taboo identifier created internally by the CDB-software		
taboo_country_meaning	VARCHAR2(1000)	NULL	
	Meaning of the taboo in the country		
url	VARCHAR2(512)	NULL	

URL to refer to where a document or file can be found.

url_title	VARCHAR2(256) NULL	
	Title of the document or file referenced by URL	

GRAPHIC_CDB

graphic_id	NUMBER(38) NOT NULL PK	
	Graphic identifier created internally by the CDB-software	
graphic_name	VARCHAR2(100) NOT NULL	UNIQUE
	Name of the graphic	
graphic_description	VARCHAR2(1000) NULL	
	Description of the graphic	
graphic_pic_loc	VARCHAR2(1000) NULL	
	Access path and file name to locate the picture of the graphic	
graphic_ftype	VARCHAR2(100) NULL	
	Type of the file containing the picture of the graphic	
free_text	VARCHAR2(2048) NULL	
	Provided for more information	
url	VARCHAR2(512) NULL	

URL to refer to where a document or file can be found.

url_title	VARCHAR2(256) NULL	
	Title of the document or file referenced by URL	

GRAPHIC_CDB_ID_ID

graphic_id	NUMBER(38) NOT NULL	
	Counter to create unique primary keys.	

KEYBOARD

keyboard_id	NUMBER(38) NOT NULL PK	
	Keyboard identifier created internally by the CDB-software	

keyboard_name VARCHAR2(100) NOT NULL UNIQUE
Name of the keyboard

keyboard_description VARCHAR2(1000) NULL
Description of keyboard

keyboard_pic_loc VARCHAR2(1000) NULL
Access path and file name to locate the picture of the keyboard

keyboard_ftype VARCHAR2(100) NULL
Type of the file containing the keyboard picture

keyboard_type VARCHAR2(100) NULL
Keyboard type

free_text VARCHAR2(2048) NULL
Provided for more information about the keyboard

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

KEYBOARD_ID_ID

keyboard_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

LANGUAGE

language_id CHAR(3) NOT NULL PK
Code for the representation of the name of language.

english_name VARCHAR2(50) NOT NULL UNIQUE
English name for the language

native_name VARCHAR2(50) NULL
Native name for the language

amount_of_speakers NUMBER(12) NULL
Total number of speakers of this language

list_separator VARCHAR2(1000) NULL

List separator

name_gender VARCHAR2(1000) NULL
Acordance with gender (e.g. Mr. Ivanov, Mrs. Ivanova)

name_order CHAR(1) NULL
Order of first and last name. Order lastname, first name is considered true. Character value ’0’ denotes false, and value ’1’ denotes true.

name_others_used VARCHAR2(1000) NULL
Paternal of other names used

name_prefix VARCHAR2(1000) NULL
Prefixes used (e.g. von, van der)

writing_system VARCHAR2(10) NULL
Writing system. Accepted values: ”character”, ”syllable”, ”word”

writing_direction CHAR(2) NULL
Coding for direction and system

display_text_style VARCHAR2(1000) NULL
Description of display text style

punctuation VARCHAR2(1000) NULL
Punctuation

hyphenation VARCHAR(1000) NULL
Special remarks on hyphenation

free_text VARCHAR2(2048) NULL
Provided for more information

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

LANGUAGE_CHARACTER_SET

language_id CHAR(3) NOT NULL PK FK
Code for the representation of the name of language.

character_set_id NUMBER(38) NOT NULL PK FK
Character set identifier created internally by the CDB-software

notes VARCHAR2(1000) NULL
Notes of the representation of this language with this character set

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

LANGUAGE_UNICODE_CHARACTER

language_id CHAR(3) NOT NULL PK FK
Code for the representation of the name of language.

unicode_value CHAR(4) NOT NULL PK FK
Integer unicode value

language_alphabet CHAR(1) NULL
Is the unicode alphabet an alphabet in the language. Value ’0’ denotes false, and ’1’ denotes true

remarks VARCHAR2(1000) NULL
Remarks of using this character in this language.

alphabetical_order NUMBER(3) NULL
Order of this character in the alphabet of this language.

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

LAW

law_id NUMBER(38) NOT NULL PK
An unique identifier created by the CDB-software.

name VARCHAR2(100) NOT NULL UNIQUE
Name of the law

category NUMBER(2) NULL
Category of the law (copyright, product protection, etc,

NOKIA

to be defined later)

document_file_path VARCHAR2(100) NULL
The filename and path of the document containing the actual law text.

document_file_type VARCHAR(50) NULL
The type of the document file

free_text VARCHAR2(2048) NULL
Provided for more information about the law

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

country_id CHAR(3) NULL FK
The country in which the law is applied

LAW_ID_ID

law_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

LOCALE

country_id CHAR(3) NOT NULL PK FK
A unique country code.

language_id CHAR(3) NOT NULL PK FK
Code for the representation of the name of language.

locale_name VARCHAR2(50) NOT NULL UNIQUE
Name of locale

separator_list VARCHAR2(20) NULL
The list separator used

separator_1000 VARCHAR2(20) NULL
Thousand separator

separator_decimal	VARCHAR2(20)	NULL
	Decimal separator	
date_format_short	VARCHAR2(1000)	NULL
	Short date format	
date_format_long	VARCHAR2(1000)	NULL
	Long date format	
time_format	VARCHAR2(1000)	NULL
	Time format	
currency_format	VARCHAR2(50)	NULL
	Format of currrency	
number_format	VARCHAR2(50)	NULL
	Number format	
address_format	VARCHAR2(1000)	NULL
	Address format	
phone_number_format	VARCHAR2(100)	NULL
	Format of phone number	
free_text	VARCHAR2(2048)	NULL
	Provided for more information	
url	VARCHAR2(512)	NULL
URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL
	Title of the document or file referenced by URL	

LOCALE_KEYBOARD

country_id	CHAR(3)	NOT NULL PK FK
	A unique country code.	
language_id	CHAR(3)	NOT NULL PK FK
	Code for the representation of the name of language.	
keyboard_id	NUMBER(38)	NOT NULL PK FK
	Keyboard identifier created internally by the CDB-software	
url	VARCHAR2(512)	NULL
URL to refer to where a document or file can be found.		

NOKIA

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

NMP_EMPLOYEE

emp_no CHAR(20) NOT NULL PK
NMP-employee identifier

email VARCHAR2(100) NULL
e-mail address

phone_number VARCHAR2(50) NULL
Phone number of employee

name VARCHAR2(100) NOT NULL
Name of the employee

expertise VARCHAR2(1000) NULL
Area of expertise

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

OPERATOR

operator_id NUMBER(38) NOT NULL PK
Operator identifier created internally by the CDB-software

operator_name VARCHAR2(100) NOT NULL UNIQUE
Name of operator

free_text VARCHAR2(2048) NULL
Provided for more information

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

OPERATOR_ID_ID

operator_id	NUMBER(38)	NOT NULL	
	Counter to create unique primary keys.		

REGION

region_id	NUMBER(38)	NOT NULL	PK
	Region identifier created internally by the CDB-software		
region_name	VARCHAR2(50)	NOT NULL	UNIQUE
	Name of region		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		
emp_no	CHAR2(20)	NULL	FK
	NMP contact person in region		

REGION_ID_ID

region_id	NUMBER(38)	NOT NULL	
	Counter to create unique primary keys.		

RELIGION

religion_id	NUMBER(38)	NOT NULL	PK
	Religion identifier created internally by the CDB-software		
religion_name	VARCHAR2(100)	NOT NULL	UNIQUE
	Name of religion		
religion_description	VARCHAR2(1000)	NULL	
	Description of religion		
number_of_people	NUMBER(11)	NULL	

Number of people in this religion

behavior VARCHAR2(1000) NULL
Description about behavior in this religion

free_text VARCHAR2(2048) NULL
Provided for more information

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

RELIGION_ID_ID

religion_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

RELIGION_ COLOUR

religion_id NUMBER(38) NOT NULL PK FK
A unique religion identifier created internally by the CDB-software

colour_id NUMBER(38) NOT NULL PK FK
Colour identifier created internally by the CDB-software

colour_meaning VARCHAR2(1000) NOT NULL
Meaning of colour in this religion

free_text VARCHAR2(2048) NULL
Provided for more information about the colour in this religion

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

RELIGION_ GRAPHIC

religion_id NUMBER(38) NOT NULL PK FK

NOKIA

A unique religion identifier created internally by the CDB-software

graphic_id NUMBER(38) NOT NULL PK FK
Graphic identifier created internally by the CDB-software

graphic_meaning VARCHAR2(1000) NOT NULL
Meaning of graphic in religion

free_text VARCHAR2(2048) NULL
Provided for more information about the graphic in religion

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

RELIGION_TABOO

religion_id NUMBER(38) NOT NULL PK FK
Religion identifier created internally by the CDB-software

taboo_id NUMBER(38) NOT NULL PK FK
Taboo identifier created internally by the CDB-software

taboo_meaning VARCHAR2(1000) NOT NULL
Meaning of the taboo

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

TABOO

taboo_id NUMBER(38) NOT NULL PK
Taboo identifier created internally by the CDB-software

taboo_name VARCHAR2(100) NOT NULL UNIQUE
Name of taboo

NOKIA

free_text VARCHAR2(2048) NULL
Provided for more information

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

graphic_id NUMBER(38) NULL FK
Graphic identifier created internally by the CDB-software

TABOO_ID_ID

taboo_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

UI_FIELD

table_name VARCHAR2(50) NOT NULL PK FK
The actual table name in the database

field_name VARCHAR2(50) NOT NULL PK
The actual table column name in the database

ui_field_name VARCHAR2(50) NOT NULL
The column name to be shown in UI

description VARCHAR2(100) NULL
Description of the field.

UI_TABLE

table_name VARCHAR2(50) NOT NULL PK
The actual table name in the database

ui_table_name VARCHAR2(50) NOT NULL
The table name to be shown in UI

description VARCHAR(100) NULL
Description of the table

UNICODE_CHARACTER

unicode_value	CHAR(4)	NOT NULL	PK
	The value of a unicode character		
unicode_name	VARCHAR2(50)	NOT NULL	
	Name of unicode character		
unic_pic_loc	CHAR(1000)	NULL	
	Access path and file name to locate the picture of the unicode character		
unic_pic_ftype	VARCHAR2(100)	NULL	
	Type of the file containing the unicode character picture		

USER_CDB

user_id	CHAR(10)	NOT NULL	PK
	Login name of user		
email	VARCHAR2(100)	NULL	
	e-mail address		
phone_number	VARCHAR2(50)	NULL	
	Phone number of user		
name	VARCHAR2(100)	NOT NULL	
Name of the user			
passwd	CHAR(10)	NOT NULL	
	User password		
access_right	NUMBER(1)	NOT NULL	
	Level of the user rights.		
emp_no	CHAR(20)	NULL	FK
	NMP employee number		

INTEGRITY CONSTRAINTS

All constraints have a unique name.

DOMAIN CONSTRAINTS

Table column domain datatype constraints are defined in the DDL statements that create tables. Column domain constraints are described in section 3.1.

PRIMARY KEY CONSTRAINTS

Primary key constraints are defined in the DDL statements that create tables. Primary key columns are described in section 3.1.

REFERENCIAL CONSTRAINTS

Referencial constraints on foreign keys are defined in the DDL statements that create tables. Foreign key columns are described in section 3.1. Foreign keys have been defined with the ON DELETE CASCADE or ON DELETE SET NULL option to maintain referential integrity when deleting referenced values in the parent table that have dependent rows in the child table thus allowing Oracle to automatically delete rows from the child table.

UNIQUE CONSTRAINTS

Table column unique constraints are defined in the DDL statements that create tables. Column unique constraints are described in section 3.1.

CHECK CONSTRAINTS

Check constraint names, validation rules and descriptions.

CHK_LANG_ALPHABET11

language_alphabet IN(‘0’,’1’)

Valid characters in table LANGUAGE_UNICODE_CHARACTER, column language_alphabet. Character belongs to the alphabets of the language (‘1’) or not (‘0’).

CHK_OFFICIAL_LANGUAGE11

official_language IN(‘0’,’1’)

Valid characters in table COUNTRY_LANGUAGE, column official_language. Language is the official language of the country (‘1’) or not (‘0’).

NOKIA

CHK_WEEK_NUMBER_USE11

week_number_use IN('0','1')

Valid characters in table COUNTRY_CALENDAR, column week_number_use are '0' or '1'.

CHK_WRITING_SYSTEM

writing_system IN('character','syllable','word')

Valid character strings in table LANGUAGE, column writing_system are "character", "syllable" or "word".

AUXILIARY OBJECTS

The following Oracle specific objects have been created in the database. These SEQUENCE objects can be used by the CDB software to create unique primary key values.

calendar_id_seq

character_set_id_seq

colour_id_seq

graphic_id_seq

law_id_seq

operator_id_seq

region_id_seq

religion_id_seq

taboo_id_seq

INDEXES

All primary key columns (PK) and foreign key columns (FK) are indexed.

DATABASE BACKUP

The Oracle8i Export is used as the database backup method, and it is run manually. The Export utility can selectively export specific database objects. Backups taken by Export are done in Full Database Mode and User Mode.

NOKIA

REFERENCES

Document References

Culture Database Project Plan

Culture Database Software Requirements

Culture Database Functional Specification

Culture Database User Interface Specification

Culture Database Technical Specification

Oracle8i Enterprise Edition Release 8.1.5 for Windows NT.

Book References

Using Oracle8, Special Edition, William G. Page Jr and Nathan Hughes, 1998

Other References

http://www.oracle.com/

NMP
Sakari Hokkanen

LIITE 4: Culture Database Technical Specification

Version: 1.1

State: Approved

Author: Sakari Hokkanen, Pekka Mäkiaho

Version history:

Version	Date	Description
0.1	2000-03-08	First draft version
0.2	2000-03-16	Second draft version
0.3	2000-03-21	Third draft version
0.4	2000-05-17	Fourth draft version
1.0	2000-05-25	Approved by the steering group
1.1	2003-12-03	Nokia confidential information removed.

NMP
Sakari Hokkanen

GLOSSARY

Abbreviation	Name
AVS	Advanced Visual Search
CDB	Culture Database
CSS	Cascading Style Sheets
CVS	Concurrent Versions System
HTML	Hypertext Markup Language
IIS	(Microsoft) Internet Information Software
IRP	Internet Response Time
JCVS	Java Concurrent Versions System
JDBC	Java Database Connectivity
JDK	Java Development Kit
NMP	Nokia Mobile Phones
PWS	Personal Web Server
SQL	Structured Query Language
TBD	To Be Determined
UI	User Interface
WinCVS	Windows Concurrent Versions System

INTRODUCTION

Purpose of this document

The purpose of this document is to explain the technical design that is used to implement the Culture Database system. This document shall explain the design architecture and the various techniques that are used in the implementation of CDB. The readers of this document are the CDB project members, who will be using this document in the implementation of CDB, and the clients of this project, that is, people at NMP using and further developing CDB.

Culture Database

Culture Database is a database system for storing and retrieving information on different countries, different cultures and different languages. This system will be developed as an exercise project for the courses Programming Project and Software Project Management at the department of computer and information sciences, University of Tampere during autumn 1999 and spring 2000. Further information of the Culture Database can be found from the Culture Database Requiremens Specification document and other project documents.

Major design constraints and limitations

Culture Database is used from a computer that has access to the NMP-intranet and has Windows NT 4.0 operating system and Netscape Navigator 4.0 (or higher) www-browser. The User Interface of Culture Database shall be able to be used without any external plugins or add-ons.

The baseline for the implementation of the UI shall be HTML 4.01 Transitional. Although HTML 4.01 Transitional contains elements that can be used to control the presentation of the HTML pages, they shall be avoided whenever possible. Instead, the desired presentation will be achieved using Cascading Style Sheets, Level 1 (CSS). The Culture Database User Interface Specification describes this in more detail.

The middleware part of Culture Database is programmed with Java programming language version 1.1. The database access parts of the middleware shall be programmed with JDBC 1.2 data access API. Queries to the database shall be made using the ANSI/ISO SQL/92 query language. The servlets will be programmed with java Servlet API (v.?). The Java code is developed using the Java Development Kit 1.1.8. The servlets are developed using the Java Servlet Development Kit 2.1.

The middleware will be running on a server computer that has IIS 4.0 server software. Culture Database will although be developed and tested using Personal Web Server (PWS) –software because it is the only server software that the project team has on the development computer. On some point of the implementation phase some tests will be run on IIS –environment, though the only difference between IIS and PWS should be the limited amount of connections in PWS. The software that we are using for running the servlets is Caucho Resin 1.1.

The Culture Database uses Oracle 8i for it's database. Although Oracle supports BLOB-type attributes, they are not used in database tables. Instead, binary objects, such as documents and pictures, are stored to database servers hard disk. The paths and filetypes of such files are stored to the database. Database Specification describes this and other database related issues in more detail.

More Information on the techniques mentioned above can be found by following the links listed in section Other References.

DESIGN DESCRIPTION

Overview of the design architecture

The technical design architecture of Culture Database system consists of three parts. They are the user interface that is implemented as a set of HTML pages, the so called middleware which runs on a web server and acts between the user interface and the database, and the database system that runs on a database server. When a user makes a query to CDB by, for example, clicking a link in the UI the query is posted to a Servlet on the web server. The Servlet passes it on for a Java class which processes an SQL statement corresponding to user's query and sends it to the database server. The results of the query that are returned by the database server are then passed on to a Java class which generates a new HTML page of them. The result page is then returned to the Servlet which sends the new HTML page to the user.

Design and implementation viewpoints

Portability

A goal has been to make CDB as portable as possible. Some exceptions to this rule, however, must have been made. In order to make the Culture Database portable to other environments Oracle specific extensions to ANSI/ISO/SQL 92 have been avoided in the design of the database.

Version control

The version control system used in the project is Concurrent Versions System (CVS) which is an open source software. For version control also WinCVS and/or JCVS, which both are graphical user interfaces for CVS, are used.

Coding

No port numbers or server numbers etc. shall be hard-coded into the code when the middleware is programmed. Separate configuration file is made for this kind of data from where it should programmatically be read. The file name is CDB.properties and it is stored to the package com.nokia.cdb. This is partly for making the JDBC –connection generic so the code needs not to be re-compiled when moving the CDB software to a different server or a different database environment.

The coding conventions that each programmer shall use are the ones presented in the “Java Coding Conventions” document (Link to the document can be found from the section Other References).

Documentation

Javadoc comments shall be written to the code by each programmer and the documents generated by the Javadoc tool are included in this document. .

MODULES

Overview of modules

The main modules of the middleware are servlets (the main servlet is actually one class, but because of it's major role it can be considered as a module of it's own), db, viewer, format and admin. Admin module, which is used for administrating the database, is somewhat separate from the other modules and it has its own servlet class, AdminServlet.

When user acts in the User Interface by clicking links or making searches using simple text search or the advanced visual search s/he produces queries. These queries are in general processed according to the following.

When user clicks on a link or a search button to make a query to CDB the UI passes a list of parameters to the main servlet. These parameters represent the action that the user has performed. Servlet then passes the parameter list to the db module which decides what kind of query should be made, produces the corresponding SQL –statement, executes it to the database and gets the results of the query. Db module then generates a view of the results by using the classes of the viewer module. The resulting view is then returned to the main servlet which passes it to the format module. Format generates a HTML page of the view and returns it to the servlet. Servlet then returns this result page to the user.

Main Servlet

When a user clicks a link or presses the search button on the UI a list of parameters are sent to the MainServlet class as a text string. MainServlet then converts the parameters to Java types. MainServlet handles the query so that it first makes an instance of db modules class QueryManager and then calls its handleQuery method with the parameters converted to Java types. HandleQuery returns a view object (views are described in more detail in the description of viewer module) to the MainServlet which then makes a new HtmlDocument (module format) of it. Then it sends the produced HtmlDocument as text string to the clients outputstream and the user gets a new HTML page to his www-browser.

Parameters

The parameter list that comes from a HTML page to the servlet has four parameters. The parameters are converted to Java types before they are passed on to db module. The types and names of the parameters are:

(`int category, Id id, int view, String text`)

Parameter `category` contains the category where the query should be made. Acceptable values for `category` can be found from the class Categories.java. They are:

```
COUNTRIES                   = 1;
REGIONS                     = 2;
TELECOM                     = 3;
LANGUAGES                   = 4;
CHARSETS                    = 5;
LOCALES                     = 6;
TABOOS                      = 7;
LAWS                        = 8;
RELIGIONS                   = 9;
CALENDARS                   = 10;
COLOURS_AND_GRAPHICS        = 11;
KEYPADS                     = 12;
OPERATORS                   = 13;
```

Parameter `id` contains the key(s) to the table where the query and/or update should be made. If the `id` string is null it is left unnoticed, that is, for example, all rows from a table are returned as a query result.

Parameter `view` contains information on what kind of query the user has made and how the results should be presented on the UI. Accepted values for `view` can be found from the class Views.java. They are:

```
LIST                    = 1;
DETAIL                  = 2;
LONG_DETAIL             = 3;
SEARCH                  = 4;
UPDATE                  = 5;
LINK                    = 6;
```

Parameter `text` contains the search string that is to be searched from the database. If the test string is null it is left unnoticed.

Db

Db module handles the connections to database, the producing and executing of database queries and the constructing of the result views. The main class of the db module is QueryManager. When MainServlet calls the QueryManagers method handleQuery QueryManager first opens a connection to the database. Then it calls the method buildQuery which makes a new Query object according to the category parameter, that is, if the category paramer is Categories.COUNTRIES a CountryQuery object is generated. QueryManager then calls Querys method makeQuery. MakeQuery returns a View object to the QueryManager which then closes the database connection and returns the View to the Main Servlet.

The Query class has four query methods. They are listQuery, detailQuery, longDetailQuery and searchQuery. According to the type of the view-parameter makeQuery calls one of these.

ListQuery makes a query to the database that returns all rows from the table that corresponds to the category. It then makes a new ListView object of the results and returns it to the QueryManager.

DetailQuery makes a query that gets the fields that are wanted from a row of the table corresponding to the category, with an id field corresponding to the id-parameter. Some fields from some tables in relation to the category are also got in some cases. For example, LanguageQuerys detailQuery gets all the fields from the language table, rows from relations country_language, locale, language_unicode_character and language_character_set. It also gets some fields from the table codes needed for the presentation of the values of some fields in the UI. DetailQuery then makes a new DetailView object of all the query results and returns it to the QueryManager.

Method longDetailQuery is used only with the country category, so it only has an implementation in class CountryQuery. LongDetailQuery is quite similar to the detailQuery, except that it gets all the fields from the country table and rows from all the tables in relation to the country. This has been made because the country table is quite big and has many relation tables. CountryQuerys detailQuery gets only some of the fields from the country table and country_language relation.

Method searchQuery executes a free text search to the table corresponding to the category and to some of its relation tables. It searches the string in the parameter `text` from the text fields of the category table and from the text fields of the relation tables. It then makes a new ListView object of the search results and returns it to the QueryManager. The ListView has links to the detail views of the rows of category table that matched in the search.

All of the query methods also use two methods, uiTable and uiField, for getting the user interface captions for the title of the result view object and the fields that are returned in it. These methods are implemented in the super class Query.

Viewer

Module viewer contains classes that are used as data storages by modules db and format. The main class of the module is abstract class View that has two subclasses, ListView and DetailView. Views are made by db module as described above. When MainServlet gets a ListView or a DetailView object from QueryManager as a query result it passes it to format module which generates a new HtmlDocument object of it.

Module viewer also contains classes Id, Link, and ExternalLink. Id represents a key that is used to find rows from database tables. It can contain one two three strings that represent keys in database tables. This is because some tables have two or even three keys that are used in getting a row. For example locale table has two primary keys, country_id and language_id.

Link and ExternalLink both represent links in CDBs user interface. The difference between them is that class Link represents a link to information that comes from the database, it contains the parameters needed to make a query to CDB.

ExternalLink, on the contrary, links to information that comes from some other place than CDBs database. It can link to a file on the hard disk of CDBs www-server or to any file anywhere in the Internet. ExternalLink has a field fileType that represents the type of the file that the field url links to. If the fileType field contains string "inline" the file is considered as a small picture and it is shown among other data fields in the user interface. If fileTyp field contains anything else than "inline" a new browser window is opened and it is left to users www-browser to decide how to show the file to the user. The links that are stored to the class ExternalLink should not be mixed with those links that are stored two View classes fields linkURL and linkText. They link in general to any web-page in the Internet or the NMP intranet as ExternalLinks link to binary files.

Format

Module format is used to generate a HTML representation of the View object returned as a query result by the db module. The main class of the module is HtmlDocument which generates the HTML using class ViewFormat. The main method of the HtmlDocument is toString which returns the fields of the view and the HTML code as a string object.

Admin

The admin module is used for administrating the database one table at time. The structure of CDBs database, the relations between tables and the types and lengths of their columns, is described in the class DataModel. The admin module also handles the user authentication, a user needs to have certain privileges to be able to administrate the database.

PACKAGES AND CLASSES

The modules described above are stored to packages with the name of the module. The package hierarchy is following:

package com.nokia.cdb
package com.nokia.cdb.admin
package com.nokia.cdb.admin.db
package com.nokia.cdb.admin.elements
package com.nokia.cdb.db
package com.nokia.cdb.format
package com.nokia.cdb.viewer

REFERENCES

Document References

Kyekyeku Opoku-Pong, Culture Database Requirements Specification

Jere Käpyaho, Culture Database User Interface Specification

Tuukka Lahtela, Niko Saastamoinen, Culture Database Functional Specification

Kaarlo Kanerva, Culture Database Database Specification

Other References

Java Development Kit, JavaSoft
http://java.sun.com/jdk/ (last accessed: 2000-03-06)

Java Servlet Development Kit, JavaSoft
http://java.sun.com/products/servlets/ (last accessed: 2000-03-06)

Element Construction Set
http://java.apache.org/ecs/index.html (last accessed: 2000-03-06)

Java Coding Conventions
http://java.sun.com/docs/codeconv/ (last accessed: 2000-03-06)

Resin 1.0, Caucho
http://www.caucho.com/products/resin1.0/index.html (last accessed: 2000-03-06)

Oracle
http://www.oracle.com (last accessed: 2000-03-06)

CVS
http://www.sourcegear.com (last accessed: 2000-03-06)

WinCVS
http://www.wincvs.org (last accessed: 2000-03-06)

JCVS
http://www.jcvs.org (last accessed: 2000-03-06)

NMP
Sakari Hokkanen

LIITE 5: Culture Database User Interface Specification

Version: 1.1

State: Approved

Author: Jere Käpyaho, Pekka Mäkiaho

Version history:

Version	Date	Description
0.1	2000-02-29	First draft version
0.2	2000-03-06	Second draft version, with revisions from the project group, Taija Björklund and Pekka Mäkiaho
0.3.	2000-03-22	Third draft version, with revisions from the project group and Pekka Mäkiaho
0.4	2000-04-26	Fourth draft version, includes several miscellaneous corrections
0.5	2000-05-17	Fifth draft version, added Administration section, removed screen captures
0.6	2000-05-23	Upgraded to proposal stage with minor corrections

NMP
Sakari Hokkanen

1.0	2000-05-25	Approved by the project steering group
1.1	2003-12-03	Nokia confidential information removed.

NMP
Sakari Hokkanen

INTRODUCTION

Purpose of this document

This document describes the user interface for the Culture Database. Its intended audience consists of the members of the project group and the steering group. The aim of this document is to provide a reference for the implementation effort of the user interface. However, this document is not a user manual for the system. A dedicated on-line help system will be produced as a part of the user interface.

Major design constraints and limitations

The user interface shall be implemented as a set of HTML pages, designed for viewing with a WWW browser application. The minimum requirement for the browser application is Netscape Communicator 4.0. However, due to the generic nature of HTML the pages shall be designed so that they do not use any Netscape-specific extensions to HTML, nor any similar extensions implemented by Microsoft in Internet Explorer.

The use of specific plug-ins or other browser extension mechanisms shall not be required. The content of the HTML pages must be viewable with the basic configuration of the required browser. This rules out, for example, most current forms embedded binary content such as Shockwave, Flash, Vivo, and VRML, but not Java applets.

The baseline specification for the HTML implementation shall be HTML 4.01 Transitional, which is a W3C Recommendation. Currently there are no mainstream browsers that implement full support for HTML 4.01, but the use of HTML 4.01 specific elements does not hinder the normal viewing of documents. In the future, as support for HTML 4 matures, it is possible to benefit from the additional elements in terms of information structure and presentation.

Although HTML 4.01 Transitional contains elements that can be used to control the presentation of the HTML pages, they shall be avoided whenever possible. Instead, the desired presentation will be achieved using Cascading Style Sheets, Level 1 (CSS). The support for CSS in Netscape Communicator is less than optimal, but basic functionality is usable. Since it is possible for every page in the user interface to utilize a common style sheet, it will be considerably easier to makes changes to the layout as need arises. In the best case the layout of the entire collection of pages can be changed by modifying a single style sheet, since style sheets are linked to the HTML page.

The Culture Database is designed to be viewed on a desktop computer. Therefore, no significant effort will be made to optimize the application for PDAs, cell phones or other alternative client devices. However, care should be taken not to rely on any client-specific features, should the need arise to adapt the user interface to a different environment. This means that there will be no arbitrary requirements for screen resolution, color depth or other issues related to computer display hardware or browser software.

NMP
Sakari Hokkanen

NOKIA

GLOSSARY

Term or abbreviation	Definition
AVS	Advanced Visual Search
browser	WWW browser application
CDB	Culture Database
CSS	Cascading Style Sheets
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
option set	Boolean operators, item list, and category list
TBD	to be determined
UI	user interface
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WWW	World Wide Web

OPEN ITEMS

The Update user interface.

THE USER INTERFACE

Intended audience

The intended audience of the user interface consists of employees of NMP, a group of technically sophisticated people. The employees may come from several different cultures, so there shall be no elements in the user interface that would be considered generally inappropriate. The language used in addressing users will be English.

The Web Site / Hypertext model

The user interface of Culture Database resembles what is commonly understood as a "web site". In fact, the CDB can be understood as a "subsite" of the NMP intranet. CDB has its own information structure, navigable from the start page.

The user interface is based on the hypertext model where some pages serve as launchpads to more complex pages generated from the information stored in the database, based on the selections and queries made by the user. The interface shall provide direct links from one type of information to the other. Relevant links to other main pages are embedded in the presentation.

The client browser communicates with the server using HTTP requests. Most of the interaction will be typical for World Wide Applications in general: the client requests a document from the server, and the server sends the document over the communication channel. Most of the result pages are constructed dynamically by programs running on the Web server. The pages contain results from database queries.

Look and feel

The "look and feel" of the CDB subsite shall be simple and formal, with no loud colors. Each page shall contain the Nokia logo in the upper left corner of the page, followed by

the name and version of the CDB application. The logo shall also function as a link to the CDB home page. The stylesheet used to format the HTML pages will be developed by the project team.

Each page of the user interface (excluding the main page) shall have a horizontal navigation menu placed at the top of the screen, below the logo and application name. This menu will give access to the main categories from every page. Every subpage's menu shall also contain a link back to the main page.

Each page shall have access to both simple and advanced search facilities. The simple search is presented as a text field to type keywords in and a button to launch the search. The advanced search shall have its own dedicated page (described later in this document).

Each page shall have a meaningful title to help distinguish pages. The title shall contain the phrase "Culture Database" prepended to it. This phrase may be followed by different phrases separated by a colon (:). These phrases include, but are not limited to: a category description, such as "Countries" or "Languages"; a detail identifier, such as "Germany" or "Christianity". Therefore, an actual title string may be something like "Culture Database: Countries: Germany" or "Culture Database: Religions: Christianity". The title shall be repeated, possibly in expanded form, in the page's main heading.

USER INTERFACE SECTIONS

Main page

Users enter the CDB subsite through a main page (exact Web address TBD, probably http://somedomain/culture or http://somedomain/cdb). The CDB main page presents a category of items for selection. These categories are chosen to reflect the priorities of the content requirements.

The categories available on the main page are:

Countries	Telecommunications	Laws
Regions	Operators	Religions
Languages	Keypads and keyboards	Calendars
Locales	Character sets	Taboos
		Colours and graphics

The categories shall be presented as hypertext links to relevant subpages, possibly as a multi-column table. Each category listed will have its own subpage.

Category pages

Countries

The Countries page presents a list of all the countries that are found in the database, in alphabetical order. The countries are presented as a table where each row contains

the name of the country and other relevant details, such as the country's native name and capital.

The names of the countries function as links. When the user follows any such link, s/he is directed to a page that contains detailed information about the selected country (see sections 5.3.1 and 5.3.2).

Regions

The Regions page presents a list of all the regions that are found in the database, in alphabetical order. The presentation is similar to the presentation of the countries.

The names of the regions function as links. When the user follows any such link, s/he is directed to a page that contains detailed information about the selected region. This page will also contain a list of all the countries in the selected region.

Telecommunications

The Telecom page contains a list of countries in the database, presented like on the Countries page. The names of the countries function as links. When the user follows a link, s/he is directed to a detail page with relevant information about the selected country's telecommunications infrastructure.

Languages

The Languages page contains a list of all the languages in the database, presented in alphabetical order. The names of the languages function as links. When the user follows a link s/he is directed to a page that contains detailed information about the selected language.

Character sets

The Character sets page contains a list of all the character sets that have information about them stored in the database. The descriptive names of the languages, sorted in alphabetical order, function as links. When the user follows a link, s/he is directed to a page that presents detailed information about the selected character set.

Locales

The Locales page contains a list of all the locales (combinations of a particular language and country) that are stored in the database in alphabetical order. The list items are names of locales in the form "Language (Country)", e.g. "French (France)". Each name functions as a link. When the user follows a link, s/he is directed to a page that presents detailed information about the selected locale.

Taboos

The Taboos page contains a list of all the taboos stored in the database, presented in alphabetical order. The short descriptions of the taboos act as links. When the user follows a link s/he is directed to a page that contains detailed information about the selected taboo.

Laws

The Laws page contains a list of all the laws stored in the database, sorted by category (type of law) and the country where the law is applicable. Laws that are not related to a country are listed first. Example listing (the numbers are for reference only):

1. General laws
2. Copyright laws
3. Copyright laws of Sweden
4. Copyright laws of China
5. Trademark laws
6. Trademark laws of Belgium
7. Trademark laws of France
8. etc.

The short law descriptions function as links. When the user follows a link s/he is directed to a page that contains detailed information about the selected law.

Religions

The Religions page contains a list of all the religions stored in the database presented in alphabetical order. The names of the religions act as links. When the user follows a link s/he is directed to a page that contains detailed information about the selected religion.

Calendars

The Calendars page contains a list of all the calendars that have information stored about them in the database. The names of the calendars act as links. When the user follows a link s/he is directed to a page that contains detailed information about the selected calendar.

Colours and graphics

The Colours and graphics page contains a list of all the colours and graphic images that have information stored about them in the database. The colours and graphic images are not presented on this page. A link is provided to a detailed descriptive view of the selected colour or graphic. Detailed views of graphics may also provide an in-line image of the graphic or a link to an external file.

Keypads and keyboards

The Keypads and keyboards page contains a list of descriptions of keypad/keyboard layouts stored in the database. A link is provided to a detailed view of the selected keypad/keyboard. This detailed view can provide an inline image of the layout or a link to an external image file. See section 5.3.12.

Detailed views

Some topics of CDB are presented with two kinds of detailed views on the information: short views present the essential information about the selected topic, and long views present additional information that may not be required by all users. Other topics contain significantly less information, so they can be presented on a single detail view page.

Each detail view also presents a hyperlink that relates to the corresponding item, stored in the database.

Country details (short view)

The short view of the country details lists information that is relevant to all users. This information includes the English and native names of the country, its associated region, the country's capital, the languages spoken, and the population. Possible referrers to this page are the Countries and Regions pages and each language detail page (see 5.3.5).

The country details are presented as a two-column table, where the first column labels the information and the second presents it. In some cases the contents of the second column may be a list of items, as in the case of official languages of a country. The contents may also appear as links to other pages.

The long view is accessed through a link at the end of the country information.

Country details (long view)

The long view of the country information duplicates the information found in the short view and adds the details omitted from the short view. This page has essentially all the information there is stored about the country, including those found through direct relations in the database.

Region details

The region detail view presents information about a region selected in a region list (see section 5.2.2).

Telecommunications details

The Telecommunications detail view presents information that is available about a certain country's telecommunications infrastructure in a tabular format. The country was selected in the Telecommunications category page (see section 5.2.3).

This page also lists the telecom operators operating in the selected country. The operator names are links to operator detail views.

Language details

The Language detail view presents information that is available about a language selected from a list in a country short or long view. in the corresponding table of the database. The countries where the language is spoken are also listed, with the country names functioning as links to the relevant (short) detail views of that country's information.

The page also contains links to locale-specific pages about the language, such as German spoken in Germany vs. German spoken in Austria etc. See 5.3.10.

Taboo details

The Taboo details view presents a selected taboo in its entirety. All the items stored in the corresponding database table are shown in a tabular format. Relevant material is presented as links to other general or detail views.

Law details

The Law details view presents detailed information about a law that was selected in the Laws category view (see 5.2.8). category.

Religion details

The Religion details view presents all the information available about a selected religion in tabular format. The information is presented as a table.

This page also lists the countries that have followers of the selected religion, the taboos associated with the religion, and the colours and graphic images

Calendar details

The Calendar details view presents all the information available about a selected calendar in tabular format. Relevant material is presented as links to other general or detail views.

Locale details

The Locale details view presents specific information about some combination of language and country, commonly referred to as locale. The information is presented tabular format.

This page also presents a list of the keypads and keyboards related to this locale.

Character set details

The Character set detail view lists information about a selected character set that is stored in the database. The languages that are usually presented using this character set are also listed.

Keypad/keyboard details

The Keypad/keyboard detail view lists information about a selected keypad or keyboard. The presentation may contain an inline image of the layout or a link to an external image file.

This page also lists the locales that are related to the selected keypad or keyboard.

Operator details

The Operator detail view presents information about a selected telecommunications operator. This page also lists the countries where this operator is operating as links to country short detail views.

Colours and graphics details

The Colours and graphics detail view presents information about a selected colour or graphic. In the case of colour the information lists the countries and religions associated with the colour. In the case of a graphic image the presentation may also contain a listing of taboos related to this image and an inline image or a link to an external image file.

The Search interface

The CDB UI has two different search interfaces: 1) a simple keyword search from a selected category and 2) an advanced search that combines categories and keywords using Boolean search operators.

Keyword search

The keyword search is intended for casual users with no time to learn more powerful but also more difficult search options. The keyword search always focuses on a specific category, not the whole database. The search interface includes a category selection item.

The keyword search is performed by typing the desired keyword into a text field and selecting a button labeled "Search". The results are presented as a HTML page with found items clearly labeled with their respective category.

The category selected as default must reflect the category page that is being viewed, if applicable.

The keyword search is case-insensitive.

Advanced Visual Search (AVS)

The Advanced Visual Search (AVS) is a way to specify more powerful search options. Initially the AVS page consists of a keyword input field, a dynamically modified item list and an initial category selection list. A button labeled "Search" starts the specified search.

If the user wants to perform just a basic search against the database, the search is essentially the same as the simple keyword search. However, the user can refine his/her search by selecting a button labeled "Refine search", which adds to the page a selection of a Boolean search operator (AND, OR, or NOT), another item list, and another category list. Together these three are termed as an "option set".

Each press of the "Refine search" button adds another option set to the page. The maximum number of option sets per query is 10, which is a sensible usability constraint (Rosenfeld 1998). Even 3-4 to option sets will force the user to scroll the page to see them all. Exactly one Boolean operator may be selected per option set (implemented as radio buttons).

Using the AVS, the user can perform complicated queries. Examples of such queries are:

(Category=Taboos, Keyword="death") NOT (Category=Religions, Item=Christianity)	Finds all the taboos containing the word "death" but which are not related

	to Christianity.
(Category=Languages, Keyword="ISO-8859-1") AND (Category=Regions, Item=Europe)	Finds all the languages spoken in Europe that can be represented using the ISO-8859-1 character set.
(Category=Languages, Item=German) AND (Category=Countries)	Finds all the countries where German is spoken.
(Category=Regions, Item=Europe) AND (Category=Countries, Item=Germany) AND (Category=Religions, Item=Catholicism) NOT (Category=Language, Item=German)	Finds all the countries in Europe that has followers of the Catholic religion but where German is not spoken.

If a keyword is entered, it is searched for within the item that was selected in the item list, and in the category that was selected in the category list. Both of these selections default to "all", i.e. the item list defaults to "All items in this category" and the category list defaults to "All categories".

If a keyword is not entered, the search will focus on the item and category selections. If these selections have their default values, the user is prompted to select an item and/or category.

The contents of an option set's item list are dynamically generated from the information available in the database. For example, if the user selected Category=Countries, the item list is populated with the countries entered in the database. If Category=Languages was selected, the list is populated with the languages, and so on. The categories in the category selection list are fixed.

Administration

The contents of the database can be changed by a user interface similar to the views. This is called the Administration Interface, or Admin for short. In order to change any data, the user needs a valid user name and a corresponsing password.

The Admin interface is not part of the regular Culture Database user interface, so not all the common elements of a typical user interface page are present. Admin is not intended for casual users, but for system administrators and other experts.

Authorization

When Admin is invoked, it prompts the user for the user name and password, together with a push button labeled “Login”. The user types this information and presses the Login button. Thereafter s/he is logged in to Admin, and stays logged in until s/he selectes the “Logout” link featured on every page.

Entering data

Admin presents the tables in the database as a list of links. Each link has a corresponding “Create new” selection. When the user selects the table name, s/he is presented with a list of data rows in the table, together with a “Remove” selection. When the user selects the “Create new” link, s/he gets an HTML form with a labeled

text field for each column in the table, together with a “Save” push button. To enter new data, the user needs to fill in these fields and press the “Save” button.

Removing data

To remove data, the user first needs to select the table containing the item to be removed. All the items in the table are listed as links with a corresponding “Remove” link. The actual removal of the data takes place when the “Remove” link is selected.

Error pages

It is feasible to suggest that from time to time a query entered by the user will produce no results. In this case the null result shall be presented to the user in a meaningful fashion. A simple statement "No results" is not acceptable; instead the error page should be proactive and offer at least the following options:

- Basic navigation menu
- Explanation of the reason (possibly malformed query)
- An option to go back and refine the search
- Links to general information about the topic of the query (countries, languages, law etc.)

Some conditions of the database software may lead to error messages from the server. These error messages may evidently be customized but this is not planned in the prototype version of the software.

On-line help

The on-line help system consists of a HTML page (or pages) describing the usage of the system from a casual user's point of view. This page should be accessible directly from the main page and from the navigation menu found on every page.

The AVS system has its own dedicated help page for advanced users. It should provide examples of complicated queries, much like those found in the description of AVS (see 5.4.2). The AVS system will not be implemented in the prototype version of the software.

REFERENCES

Document References

Kyekyeku Opoku-Pong, Culture Database Requirements Specification

Niko Saastamoinen, Tuukka Rantala, Culture Database Functional Specification

Book References

Rosenfeld 1998: Louis Rosenfeld, Peter Morville, Information Architecture for the World Wide Web. O'Reilly & Associates, 1998.
http://www.ora.com/catalog/infotecture/index.html (last accessed: 2000-04-26)

Other References

Jakob Nielsen, Search and You May Find. Alertbox, July 15, 1997.
http://www.useit.com/alertbox/9707b.html (last accessed: 2000-04-26)

World Wide Web Consortium: HTML 4.01 Specification
http://www.w3.org/TR/html4/ (last accessed: 2000-04-26)

World Wide Web Consortium: Cascading Style Sheets, Level 1
http://www.w3.org/TR/REC-CSS1 (last accessed: 2000-04-26)

NMP
Sakari Hokkanen

LIITE 6: Pandora content creation plan

NMP
Sakari Hokkanen

Introduction

The Pandora project is the continuation of the Culture Database project, the purpose of which was to investigate the possibility of building a database that would contain cultural information that could be used by NMP localisation. For further information about the Culture Database project, see /1/.

The goal of the Pandora project is to modify the database software so that the database will contain relevant localisation and internationalisation related information and can serve the needs of both NMP localisation and application or platform developers.

The purpose of this document is to plan the content creation for the Phase I of Pandora. This means describing and listing the content intended for the first phase, estimating the effort required and documenting the division of work. This plan does not contain information about the technical solutions required for modifying the existing software or the database structure. They are discussed in /2/.

PHASE 1 DATABASE CONTENT

CALENDAR . (Task 1.1)

- Description of what the calendar is based on. For example lunar, solar or imperial.
- Year 0 (zero) compared to the Gregorian calendar
- month (i.e. months/year, length of one month, when does a month change)
- Length of a year
- First day of the year (compared to the Gregorian calendar)
- Desciption of how errors are fixed (i.e. leap days, leap years etc.)
- links

Required calendars: Gregorian, Hijri, Hebrew, Chinese lunar, Japanese Imperial, Thai

Most of this information needs to be gathered from various sources (Internet, books about calendars and calendrical calculations), requires quite an effort in gathering and writing the content

LOCALE – CALENDAR (Task 1.2)

- Is this the default calendar of the locale or an optional calendar*
- Names of months (full name and abbreviation)* [1]
- Names of weekdays (full name and abbreviation)* [1]
- Are week numbers used?
- What is the first week of the year?*
- What is the first day of the week?*
- The purpose of use of this calendar (civil, religious etc.)

NOKIA

* Included in IBM ICU (=International Components for Unicode), might not be stored into the Pandora database, but retrieved directly from ICU data
[1] Will be validated

The rest of the information needs to be gathered from other sources.

CHARACTER (Task 1.11)

- code point*
- name*
- script in which the character is used*
- general category of the character*
- bidi category of the character*
- numeric value of the character*
- mirrored*
- upper case equivalent*
- lower case equivalent*
- link

* The data can be retrieved from the Unicode character database

LANGUAGE-CHARACTER (Task 1.10)

- The relation between the character and the language (i.e. unknown, non-essential, desirable, essential).
- Remarks on using this character in this language.

CHARACTER_SET (Task 1.3)

- Name of the charset + alternative names for it
- MIME name
- MIB enum
- Encoding (7, 8 or 16 bit, multibyte)
- (The characters it contains, their code points and order - to be left for Phase II)
- Description of the character set (purpose etc.)
- link

Required character sets: <Nokia confidential information removed>

Most of this information needs to be gathered from various sources (Internet, technical literature), requires quite an effort in gathering and writing the content.

LANGUAGE-CHARACTER_SET(Task 1.3)

- Support for the representation of this language using this charset, 1 = full, 2 = partial
- Notes of the representation of this language with this character set (e.g. limitations, usage, i.e. is this charset commonly used (on webpages, in mail messages…)

See above.

COUNTRY(Task 1.9)

- the name of the country in English
- native name of the country (if there are several official or major languages in the country, the native name of the country in all the languages)
- link (e.g. to CIA World Factbook)

Some of the data required has already been gathered by Sakari Hokkanen, needs to be checked and completed.

KEYBOARD(Task 1.4)

- Description of keyboard layout
- picture of the keyboard
- Keyboard type (or where is this used)
- status of the keyboard layout (e.g. draft, approved, obsolete, recommended)
- link (e.g. to Microsoft keyboard layouts page)

Required keyboard layouts: Hermes and Maverick keyboards, Linda keyboards, Nokia CMT phone keypads

LOCALE-KEYBOARD(Task 1.4)

A table for linking the keyboard to a certain locale / locales

LANGUAGE(Task 1.7)

- Punctuation (Does the language have punctuation marks of its own?)
- Does the language have genders? (yes/no)
- (line breaking)
- How is plural formed?
- Are other than Latin digits used in this language?
- Description of the dipslay text style (e.g. are articles used, how do you address a person)
- link (e.g. to Ethnologue)

Required languages: <Nokia confidential information removed>
This information should be gathered from the translators with the assistance of language coordinators and validated by the validators.

NMP
Sakari Hokkanen

NOKIA

LOCALE(Task 1.5)

- sorting tool
- list separator
- EPOC ID
- Java ID
- Windows ID
- 1000 separator
- Decimal separator
- Short date format
- Long date format
- Time format
  - pattern (separators, AM/PM separated by space, leading or trailing AM/PM, are leading zeros used)
  - 12- or 24-hour clock
  - AM/PM symbols
- currency format
  - pattern (currency symbol, leading or trailing currency symbol, currency symbol separated by space)
  - international currency symbol
  - negative currency
- Number format
  - pattern (grouping, separators)
  - per cent
- (Phone number format - to be left for Phase II)
  - (domestic - grouping, separators)
  - (international - grouping, separators)

Most of this information can be obtained from Java locales. The result should be compared to other sources, such as Microsoft locales. When the data from various sources have been made comparable, a decision about locales to be included needs to be made and the data received from the various sources also needs to be compared.

This type of information is also being collected for the Calypso project.

SCRIPT(Task 1.6)

- script name*
- script type (alphabetic, syllabic, semi-syllabic, ideographic, uncategorized)
- writing order
- context sensitive glyphs
- line breaking

- native digits
- case distinction
- character width distinction
- composition rules
- link

* included in the Unicode Technical report #24

Scripts required: Latin, Cyrillic, Greek, Arabic, Hebrew, Thai, Katakana, Hiragana, Kanji, Devanagari, Hangul, Hanja

SCRIPT-LANGUAGE(Task 1.6)

Linking the script to languages using the script.

INFORMATION STORAGE FORMAT(TASK 5.1)

The information storage format means the tool and document template to which the information collected is stored (e.g. Excel sheet) so that the data can be easily imported to the database.

The prerequisite for planning the information storage format means that certain phases of the technical planning have been completed. The database specification and database structure need to have been modified and the tool for importing data to the database specified. When the tool has been specified, it might dictate the tool to be used for storing the information. And the structure of the database and the content of the database tables and the fields included in the tables will specify what kind of information is to be collected and what is the format of the data.

TASK LIST

Note: to be further defined in the March meeting

Task area	**Task**	**Responsibility of**	**Amount of work required in days**
Information storage format	defining the format and making the template for each information category	NN : <Nokia confidential information removed>	4
Calendar	collecting calendar information	NN : <Nokia confidential information removed>	7
Locale – calendar	retrieving information from MS, Java or other similar sources	NN : <Nokia confidential information removed>	
	collecting calendar – locale information	NN : <Nokia confidential information removed>	2

Character	retrieving information from the Unicode character database	NN : <Nokia confidential information removed>	2
Language-character	creating the relationship tables between characters and languages on the basis of CharRepertoire.xls	NN : <Nokia confidential information removed>	2
Character set & Language – character set	collecting character set information	NN : <Nokia confidential information removed>	8
Country	checking and completing country names data already collected	NN : <Nokia confidential information removed>	1
Keyboard layout & Locale - keyboard	collecting information	NN : <Nokia confidential information removed>	3
Language	collecting the information from language coordinators and translators / validators, validating the information	NN : <Nokia confidential information removed>	14
Locale	comparing locale data	NN : <Nokia confidential information removed>, Taija Björklund	5
Script	Defining the content of script information	NN : <Nokia confidential information removed>	Done.
Language	Collecting the information	NN : <Nokia confidential information removed>	1 (one-day workshop)

SCHEDULE

The phase I of Pandora will be completed by November 30, 2001. The schedule for the phase 1 information gathering will be in /2/.

REFERENCES

/1/ Culture Database project plan, Markku Hanhiniemi.
/2/ Pandora project plan, Pekka Mäkiaho.

NMP
Sakari Hokkanen

PANDORA PROJECT PLAN

Version: 0.4

State: PROPOSAL

Author: Pekka Mäkiaho

Version history:	Version	Date	Description
	0.1	2001-02-26	First draft version
	0.2	2001-03-01	Changes after comments, Amount of work estimated
	0.3	2001-03-01	Some correctives

NMP
Sakari Hokkanen

0.4	2001-06-25	Schedule added References to other documents added

INTRODUCTION

1.2 Project

The goal of the Pandora project is to enhance the database software Culture Database that was developed together with Department of the Computer and Information Sciences ,University of Tampere at spring 2000. See Culture Database Project Plan /1/.
The software is now up and running in the NMP's Intranet. However, there is a need to improve the software technically, gather more information to the database and clarify the focus of the content. A document *Pandora content creation plan* /2/ written by Taija Björklund tels about gathering the information and a *document Pandora Technical Specification* /3/ by Jere Käpyaho consentrates on the technical improvements.The software will be finalised and the content gathered by the end of October 2001. More detailed schedule can be found from an Excel –document *Pandora Schedule* /4/ This and the other related to this project are stored in a directory \\nmpprj07tr\Common\internationalisation\pandora\docs.

GLOSSARY

Name	Abbreviation
CDB	Culture Database

PROJECT ORGANISATION

Structure of the organisation

The project organisation consists of the project group and the project steering group. The project group has five members and a project manager Pekka Mäkiaho. The steering group consists of Taija Björklund, Ilkka Kari, Jere Käpyaho Pekka Mäkiaho and Pirkko Riepponen. There is also a possibility that we use an external employee from our sub-contractor Tieto-X. The project members are.

- Taija Björklund - coordinating of gathering the information
- Jere Käpyaho – designing the technical architecture, implementation, collecting locale spesific data
- Jyrki Lassila -collecting information on calendars
- Jarmo Pikkujämsä – collecting information on languages
- Heikki Rasimäki (?) designing and implementing the Admin Tool

The responsibilities of each members are specified more specific in Pandora Schedule /4/

NMP
Sakari Hokkanen

SCHEDULE

The phase 1 of this project should be ready by the end of November 2001. Here are listed the tasks that need to be done. The unique id-numbers of each task refers to the Schedule /4/ Notice that the task listed in this document are mainly technical – the task related to collecting the information are listed in Pandora content creation plan /2/

Rewriting the database specification

The database specification must be gone over according to the new content requirements (task 4.1) The changes will be documented and new scripts for creating database and database restrictions will be written.

Clearing the database

The current database has data only for testing purposes. However, there may be some data that we want to conserve and therefore all the current data needs to be exported from the database. A backup of the current database software will be taken first. Then the data will be exported to Excel-sheets table by table. Finally the whole database will be dropped. (task 2.1)

Creating a new database

The database must be re-created and a part of the old contents have to be restored. The rewritten database scripts will be executed and some of the data will be restored from the Excel sheets. (task 2.2)

Tools for inserting data

Methods and tools for gathering and inserting data to Pandora will be planned and implemented. (Excel macros maybe) (task 2.3)

Changing the functionality and the UI according to the new database specification

Besides the database modification, there is a need to change the user interface and the functionality of the software. In practise, this means the changing the java source code and the HTML-pages. The technical sub-tasks are: (tasks 3.1and 3.3)

- Going through the existin source code
- Creating an building an environment for Pandora
- Modifying the existing source code
- Updating the current specifications
- Testing

NMP
Sakari Hokkanen

A program for sorting data according to a given locale

A cool feature of Pandora might be a possibility to sort a given file according to a specific locale. According to a preliminary plan, this will be a servlet which will be launched from the locale information page. A file would be browsed and then uploaded to the servlet, get sorted, downloaded back and finally it could be save to a local computer. This was left out from the Phase 1.

Admin tool

There must be an easy way handle (insert, update, delete) the data in Pandora using a graphical interface.

Currently there is an existing admin tool but this is not in very good shape and not properly documented either.

There are at least five possibilities to make a tool for inserting and updating the content of the database.

- Rewriting the existing admin tool, which was java servlets to generate HTML.
- Programming a new admin tool using java and servlets
- Creating a new tool with Microsoft Access
- Using Oracle client tools
- Using Excel sheets and the import feature of Oracle

Benefits and disadvantages of each possibility have to be evaluated. Writing a Java application might take the longest time. The easiest solution would be to use some graphical form developing tools for creating the UI and integrating it to the database. Microsoft Access appeared first very suitable for this purpose. However, it seems that Access does not support Unicode. I have ordered an evaluating version and an offer for Oracle's Internet Developer Suite. A person from Tieto-X might implement this separate tool with the help of Heikki Rasimäki. (Task 3.2)

Exporting locale specific data from Locale Browser and importing that to the database

We are going to put locale specific information to the database. Jere Käpyaho will implement some utility programs which will be used to export data from Java 2 runtime and Windows. (Task 1.5)

Administration

During the content creation, some amount of work will be needed to import data to Pandora, create user accounts, take backups etc. This will be done by Pekka Mäkiaho (Task 6.2)

NOKIA

REFERENCES

Document references

7. Culture Database Project Plan, Markku Hanhimiemi
8. Pandora content creation plan, Taija Björklund
9. Pandora Technical Specification, Jere Käpyaho
10. Pandora Schedule, Pekka Mäkiaho

NMP
Sakari Hokkanen

LIITE 8: Pandora Database Specification

Version: 1.2

State: Final

Authors: Jere Käpyaho, Taija Björklund, Pekka Mäkiaho (since 1.2)
Sakari Hokkanen (1.1)
Kaarlo Kanerva (up to 1.0)

Version history:

Version	Date	Description
0.1	2000-03-06	First draft version
0.2	2000-03-13	First proposal with revisions from the project group, Taija Bjorklund and Pekka Mäkiaho.
0.3	2000-03-17	Proposal with revisions from PekkaMäkiaho.
0.4	2000-03-20	Proposal with revisions from Pekka Mäkiaho and Sakari Hokkanen
0.5	2000-04-02	Proposal with revisions from Pekka Mäkiaho and Sakari Hokkanen
0.6	2000-04-12	Proposal with revisions from Pekka Mäkiaho
0.7	2000-04-19	Proposal with revisions from Pekka Mäkiaho and Taija Björklund
0.8	2000-05-08	Proposal with additions made by the project group
0.9	2000-05-22	Proposal with revisions from Kyekyeku Opoku-Pong
1.0	2000-05-24	Approved by the project steering group
1.1	2000-07-19	Added table UPDATELOCK
1.2	2001-05-17	Several tables added and amended for Pandora Phase 1.
1.3	2003-04-14	Nokia confidential information removed

NOKIA

GLOSSARY

Abbreviation	Name
ANSI	American National Standards Institute
DDL	Data Definition Language
ERM	Entity Relationship Modelling
FK	Foreign key
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
NMP	Nokia Mobile Phones
PK	Primary key
PL/SQL	Procedural language extension to SQL
SQL	Structured Query Language
TBD	To be determined
UI	User Interface
URL	Universal Resource Locator
UTF-8	Unicode Transformation Format 8-bit

INTRODUCTION

Purpose

This document is one of a set of documents describing the Pandora software. Conceptual schema of the object domain is represented by using entity-relationship modeling technique (ERM). The objective of this document is to describe the structure and information content of Pandora software as the tables and their associated columns and relationships as they appear in the database.

The primary audience of this document is the project group that is involved in the design, implementation and testing of the Pandora application. This document serves as a basis for possible further development, enhancement and extension of the database information contained in the Pandora application prototype for NMP personnel who will participate in the maintenance of the product.

Scope

The scope of this document is the description of the database tables and columns and their relationships. The end users of the Pandora application have access to information outside the database table columns via links provided in the columns of the tables. The information structure and content beyond the link columns is not included in this document.

Major design constraints and limitations

The Pandora uses Oracle 8i version 8.1.7 as the database server. In the prototype version of Pandora ("Culture Database 1.0") the requirements specification paid special attention to portability to other environments. In practice this portability has not been adhered to in the database, and for practical reasons Oracle-specific datatypes such as VARCHAR2 are now endorsed. The database specification no longer attempts to avoid Oracle-specific extensions to the ANSI/ISO SQL standards, but features such as PL/SQL and triggers are currently not used.

One significant Oracle-specific extension used is the CREATE SEQUENCE statement, which creates an object to the database. It can be used to create primary key values to columns by which an entity is uniquely identified. The VARCHAR2 data type is used because Oracle recommends it instead of VARCHAR (see Oracle SQL documentation for details).

DESIGN DESCRIPTION

DATABASE CHARACTER SET

The character set of the Oracle database shall be UTF-8, because the database will contain data in various different languages. UTF-8 is the most practical encoding to use and also supported by Oracle 8i.

TABLES AND COLUMNS

Column names are followed by column datatype, null option, primary key, foreign key and unique constraints and a description of the meaning of the column.

The tables are listed in alphabetical order.

CALENDAR

calendar_id	NUMBER(38) NOT NULL PK Calendar identifier created internally by the software.	
calendar_name	VARCHAR2(100) NOT NULL UNIQUE The display name of this calendar in English.	
calendar_type	NUMBER(3) NOT NULL Numeric calendar type used by the Win32 API.	
basis	VARCHAR2(1000) NULL Description of what the calendar is based on (e.g. lunar, solar or imperial).	
chronology	NUMBER(5) NULL Year 0 (zero) compared to the Gregorian calendar.	
~~y2k_date~~	~~VARCHAR2(20) NULL~~ ~~Which date is the date 1.1.2000 of the Gregorian calendar in this calendar?~~	
month	VARCHAR2(1000) NULL Free text for the description of month. (i.e. months/year, length of one month)	
year	VARCHAR2(1000) NULL Description of the definition of the year (i.e. length in days)	

year_first_day VARCHAR2(1000) NULL
First day of the year compared to the Gregorian calendar.

errors_fixed VARCHAR2(1000) NULL
Description of how errors are fixed (i.e. leap days etc.)

free_text VARCHAR2(2000) NULL
Provided for more information about the calendar.

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by an URL.

CALENDAR_ID_ID

calendar_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

CHARACTER

code_point NUMBER(8) NOT NULL PK
The code point of this Unicode character in Unicode 3.1.

script_name CHAR(20) NOT NULL FK
The name of the script that this character belongs to, according to Unicode Technical Report #24.

name VARCHAR2(100) NULL
The name of this character according to Unicode 3.1.

general_category CHAR(2) NULL
The general category of this character.

bidi_category CHAR(3) NULL
The bi-directional category of this character.

numeric_value CHAR(10) NULL
The numeric value of this character, if applicable. Stored as text in expression form, e.g. 1 or ¾.

mirrored NUMBER(1) NULL
Is this character mirrored? 1 = yes, 0 = no.

upper_case NUMBER(8) NULL

The code point of the upper case equivalent of this character, if applicable.

lower_case NUMBER(8) NULL
The code point of the lower case equivalent of this character, if applicable.

CHARSET

charset_id NUMBER(38) NOT NULL PK
Character set identifier created internally by the software.

charset_name VARCHAR2(50) NOT NULL UNIQUE
Name of the character set.

alternative_names VARCHAR2(100) NULL
Alternative names for the character set

mime_name VARCHAR2(50) NULL
The registered MIME name of this character set.

mibenum NUMBER(5) NULL
The MIBenum of this character set, if it exists.

encoding VARCHAR2(100) NULL
Description of the encoding system used (single byte, multibyte, double byte, 7-bit, 8-bit etc.).

charset_desc VARCHAR2(1000) NULL
Description of the character set.

free_text VARCHAR2(2000) NULL
Provided for more information about the character set.

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by URL.

CHARSET_ID_ID

charset_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

CODES

ctype	VARCHAR2(3)	NOT NULL	
	The type of the code, the table that the code refers to.		
ccode	VARCHAR2(5)	NOT NULL	
	The code itself.		
cexplanation	VARCHAR2(100)	NOT NULL	
	Explanation of the code (for example: 01 = official language).		

COLOUR (deprecated)

colour_id	NUMBER(38)	NOT NULL PK	
	Colour identifier created internally by the software.		
colour_name	VARCHAR2(50)	NOT NULL	UNIQUE
	Name of colour		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information about the colour		
url	VARCHAR2(512)	NULL	
URL to refer to where a document or file can be found.			
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COLOUR_ID_ID (deprecated)

colour_id	NUMBER(38)	NOT NULL	
	Counter to create unique primary keys.		

COUNTRY

country_id	CHAR(3)	NOT NULL PK	
	A unique country code.		
country_name	VARCHAR2(100)	NOT NULL	UNIQUE
	Name of the country in English.		
native_name	VARCHAR2(100)	NULL	

The native name of the country. If there are several official or major languages used in the country, list the name in all of them.

~~currency CHAR(3) NULL~~
~~Abbreviation of currency.~~

~~currency_name VARCHAR2(100) NULL~~
~~Currency name~~

~~capital VARCHAR2(100) NULL~~
~~The capital city of the country~~

~~population NUMBER(11) NULL~~
~~Population of country~~

~~gnp_capita NUMBER(9) NULL~~
~~GNP per capita~~

~~gnp NUMBER(15) NULL~~
~~GNP in dollars per year~~

~~infrastructure VARCHAR2(1000) NULL~~
~~General information on infrastructure of the country~~

~~education VARCHAR2(1000) NULL~~
~~Information on education of people in the country~~

~~literacy_rate NUMBER(4,1) NULL~~
~~Literacy rate~~

~~fixed_line_subscrs NUMBER(11) NULL~~
~~Total amount of subscribers of fixed lines~~

~~cellular_subscrs NUMBER(11) NULL~~
~~Number of cellular subscribers~~

~~internet_usage VARCHAR2(1000) NULL~~
~~Information on Internet usage~~

~~cellular_growth_frc VARCHAR2(1000) NULL~~
~~Forecast on cellular growth in the country~~

~~cellular_techs_used VARCHAR2(1000) NULL~~
~~Cellular technologies used in the country (GSM, TDMA etc.)~~

~~traveller_advice VARCHAR2(1000) NULL~~
~~Advice for travellers in the country.~~

~~behavior VARCHAR2(1000) NULL~~

~~Information on behavior in the country~~

~~flag_pic_loc VARCHAR2(1000) NULL~~
~~Access path and file name to locate the picture of the country flag~~

~~flag_ftype VARCHAR2(100) NULL~~
~~Type of the file containing the flag picture~~

~~national_holidays VARCHAR2(4000) NULL~~
~~Description and a list of national holidays of the country~~

free_text VARCHAR2(2000) NULL
Provided for more information

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by the URL.

~~region_id NUMBER(38) NULL FK~~
~~Region identifier created internally by the software.~~

COUNTRY_COLOUR (deprecated)

country_id CHAR(3) NOT NULL PK FK
A unique country code.

colour_id NUMBER(38) NOT NULL PK FK
Colour identifier created internally by the CDB-software

colour_meaning VARCHAR2(1000) NOT NULL
Meaning of the colour in the country

free_text VARCHAR2(2048) NULL
Provided for more information about the colour in the country

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

COUNTRY_GRAPHIC (deprecated)

country_id	CHAR(3)	NOT NULL	PK FK
	A unique country code.		
graphic_id	NUMBER(38)	NOT NULL	PK FK
	Graphic identifier created internally by the CDB-software.		
graphic_meaning	VARCHAR2(1000)	NOT NULL	
	Meaning of the graphic in the country		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information on the graphic in the country		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COUNTRY_LANGUAGE (deprecated)

country_id	CHAR(3)	NOT NULL	PK FK
	A unique ISO country code.		
language_id	CHAR(3)	NOT NULL	PK FK
	A unique ISO language code.		
language_speakers	NUMBER(4,1)	NULL	
	Percentage of the amount of people speaking the language in the country		
official_language	CHAR(1)	NULL	
	Is this the official language of the country (1 = yes, 0 = no)		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information.		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COUNTRY_OPERATOR (deprecated)

country_id	CHAR(3)	NOT NULL	PK FK
	A unique country code.		
operator_id	NUMBER(38)	NOT NULL	PK FK
	Operatior identifier created internally by the software		
free_text	VARCHAR2(2048)	NULL	
	Provided for more information		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COUNTRY_ RELIGION (deprecated)

country_id	CHAR(3)	NOT NULL	PK FK
	A unique country code.		
religion_id	NUMBER(38)	NOT NULL	PK FK
	Religion identifier created internally by the CDB-software		
special_information	VARCHAR2(1000)	NULL	
	Special information on the religion in the country		
population_percentage	NUMBER(4,1)	NULL	
	Percentage of the country's population following this religion		
url	VARCHAR2(512)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(256)	NULL	
	Title of the document or file referenced by URL		

COUNTRY_ TABOO (deprecated)

country_id	CHAR(3)	NOT NULL	PK FK
	A unique country code.		
taboo_id	NUMBER(38)	NOT NULL	PK FK
	Taboo identifier created internally by the CDB-software		

taboo_country_meaning VARCHAR2(1000) NULL
Meaning of the taboo in the country

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

GRAPHIC_CDB (deprecated, use BFILE)

graphic_id NUMBER(38) NOT NULL PK
Graphic identifier created internally by the software.

graphic_name VARCHAR2(100) NOT NULL UNIQUE
Name of the graphic.

graphic_description VARCHAR2(1000) NULL
Description of the graphic.

graphic_pic_loc VARCHAR2(1000) NULL
Access path and file name to locate the picture of the graphic.

graphic_ftype VARCHAR2(100) NULL
Type of the file containing the picture of the graphic.

free_text VARCHAR2(2048) NULL
Provided for more information

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by the URL.

GRAPHIC_CDB_ID_ID (deprecated, use BFILE)

graphic_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

KEYBOARD

keyboard_id	NUMBER(38)	NOT NULL	PK
	Keyboard identifier created internally by the software.		
keyboard_name	VARCHAR2(100)	NOT NULL	UNIQUE
	Name of the keyboard. If Nokia keyboard, indicate product.		
keyboard_description	VARCHAR2(1000)	NULL	
	Description of the keyboard.		
keyboard_type	VARCHAR2(100)	NULL	
	Keyboard type: alphabetic keyboard (e.g. QWERTY), numeric keypad (e.g. ITU-T) etc.		
keyboard_image	BFILE	NULL	
	Reference to an external image file representing this keyboard.		
~~keyboard_graphic_id~~	~~NUMBER(38)~~	~~NULL~~	~~FK~~
	~~Identifier for a graphic representing this keyboard.~~		
keyboard_status	VARCHAR(1000)	NULL	
	Description of the status of this keyboard (draft, approved, obsolete, recommended etc.)		
~~keyboard_pic_loc~~	~~VARCHAR2(1000)~~	~~NULL~~	
	~~Access path and file name to locate the picture of the keyboard~~		
~~keyboard_ftype~~	~~VARCHAR2(100)~~	~~NULL~~	
	~~Type of the file containing the keyboard picture~~		
free_text	VARCHAR2(2000)	NULL	
	Provided for more information about the keyboard.		
url	VARCHAR2(500)	NULL	
URL to refer to where a document or file can be found.			
url_title	VARCHAR2(200)	NULL	
	Title of the document or file referenced by URL.		

KEYBOARD_ID_ID

keyboard_id	NUMBER(38)	NOT NULL	
	Counter to create unique primary keys.		

LANGUAGE

language_id	CHAR(3)	NOT NULL PK	
	Unique ISO 639 code for the representation of the name of language.		
english_name	VARCHAR2(50)	NOT NULL	~~UNIQUE~~
	English name for the language.		
native_name	VARCHAR2(50)	NULL	
	Native name for the language.		
~~amount_of_speakers~~	~~NUMBER(12)~~	~~NULL~~	
	~~Total number of speakers of this language.~~		
~~list_separator~~	~~VARCHAR2(1000)~~	~~NULL~~	
	~~List separator~~		
~~name_gender~~	~~VARCHAR2(1000)~~	~~NULL~~	
	~~Accordance with gender (e.g. Mr. Ivanov, Mrs. Ivanova)~~		
~~name_order~~	~~CHAR(1)~~	~~NULL~~	
	~~Order of first and last name. Order lastname, first name is considered true. Character value '0' denotes false, and value '1' denotes true.~~		
~~name_others_used~~	~~VARCHAR2(1000)~~	~~NULL~~	
	~~Paternal of other names used~~		
~~name_prefix~~	~~VARCHAR2(1000)~~	~~NULL~~	
	~~Prefixes used (e.g. von, van der)~~		
~~writing_system~~	~~VARCHAR2(10)~~	~~NULL~~	
	~~Writing system. Accepted values: "character", "syllable", "word"~~		
~~writing_direction~~	~~CHAR(2)~~	~~NULL~~	
	~~Coding for direction and system~~		
display_text_style	VARCHAR2(2000)	NULL	
	Description of display text style.		
punctuation	VARCHAR2(1000)	NULL	
	Description of punctuation conventions.		
gender	NUMBER(1)	NULL	
	Does the language have genders? 1 = yes, 0 = no		
plural	VARCHAR2(2000)	NULL	
	Description of the formation of plural forms in this language.		
line_breaking	VARCHAR(1000)	NULL	

Special remarks on line breaking (e.g. hyphenation) in this language.

free_text VARCHAR2(2000) NULL
Provided to store more information about this language.

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by the URL

LANGUAGE_CHARACTER (renamed from LANGUAGE_UNICODE_CHARACTER)

language_id CHAR(3) NOT NULL PK FK
Code for the representation of the name of language.

code_point NUMBER(8) NOT NULL PK FK
The code point of this character in the Unicode standard.

usage NUMBER(1) NULL
The relation between the Unicode character associated with the code point and the associated language. Values: 0 = unknown, 1 = non-essential, 2 = desirable, 3 = essential.

remarks VARCHAR2(1000) NULL
Remarks on using this character in this language.

~~alphabetical_order NUMBER(3) NULL~~
~~Order of this character in the alphabet of this language.~~

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by the URL.

LANGUAGE_CHARSET

language_id CHAR(3) NOT NULL PK FK
Code for the representation of the name of language.

character_set_id NUMBER(38) NOT NULL PK FK
Character set identifier created internally by the software.

support_type NUMBER(1) NOT NULL
Support for the representation of this language using this character set: 1 = full, 2 = partial.

notes VARCHAR2(1000) NULL
Notes on the representation of this language with this character set, e.g. limitations, usage.

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by URL

LAW (deprecated)

law_id NUMBER(38) NOT NULL PK
An unique identifier created by the CDB-software.

name VARCHAR2(100) NOT NULL UNIQUE
Name of the law

category NUMBER(2) NULL
Category of the law (copyright, product protection, etc, to be defined later)

document_file_path VARCHAR2(100) NULL
The filename and path of the document containing the actual law text.

document_file_type VARCHAR(50) NULL
The type of the document file

free_text VARCHAR2(2048) NULL
Provided for more information about the law

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

country_id CHAR(3) NULL FK
The country in which the law is applied

LAW_ID_ID (deprecated)

law_id	NUMBER(38) NOT NULL Counter to create unique primary keys.	

LOCALE

country_id	CHAR(3) NOT NULL PK FK A unique ISO 3166 country code.	
language_id	CHAR(3) NOT NULL PK FK A unique ISO 639 language code.	
windows_id	NUMBER(11) NULL The Win32 numeric locale ID ("LCID", such as 0x040C but in decimal format.).	
java_id	VARCHAR2(40) NULL The Java locale ID string.	
symbian_id	VARCHAR2(40) NULL The Symbian platform identifier (ELangFinnish etc.).	
locale_name	VARCHAR2(100) NOT NULL The display name of the locale in English.	~~UNIQUE~~
separator_list	CHAR(20) NULL The list separator used.	
separator_1000	CHAR(20) NULL Thousands separator	
separator_decimal	CHAR(20) NULL Decimal separator.	
date_format_short	VARCHAR2(200) NULL Short date format pattern (in the description language of the Java API).	
date_format_long	VARCHAR2(200) NULL Long date format pattern (in the description language of the Java API).	
time_format	VARCHAR2(200) NULL Time format (in the description language of the Java API, “short” variant HH:MM).	

clock_format	NUMBER(1)	NULL	
	0 = 12-hour format, 1 = 24-hour format		
am_symbol	CHAR(20)	NULL	
	A.M. symbol (NULL if not used in this locale)		
pm_symbol	CHAR(20)	NULL	
	P.M. symbol (NULL if not used in this locale)		
currency_format	VARCHAR2(200)	NULL	
	Currency format pattern (in the description language of the Java API). Includes negative currency format.		
iso_currency_symbol	CHAR(3)	NULL	
	A unique ISO 4217 standard currency symbol for this locale's currency.		
number_format	VARCHAR2(200)	NULL	
	Number format pattern (in the description language of the Java API). Includes negative number format.		
address_format	VARCHAR2(1000)	NULL	(deferred to Phase 2)
	Address format.		
phone_number_format	VARCHAR2(1000)	NULL	(deferred to Phase 2)
	Format of phone number		
free_text	VARCHAR2(2000)	NULL	
	Provided for more information.		
url	VARCHAR2(500)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(200)	NULL	
	Title of the document or file referenced by the URL.		

LOCALE_CALENDAR

country_id	CHAR(3)	NOT NULL	PK	FK
	A unique ISO country code.			
language_id	CHAR(3)	NOT NULL	PK	FK
	A unique ISO language code.			
calendar_id	NUMBER(38)	NOT NULL	PK	FK

	Calendar identifier created internally by the software. See CALENDAR_ID_ID.	
locale_default	NUMBER(1)	NULL
	Is this the default calendar for this locale (1 = yes, 0 = no).	
first_day_of_week	NUMBER(1)	NULL
	Ordinal number of the first day of the week (as per Win32: 1 = Monday, …, 7 = Sunday).	
week_numbers_used	NUMBER(1)	NULL
	Are week numbers used (1 = yes, 0 = no).	
first_week_of_year	VARCHAR2(1000)	NULL
	Description of how the first week of the year is defined in this calendar used in this country.	
weekday_name_1	VARCHAR2(50)	NULL
	Name of weekday 1.	
weekday_name_2	VARCHAR2(50)	NULL
	Name of weekday 2.	
weekday_name_3	VARCHAR2(50)	NULL
	Name of weekday 3.	
weekday_name_4	VARCHAR2(50)	NULL
	Name of weekday 4.	
weekday_name_5	VARCHAR2(50)	NULL
	Name of weekday 5.	
weekday_name_6	VARCHAR2(50)	NULL
	Name of weekday 6.	
weekday_name_7	VARCHAR2(50)	NULL
	Name of weekday 7.	
weekday_abbr_name_1	VARCHAR2(20)	NULL
	Abbreviated name of weekday 1.	
weekday_abbr_name_2	VARCHAR2(20)	NULL
	Abbreviated name of weekday 2.	
weekday_abbr_name_3	VARCHAR2(20)	NULL
	Abbreviated name of weekday 3.	
weekday_abbr_name_4	VARCHAR2(20)	NULL
	Abbreviated name of weekday 4.	

weekday_abbr_name_5	VARCHAR2(20)	NULL
	Abbreviated name of weekday 5.	
weekday_abbr_name_6	VARCHAR2(20)	NULL
	Abbreviated name of weekday 6.	
weekday_abbr_name_7	VARCHAR2(20)	NULL
	Abbreviated name of weekday 7.	
month_name_1	VARCHAR2(50)	NULL
	Name of month 1.	
month_name_2	VARCHAR2(50)	NULL
	Name of month 2.	
month_name_3	VARCHAR2(50)	NULL
	Name of month 3.	
month_name_4	VARCHAR2(50)	NULL
	Name of month 4.	
month_name_5	VARCHAR2(50)	NULL
	Name of month 5.	
month_name_6	VARCHAR2(50)	NULL
	Name of month 6.	
month_name_7	VARCHAR2(50)	NULL
	Name of month 7.	
month_name_8	VARCHAR2(50)	NULL
	Name of month 8.	
month_name_9	VARCHAR2(50)	NULL
	Name of month 9.	
month_name_10	VARCHAR2(50)	NULL
	Name of month 10.	
month_name_11	VARCHAR2(50)	NULL
	Name of month 11.	
month_name_12	VARCHAR2(50)	NULL
	Name of month 12.	
month_name_13	VARCHAR2(50)	NULL
	Name of month 13.	

month_abbr_name_1 VARCHAR2(20) NULL
Abbreviated name of month 1.

month_abbr_name_2 VARCHAR2(20) NULL
Abbreviated name of month 2.

month_abbr_name_3 VARCHAR2(20) NULL
Abbreviated name of month 3.

month_abbr_name_4 VARCHAR2(20) NULL
Abbreviated name of month 4.

month_abbr_name_5 VARCHAR2(20) NULL
Abbreviated name of month 5.

month_abbr_name_6 VARCHAR2(20) NULL
Abbreviated name of month 6.

month_abbr_name_7 VARCHAR2(20) NULL
Abbreviated name of month 7.

month_abbr_name_8 VARCHAR2(20) NULL
Abbreviated name of month 8.

month_abbr_name_9 VARCHAR2(20) NULL
Abbreviated name of month 9.

month_abbr_name_10 VARCHAR2(20) NULL
Abbreviated name of month 10.

month_abbr_name_11 VARCHAR2(20) NULL
Abbreviated name of month 11.

month_abbr_name_12 VARCHAR2(20) NULL
Abbreviated name of month 12.

month_abbr_name_13 VARCHAR2(20) NULL
Abbreviated name of month 13.

purpose VARCHAR2(1000) NULL
The purpose of the use of this calendar in the associated locale (civil, religious etc.).

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL

Title of the document or file referenced by URL

LOCALE_KEYBOARD

country_id	CHAR(3)	NOT NULL	PK FK
	A unique ISO 3166 country code.		
language_id	CHAR(3)	NOT NULL	PK FK
	A unique ISO 639 language code.		
keyboard_id	NUMBER(38)	NOT NULL	PK FK
	Keyboard identifier created internally by the software.		
free_text	VARCHAR2(2000)	NULL	
	Provided for more information.		
url	VARCHAR2(500)	NULL	
	URL to refer to where a document or file can be found.		
url_title	VARCHAR2(200)	NULL	
	Title of the document or file referenced by the URL.		

OPERATOR

~~operator_id~~	~~NUMBER(38)~~	~~NOT NULL~~	~~PK~~
	~~Operator identifier created internally by the CDB-software~~		
~~operator_name~~	~~VARCHAR2(100)~~	~~NOT NULL~~	~~UNIQUE~~
	~~Name of operator~~		
~~free_text~~	~~VARCHAR2(2048)~~	~~NULL~~	
	~~Provided for more information~~		
~~url~~	~~VARCHAR2(512)~~	~~NULL~~	
	~~URL to refer to where a document or file can be found.~~		
~~url_title~~	~~VARCHAR2(256)~~	~~NULL~~	
	~~Title of the document or file referenced by URL~~		

OPERATOR_ID_ID

~~operator_id~~	~~NUMBER(38)~~	~~NOT NULL~~
	~~Counter to create unique primary keys.~~	

NOKIA

1.2.1 REGION

region_id NUMBER(38) NOT NULL PK
Region identifier created internally by the CDB-software

region_name VARCHAR2(50) NOT NULL UNIQUE
Name of region

free_text VARCHAR2(2048) NULL
Provided for more information

url VARCHAR2(512) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(256) NULL
Title of the document or file referenced by URL

emp_no CHAR2(20) NULL FK
NMP contact person in region

REGION_ID_ID

region_id NUMBER(38) NOT NULL
Counter to create unique primary keys.

RELIGION

religion_id NUMBER(38) NOT NULL PK
Religion identifier created internally by the CDB-software

religion_name VARCHAR2(100) NOT NULL UNIQUE
Name of religion

religion_description VARCHAR2(1000) NULL
Description of religion

number_of_people NUMBER(11) NULL
Number of people in this religion

behavior VARCHAR2(1000) NULL
Description about behavior in this religion

free_text VARCHAR2(2048) NULL
Provided for more information

NOKIA

~~url VARCHAR2(512) NULL~~
~~URL to refer to where a document or file can be found.~~

~~url_title VARCHAR2(256) NULL~~
~~Title of the document or file referenced by URL~~

RELIGION_ID_ID

~~religion_id NUMBER(38) NOT NULL~~
~~Counter to create unique primary keys.~~

RELIGION_ COLOUR

~~religion_id NUMBER(38) NOT NULL PK FK~~
~~A unique religion identifier created internally by the CDB-software~~

~~colour_id NUMBER(38) NOT NULL PK FK~~
~~Colour identifier created internally by the CDB-software~~

~~colour_meaning VARCHAR2(1000) NOT NULL~~
~~Meaning of colour in this religion~~

~~free_text VARCHAR2(2048) NULL~~
~~Provided for more information about the colour in this religion~~

~~url VARCHAR2(512) NULL~~
~~URL to refer to where a document or file can be found.~~

~~url_title VARCHAR2(256) NULL~~
~~Title of the document or file referenced by URL~~

RELIGION_ GRAPHIC

~~religion_id NUMBER(38) NOT NULL PK FK~~
~~A unique religion identifier created internally by the CDB-software~~

~~graphic_id NUMBER(38) NOT NULL PK FK~~
~~Graphic identifier created internally by the CDB-software~~

~~graphic_meaning VARCHAR2(1000) NOT NULL~~
~~Meaning of graphic in religion~~

~~free_text VARCHAR2(2048) NULL~~
~~Provided for more information about the graphic in religion~~

~~url VARCHAR2(512) NULL~~
~~URL to refer to where a document or file can be found.~~

~~url_title VARCHAR2(256) NULL~~
~~Title of the document or file referenced by URL~~

RELIGION_TABOO

~~religion_id NUMBER(38) NOT NULL PK FK~~
~~Religion identifier created internally by the CDB-software~~

~~taboo_id NUMBER(38) NOT NULL PK FK~~
~~Taboo identifier created internally by the CDB-software~~

~~taboo_meaning VARCHAR2(1000) NOT NULL~~
~~Meaning of the taboo~~

~~url VARCHAR2(512) NULL~~
~~URL to refer to where a document or file can be found.~~

~~url_title VARCHAR2(256) NULL~~
~~Title of the document or file referenced by URL~~

SCRIPT

script_name CHAR(20) NOT NULL PK
The name of the script according to the Unicode Technical Report #24.

script_type NUMBER(1) NOT NULL
The type of (the characters in) the script, according to the following categorisation: 1 = alphabetic, 2 = syllabic, 3 = semi-syllabic, 4 = ideographic, 5 = uncategorized.

writing_order NUMBER(3) NOT NULL
The writing order of the script, according to the following categorisation: 0x00 = unknown, 0x01 = LRTB, 0x02 = RLTB, 0x04 = TBLR, 0x08 = TBRL. (Legend: L = left, R = right, T = top, B = bottom.) If you wish to indicate that several of these are possible, use a logical

OR of the values, e.g. both TBRL and LRTB is specified as 0x04 OR 0x01 = 0x05.

csglyphs NUMBER(1) NOT NULL
Does this script use context-sensitive glyphs: 1 = yes, 0 = no.

csglyphdesc VARCHAR2(1000) NULL
If csglyphs = 1, include a description of the context-sensitive glyphs, including the use of ligatures and glyph selection.

line_breaking NUMBER(1) NULL
The line breaking convention of this script: 1 = delimited, 2 = free (unless prohibited), 3 = dictionary-based.

native_digit_use VARCHAR2(1000) NULL
Description of the use of native digits in this script. If native digits are used, include their Unicode range and information about typical conventions.

case_distinction NUMBER(1) NOT NULL
Does this script have distinct upper and lower case characters (1 = yes, 0 = no).

width_distinction NUMBER(1) NOT NULL
Does this script use separate full-width and half-width characters (1 = yes, 0 = no).

composition_rules VARCHAR2(1000) NULL
If this script is compositional, include a description of the rules used in composing the characters.

free_text VARCHAR2(2000) NULL
Optional field used to provided additional information. If the script name is COMMON or INHERITED, provide an explanation.

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by the URL.

SCRIPT_LANGUAGE

script_name CHAR(20) NOT NULL PK FK
The name of the script associated with this language.

language_id	CHAR(3)	NOT NULL PK FK
	The ISO language code for the associated language.	
free_text	VARCHAR2(2000)	NULL
	Optional field used to provided additional information.	

TABOO

~~taboo_id~~	~~NUMBER(38)~~	~~NOT NULL PK~~
	~~Taboo identifier created internally by the CDB-software~~	
~~taboo_name~~	~~VARCHAR2(100)~~	~~NOT NULL UNIQUE~~
	~~Name of taboo~~	
~~free_text~~	~~VARCHAR2(2048)~~	~~NULL~~
	~~Provided for more information~~	
~~url~~	~~VARCHAR2(512)~~	~~NULL~~
~~URL to refer to where a document or file can be found.~~		
~~url_title~~	~~VARCHAR2(256)~~	~~NULL~~
	~~Title of the document or file referenced by URL~~	
~~graphic_id~~	~~NUMBER(38)~~	~~NULL FK~~
	~~Graphic identifier created internally by the CDB-software~~	

TABOO_ID_ID

~~taboo_id~~	~~NUMBER(38)~~	~~NOT NULL~~
	~~Counter to create unique primary keys.~~	

UI_FIELD

table_name	VARCHAR2(50)	NOT NULL PK FK
	The actual table name in the database.	
field_name	VARCHAR2(50)	NOT NULL PK
	The actual table column name in the database.	
ui_field_name	VARCHAR2(50)	NOT NULL
	The column name to be shown in UI.	
description	VARCHAR2(100)	NULL

Description of the field.

UI_TABLE

table_name	VARCHAR2(50)	NOT NULL PK The actual table name in the database.
ui_table_name	VARCHAR2(50)	NOT NULL The table name to be shown in UI.
description	VARCHAR2(100)	NULL Description of the table.

UNICODE_CHARACTER

~~unicode_value~~	~~CHAR(4)~~	~~NOT NULL PK~~ ~~The value of a Unicode character.~~
~~unicode_name~~	~~VARCHAR2(50)~~	~~NOT NULL~~ ~~Name of Unicode character.~~
~~unic_pic_loc~~	~~CHAR(1000)~~	~~NULL~~ ~~Access path and file name to locate the picture of the Unicode character.~~
~~unic_pic_ftype~~	~~VARCHAR2(100)~~	~~NULL~~ ~~Type of the file containing the Unicode character picture.~~

UPDATE_LOCK

table_name	VARCHAR2(50)	NOT NULL PK The table that is being updated.
user_id	CHAR(10)	NOT NULL PK Login name of the user doing the update.
id1	CHAR(10)	NOT NULL PK Id1 of the row that is being updated, if a new row is being added id1 = "foo"
id2	CHAR(10)	NOT NULL PK Id2 of the row that is being updated, if a new row is being added or table has only one id id2 = "foo"
id3	CHAR(10)	NOT NULL PK

Id3 of the row that is being updated, if a new row is being added or table has only one or two ids id3 = foo

time_stamp VARCHAR2(20) NOT NULL
Timestamp of when the update lock has been set.

USER (formerly NMP-EMPLOYEE)

username CHAR(40) NOT NULL PK
Pandora user name.

password VARCHAR2(100) NOT NULL
User password.

emp_no CHAR(20) NULL
NMP unique employee identifier.

email VARCHAR2(100) NULL
The e-mail address of the user.

phone_number VARCHAR2(50) NULL
Phone number of the user.

user_level NUMBER(3) NOT NULL
0 = user, 10 = administrator (allows for intermediate user levels).

first_name VARCHAR2(50) NULL
First name of the employee.

last_name VARCHAR2(100) NOT NULL
Last name of the employee.

~~expertise VARCHAR2(1000) NULL~~
~~Area of expertise~~

url VARCHAR2(500) NULL
URL to refer to where a document or file can be found.

url_title VARCHAR2(200) NULL
Title of the document or file referenced by the URL.

USER_CDB (deprecated)

user_id	CHAR(10)	NOT NULL	PK
	Login name of user		
email	VARCHAR2(100)	NULL	
	e-mail address		
phone_number	VARCHAR2(50)	NULL	
	Phone number of user		
name	VARCHAR2(100)	NOT NULL	
	Name of the user		
passwd	CHAR(10)	NOT NULL	
	User password		
access_right	NUMBER(1)	NOT NULL	
	Level of the user rights.		
emp_no	CHAR(20)	NULL	FK
	NMP employee number		

INTEGRITY CONSTRAINTS

All constraints have a unique name.

DOMAIN CONSTRAINTS

Table column domain datatype constraints are defined in the DDL statements that create tables. Column domain constraints are described in section 3.1.

PRIMARY KEY CONSTRAINTS

Primary key constraints are defined in the DDL statements that create tables. Primary key columns are described in section 3.1.

REFERENTIAL CONSTRAINTS

Referential constraints on foreign keys are defined in the DDL statements that create tables. Foreign key columns are described in section 3.1. Foreign keys have been defined with the ON DELETE CASCADE or ON DELETE SET NULL option to maintain referential integrity

when deleting referenced values in the parent table that have dependent rows in the child table, thus allowing Oracle to automatically delete rows from the child table.

UNIQUE CONSTRAINTS

Table column unique constraints are defined in the DDL statements that create tables. Column unique constraints are described in section 3.2.

CHECK CONSTRAINTS

Check constraint names, validation rules and descriptions.

CHK_LANG_ALPHABET11

~~language_alphabet IN('0','1')~~

~~Valid characters in table LANGUAGE_UNICODE_CHARACTER, column language_alphabet. Character belongs to the alphabets of the language ('1') or not ('0').~~

CHK_OFFICIAL_LANGUAGE11

~~official_language IN('0','1')~~

~~Valid characters in table COUNTRY_LANGUAGE, column official_language. Language is the official language of the country ('1') or not ('0').~~

CHK_WEEK_NUMBER_USE11

~~week_number_use IN('0','1')~~

~~Valid characters in table COUNTRY_CALENDAR, column week_number_use are '0' or '1'.~~

CHK_WRITING_SYSTEM

~~writing_system IN('character','syllable','word')~~

~~Valid character strings in table LANGUAGE, column writing_system are "character", "syllable" or "word".~~

Auxiliary objects

The following Oracle-specific objects have been created in the database. These SEQUENCE objects can be used to create unique primary key values.

calendar_id_seq

charset_id_seq

~~colour_id_seq~~

graphic_id_seq

~~law_id_seq~~

~~operator_id_seq~~

~~region_id_seq~~

~~religion_id_seq~~

~~taboo_id_seq~~

JK: If we have a sequence created with "CREATE SEQUENCE calendar_id_seq", what is the purpose of the CALENDAR_ID_ID table? In my opinion they are redundant; when inserting a new calendar, you just use INSERT and pass calendar_id_seq.NEXTVAL as the primary key. The same applies to any other table with a generated PK. What am I missing?

INDEXES

All primary key columns (PK) and foreign key columns (FK) are indexed.

DATABASE BACKUP

The Oracle8i Export is used as the database backup method, and it is run manually. The Export utility can selectively export specific database objects. Backups taken by Export are done in Full Database Mode and User Mode.

IMPORTING DATA

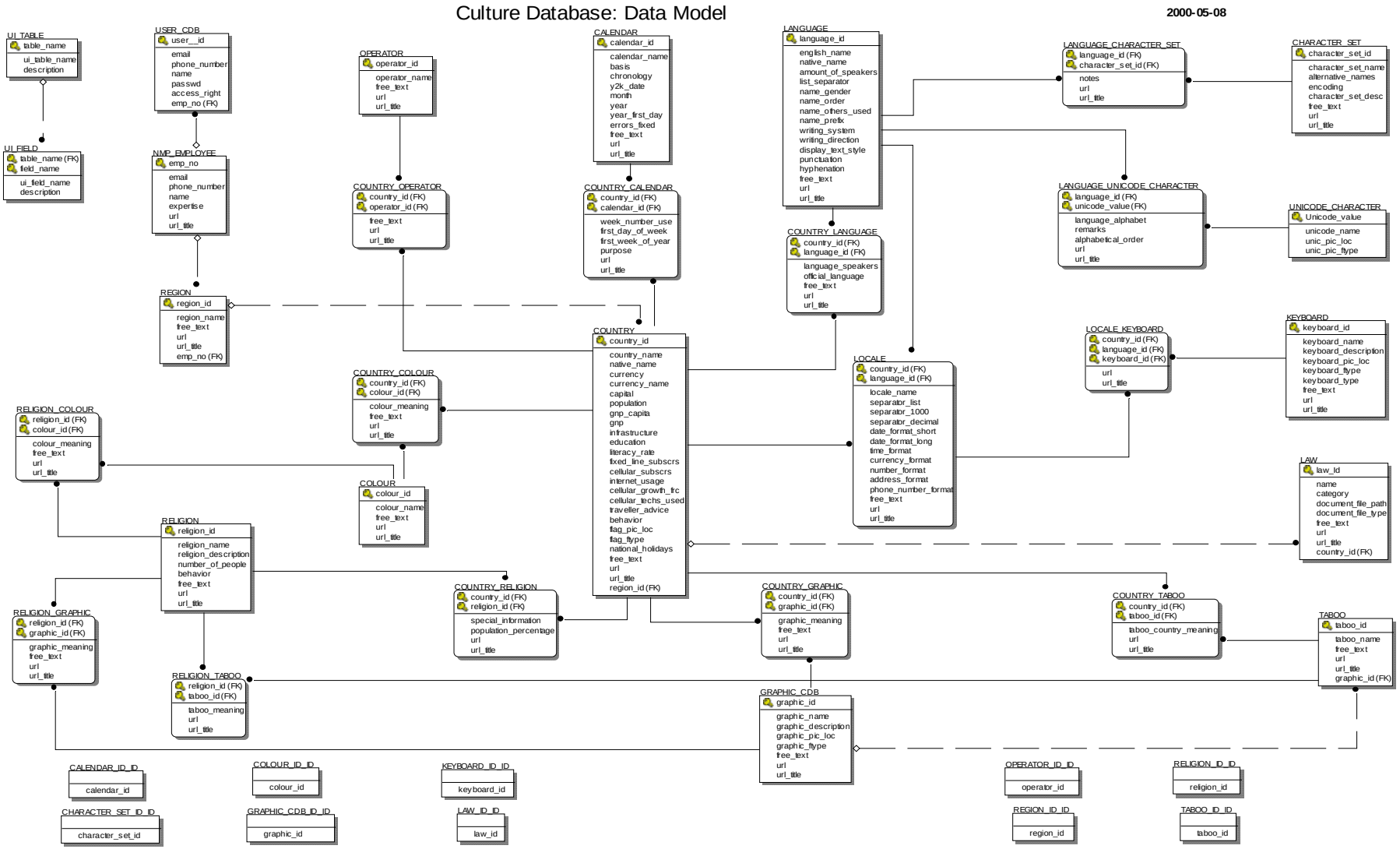
Delimited data can be imported to the database using the Oracle SQL*Loader utility, described in "Oracle 8i Utilities" (Part Number A76955-01).

NMP

Sakari Hokkanen

APPENDICES

APPENDIX A



NOKIA

REFERENCES

Document References

Pandora Project Plan

Culture Database Requirements Specification

Pandora Functional Specification

Pandora Technical Specification

“Oracle8i Enterprise Edition Release 8.1.7 for Windows NT” system documentation.

Book References

Using Oracle8, Special Edition, William G. Page Jr and Nathan Hughes, 1998

Other References

http://www.oracle.com/

NMP
Sakari Hokkanen

LIITE 9: Pandora Technical Specification

Version: 1.3

State: Approved

Author: Jere Käpyaho, Sakari Hokkanen, (Pekka Mäkiaho)

Version history:

Version	Date	Description
0.1	2000-03-08	First draft version
0.2	2000-03-16	Second draft version
0.5	2000-03-21	Third draft version
0.6	2000-05-17	Fourth draft version
0.7	2000-05-22	First proposal
1.0	2000-05-25	Approved by the project steering group.
1.1	2000-07-10	Updated the module descriptions due to changes in code.
1.2	2001-12-07	Updated to reflect major changes in Pandora Phase 1 architecture.
1.3	2003-12-06	Nokia confidential information removed

NOKIA

GLOSSARY

CSS	Cascading Style Sheets
CVS	Concurrent Versions System
HTML	Hypertext Markup Language
IRP	Internet Response Time
JCVS	Java Concurrent Versions System
JDBC	Java Database Connectivity
JDK	Java Development Kit
NMP	Nokia Mobile Phones
SQL	Structured Query Language
UI	user interface
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language
XSLT	XSL Transformations
XSQL	Oracle technology for expressing SQL queries in XML

INTRODUCTION

Purpose of this document

This document outlines the technical design of Pandora (formerly Culture Database). It explains the architecture and technologies used in the implementation. The intended audience of this document consists of the Pandora team and the clients of the project, i.e. developers and users of Pandora at NMP.

Pandora

Pandora is a database application for storing and retrieving information about cultural conventions needed in NMP product development. The prototype of the application was originally developed as an assignment for the Programming Project and Software Project Management courses at the Department of Computer and Information Sciences, University of Tampere during the academic year 1999-2000. Further information about the prototype is in the project documentation, in document files named cdb_xxxx.doc.

The project was renamed Pandora and further developed by Sakari Hokkanen until August 2000. Further development was discontinued until early 2001, when interest in the project was revived. It was decided to implement Pandora in two phases, Phase 1 with limited database content and functionality, and Phase 2 with additional content and features.

Major design constraints and limitations

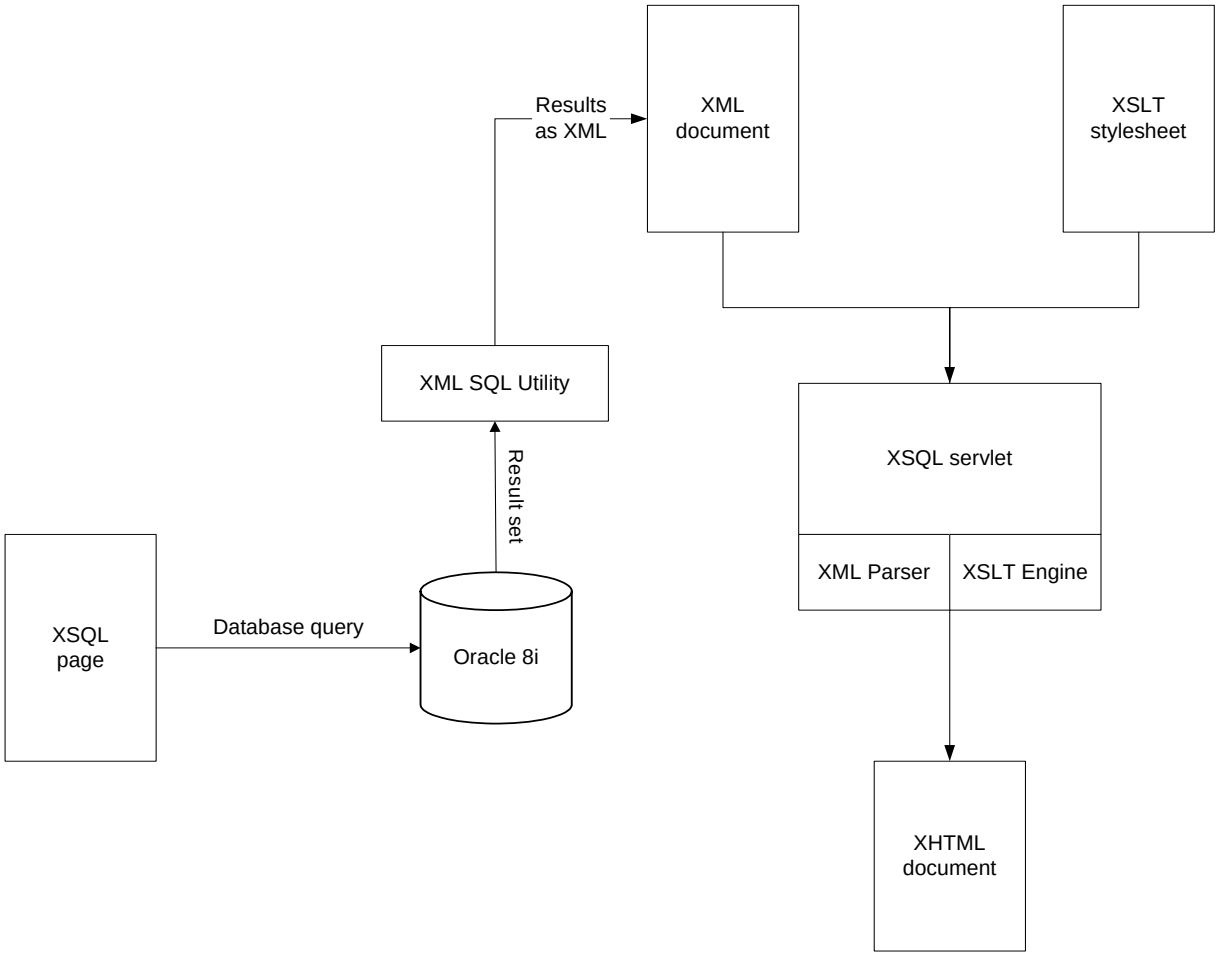
Pandora is an intranet application, running in the NMP intranet. It is designed to be accessed using a World Wide Web browser, mostly from a Windows NT 4.0 or Windows 95/98/Me environment. The primary client platform is Windows NT 4.0, while the target browsers are Netscape Navigator 4.0 or later and Internet Explorer 4.0 or later. No special browser plug-ins are required to use the Pandora WWW interface, but a font with glyphs for most of the characters in the Unicode standard is essentially required for the client.

The back-end of the application is an Oracle 8i database with UTF-8 as the database character set. This specification of character set is very important, since the database will have content in various

languages, and the only (practical) way they can be mixed on the same HTML page is to use UTF-8. Database content is stored as UTF-8-encoded text with the exception of references to external binary files, which are stored as is. A text column in the corresponding database field contains the relative filename.

The user interface utilises HTML 4.01 Transitional, with presentational elements avoided whenever possible. Instead the presentation will be controlled using Cascading Style Sheets, Level 1 (CSS). The Pandora User Interface Specification describes this in more detail.

The user interface is constructed programmatically using Oracle XSQL pages and XSL transformations. An XSQL page is an XML document containing an embedded SQL query to the database. The Oracle XSQL engine translates the results of the query to an XML document containing rowsets and rows with elements corresponding to the fields of the queried table(s). This result document is then further transformed into HTML 4.01 by the Oracle XSLT processor using an XSL stylesheet. A conceptual diagram of this process follows.



For a more detailed explanation of the sequence of events, please refer to the "XSQL Servlet" section of /5/.

The prototype of Culture Database used Java servlets exclusively to extract data from the database and transform it to HTML. The use of XSQL and XSLT dramatically reduces the programming effort required. The same result can be achieved essentially without programming, using declarative XSLT statements.

In cases where XSQL and XSLT do not provide enough power of expression, Java extensions to XSLT, shall be used as middleware components.

The following versions of Java technology and related APIs are used:

- The Java programming language: Java 2, version 1.2.2
- The JDBC Data Access API: 1.1

The JDBC Data Access API is used for communication between Oracle XSQL and the database.

Since the database specification has long since sacrificed independence of Oracle, the queries issued in XSQL pages may use Oracle-specific features of SQL.

The WWW server component used is the Oracle HTTP Server, which is based on Apache 1.3 for Windows NT. In addition to the Web server, the Oracle XSQL servlet v9.0.1.0.0 is required to use XSQL pages.

More information on the techniques mentioned above can be found in the documents listed in References.

DESIGN DESCRIPTION

Architecture overview

The Pandora technical architecture consists of the following parts:

1. The user interface, implemented as a set of HTML pages.
2. The middleware, implemented as XSQL pages, XSLT styleheets and Java XSLT extensions.
3. The database back-end.

The user interface consists of dynamically generated HTML pages. Embedded hyperlinks in the user interface point to XSQL page URLs on the server. When the user selects a link, the request is sent to the WWW server and directed to invoke the Oracle XSQL servlet. The Oracle XSQL servlet engine interprets the XSQL page and sends the query or queries to the database.

The XSQL processor formulates the result set of the query as an XML document, which is then handed over to the XSLT engine. The XSLT engine transforms the XML document into HTML using stylesheets. The resulting HTML page is then delivered to the web server. The web server then sends the HTML page to the client.

Design and implementation viewpoints

Portability

The original goal of the Culture Database prototype was to make the application as generic as possible with regard to databases and servers. However, the database design has required the use of Oracle-specific features which are reflected in the application. Because the application was already tied to Oracle anyway, the decision was made to use Oracle-specific tools such as XSQL to get the most of the

Oracle 8i database. Similar XML-based publishing solutions exist, notably the Apache Project's Cocoon. However, The Oracle XSQL utility can also be used with other web servers besides the Oracle HTTP Server.

Version control

Phase 1 of the Pandora project used no version control system. The reason for this was that there was only a single developer, and the application size was not very big.

Programming conventions

All configurable server names and port numbers shall be placed in external configuration files instead of source files.

All Java code shall follow the programming conventions found in the document “Java Coding Conventions” by JavaSoft (see Other References section).

Documentation

The XSQL pages and XSLT files shall have in-source comments about the purpose of the file, including author information and modification log.

Java source code shall contain JavaDoc documentation comments. Source documentation shall be generated using the JavaDoc tool included in the Java SDK. The latest JavaDoc documentation shall accompany this document (see Appendix 5).

Deployment

In order to deploy Pandora Phase 1, the XSQL and XSLT files need to be copied to the application root directory. This shall be a directory called `pandora` on the web server, although it can be a virtual directory.

The Oracle XSQL servlet needs to be configured with the Pandora database address and user information in the configuration file "XSQLConfig.xml", located in the directory %ORACLE_HOME%\lib. (If there is no %ORACLE_HOME%, look in xdk\lib under the Oracle home directory.) Since this file contains the database user name and password, care must be taken not to expose this file or any other configuration files to clients via the web server.

The configuration information is added to the "connectiondefs" element as follows:

```
<connection name="ph1test">
  <username>pandora</username>
  <password>box</password>
  <dburl>jdbc:oracle:thin:@trepc1744:1521:cdbphs1</dburl>
  <driver>oracle.jdbc.driver.OracleDriver</driver>
</connection>
```

The Java XSLT extensions use a configuration file called "pandoraext.properties" to read the JDBC connect information. This file is currently located in the root directory of the server machine. Need to find out where to put it and update the XSLT extension code. The contents of the file are as follows:

```
database=jdbc:oracle:thin:@trepc1744:1521:cdbphs1
user=pandora
password=box
```

This is essentially the same information as in the XSQLConfig.xml file, but it needs to be duplicated because the Java XSLT code cannot easily get at the XSQL configuration. (It would be possible, however, to use an XML parser to read the file. Maybe in a future version.)

The JServ part of the Oracle HTTP Server also needs to be configured. The file "jserv.properties" in the directory "%ORACLE_HOME%\Apache\jserv\conf" needs an additional line:

```
wrapper.classpath=C:\Oracle\Ora81\xdk\lib\pandoraext.jar
```

Replace "C:\Oracle\Ora81" with the correct value of %ORACLE_HOME% if necessary.

Also check that the wrapper.classpath values for "Oracle XML SQL Components for Java" and the XSQLConfig.xml file location are correct for XDK 9.0.0.1, depending on where it was installed. (It may be found in exotic locations such as %ORACLE_HOME%\rdbms\jlib. Just make sure it works.)

Remember to restart the Oracle HTTP Server after these configuration changes (go to Control Panel, select Services, select OracleOraHome81HTTPServer, click Stop, then click Start).

COMPONENTS

The structure of the Pandora application has been changed significantly since the Culture Database prototype. What was previously handled with Java servlets alone is now achieved using XSQL pages and XSLT stylesheets. This section outlines the components that make up the Pandora architecture.

XSQL pages

The application uses XSQL pages to extract data from the database. The XSQL pages contain embedded SQL queries that are transformed into an XML dataset by the Oracle XML SQL Utility. Each XSQL page can contain multiple SQL queries, with the results embedded into one common root element.

XSLT stylesheets

The XML datasets are transformed into HTML 4.01 by the XSQL page processor, using XSLT stylesheets. These stylesheets contain declarative instructions for replacing XML elements with HTML markup.

XSQL/XSLT combinations

Each identified category of data has a combined pair of an XSQL page and an XSLT stylesheet to be called upon by the application logic. These data categories are acessible through the user interface using hypertext links. The targets of these links are XSQL pages with the extension .xsql. They are mapped to the Oracle XML SQL Utility in the servlet engine configuration. The servlet engine invokes the XML SQL Utility to process the queries. The XML SQL Utility in turn invokes the XSLT engine to transform the XML data according to the instructions in the stylesheets.

XSLT extensions

Some XSLT stylesheets use Java extension classes to achieve computations that are not possible in XSLT. These extensions shall be placed in their own Java package and made available in the classpath of the Java servlet engine. They are invoked from the XSLT stylesheet.

NOKIA

REFERENCES

/1/ Pekka Mäkiaho, Pandora Requirements Specification

/2/ Jere Käpyaho, Pandora User Interface Specification 1.3

/3/ *NN*, Pandora Functional Specification

/4/ Jere Käpyaho, Pandora Database Specification 1.2

/5/ Using XML in Oracle Database Applications, Part 2: About Oracle XML Products. Oracle Technology Network, http://otn.oracle.com/tech/xml/info/htdocs/otnwp/about_oracle_xml_products.htm (last accessed 2001-07-17)

/6/ Steve Muench, Building Oracle XML Applications. O'Reilly, 2000.

/7/ Oracle Technology Network: Oracle XML Development Kit. http://otn.oracle.com/tech/xml/ (last accessed 2001-07-17)

/8/ JavaSoft, Java Coding Conventions. http://java.sun.com/docs/codeconv/ (last accessed: 2000-03-06)

/9/ Java Development Kit, JavaSoft. http://java.sun.com/jdk/ (last accessed: 2000-03-06)

/10/ Java Servlet Development Kit, JavaSoft. http://java.sun.com/products/servlets/ (last accessed: 2000-03-06)

NMP
Sakari Hokkanen

LIITE 10: Pandora Phase 2 Feature List

Version: 0.3

Status: FINAL

Author: Jere Käpyaho, Pekka Mäkiaho (Version 0.3)

Version history:

Version	Date	Description
0.1	2000-10-04	First draft version
0.2	2001-11-27	Added language variant scheme.
0.3	2003-04-22	Nokia confidential information removed

NMP
Sakari Hokkanen

NOKIA

INTRODUCTION

This document lists the planned new features of Pandora Phase 2. Some of the features are carried over from Phase 1, and some are completely new. For information about Phase 1, please see the corresponding specifications.

The technical requirements of adding these features are not considered in this document. That will be in the domain of the Phase 2 requirements specifications.

USER COMMENTS

A preliminary version of Pandora was released and comments about the user interface were requested. We received only a few comments, and mostly about the content. This section lists wishes from those users who tried out this early version of Pandora Phase 1.

Content

Request	Date that information was updated. Has it been validated by language professionals/Nokia marketing validators?
Response	Update time: this either requires a field or fields in all tables, or it could be extracted from the database metadata, which could possibly contain information about the last change to the data. Validation: would require a list of validators used and adding a field for it to each data item. Too extensive.

Request	Information about the use of T9 predictive text input in a given language (used/not used)
Response	Definitely worth adding a field for this to the LANGUAGE table of the database.

Request	Store of test data for T9 (doesn't need to be big): a list of words that functional testers could use when testing a language that they do not know.
Response	Could be included if such test data is already available. Requires more study.

Request	Store of test data for testing character sorting (e.g. for phonebook application). Could include native **and** foreign names.
Response	One planned feature of Phase 2 is a generic sorting apparatus. See below.

Request	List of meanings of the main colours that Nokia may use in a phone for symbols and icons.
Response	Colour data were part of the Culture Database prototype, but they were dropped from Pandora along with other similar data (such as taboos) because of the effort to collect them was considered too extensive.

Request	Information about the implementation status of different scripts in different platforms, including keyboard layouts and input methods used.
Response	Useful information that could even be easily collected and a field added to the SCRIPT table. The link to keyboard information requires study. Need to create a table for input methods (T9/ITU-T/stylus etc.)?

Request	Description of the effects of a selected language in a phone, e.g. editors.
Response	Out of scope.

User interface

Request	Ability to print out the locale information for testing purposes.
Response	Any page of Pandora can be printed using the web browser's built-in printing features. The printout varies somewhat between Internet Explorer and Netscape Communicator. This could be improved with style sheets, although older browser versions probably do not support those parts of CSS.

Request	Explanations for unfamiliar terms in the user interface, i.e. a glossary.
Response	A glossary would indeed be a useful addition to the online help.

PLANNED NEW FEATURES

This section lists those new features that were planned by the Pandora team.

Content

Character/language relation: the list of desired characters in a language needs to be collected, in addition to the essential characters that are already in Phase 1.

Character set: need to add the characters that each set contains, along with their Unicode code points and their codes in the character set. This may be feasible only for 8-bit single-byte character sets…

Locale: need to add information about formatting phone numbers (including grouping and separator characters for both domestic and international phone numbers).

User interface

Sorting

A tool that allows the user to upload an arbitrary text file to the Pandora server for sorting. The sorter, implemented as a Java servlet, returns the contents of the file sorted by HTTP so that the user can view it and possibly save it on their workstation.

The sorting servlet also needs to take into account the character encoding of the file. Since this probably cannot be detected automatically, allow the user to select it from a list before uploading the file.

Administration

The administration tool will probably be moved from Phase 1 to Phase 2, or implemented somewhere between them.

FUTURE ENHANCEMENTS

Language variant support

We need to support variants of languages, such as Simplified and Traditional Chinese. These are actually (technically) variants of the script, but there are also issues with vocabulary.

Language variants could be implemented with a table that describes the association between language and variant, for example:

LANGUAGE_VARIANT
language_id
variant_id
display_name
description
etc.

This would require an addition to the LOCALE table: a "variant_id" field would identify which language variant is associated with the country.

There will also be implications for character sets, but no other change needs have been detected at the time of writing.