Lappeenrannan teknillinen yliopisto
Teknistaloudellinen tiedekunta
Tietotekniikan koulutusohjelma

Diplomityö

Timo Rauta

TIEDONHALLINTAJÄRJESTELMÄN KÄYTTÖÖNOTTO JA SOVELLUSTEN INTEGROINTI PROSESSITEOLLISUUDEN SUUNNITTELU- JA KONSULTOINTIYRITYKSESSÄ

Työn tarkastajat: Professori Jari Porras
Diplomi-insinööri Timo Kiminki

Työn ohjaaja: Diplomi-insinööri Timo Kiminki

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto
Teknistaloudellinen tiedekunta
Tietotekniikan koulutusohjelma

Timo Rauta

Tiedonhallintajärjestelmän käyttöönotto ja sovellusten integrointi prosessiteollisuuden suunnittelu- ja konsultointiyrityksessä

Diplomityö

2010

99 sivua, 20 kuvaa, 4 taulukkoa ja 1 liite

Tarkastajat: Professori Jari Porras
Diplomi-insinööri Timo Kiminki

Hakusanat: tiedonhallintajärjestelmä, tietokannanhallintajärjestelmä, tietokanta, käyttöönotto, sovellusintegraatio, ALMA

Kilpailukykyisyyden säilyttäminen alati kehittyvillä markkinoilla vaatii ajanmukaisten tietojärjestelmien käyttöä. Eräs tärkeimmistä tällaisista järjestelmistä on organisaatiossa käytössä oleva tiedonhallintajärjestelmä, jota käytetään yrityksessä kulutettavan ja tuotettavan tiedon hallitsemiseen. Käytössä olevan tiedonhallintajärjestelmän vaihtaminen modernimpaan on monimutkainen prosessi, joka alkaa uuden järjestelmän valinnasta ja jatkuu järjestelmän käyttöönottamisella. Käyttöönottoon kuuluu tarpeellisten vanhojen sovellusten integroiminen osaksi uutta järjestelmää sekä käyttäjien kouluttaminen uuden järjestelmän vaatimiin työtapoihin. Diplomityössä on perehdytty uuden ALMA-tiedonhallintajärjestelmän käyttöönottoon prosessiteollisuuden suunnittelu- ja konsultointiyritys CTS Engtec Oy:ssä. Työn puitteissa liitettiin kaksi CTS:llä käytössä olevaa sovellusta, CTS Pine ja PMMATE, osaksi uutta tiedonhallintajärjestelmää. Lisäksi työssä on tutustuttu tiedonhallintajärjestelmiin liittyviin käsitteisiin.

ABSTRACT

Lappeenranta University of Technology
Faculty of Technology Management
Degree Program of Information Technology

Timo Rauta

Commissioning of a information management system and integration of applications in an engineering and consulting company of the process industry

Master's thesis

2010

99 pages, 20 figures, 4 tables and 1 appendice

Examiners: Professor Jari Porras
M. Sc. (Tech) Timo Kiminki

Keywords: information management system, database management system, database, commissioning, application integration, ALMA

In order to remain competitive in the constantly developing market, the use of contemporary information systems is required. One of the most important systems of this kind is the information management system of the company, which is used to manage the data both produced and consumed by the organization. The replacement of information management system currently in use to a more modern one is a complicated process, which begins from the selection of a new system and continues to a commissioning of that system. The commissioning includes integrating the old applications to the new system and training the users to accustom the new working habits required by it. This master's thesis considers the commissioning of new ALMA information management system in the engineering and consulting company of process industry, CTS Engtec Oy. During the course of this thesis, two applications used in CTS, CTS Pine and PMMATE, were integrated as a part of the new information management system. The thesis also covers the concepts related to the information management systems.

ALKUSANAT

Tämän diplomityön tiimoilta haluan kiittää CTS Engtec Oy:n Timo Kiminkiä, Jukka Kannelta ja Antti Lukkaa, joiden ansiosta minulle avautui mahdollisuus työn tekemiseen. Erityiskiitokset Timo Kimingille työn ohjauksesta, tarkistamisesta sekä muusta vaivannäöstä vuosien varrella. Kiitos myös kaikille työtovereilleni, jotka teitte ajastani CTS Engtec Oy:llä mieluisan kokemuksen.

Lappeenrannan teknillisen yliopiston henkilökunnasta haluaisin kiittää professori Jari Porrasta työn ohjauksesta ja tarkistamisesta. Viimeisimpänä, muttei vähäisimpänä, kiitokset perheelleni, ystävilleni ja kaikille heille, jotka ovat tukeneet ja kannustaneet minua opintojeni etenemisessä.

Kouvolassa 10.6.2010
Timo Rauta

SISÄLLYSLUETTELO

LYHENNELUETTELO

ACID	Atomicity, Consistency, Isolation, Durability. Tietokantoihin liittyvien transaktioiden perusominaisuudet.
API	Application Programming Interface. Sovelluksen tarjoama rajapinta, joka mahdollistaa sovellusten välisen interaktion.
ANSI/SPARC	American National Standards Institute / Standards Planning and Requirement Committee.
CSS	Cascading Style Sheets. Tyylisivukieli, jonka avulla voidaan määritellä jollakin kuvauskielellä määritellyn dokumentin ulkoasu.
DBA	Database Administrator. Henkilö, joka ylläpitää ja hallinnoi tietokantaratkaisua.
DBMS	Database Management System. Ohjelmisto, jota käytetään yhden tai useamman tietokannan hallitsemiseen.
DDL	Data Definition Language. Tietorakenteiden luomiseen käytettävän ohjelmointikielen yleisnimitys.
DLL	Dynamic Link Library. Jaettu ja dynaamisesti ladattava ohjelmointikirjasto Microsoftin Windows-käyttöjärjestelmissä.
DML	Data Manipulation Language. Tiedon hakemiseen ja muokkaamiseen käytettävän ohjelmointikielen yleisnimitys.
EAV	Entity-Attribute-Value. Tiedonmallinnustekniikka, jossa entiteettiin sisältyvä tieto esitetään attribuutti-arvo –parein.
GPL	GNU General Public License. Yleinen avoimen lähdekoodin ohjelmistoissa käytetty lisenssi.
HTML	Hypertext Markup Language. Kuvauskieli, jota käytetään verkkosivujen toteuttamisessa.
IIS	Internet Information Services. Microsoftin kehittämä web-palvelinohjelmisto.
IMS	Information Management System. Hierarkiatietomalliin pohjautuva IBM:n tietokantaratkaisu.
IP	Internet Protocol. Pakettien reitityksessä käytetty IP-osoitteisiin perustuva protokolla.

JET	Joint Engine Technology. Microsoftin kehittämä tietokantamoottori, joka on käytössä esim. Access-tiedonhallintajärjestelmissä versioon 2007 asti.
JDBC	Java Database Connectivity. Java-ohjelmointikielen rajapinta, joka mahdollistaa erilaisten tietokantojen käyttämisen Java-sovelluksista.
PHP	PHP: Hypertext Preprocessor. Ohjelmointikieli, jota käytetään erityisesti dynaamista sisältöä sisältävien verkkosivujen toteuttamiseen.
PLIM	Production Line Information Management. Nimike, jolla Alma Software Oy kuvailee Alma-tiedonhallintajärjestelmää.
RAID	Redundant Array of Inexpensive Disks. Redundantin tiedon tallentamiseen perustuva vikasietoinen levyjärjestelmä.
RAM	Random Access Memory. Tietokoneen keskusmuisti, joka on tyypiltään haihtuvaa eli tarvitsee sähkövirtaa tietojen säilyttämiseksi.
SQL	Structured Query Language. Alunperin IBM:n kehittämä standardoitu kyselykieli relaatiotietokantojen hallintaan.
SSL	Secure Sockets Layer. Kuljetuskerroksella toimiva protokolla, jonka tarkoituksena on mahdollistaa turvallinen päästä-päähän -tiedonsiirto.
TKHJ	Tietokannanhallintajärjestelmä. Ohjelmisto, jota käytetään yhden tai useamman tietokannan hallitsemiseen.
VBA	Visual Basic for Applications. Microsoftin toimisto-ohjelmiin (kuten Word, Excel, Powerpoint) sekä esimerkiksi Autodeskin AutoCAD-sovellukseen integroitu Visual Basic-pohjainen sovelluskehitysympäristö.
VPN	Virtual Private Networking. Tekniikka, jonka avulla voidaan muodostaa suojattu yhteys kahden tai useamman yksityisen verkon välille julkista verkkoa (kuten Internetiä) hyödyntäen.
WWW	World Wide Web. Internetissä toimiva, linkitettyihin hypertekstidokumentteihin perustuva järjestelmä.
XML	Extensible Markup Language. Kuvauskieli, jonka avulla voidaan kuvata tietoa sekä tiedon ominaisuuksia.

XSLT	Extensible Stylesheet Language Transformations. Kuvauskieli, jota käytetään XML-dokumenttien muunnoksiin.

1 JOHDANTO

Tiedon varastoimiseen ja hallintaan käytettävä tiedonhallintaratkaisu on kriittinen komponentti teollisuudessa käytettävissä suunnittelujärjestelmissä. Käytössä olevan tiedonhallintaratkaisun vaihtaminen modernimpaan on työläs prosessi, joka alkaa uuden järjestelmän valinnasta ja jatkuu käyttöönottoon, johon kuuluu mm. henkilöstön koulutus sekä vanhojen sovelluksien integroiminen osaksi uutta järjestelmää. Tässä diplomityössä on tutkittu uuden, teknisiltä ominaisuuksiltaan erilaisen tiedonhallintaratkaisun käyttöönottoa sekä siihen liittyviä ongelmia keskisuuressa suunnittelualan yrityksessä, CTS Engtec Oy:ssä.

1.1 Tausta

CTS Engtec Oy on Kouvolassa toimiva metsä- ja prosessiteollisuuden suunnittelu- ja konsultointiyritys. Yhtiö on perustettu 1973 ja se työllistää kirjoitushetkellä vajaat 160 työntekijää (CTS Engtec Oy, 2010). Sen tarjoamiin palveluihin kuuluvat metsä- ja prosessiteollisuuden projektien hallinta esiselvityksistä suunnittelun kautta asennusvalvontaan ja käyttöönottoon. Yritys koostuu hallinnon ja markkinoinnin lisäksi viidestä eri osastosta, jotka ovat teräsrakenne-, automaatio-, sähkö-, prosessi- ja laitossuunnittelu. Yhtiön liikevaihto vuonna 2008 oli noin 11,2 miljoonaa euroa.

CTS Engtec Oy:llä on ollut käytössä pitkälti yrityksen tarpeisiin kehitetty Intecro-tiedonhallintasovellus, jolla suunnittelussa käytettävää tietoa voidaan hallita. Tälläistä tietoa ovat mm. erilaisten automaatiopiirien, prosessilaitteiden ja putkilinjojen ominaisuudet, kuten valmistaja, virtaava aine, moottorin kierrosluku jne. Koska kyseessä on pääasiallisesti CTS:lle kehitetty tuote, on sen myyminen asiakkaalle projektissa käytettävänä tietokantana hankalaa, koska tällöin asiakas joutuisi käytännössä sitoutumaan CTS Engtec Oy:n palveluihin myös tulevaisuudessa. Lisäksi Intecro-järjestelmän ikä alkoi näkyä sen yhteensopivuudessa uudempien käyttöjärjestelmien, kuten Microsoft Windows Vistan kanssa. Internetin yleistymisen myötä yhä useammat sovellukset on kehitetty

hyödyntämään webin tarjoamia mahdollisuuksia ja myöskään tällä saralla Intecro ei enää ollut ajan tasalla. Koska Intecron kehityksestä CTS:llä vastasi pääosin yksi henkilö, vastuu kehittämisestä ja toiminnasta oli pitkälti ja liiallisesti tämän yhden henkilön varassa. Tiedonhallintajärjestelmän kehittäminen tai vaihtaminen oli siis perusteltua kilpailukykyisyyden säilyttämiseksi.

Kartoitetuista vaihtoehdoista uudeksi tiedonhallintajärjestelmäksi valittiin Kokkolalaisen ALMA Software Oy:n ALMA-tiedonhallintajärjestelmä. Se koostuu asiakas- ja palvelinohjelmistosta sekä MySQL- tai Oracle -tietokannasta. Koska Alma-tietokannan rakenne poikkeaa aikaisemmin käytössä olleen Intecro-suunnittelutietokannan rakenteesta, kyseessä olevaa suunnittelutietokantaa hyödyntäneet ohjelmistot eivät enää toimineet uuden tiedonhallintajärjestelmän kanssa. Suurin osa näistä ohjelmistoista on toteutettu talon sisäisesti. Tällaisia ohjelmistoja ovat mm. prosessi- ja instrumentointikaavioiden luomiseen tarkoitettu CTS PI-CAD, piirikohtaisten toimintakuvausdokumenttien luomiseen käytettävä CTS Pine sekä putkimateriaalien hallintaan käytettävä PMMATE. Nämä ohjelmistot haluttiin pitää käytössä myös tiedonhallintajärjestelmän muutoksen jälkeen, sillä ne olivat käytännössä hyväksi havaittuja sekä yrityksen tarpeeseen räätälöityjä.

1.2 Tavoitteet ja rajaukset

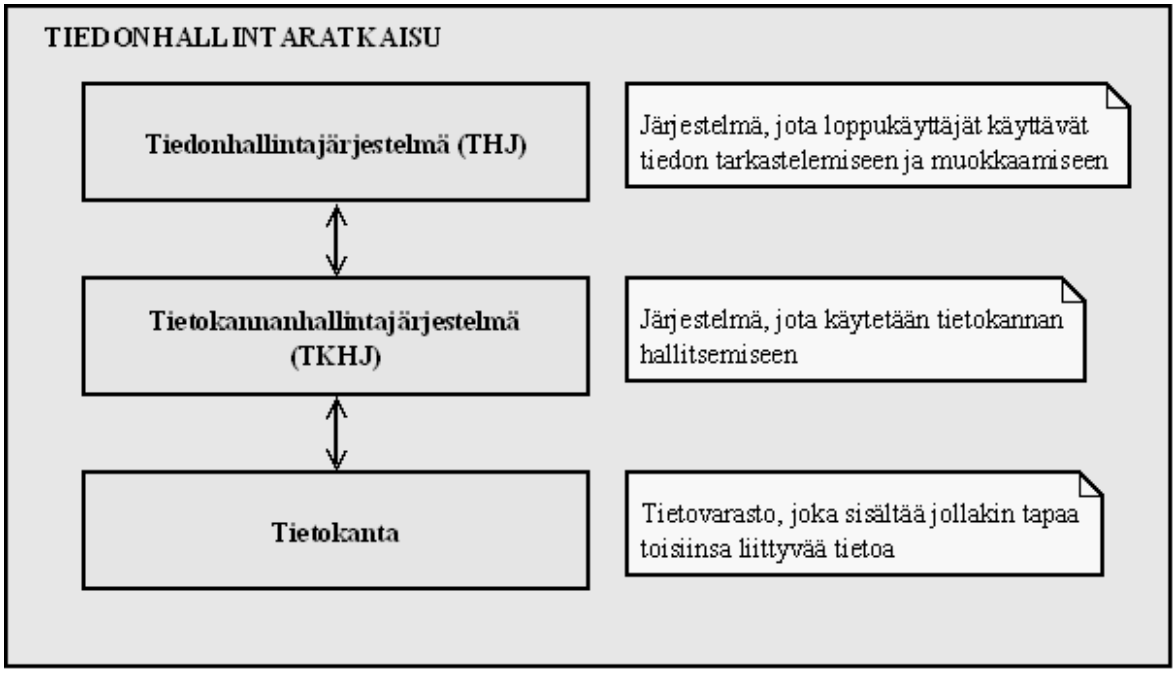
Diplomityön tavoitteena oli tutustua ALMA-tiedonhallintajärjestelmään sekä auttaa sen käyttöönotossa ja ylläpidossa. Käyttöönottoon kuului CTS Pine- ja PMMATE-sovellusten integrointi Almaan, ja ylläpitoon Alma-sovellusten asentaminen, päivittäminen sekä varmuuskopiointi. Alun perin tarkoituksena oli myös laatia ohjeistusta sekä koulutusta Alman käyttöön, mutta nämä jäivät lopulta pois aikatauluongelmien vuoksi. Aikataulun pettämisen aiheutti CTS Pine -ohjelmiston ongelmat, jotka vaikeuttivat sovelluksen toimintaan saattamista Alma-järjestelmän kanssa. Työn teoriaosuudessa tavoitteena oli perehtyä yleisesti tietokantoihin ja tiedonhallintajärjestelmiin sekä niiden ominaisuuksiin ja toiminnallisuuksiin.

1.3 Työn rakenne

Diplomityön aluksi käydään läpi yleisiä tiedon hallintaan liittyviä käsitteitä, kuten tietokanta, tietokannanhallintajärjestelmä ja tiedonhallintajärjestelmä. Näihin käsitteisiin on perehdytty luvussa 2. Luvussa 3 tutustutaan puolestaan tarkemmin CTS Engtec Oy:n vanhaan Intecro-tiedonhallintajärjestelmään sekä sen korvanneeseen Alma-tiedonhallintajärjestelmään. Lisäksi tässä luvussa käsitellään järjestelmän vaihtoon johtaneita syitä sekä valintaperusteita, jotka johtivat juuri ALMA Software Oy:n tarjoaman tiedonhallintajärjestelmän valintaan. Diplomityössä toteutettiin myös kolme sovellusta: CTS Pineen liittyvät CTS Database Creator for Pine ja CTS Pine Management Utility, sekä PMMATEen liittyvä CTS XML Creator for PMMATE. Näiden sovellusten tarkoituksena on mahdollistaa Pinen ja PMMATEn yhteistyö uuden tiedonhallintajärjestelmän kanssa. Niiden toteutukseen, rakenteeseen ja ongelmiin on perehdytty luvuissa 4 ja 5. Lopuksi luvussa 6 esitetään työn lopulliset tulokset sekä niistä seuranneita johtopäätöksiä.

2 TIEDON HALLINTA

Diplomityön painopistealueena ovat erilaiset tiedon hallintaan liittyvät sovellukset, kuten Alma, Intecro, Microsoft SQL Server, MySQL ja Microsoft Access. Tässä työssä tiedon hallinnan katsotaan koostuvan kolmesta peruskomponentista, jotka ovat tietokanta, tietokannanhallintajärjestelmä (TKHJ) ja tiedonhallintajärjestelmä (THJ). Yhdessä nämä komponentit muodostavat yrityksen tai organisaation tiedonhallintaratkaisun, joka on esitetty kuvassa 1. Työtä tehdessä tuli esille, että käytännössä näiden erilaisten käsitteiden käyttö on usein epämääräistä ja osittain päällekkäistä. Esimerkiksi CTS:llä Almaa ja Intecroa kutsutaan yleisesti tietokannoiksi, vaikka tarkemmin määriteltynä kummatkin ohjelmistot ovat pikemminkin tiedonhallintajärjestelmiä, joissa tietokanta on tosin oleellisena osana. Kuvassa 1 esitetty tiedonhallintaratkaisun arkkitehtuuri ei ole niinkään mikään universaali käytössä oleva malli, vaan pikemminkin tätä työtä varten luotu hahmotelma siitä, miten erilaisia tiedon hallintaan liittyviä ohjelmistoja voidaan jaotella. Tämän luvun tarkoituksena on pyrkiä selkeyttämään kuvan 1 tiedonhallintaratkaisun eri komponenttien vastuuta ja tarkoitusta.



Kuva 1 - Yksinkertaisen tiedonhallintaratkaisun komponentit

2.1 Tietokanta

Tiedonhallintaratkaisun pohjimmainen komponentti on itse tietokanta. Se on tietovarasto, joka sisältää joillakin tapaa toisiinsa liittyvää tietoa. Jotta tiedosta olisi jotain hyötyä, sen tulisi olla selkeästi jäsennettyä ja järjesteltyä. Lisäksi tietokantaan varastoidulla tiedolla tulee olla jokin käyttötarkoitus. Randy Yarger, George Reese ja Tim King määrittävätkin tietokannan ”kokoelmaksi tietoa, joka on jäsenneltyä ja varastoitu jotakin tarkoitusta varten” (Yarger et al., 1999). Suunnittelutietokannoissa tällaista tietoa on yleensä suunnittelussa käytettävät komponentit ja niiden ominaisuudet. Putkilinja on esimerkki suunnittelussa usein käytetystä komponentista, jolla voi olla useita ominaisuuksia kuten putken koko, paine ja virtaava aine. Tietokannan perustoimintoja ovat tiedon lisääminen, hakeminen, muuttaminen ja poistaminen. Yleensä tietokannoista puhuttaessa tietokanta käsitetään sähköisenä, tietokoneella sijaitsevana tietovarastona.

Ominaista tietokannoille on myös se, että niissä oleva tieto on pysyvää (Date, 2004). Tällä tarkoitetaan sitä, että kun tieto on kerran syötetty tietokantaan, se ei häviä sieltä ilman tietokannan käyttäjien eksplisiittisiä toimenpiteitä. Tämä ominaisuus erottaa tietokannan mm. sellaisesta tiedosta, joka tallentuu muistiin jonkin ohjelman ajon aikana - kun ohjelma suljetaan, tietokin häviää muistista.

Puhuttaessa tietokannasta sillä tarkoitetaan yleensä suurpiirteisesti koko tiedonhallintaratkaisua tietokannanhallintajärjestelmä ja tiedonhallintajärjestelmä mukaanlukien. Kun määritelmää tarkennetaan, voidaan tietokanta nähdä tietokannanhallintajärjestelmän sisältönä, ja edelleen tarkennettuna sillä voidaan tarkoittaa käytettävää tietokantamoottoria (Haigh, 2006). Esimerkiksi tietokannanhallintajärjestelmäksi luokiteltavassa Microsoft Accessissa luotuja ja hallinnoitavia tietokantoja kutsutaan usein Access-tietokannoiksi, vaikka itse tietokantamoottorina toimiikin Microsoft JET (Joint Engine Technology) tai Access Database Engine. Samoin MySQL-tietokannanhallintajärjestelmää käytettäessä puhutaan MySQL-tietokannasta, vaikka itse tietokantamoottori onkin yleensä joko InnoDB tai MyISAM (MySQL, 2009).

2.1.1 Käsitteelliset komponentit

Tietokantaan mallinnettua asiaa tai ilmiötä kutsutaan entiteetiksi (engl. entity) (Oppel, 2004). Entiteetti voi olla esimerkiksi henkilö, yritys, tilaus tai teollisuussuunnittelun puolella esimerkiksi putkilinja, prosessikaavio, automaatiopiiri jne. Jokaisella entiteetillä on ominaisuuksia, joita kutsutaan attribuuteiksi. Attribuutteja ovat esimerkiksi henkilön tapauksessa mm. nimi, ikä sekä sukupuoli, ja putkilinjan tapauksessa mm. edellä mainitut koko, paine ja virtaava aine. Kolmas peruskomponentti tiedon mallintamisessa on entiteettien väliset suhteet (engl. relationships). Suhteita on kolmentyyppisiä: yksi-yhteen, yksi-moneen ja monta-moneen. Esimerkiksi automaatiopiiri koostuu useista automaatiolaitteista, ja yksi automaatiolaite voi kuulua vain yhteen automaatiopiiriin. Tällöin automaatiopiiri- ja automaatiolaite-entiteettien välillä on yksi-moneen suhde.

Näiden kolmen peruskomponentin lisäksi tietokanta sisältää eheyssääntöjä, joilla voidaan osaksi varmistaa että tieto on syötetty oikein, kuten esimerkiksi yrityksessä tai standardeissa sovittujen käytäntöjen mukaisesti. Yksinkertaisimmillaan säännöillä voidaan mm. varmistaa että numeroarvollisiin kenttiin syötetään vain numeroarvoja eikä tekstiä, tai että päivämäärä-kenttään syötetty päivämäärä on oikeassa muodossa. Esimerkiksi CTS:n suunnittelutietokannassa lähes jokaisella tietokantaan syötetyllä objektilla on projektin sisällä yksilöllinen positiotunnus, jonka avulla kukin projektin objekti pystytään yksilöimään. Muun muassa tämän positiotunnuksen yksilöllisyys voitaisiin varmistaa eheyssääntöjä hyödyntäen. Näin ollen käyttäjä ei voisi vahingossa tai tahallaan syöttää tietokantaan useampia objekteja, joilla on sama positiotunnus.

2.1.2 Tietomallit

Tietokannan rakennetta kuvataan tietomallilla, joka on abstrakti malli siitä, miten tietokantaobjektit ja niiden väliset suhteet talletetaan tietokantaan. Tietomalli ei ota kantaa fyysiseen toteutukseensa vaan se voidaan toteuttaa usealla eri tavalla. Tietokannan käyttäjien ei kuitenkaan tarvitse olla tietoisia siitä, miten tietomalli on fyysisesti toteutettu, vaan heille riittää, että he tietävät mallin toiminnan korkeammalla abstraktiotasolla.

Tietomalleilla mahdollistetaan näin tietokannan fyysisen toteutuksen yksityiskohtien kätkeminen käyttäjiltä. Fyysisellä toteutuksella tarkoitetaan tässä yhteydessä sitä, miten tietomalli on implementoitu tietokoneelle. Yleisin käytössä oleva tietomalli on IBM:n tutkija E.F. Coddin 1970 esittelemä relaatiomalli (Codd, 1970). Muita tunnettuja tietomalleja ovat hierarkiamalli, verkkomalli, oliomalli sekä olio-relaatiomalli.

Hierarkiamalli oli käytössä ensimmäisissä tietokannoissa, ja tunnetuin siihen pohjautuva tuote oli IBM:n IMS (Information Management System) (Oppel, 2004). Hierarkiamallissa tietueet yhdistetään toisiinsa osoittimia käyttämällä. Tietokantaan syntyy näin ollen lapsi-vanhempi -suhteita, eli käytännössä yksi-moneen suhteita. Mallin ongelmana on se, ettei se salli monta-moneen tyyppisiä suhteita, vaan jokaisella lapsitietueella on ainoastaan yksi vanhempi. Verkkomallissa tämä ongelma on korjattu, ja yksittäisellä tietueella pystyy olemaan useampi vanhempi. Tämän korjauksen johdosta verkkomallista tuli kuitenkin tarpeettoman monimutkainen ja vaikeaselkoinen. Lisäksi hierarkia- ja verkkomallin osoittimiin pohjautuva toteutus johti mallien skaalautumattomuuteen ja joustamattomuuteen, jossa ad hoc -tyyliset haut vaativat käytännössä koko tietokannan läpikäymisen haettavan tiedon löytämiseksi. Tämä johtuu siitä, että osoitinpohjainen toteutus rajoittaa käytössä olevia hakupolkuja.

Relaatiomallin eräänä perusajatuksena oli päästä eroon osoittimiin pohjautuvien mallien joustamattomuudesta. Osoitinpohjaisissa toteutuksissa varastoitavan tiedon käyttötarkoitus tuli tietää etukäteen, jotta osoitinpolut saataisiin optimoitua ja turhalta tietueiden läpikahlaukselta vältyttäisiin (Oppel, 2004). Vaikka tietokantaa suunnitellessa sinne talletettavan tiedon käyttötarkoituksen tulisikin olla hyvin selvillä, saattaa uusia käyttötarkoituksia ilmetä myös myöhemmässä vaiheessa. Tällöin on oleellista, että tietokantamalli ei nojaudu liikaa ennalta määriteltyihin tapoihin yhdistää tietueita, vaan että näitä tapoja voidaan tarvittaessa luoda uusia. Vaikka relaatiomallissa ei olekaan osoittimia loogisella tasolla, mallin fyysinen toteutus voi hyvinkin sisältää niitä. Käyttäjälle nämä osoittimet eivät kuitenkaan näy, sillä tietomallien perusajatuksena on nimenomaan monimutkaisen fyysisen toteutuksen piilottaminen käyttäjältä.

Relaatiomallissa tieto näkyy tietokannan käyttäjille tauluina ja vain tauluina (Date, 2004). Käyttäjälle tarjolla olevilla relaatioalgebraan perustuvilla operaattoreilla pystytään muodostamaan uusia tauluja vanhojen taulujen pohjalta. Esimerkiksi SQL-kielen (Structured Query Language) peruskäsky ”SELECT” luo hetkellisesti uuden taulun jo olemassa olevien taulujen halutuista tiedoista. Taulut ovat looginen abstraktio, joilla tietokannan rakennetta pyritään selventämään käyttäjälle. Fyysisesti tietoa ei välttämättä tallenneta taulukkomuodossa, vaan tietokannanhallintajärjestelmä voi tallentaa sen parhaaksi katsomallaan tavalla.

Relaatiotietokannan taulut ovat kaksiulotteisia ja koostuvat riveistä ja sarakkeista. Jokainen taulun rivi kuvastaa yhtä taulun esittämän entiteetin ilmentymää. Esimerkiksi ”Henkilö”-taulussa yksittäinen rivi kuvastaa yksittäistä henkilöä ja häneen liittyviä tietoja. Taulussa olevaa riviä kutsutaan usein tietueeksi (engl. record). Jokainen sarake puolestaan sisältää yhden attribuutin, joka kuvaa jotakin entiteettiin liittyvää tietoa. Esimerkiksi henkilön nimi voisi olla attribuutti ”Henkilö”-taulussa. Yleensä tietokantaa suunniteltaessa attribuuteissa oleva tieto tulisi olla jakamatonta siten, että sitä ei voi jakaa enää pienempiin osiin. Edellä mainittu henkilön nimi on siinä mielessä huono attribuutti, että se voidaan jakaa etu- ja sukunimeen. Jos attribuuttina käytetään koko nimeä, ei henkilöitä voida esimerkiksi järjestellä etu- tai sukunimien mukaan. Jokaisella attribuutilla on arvon lisäksi tietotyyppi, joka määrittää, ja jolla voidaan rajoittaa, minkä tyylistä tietoa attribuutti voi sisältää, kuten esimerkiksi teksti, kokonaisluku, päivämäärä jne. Jos attribuutin arvo ei ole tiedossa, käytetään sen arvona yleensä ”NULL”-arvoa, joka on eri asia kuin tyhjä merkkijono tai nolla.

Taulun rivin yksilöivää attribuuttia kutsutaan perusavaimeksi (engl. Primary key). Tämän attribuutin tulee luonnollisesti olla yksilöllinen, jotta se pystyy yksilöimään rivit. Yleensä perusavaimessa käytetään juoksevaa numerointia, jolloin tietokannanhallintajärjestelmä huolehtii siitä, että perusavain pysyy yksilöllisenä. Perusavainta voidaan käyttää hyödyksi tauluja yhdistettäessä tallentamalla se yhdistettävään tauluun. Saraketta tai sarakkeita, joiden avulla taulu voidaan yhdistää toiseen tauluun, kutsutaan vierasavaimeksi (engl. foreign key).

Relaatiomallin ongelmana on sen kyvyttömyys käsitellä monimutkaisempia tietotyyppejä, kuten valokuvia, ääntä ja videota, teknisiä piirustuksia ja niin edelleen (Oppel, 2004). Tätä puutetta paikkaamaan kehitettiin oliopohjainen tietokantamalli, jossa tieto koostuu tietueiden sijaan olioista. Olio on kuten entiteetti, eli jokin tosimaailman mallinnettava asia, joka koostuu loogisesti yhteenkuuluvasta tiedosta ja toiminnallisuudesta. Olioterminologiassa attribuutteja kutsutaan olion jäsenmuuttujiksi ja toiminnallisuuksia metodeiksi. Esimerkiksi pankkitili voisi olla olio, jolla on jäsenmuuttujana tilin saldo, ja metodina funktio, joka palauttaa tilin saldon. Jäsenmuuttujiin tulisi päästä käsiksi ainoastaan metodien kautta. Tätä ominaisuutta kutsutaan enkapsuloinniksi, ja sen pääasiallisena tavoitteena on metodien yksityiskohtien piilottaminen käyttäjältä sekä ylläpidettävyyden parantaminen. Jos metodin toteutusta halutaan muuttaa, riittää että se muutetaan yhteen paikkaan sen sijaan, että kaikkiin metodia käyttäviin sovellusohjelmiin tarvitsisi tehdä muutosta. Oliopohjainen tietokantamalli soveltuu myös hyvin käytettäväksi oliopohjaisten ohjelmointikielien kanssa, sillä tiedon esitystapa on tällöin samankaltainen sekä tietokannassa että sovellusohjelmissa, eikä tietoa tarvitse näin ollen muuntaa muodosta toiseen. Kunnollisten ad hoc -hakujen puute rajoittaa kuitenkin oliomallin hyödyllisyyttä, ja sitä käytetäänkin vain sellaisissa erityisissä sovellutuksissa, jossa monimutkaisten tietotyyppien tuki on tarpeellinen, mutta ad hoc -tyyliset haut eivät. Olio- ja relaatiomalleja on myös pyritty yhdistämään, jotta käyttöön saataisiin molempien mallien parhaat puolet. Tällaista tietokantamallia kutsutaan oliorelaatio-malliksi.

2.1.3 EAV-esitystapa

Tavanomaisessa tietokantasuunnittelussa kukin yksittäistä entiteettiä kuvaava attribuutti esitetään sarakkeena relaatiotietokannan taulussa siten, että yhtä attribuuttia vastaa aina yksi sarake. Tämä lähestymistapa on soveltuva silloin, kun attribuuttien määrä on suurin piirtein kiinteä, ja kun jokaisella attribuutilla on jokin arvo suurimmassa osassa taulun riveistä (Dinu & Nadkarni, 2007). Se sopii kuitenkin huonosti tapauksiin, joissa attribuutteja on paljon, ja joissa vain osa näistä attribuuteista saa jonkun muun arvon kuin tyhjän jokaista taulun riviä kohti (vrt. harva matriisi matematiikassa). Tällöin tietokannassa joudutaan käyttämään paljon turhaa tilaa pelkästään tyhjien arvojen tallentamiseen. Lisäksi

tietokannanhallintajärjestelmissä on toimittajakohtaisia rajoituksia siitä, kuinka monta saraketta yksittäinen taulu voi sisältää. Esimerkiksi MySQL-tietokannanhallintajärjestelmän versiossa 5.0 tämä yläraja on 4096 saraketta taulua kohden ja Oraclen 10g-versiossa 1000. (MySQL, 2010; Oracle, 2004).

Taulukossa 1 on esitetty tavanomaisesti mallinnettu relaatiotietokannan taulu, joka sisältää tietoja putkilinjoista. Entiteettinä tässä taulussa toimii siis putkilinja, ja attribuutteina eli taulun sarakkeina putkilinjan ID, positiotunnus, nimelliskoko ja virtaava aine. Putkilinjalle, jonka ID on 1, ei ole esimerkin vuoksi määritetty virtaavaa ainetta.

ID	Positiotunnus	Nimelliskoko	Virtaava aine
1	736-028-VRA-40-10H1A	40	
2	726-102-VRA-40-10H1A	100	Vesi

Taulukko 1 - Putkilinjojen tiedot tavanomaisen esitystavan mukaisesti

EAV-mallinnuksessa (Entity-Attribute-Value) taulu koostuu yksinkertaisimmillaan kolmesta sarakkeesta, jotka ovat entiteetti, attribuutti ja arvo. Entiteetti-sarakkeeseen talletetaan entiteetin yksilöllinen tunniste, attribuutti-sarakkeeseen attribuutin nimi tai tunniste, ja arvo-sarakkeeseen attribuutin arvo. EAV-esitystavan mukaisessa taulussa on yksi rivi jokaista attribuutti-arvo -paria kohden, ja yksi rivi sisältää aina yhden tiedon, kun taas tavanomaisessa taulussa yksi rivi sisältää useampia tietoja. Taulukossa 2 on esitetty edellä oleva putkilinjataulu EAV-esitystavan mukaisesti. Koska putkilinjalle, jonka ID on 1, ei ole määritetty virtaavaa ainetta, ei vastaavaa riviä tarvitse tallentaa EAV-esitystavan mukaiseen tauluun.

ID	Attribuutti	Arvo
1	Positiotunnus	736-028-VRA-40-10H1A
1	Nimelliskoko	40
2	Positiotunnus	726-102-VRA-40-10H1A
2	Nimelliskoko	100
2	Virtaava aine	Vesi

Taulukko 2 - Putkilinjojen tiedot EAV-esitystavan mukaisesti

Tietokantaa, jossa suurin osa tiedosta on mallinnettu EAV-esitystavan mukaisesti, kutsutaan yleensä EAV-tietokannaksi. Kaikkien tietokannan taulujen ei kuitenkaan välttämättä tarvitse noudattaa EAV-esitystapaa, vaan osa tauluista voidaan mallintaa myös tavanomaista lähestymistapaa hyödyntäen. EAV-malli soveltuu parhaiten tilanteisiin, joissa tietokantaan tallennettaviin entiteetteihin liittyvien attribuuttien määrä on potentiaalisesti suuri, mutta yksittäiseen entiteettiin liittyvien attribuuttien määrä on kohtuullisen pieni. Se on tehokas mm. lääketieteellisissä tietokannoissa, joihin tallennetaan esimerkiksi potilastietoja (Nadkarni et al., 1999).

EAV-mallin suurimmat hyödyt ovat sen tarjoama joustavuus sekä tehokas tilankäyttö tilanteissa, joissa attribuutteja on paljon, mutta niille määritettyjä arvoja vain harvassa (Anhøj, 2003). Joustavuudella tarkoitetaan sitä, että entiteettiä koskevan attribuuttien määrää ei teoriassa ole rajoitettu mitenkään. Lisäksi attribuutteja voidaan tarvittaessa lisätä ilman, että tietokannan loogista skeemaa joudutaan muokkaamaan. Esimerkiksi jos tavanomaisesti mallinnettuun putkilinjatauluun halutaan lisätä uusia attribuutteja, kuten esimerkiksi paineluokka, taulun rakennetta ja näin ollen myös tietokannan skeemaa joudutaan muokkaamaan. Skeeman muutokset voivat puolestaan aiheuttaa muutostarvetta myös tietokantaa hyödyntäviin sovelluksiin sekä niiden käyttöliittymiin. Tilankäyttö EAV-mallissa on puolestaan tehokasta siksi, että tyhjille attribuuteille ei tarvitse varata turhaan tilaa.

EAV-mallin haittoja ovat mm. lisääntynyt etukäteisohjelmoinnin tarve sekä monimutkaisten attribuuttikeskeisten hakujen tehottomuus ja hankaluus (Anhøj, 2003). Etukäteisohjelmoinnilla viitataan siihen, että monet sellaiset toimenpiteet, jotka tietokannanhallintajärjestelmä suorittaa automaattisesti tavanomaista lähestymistapaa käytettäessä, joudutaan toteuttamaan manuaalisesti. Tällaisia toimenpiteitä ovat esimerkiksi viite-eheyksien ja tietotyyppien tarkistukset. Attribuuttikeskeisten hakujen tehottomuus johtuu puolestaan siitä, että niiden suorittamiseen vaaditaan EAV-mallissa joukko-operaatioiden käyttöä, mikä on puolestaan hitaampaa kuin yksinkertaisien SELECT -käskyjen käyttäminen. Ennen kuin EAV-mallia aletaan hyödyntää tiedon esittämisessä, on syytä harkita tarkkaan, ovatko mallin käytöstä seuraavat hyödyt suurempia kuin siitä seuraavat haitat.

2.1.4 Tietokantaratkaisun hyödyt

C.J. Date listaa teoksessaan “Introduction to Database Systems” joitakin tietokantapohjaisen ratkaisun tarjoamia etuuksia, joita ovat mm: (Date, 2004)

- Tieto on jaettua ja keskitetysti hallittavissa
- Päällekkäisen tiedon määrää saadaan vähennettyä
- Tiedon oikeellisuutta voidaan kontrolloida
- Tarjoaa oikein hallittuna tietoturvaa
- Tarjoaa ns. tietoriippumattomuuden
- Tuki transaktioille

Tiedon jakamisella Date viittaa siihen, että sekä uudet että vanhat sovellukset voivat käyttää yhtä ja samaa tietovarastoa sen sijaan, että jokaisella sovelluksella olisi oma yksityinen tietovarastonsa. Tiedon keskittäminen yhteen paikkaan helpottaa myös sen hallinnoimista. Ilman keskitettyä tietokantaa kullakin tiedon käyttäjällä olisi oma kopionsa tiedosta, joka puolestaan hankaloittaa tiedon hallitsemista. Tällainen keskittämättömäksi ratkaisuksi kutsuttu tapa johtaa siihen, että sama tieto sijaitsee useissa eri paikoissa. Jos tieto tällöin päivittyy yhdessä paikassa, tulisi se päivittää myös kaikkiin muihin paikkoihin, jotta kokonaisvaltainen tiedon eheys säilyisi. Tämä puolestaan on hankalasti toteutettavissa, jos käytössä ei ole keskitettyä ratkaisua. Myös keskitetyssä tietokantaratkaisussa sama tieto saattaa jostakin syystä esiintyä useammassa eri paikassa, mutta keskitetty ratkaisu antaa kuitenkin mahdollisuuden hallita tällaisia tilanteita olettaen kuitenkin, että tietokannanhallintajärjestelmä on tietoinen mahdollisista päällekkäisyyksistä. Tällöin tietokannanhallintajärjestelmä pystyy automaattisesti päivittämään muutetun tiedon myös niihin paikkoihin, joihin käyttäjä ei ole sitä eksplisiittisesti määrittänyt päivitettäväksi.

Tietokantaratkaisu mahdollistaa myös jossakin määrin tiedon oikeellisuuden kontrolloimisen. Tietokannanhallintajärjestelmän avulla voidaan valvoa tietokantaan syötettävää tietoa sekä havaita mahdollinen väärä ja epärealistinen tieto. Esimerkiksi ei ole realistista olettaa, että henkilön ikä olisi 400 vuotta, kun taas 40 vuotta on realistinen

oletus. Tällaiset huolimattomuusvirheet on mahdollista huomata tietokannanhallintajärjestelmän sääntöjen avulla. Myös tiedon käyttöä voidaan kontrolloida määrittelemällä tietokannan käyttäjille erilaisia oikeuksia, kuten luku- ja muokkausoikeudet. Keskitetyn luonteensa ansiosta tietokanta on usein houkutteleva tietomurtoyritysten kohde, ja se vaatii huolellista hallintaa ollakseen tietoturvallinen.

Tietoriippumattomuus on tärkeä tietokantojen ominaisuus, joka tarkoittaa käytännössä sitä, että tietoa käyttävät sovellukset eivät ole riippuvaisia tiedon esitys- tai hakutavasta. Näin ollen mikäli tiedon esitys- tai hakutapa muuttuu, sovelluksiin ei tarvitse tehdä muutoksia. Tietoriippumattomuutta on käsitelty tarkemmin ANSI/SPARC -arkkitehtuurin esittelyn yhteydessä kappaleessa 2.2.1. Transaktio-tuki on puolestaan ominaisuus, jonka avulla pyritään varmistamaan tiedon eheys tietokantaoperaatioita suoritettaessa. Transaktioihin on perehdytty tarkemmin tietokannanhallintajärjestelmien yhteydessä kappaleessa 2.2.3.

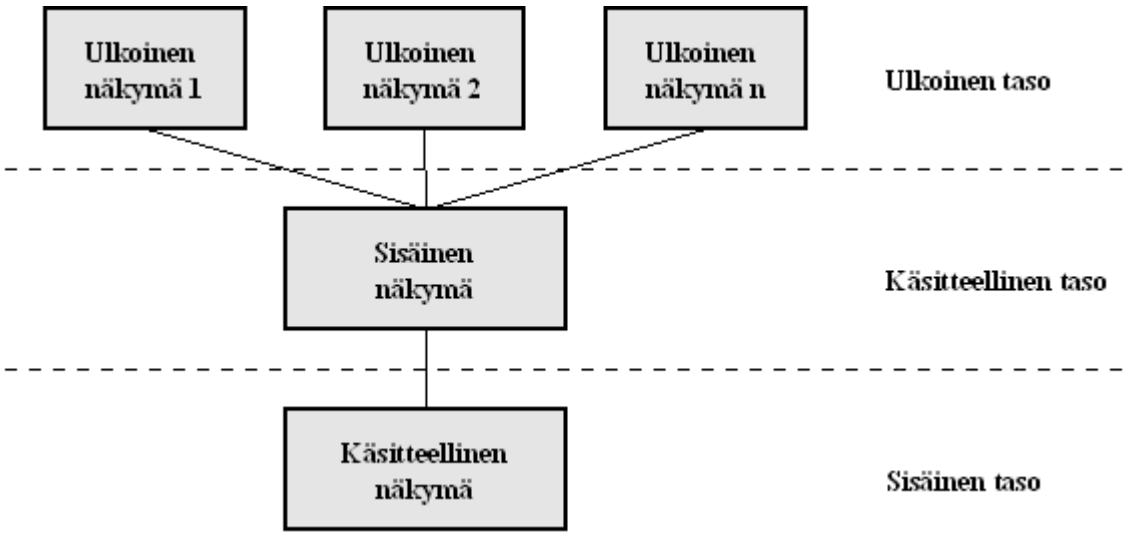
2.2 Tietokannanhallintajärjestelmä

Tietokannanhallintajärjestelmällä (TKHJ, engl. Database Management System, DBMS) tarkoitetaan nimensä mukaisesti järjestelmää (ohjelmistoa), jolla tietokannassa olevaa informaatiota sekä itse tietokantaa voidaan hallita. Kaikki interaktio sovellusten ja tietokannan välillä tapahtuu tietokannanhallintajärjestelmän kautta, ja se toimiikin eräänlaisena rajapintana tietokantaan, jonka takana kaikki toiminnallisuus on näkymätöntä käyttäjälle ja sovellusohjelmille. Yksittäisellä tietokannanhallintajärjestelmällä voidaan hallita useita eri tietokantoja samanaikaisesti.

2.2.1 ANSI/SPARC -arkkitehtuuri

ANSI/SPARC -arkkitehtuuri (American National Standards Institute / Standards Planning and Requirements Committee) on ANSI/X3/SPARC -työryhmän vuonna 1975 ensimmäisen kerran esittelemä abstrakti kehysrakenne (engl. framework) tietokannanhallintajärjestelmien kuvaamiseen. Se koostuu kolmesta eri tasosta, jotka ovat ulkoinen taso, käsitteellinen taso ja sisäinen taso (Brodie & Schmidt, 1981). Tasoja on

havainnollistettu kuvassa 2. Abstraktoinnin tarkoituksena on mahdollistaa tietokannoille tärkeä ominaisuus, jota kutsutaan tietoriippumattomuudeksi (engl. data independence). Tietoriippumattomuus voidaan jakaa kahtia fyysiseen ja loogiseen tietoriippumattomuuteen. Perusajatuksena molemmissa on se, että itse tietoa ei ole sidottu mihinkään tiettyyn esitystapaan (McLeod & Smith, 1980). Näin ollen tiedon esitystapaa tietokannassa voidaan tarvittaessa muuttaa, ilman että se vaikuttaa tietokantaa käyttävien sovelluksien toimintaan. Jos tietokantaratkaisu ei olisi tietoriippumaton, muutokset tiedon fyysisessä tai loogisessa esitystavassa aiheuttaisivat muutoksia myös kaikkiin sovellusohjelmiin (Date, 2004). Tästä puolestaan seuraa sovellusohjelmien ylläpidettävyyden huomattava heikentyminen.



Kuva 2 - ANSI/SPARC -arkkitehtuuri

ANSI/SPARC -arkkitehtuurin sisäinen taso pitää sisällään sisäisen näkymän, joka puolestaan sisältää tiedon siitä, kuinka tietokanta on varastoitu jollekin tallennusmedialle, kuten kiintolevyille. Tasoa voidaan kutsua myös fyysiseksi tasoksi. Kiintolevyllä tieto on tallennettu tiedostoihin, joiden sisäinen rakenne vaihtelee eri tietokantahallintajärjestelmien välillä. Tiedostojen käsittelyä voidaan nopeuttaa hajauttamalla tieto useampaan tiedostoon ja tiedostot useammille levyasemille, jolloin niitä voidaan lukea rinnakkain (Oppel, 2004). Suuremmissa palvelinpohjaisissa tietokantaratkaisuissa, kuten CTS:llä käytössä olevassa Microsoft SQL Serverissä, käyttäjän ei tarvitse huolehtia tiedon fyysisestä tallennustavasta, vaan tietokannanhallintajärjestelmä hoitaa tiedostojen käsittelyn automaattisesti. Joissakin tietokantaratkaisuissa käyttäjä joutuu puolestaan käsittelemään myös itse tiedon sisältävää

tiedostoa. Esimerkki tällaisesta ratkaisusta on Microsoft Access, jossa käyttäjä hoitaa manuaalisesti mm. tiedoston avaamisen ja tallentamisen.

Käsitteellinen taso, jota joskus kutsutaan myös loogiseksi tasoksi, sisältää käsitteellisen näkymän tietokantaan. Käsitteellinen näkymä kuvaa koko tietokannan rakenteen (skeeman) jonkin tietomallin, kuten relaatiomallin avulla. Se on siis korkeampitasoinen abstraktio sisäisellä tasolla olevasta tiedosta. Sisäisen tason mahdollisesti monimutkaisen, mutta tietokoneelle paremmin sopivan esitystavan sijaan tieto esitetään ihmiselle helpommin ymmärrettävässä yksinkertaistetussa muodossa, kuten esimerkiksi relaatiomallin tauluina. Tämä sisäisen tason erottaminen käsitteellisestä tasosta mahdollistaa edellä mainitun fyysisen tietoriippumattomuuden, sillä nyt tiedon fyysisen tallennuksen yksityiskohtia voidaan muuttaa, ilman että ne vaikuttavat abstraktiin loogisen tason malliin (Oppel, 2004).

Ulkoinen taso sisältää käyttäjän näkymät tietokantaan. Yleensä käyttäjää ei kiinnosta koko tietokannan sisältö vaan pelkästään rajattu osa siitä. Näkymien avulla voidaan rajoittaa näytettävää informaatiota joko selkeyden vuoksi tai tietoturvallisuussyistä. Näkymiä voidaan määritellä tietokantaan joko kiinteästi tai ad hoc -tyylisesti. Esimerkiksi SQL:n SELECT -lauseella muodostetaan periaatteessa ad hoc -näkymä tietokantaan. Ulkoisen tason ja käsitteellisen tason erottaminen toisistaan mahdollistaa loogisen tietoriippumattomuuden.

Tasojen välisien vastaavuuksien esityksiä kutsutaan kuvauksiksi (engl. mapping). Kuvauksien avulla tietokannanhallintajärjestelmä kykenee muuntamaan tietoa muodosta toiseen, esimerkiksi sisäisen tason esitystavasta relaatiomallin tauluiksi. Mikäli johonkin esitystapaan tehdään muutoksia, vastaava muutos tulee tehdä myös tasoon liittyviin kuvauksiin jotta järjestelmän toiminta säilyisi muuttumattomana.

Kullakin tasolla tietoa kuvaavaa määrittelyä kutsutaan skeemaksi (engl. schema) (Date, 2004). Sisäisellä tasolla tietoa kuvaa sisäinen skeema, käsitteellisellä tasolla käsitteellinen skeema ja ulkoisella tasolla ulkoinen skeema. Yleensä tietokannan skeemasta puhuttaessa tarkoitetaan käsitteellisen tason skeemaa, eli koko tietokannan rakennetta kuvattuna jonkin

tietomallin, kuten relaatiomallin avulla. Ulkoisen tason näkymät puolestaan muodostavat aliskeeman, joka on jollakin tavoin rajattu osa koko tietokannan skeemasta. Skeemojen määrittämiseen käytettäviä ohjelmointikieliä kutsutaan tiedonmäärittelykieliksi (engl. data definition language, DDL), ja skeemojen käsittelyyn käytettäviä ohjelmointikieliä tiedonkäsittelykieliksi (engl. data manipulation language, DML).

Relaatiotietokantojen yhteydessä käytetyin tiedonmäärittely- ja käsittelykieli on IBM:n 1970-luvulla kehittämä SQL-kyselykieli. SQL sisältää käskyjä tiedon luomiseen, muokkaamiseen, hakemiseen ja hallintaan. Se toimii relaatiotietokantojen yhteydessä edellä mainittuina tiedonmäärittely ja -käsittelykielenä, jolla voidaan siis luoda relaatiomallin mukainen käsitteellinen skeema. Lisäksi SQL:n avulla voidaan tehdä kyselyitä (hakuja) tietokantaan sekä määritellä käyttäjille erilaisia käyttöoikeuksia tietokantaobjekteihin.

2.2.2 Toiminnallisuus

Tietokannanhallintajärjestelmän tehtävänä on tarjota perustoiminnot tietokannan luomiseen, käsittelyyn ja ylläpitoon. Sen tarjoamia toimintoja ovat mm: (Date, 2004)

- Tiedon määrittely
- Tiedon käsittely
- Kyselyjen optimointi ja suoritus
- Tietoturva
- Tiedon eheys
- Yhdenaikaisten kyselyjen hallinta
- Tiedon palautus virhetilanteen jälkeen

Tietokannanhallintajärjestelmän perustoiminto on käyttäjien joko suoraan tai sovellusohjelmien kautta antamien syötteiden käsittely ja suoritus. Syötteet annetaan tietokannanhallintajärjestelmälle jollakin sen ymmärtämällä kielellä. Tietokannan luomisessa ja tiedon määrittelyssä käytetään jotakin tiedonmäärittelykieltä.

Tietokannanhallintajärjestelmän on osattava kääntää tiedonmäärittelykielellä (DDL) kuvattu skeema paremmin tietokoneella käsiteltävään muotoon, eli sen on tarjottava jonkinlainen DDL-käsittelijä. Vastaavasti tietokannanhallintajärjestelmän on tarjottava myös DML-käsittelijä, joka käsittelee tiedonkäsittelykielen (DML) avulla annetut käskyt. Relaatiotietokannanhallintajärjestelmän tapauksessa tämä tarkoittaa käytännössä sitä, että järjestelmä osaa käsitellä SQL-kielen komentoja, sillä niihin sisältyy sekä tiedonmäärittely- että tiedonkäsittelykieli.

Tietokannanhallintajärjestelmän on luonnollisesti hallittava myös tiedon hakeminen tietokannasta. Tiedon hakuun käytetään kyselyitä, jotka annetaan jollakin kyselykielellä - relaatiotietokantojen yhteydessä yleensä SQL:llä. Kyselyt voidaan jakaa kahteen eri tyyppiin, jotka ovat suunnitellut kyselyt ja suunnittelemattomat kyselyt. Suunnitellulla kyselyllä tarkoitetaan sellaista tiedonhakua tietokannasta, joka on otettu huomioon tietokannan rakennetta suunnitellessa. Suunnitellut kyselyt ovat näin ollen tehokkaita suorittaa, koska tietokannan rakennetta on voitu optimoida niitä silmälläpitäen. Kaikkia tietokannan sisältämän tiedon käyttökohteita ei kuitenkaan välttämättä tiedetä etukäteen, ja näin ollen tietokantaan voidaan tehdä myös suunnittelemattomia hakuja. Tällaisia ad hoc -tyylisiä hakuja ei ole otettu huomioon tietokantaa suunnitellessa, ja tästä johtuen niiden suorittaminen ei välttämättä ole kovin tehokasta. Tietokannanhallintajärjestelmään kuuluu optimoijaksi kutsuttu komponentti, jonka tarkoituksena on muokata käyttäjän antamaa hakua siten, että sen suoritus olisi mahdollisimman tehokasta.

Käyttäjän antamia käskyjä suorittaessa tietokannanhallintajärjestelmän on otettava huomioon myös tietoturvallisuusnäkökohdat. Tietokannanhallintajärjestelmän ei tule suorittaa sellaisia käskyjä, joiden seurauksena käyttäjälle päätyy hänelle kuulumatonta informaatiota. Joissakin tilanteissa tällaisten käskyjen suodattaminen voi olla hankalaa, kuten esimerkiksi päättelyhyökkäyksissä (engl. inference attack). Päättelyhyökkäyksessä käyttäjä saattaa pystyä kiertämään tietokantaan asetettuja turvallisuusrajoitteita päättelemällä salatun informaation saatavilla olevasta informaatiosta (Morgenstern, 1987). Tietoturvaa uhkaavien käskyjen lisäksi tietokannanhallintajärjestelmän ei tule suorittaa myöskään sellaisia käskyjä, jotka rikkovat tietokantaan asetettuja eheyssääntöjä. Näin ollen sen avulla voidaan osittain ehkäistä tietokantaan syötettävä virheellinen tieto.

Tietokantaratkaisun keskitetyn luonteen takia sillä on yleensä useita yhtaikaisia käyttäjiä, jotka saattavat käsitellä samaa tietoa yhtaikaisesti. Tällöin on olemassa riski, että tietokantaan päätyy virheellistä tietoa esimerkiksi kahden päällekkäisen päivitysoperaation vuoksi, koska toinen niistä on vaarassa hukkua. Tietokannanhallintajärjestelmän, tai tarkemmin ottaen tietokantamoottorin tehtävänä on huolehtia yhtaikaisten eli rinnakkaisten käskyjen sarjoittamisesta (Bernstein, 1990). Yleensä sarjoittaminen tapahtuu lukitusten avulla.

Rinnakkaisten tietokantaoperaatioiden lisäksi tietokantaa uhkaa sovellus- ja järjestelmävirheet, joiden seurauksena tietokanta ja siellä oleva informaatio saattavat jäädä väärään tilaan. Tämä on uhkana esimerkiksi silloin, kun useasta komennosta koostuvan tietokantaoperaation suoritus jää kesken esimerkiksi järjestelmän kaatumisen takia. Koska annetuista komennoista on ehditty suorittamaan vain osa, ei tietokannassa oleva tieto näin ollen ole välttämättä enää paikkansa pitävää. Klassinen esimerkki tällaisesta tilanteesta on tilisiirto henkilöitten A ja B välillä, jonka voidaan katsoa koostuvan kahdesta operaatiosta. Ensimmäiseksi vähennetään siirrettävä summa henkilön A tililtä, ja toiseksi lisätään siirrettävä summa henkilön B tilille. Jos järjestelmä nyt kaatuu heti sen jälkeen kun A:n tililtä on vähennetty siirrettävä summa, A menettää siirrettävän summan verran rahaa ilman, että B saa mitään. Vastaavanlaisten tilanteiden välttämiseksi tietokannassa voidaan käyttää niin sanottuja transaktioita, joiden ominaisuuksia on esitelty tarkemmin seuraavassa kappaleessa. Transaktioiden ja kirjaamisen avulla tietokannanhallintajärjestelmä pystyy toipumaan tämänkaltaisista virhetilanteista.

2.2.3 Transaktiot

Transaktiolla tarkoitetaan joukkoa toimintoja, jotka suoritetaan joko kokonaisuudessaan tai ei ollenkaan (Bernstein, 1990). Niiden tarkoituksena on säilyttää tietokannan tiedon eheys virhetilanteissa ja eristää rinnakkaiset tietokantaoperaatiot toisistaan. Transaktio voidaan määritellä myös työn yksikkönä, jolloin sen ”kaikki tai ei mitään” -ominaisuus korostuu. Mikäli transaktiossa suoritetut operaatiot kohdistuvat useampaan kuin yhteen tietokantaan,

tarvitaan niiden hallitsemiseen erillinen ohjelmistokomponentti, jota kutsutaan transaktiomanageriksi.

Transaktioilla on neljä tärkeää ominaisuutta, jotka ovat atomisuus, eheys, eristys ja kestävyys (Härder & Reuter, 1983). Näitä kutsutaan usein ACID-ominaisuuksiksi (Atomicity, Consistency, Isolation, Durability). Atomisuudella tarkoitetaan sitä, että transaktio suoritetaan joko kokonaan tai ei ollenkaan, ”kaikki tai ei mitään” -periaatteella. Transaktion eheys-ominaisuus takaa puolestaan sen, että mikäli tietokannassa oleva tieto on oikeellista (eheää) ennen transaktion suoritusta, se on sitä myös transaktion suorituksen jälkeen. Transaktion suorituksen aikana tiedon ei kuitenkaan välttämättä tarvitse olla oikeellista. Eristys-ominaisuuden avulla transaktio eristetään muista transaktioista siten, että sen aikana tehdyt muutokset eivät näy toisille transaktioille ennen kuin transaktio on kokonaan suoritettu. Transaktion kestävyydellä tarkoitetaan puolestaan sitä, että kun transaktio on hyväksytty, sen tekemät muutokset pysyvät tietokannassa, vaikka järjestelmä heti hyväksymisen jälkeen kaatuisi. Toisin sanoen transaktion tekemät muutokset on siis tallennettu johonkin pysyväismuistiin, kuten kiintolevylle.

Kuvassa 3 on esitetty edellä mainittu tilisiirtoesimerkki sekä transaktiota käyttäen että ilman. Tapauksessa 1 varainsiirto suoritetaan ilman transaktio-ominaisuuden käyttöä. Mikäli järjestelmä kaatuu UPDATE-komentojen välissä, tilin A saldo vähenee ilman, että tilin B saldo kasvaa, jolloin lopputuloksena tietokantaan jää väärää tietoa. Tapauksessa 2 tilisiirtoon käytetään transaktiota, jolloin joko kaikki siihen kuuluvat käskyt suoritetaan tai mitään niistä ei suoriteta. Transaktio hyväksytään COMMIT-käskyllä. Tämän jälkeen sen tekemät muutokset on pysyvästi tallennettu tietokantaan. Mikäli järjestelmä kaatuu UPDATE-komentojen välissä, tietokantaan ei joudu väärää tietoa kuten tapauksessa 1.

TAPAUS 1 - Tilisiirto ilman transaktiota

```
UPDATE tilit SET Saldo = Saldo - 100 WHERE TILI = 'A';
// Mahdollinen häiriö
UPDATE tilit SET Saldo = Saldo + 100 WHERE TILI = 'B';
```

TAPAUS 2 - Tilisiirto transaktiolla

```
START TRANSACTION;
  UPDATE tilit SET Saldo = Saldo - 100 WHERE TILI = 'A';
  // Mahdollinen häiriö
  UPDATE tilit SET Saldo = Saldo + 100 WHERE TILI = 'B';
COMMIT;
```

Kuva 3 - Esimerkki transaktion käytöstä

2.2.4 Ylläpito

Tiedonhallintajärjestelmää ja siihen liittyviä komponentteja, kuten tietokantaa ja tietokannanhallintajärjestelmää ylläpitävää henkilöä kutsutaan tietokannan ylläpitäjäksi (engl. Database Administrator, DBA). Ylläpitäjän vastuulla on tiedonhallintaratkaisun moitteettoman toiminnan takaaminen. Tämän vuoksi kyseisen henkilön tulisi olla hyvin perehtynyt käytettävään laitteistoon, tietokannanhallintajärjestelmään sekä itse ylläpidettävään tietokantaan ja sen rakenteeseen. Ylläpitäjän tehtäviä ovat mm. seuraavat: (Date, 2004)

- Käsitteellisen ja sisäisen skeeman määrittäminen
- Yhteistoiminta käyttäjien kanssa
- Varmuuskopiointi
- Suorituskyvyn valvonta

Daten hierarkiassa tietokannan ylläpitäjä on vastuussa tietokannan käsitteellisen ja sisäisen skeeman määrittämisestä, eli käytännössä tietokannan rakenteen määrittämisestä. Näin ollen kyseinen henkilö on myös perillä tietokannan käyttötarkoituksesta ja sen sisällöstä. Tarvittaessa tietokannan ylläpitäjä osaa myös antaa teknistä tukea, ja auttaa käyttäjiä tietokantaan liittyvissä ongelmatilanteissa. Ylläpitäjän on myös pyrittävä optimoimaan

tietokannan rakenne siten, että se täyttää vaaditut suorituskykyvaatimukset esimerkiksi hakuaikojen osalta. Tärkeä osa ylläpitäjän tehtävää on varmistua, että tietokanta toipuu erilaisista häiriöistä.

Tietokantaan kohdistuvat häiriöt voidaan jaotella järjestelmän ja tallennusmedian häiriöihin (Date, 2004). Järjestelmähäiriöt ovat häiriöitä, joissa järjestelmä lakkaa toimimasta, mutta tietokanta pysyy fyysisesti vaurioitumattomana. Tällaisia häiriöitä ovat mm. sähkökatkokset. Järjestelmähäiriöiden tyypillinen piirre on se, että kaikki niin sanotussa haihtuvassa muistissa sijaitseva tieto menetetään. Haihtuvalla muistilla tarkoitetaan muistia, joka vaatii sähkövirtaa tiedon ylläpitämiseen. Tämän tyyppistä muistia on esimerkiksi tietokoneen keskusmuisti (RAM, Random Access Memory). Virransyötön katketessa kaikki haihtuvaismuistissa oleva tieto menetetään. Tästä seuraa se, että tietokannassa meneillään olevien transaktioiden tila ei ole enää tiedossa, ja kaikki tällaisten transaktioiden tietokantaan tekemät muutokset on kumottava järjestelmän käynnistyessä uudelleen. On myös mahdollista, että transaktio on suoritettu loppuun, mutta sen tekemiä muutoksia ei ole ehditty kirjoittaa pysyväismuistiin ennen järjestelmän kaatumista. Tällaiset transaktiot on puolestaan suoritettava uudelleen järjestelmän uudelleenkäynnistyessä. Tietokannanhallintajärjestelmä ei ole käyttäjien käytettävissä ennen kuin epäonnistuneet transaktiot on kokonaan käsitelty järjestelmän uudelleenkäynnistyksen yhteydessä. Jotta tietokantaoperaatioita pystyttäisiin perumaan tai suorittamaan uudelleen, tulee jokainen suoritettava operaatio tallentaa lokitiedostoon ennen varsinaisen operaation suorittamista. Lokitiedoston avulla tietokanta pystytään palauttamaan eheään tilaan järjestelmähäiriön jälkeen.

Tallennusmedian häiriöissä, kuten kiintolevyn hajoamisessa, tyypillistä on tiedon fyysinen tuhoutuminen, joko osittain tai kokonaan. Tällaisessa tilanteessa tiedonpalautuksen apuna voidaan käyttää tietokannasta tai koko järjestelmästä tasaisin väliajoin otettuja varmuuskopioita. Tietokannasta otettua varmuuskopiota kutsutaan yleensä vedokseksi tai dumpiksi. Varmuuskopion avulla järjestelmä voidaan palauttaa sen ottamista edeltävään tilaan. Tietokannan tapauksessa on mahdollista lokitiedostoa hyödyntäen palauttaa tietokanta viimeisimpään tilaansa, suorittamalla uudelleen kaikki varmuuskopion ottamisen jälkeen suoritetut toimenpiteet. Tämä edellyttää luonnollisesti sitä, että kaikki

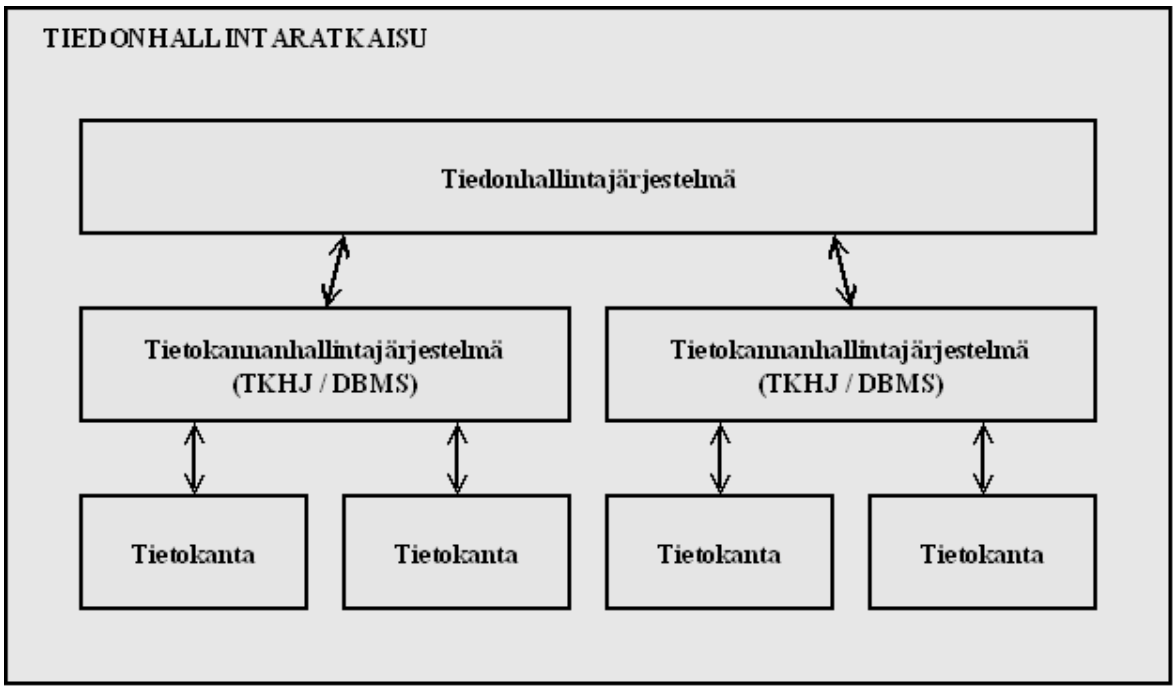
tietokantaoperaatiot on kirjattu lokitiedostoon, ja että lokitiedosto ei ole vaurioitunut tallennusmedian häiriön yhteydessä. Tästä syystä lokitiedosto tulisikin varastoida eri tallennusmedialle kuin datatiedosto, jolloin molempia ei menetetä yksittäisen tallennusmedian pettäessä. Tietokannasta otettua varmuuskopiota voidaan käyttää myös tilanteissa, joissa kannassa oleva tieto jostain syystä korruptoituu tai tuhoutuu esimerkiksi käyttäjien huolimattomien toimenpiteiden seurauksena. Tietokannan ylläpitäjän vastuulla on hoitaa järjestelmän varmuuskopiointi sekä varmistua siitä, että varmuuskopio voidaan tarvittaessa myös palauttaa.

2.3 Tiedonhallintajärjestelmä

Tässä työssä tiedonhallintajärjestelmällä tarkoitetaan sovellusta tai sovelluksia, joita järjestelmän peruskäyttäjät käyttävät tiedon käsittelemiseen. Peruskäyttäjät eivät yleensä ole suoraan tekemisissä tietokannanhallintajärjestelmän kanssa, vaan tiedon tarkasteluun ja muokkaamiseen on luotu erilaisia ohjelmia, joiden tarkoituksena on yksinkertaistaa ja helpottaa tietojen lisäämistä, tarkastelua, muokkaamista ja poistamista. Tiedonhallintajärjestelmä käyttää yleensä jonkin tietokannanhallintajärjestelmän tarjoamia palveluita, mutta sen avulla monimutkaisen tietokannanhallintajärjestelmän toiminnallisia yksityiskohtia saadaan piilotettua käyttäjiltä. Näin ollen käyttäjät voivat tarkastella ja muokata tietoa esimerkiksi graafisen käyttöliittymän avulla monimutkaisten SQL-lauseiden sijaan. Tiedonhallintajärjestelmä voidaankin näin ollen nähdä tietokannanhallintajärjestelmän korkeampana abstraktiona. Yhdessä tietokannat, tietokannanhallintajärjestelmät ja tiedonhallintajärjestelmä muodostavat tiedonhallintaratkaisun, joka on esitetty aiemmin kuvassa 1.

Kuvassa 4 on esitetty yksinkertaistettua tiedonhallintaratkaisua monimutkaisempi tiedonhallintaratkaisu. Kuten kuvasta ilmenee, tiedonhallintaratkaisu voi koostua useista tietokannoista, joiden hallinnoimiseen voidaan käyttää yhtä tai useampaa tietokannanhallintajärjestelmää. Optimaalisessa tapauksessa useammistakin tietokannoista ja tietokannanhallintajärjestelmistä koostuvaa ratkaisua voitaisiin hallita ja käyttää yhdellä tiedonhallintajärjestelmällä, joka peittää alapuolellaan olevan järjestelmän tekniset

yksityiskohdat käyttäjiltä ja toimii tarvittaessa koordinaattorina eri tietokannanhallintajärjestelmien välillä. Esimerkiksi jos käyttäjä päivittää tietoa, joka jostain syystä sijaitsee useammassa kuin yhdessä tietokannassa, tiedonhallintajärjestelmän tehtävänä on varmistaa että tieto päivittyy jokaiseen näistä tietokannoista, vaikka ne sijaitsisivatkin eri tietokannanhallintajärjestelmien alla.



Kuva 4 - Monimutkaisemman tiedonhallintaratkaisun komponentit

3 TIEDONHALLINTAJÄRJESTELMÄT CTS ENGTEC OY:SSÄ

CTS Engtec Oy:ssä tiedonhallintajärjestelmää tarvitaan esimerkiksi suunnittelutyössä tarvittavan tiedon varastointiin ja hallintaan. Tällaista tietoa ovat mm. suunnittelutyössä käytetyt komponentit sekä niiden ominaisuudet. Komponentteja ovat esimerkiksi automaatiopiirit, prosessilaitteet ja putkilinjat, ja niiden ominaisuuksia puolestaan mm. positiotunnus, moottorin kierrosluku ja virtaava aine. Tässä luvussa perehdytään CTS:n vanhaan tiedonhallintajärjestelmään, Intecroon, sekä sen korvanneeseen Alma-tiedonhallintajärjestelmään.

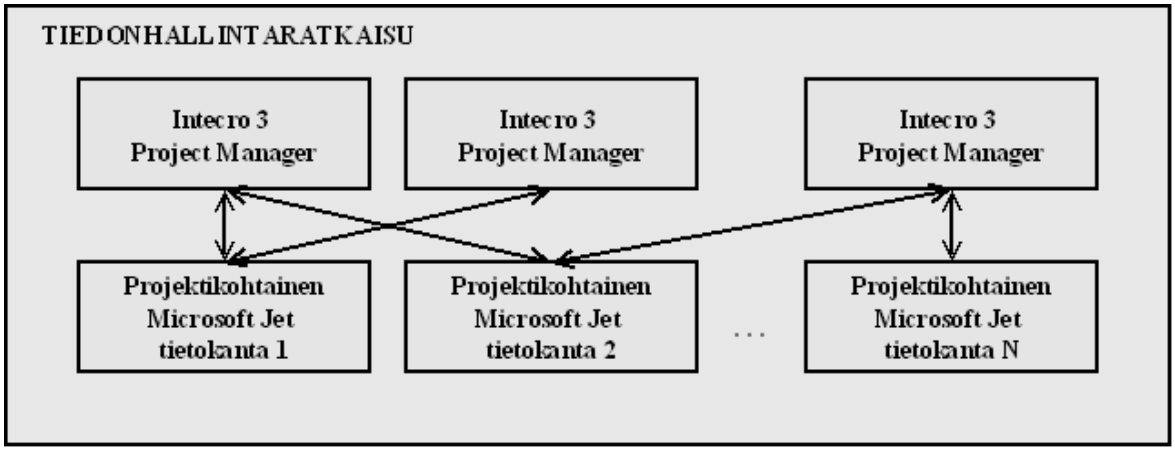
3.1 Intecro

Intecro on Kouvolalaisen IntecroSoft Oy:n kehittämä tiedonhallintaohjelma, ja se on toiminut CTS Engtec Oy:n pääasiallisena suunnittelutiedonhallintajärjestelmänä vuoteen 2008 asti. Järjestelmän kehitys aloitettiin vuonna 1993 silloisen CTS Engineeringin työntekijöiden Seppo Turusen ja Toivo Blombergin johdolla. CTS Engineering toimi IntecroSoftin pääsääntöisenä rahoittajana, ja ensimmäinen Intecro-tiedonhallintasovellus luotiinkin pitkälti juuri CTS:n tarpeisiin. IntecroSoft toimi alkuaikoinaan samoissa tiloissa CTS:n kanssa, joka osaltaan mahdollisti tiiviin yhteistyön. Keväällä 1994 CTS:llä otettiin käyttöön ensimmäinen Intecro-versio, Intecro 1. Ensimmäisen version valmistumisen myötä IntecroSoft siirtyi omiin toimitiloihin, ja siitä tuli selkeämmin oma yritys. Samalla myös yhtiön asiakaskunta laajeni. Ohjelmiston toinen versio, Intecro 2 näki päivänvalon loppuvuodesta 1995. Kolmas ja viimeinen versio, Intecro 3, ilmestyi alkuvuodesta 1997, ja samana vuonna IntecroSoft Oy hajosi. IntecroSoftin hajottua CTS Engineering osti suurimpana yksittäisenä asiakkaana oikeudet Intecro-tuotenimeen ja palkkasi palvelukseensa toisen järjestelmän kehittäjistä. Tästä eteenpäin Intecron kehitys oli täysin yhden henkilön varassa eikä varsinaisia uusia versioita ohjelmasta enää ilmestynyt. Intecron kehitys päättyi vuonna 2007.

3.1.1 Rakenne

Intecro 3 koostuu tietokannan ja sen rakenteen luomiseen käytettävästä Intecro Project Manager -ohjelmasta sekä tiedon lisäämiseen, tarkasteluun ja muokkaamiseen tarkoitetusta Intecro Data Viewer -sovelluksesta. Project Manager -ohjelmalla määritellään tietokannan taulut ja niiden sisältämät kentät sekä hallitaan käyttöoikeuksia. Data Viewer -ohjelmalla voidaan puolestaan tarkastella tietokannan sisältöä, ja lisäksi se tarjoaa työkaluja kyselyjen, näkymien ja raporttien tekemiseen sekä tiedon lisäämiseen ja muokkaamiseen. Intecro sisältää myös ohjelmointirajapinnan, jonka avulla sovellukset voidaan yhdistää Intecron tietokantaa. Esimerkiksi CTS:llä prosessi- ja instrumentointikaavioiden tuottamiseen käytetty CTS PI-CAD -sovellus hyödyntää suoraan Intecron tietokantaa. PI-CAD:lla voidaan viedä kaavioihin piirrettyjen komponenttien tietoja tietokantaan sekä lukea niitä tarvittaessa takaisinpäin.

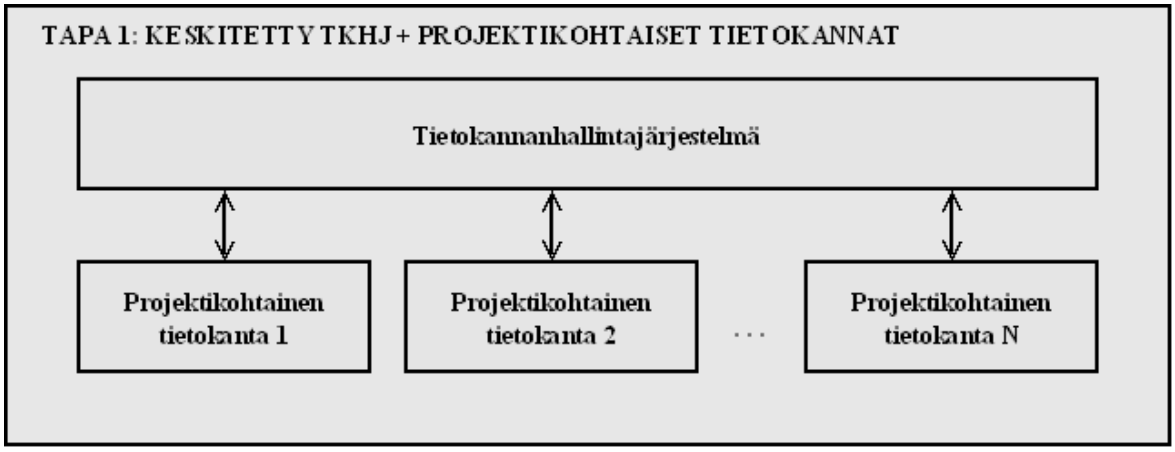
Intecro 3 -järjestelmässä jokainen projekti on erillinen tietokantansa, kuten Intecro-tiedonhallintaratkaisun karkeaa rakennetta esittävästä kuvasta 5 voidaan nähdä. Lisäksi näiden tietokantojen hallintaan ei ole olemassa keskitettyä tiedonhallintasovellusta, vaan tietokantoja voidaan hallita kaikilla koneilla, joihin on asennettu Intecro 3 Project Manager -sovellus, ja joissa varsinaiset tietokantatiedostot ovat käytettävissä. Intecro 3 -tietokannat käyttävät Microsoftin JET-tietokantamoottoria, joka on käytössä myös vanhemmissa Microsoft Accessilla luoduissa tietokannoissa. Tästä syystä Intecro 3 -tietokantoja kutsutaan usein myös Access-tietokannoiksi, mikä on siinä mielessä epätarkka ilmaisu että itse tietokantamoottori on JET. Access puolestaan on tarkalleen ottaen työkalu tietokantojen luomiseen ja hallintaan.



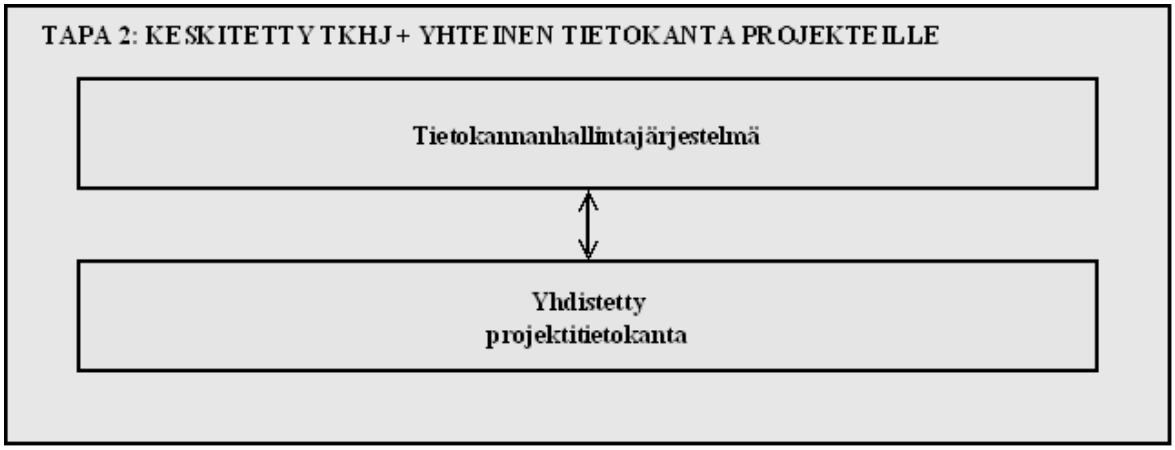
Kuva 5 - Intecro 3 -tiedonhallintaratkaisu

3.1.2 Vahvuudet ja heikkoudet

Kuvissa 6 ja 7 on esitetty kaksi vaihtoehtoista tapaa toteuttaa Intecron tyylinen tiedonhallintaratkaisu. Tässä kappaleessa esitetään Intecro-tyylisen tiedonhallintaratkaisun vahvuuksia ja heikkouksia näihin kahteen vaihtoehtoiseen toteutustapaan nähden. Molemmissa vaihtoehtoisissa tavoissa on käytetty keskitettyä tietokannanhallintajärjestelmää, toisin kuin Intecrossa. Intecrossa tietokantojen hallinta tapahtuu siinä mielessä hajautetusti, että hallintaohjelma voi olla asennettuna useampaan eri koneeseen yhtäaikaisesti. Toisistaan vaihtoehtoiset tavat poikkeavat siten, että tavassa 1 jokainen projekti on Intecron tyyliin oma erillinen tietokantansa, kun taas tavassa 2 kaikki projektit ovat samassa tietokannassa. Intecrossa käytetty ”hajautetusti hallittujen” erillisten tietokantojen ratkaisu sisältää sekä etuja että haittoja verrattuna vaihtoehtoisiin tapoihin.



Kuva 6 - Keskitetty TKHJ ja projektikohtaiset tietokannat



Kuva 7 - Keskitetty TKHJ ja yksittäinen tietokanta projekteille

Intecrossa jokainen projekti on oma tietokantansa, ja näin ollen yksittäisen tietokannan koko ei pääse kasvamaan yhtä suureksi kuin käytettäessä vaihtoehtoista tapaa 2, jossa kaikki projektit ovat samassa tietokannassa. Pienempikokoinen tietokanta on puolestaan yleensä suurikokoista tietokantaa suorituskykyisempi, koska haut ja muut operaatiot kohdistuvat pienempään joukkoon tietokantaobjekteja. Toisaalta tavoissa 1 ja 2 käytetyt keskitetyt tietokannanhallintajärjestelmät voivat myös osaltaan muodostaa pullonkaulan järjestelmän suorituskykyyn, mikä on ainakin osittain vältettävissä Intecron tyylisessä hajautetusti hallittavassa ratkaisussa. Lisäksi tietokannan toimittaminen asiakkaalle on helpompaa järjestelmissä, joissa kaikki projektit eivät sijaitse tavan 2 tyylisesti yhdessä tietokannassa, koska tällöin muita projekteja ei tarvitse käydä karsimaan tietokannasta pois toimituksen yhteydessä vaan tietokanta voidaan toimittaa sellaisenaan.

Intecron huonoihin puoliin lukeutuu hajautetuista hallintajärjestelmistä ja projektikohtaisista tietokannoista johtuva huono hallittavuus esimerkiksi tilanteissa, joissa tietokantojen rakenteeseen tai näkymiin pitää tehdä muutoksia. Esimerkiksi CTS PI-CAD -ohjelmaan lisätyt uudet ominaisuudet vaativat toisinaan uusia attribuutteja ja päivitettyjä näkymiä tietokantaan, ja näin ollen nämä muutokset on tehtävä kaikkiin niihin projektitietokantoihin, joissa uusia ominaisuuksia halutaan käyttää. Muutoksien tekeminen projektitietokantoihin on tavallisesti hoidettu manuaalisesti, ja se on aikaa vievä sekä monotoninen että sitä kautta osittain myös virhealtis prosessi. Vaihtoehtoisissa tavoissa 1 ja 2 muutoksien tekeminen on selkeämpää keskitetyn tietokannanhallintajärjestelmän ansiosta. Varsinkin tavassa 2 muutos tarvitsee tehdä vain kerran, jonka jälkeen se on käytettävissä kaikissa projekteissa, olettaen että yhteinen projektitietokanta on suunniteltu helposti muokattavaksi.

Intecron arkkitehtuurin vuoksi myös vanhan tiedon uudelleenkäytettävyys, eli lähinnä olemassa olevien tietojen kopiointi projektista toiseen, on hankalaa. Vanhan tietokannan kopiointi uuden projektin pohjaksi onnistuu helposti kopioimalla tietokantatiedostot ja nimeämällä ne uudestaan uuden projektin mukaiseksi, mutta yksittäisten piirien tai positioiden kopioimiseen projektitietokantojen välillä Intecro ei tarjoa työkaluja.

Asiakasprojektien lopussa on tapana luovuttaa asiakkaalle kaikki projektiin liittyvä dokumentaatio sekä itse tietokanta. Jotta asiakas saisi tietokannasta suurimman mahdollisen hyödyn, on melko yleistä että projektin tietokantaa ei käydä muuntamaan asiakkaan tietokantaan sopivaksi, vaan että asiakas hankkii koko projektissa käytetyn tiedonhallintajärjestelmän itselleen hallinnoitavaksi. Asiakkaan kannalta ideaalisessa tapauksessa suunnittelutoimistot käyttäisivät samoja tiedonhallintaratkaisuja, jolloin tietoa ei tarvitsisi tarpeettomasti muunnella muodosta toiseen siirryttäessä käyttämään jonkin toisen suunnittelutoimiston palveluita. Koska Intecro oli alun perin pitkälti kehitetty CTS:ää varten (ja lopulta itse kehityskin tapahtui CTS:llä), sen myyminen asiakkaalle projektin tietokantana oli hankalaa, sillä asiakkaat pelkäsivät tällöin sitoutuvansa CTS:än palveluihin myös tulevaisuudessa.

3.1.3 Syitä järjestelmän vaihtoon

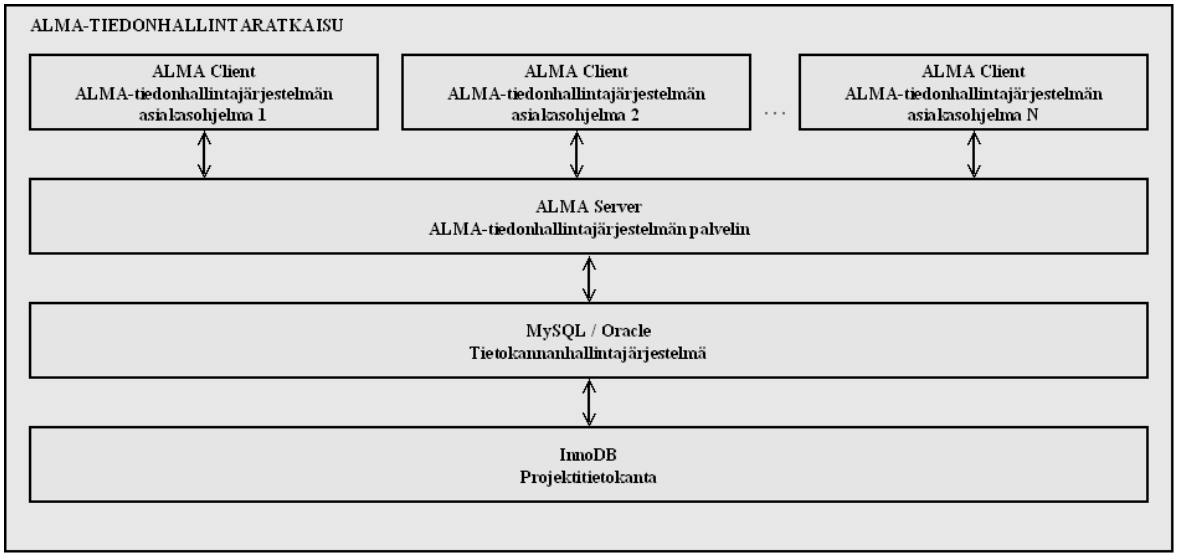
Kilpailukyvyn säilyttämiseksi tiedonhallintajärjestelmän kehittäminen oli välttämätöntä, ja näin ollen vaihtoehtoina oli joko Intecron kehityksen jatkaminen uuden tehtävään asetetun henkilön turvin, tai kokonaan uuden tiedonhallintajärjestelmän hankkiminen. Näistä vaihtoehdoista päädyttiin lopulta uuden tiedonhallintajärjestelmän hankkimiseen pääosin siitä syystä, että Intecron viimeisestä suuremmasta päivityksestä oli kulunut aikaa lähes kymmenen vuotta, ja sen toiminnassa epäiltiin esiintyvän ongelmia uudempien käyttöjärjestelmien, kuten Microsoft Windows Vistan kanssa. Järjestelmän modernisointi olisi vaatinut huomattavien muutoksien ja lisäyksien tekemistä siihen, eikä sen sisäinen toiminta ollut kenelläkään kattavasti tiedossa järjestelmän kehityksen lopettamisen jälkeen. Tästä syystä muutosten ja lisäysten tekeminen olisi ollut hankalaa.

3.2 ALMA

Alma on Kokkolalaisen Alma Software Oy:n kehittämä tiedonhallintajärjestelmä tuotantolaitoksen koko elinkaaren aikaisen informaation hallintaan. Alma Software itse määrittelee tuotteensa tuotantolinjan tiedonhallintajärjestelmäksi (PLIM, Production Line Information Management) (Alma Software Oy, 2009). Ohjelmiston kehitys sai alkunsa vuonna 1986, ja se oli alun perin tehty ensisijaisesti automaatiopuolen kunnossapitojärjestelmäksi. Tässä työssä käytössä oli ohjelmiston 10. pääversio useine eri aliversioineen.

Almaan pohjautuvan tiedonhallintaratkaisun rakenne on esitetty kuvassa 8. Ratkaisu koostuu Alma-palvelin- ja asiakasohjelmistoista, MySQL- tai Oracle-relaatiotietokannanhallintajärjestelmästä, sekä itse projektitietokannasta. Lisäksi Alma-palvelinohjelmisto sisältää sisäänrakennetun web-palvelimen, jonka avulla projektien tietoja pystytään tarkastelemaan myös web-selaimella. CTS:n Alma-ratkaisussa käytössä on MySQL-tietokannanhallintajärjestelmä ja InnoDB-tietokantamoottoriin pohjautuva tietokanta. Yhdenaikaisten Alma-asiakasohjelmalla luotujen yhteyksien lukumäärä on nykyisessä lisenssissä rajoitettu 21 käyttäjään, ja yhdenaikaisten web-käyttäjien lukumäärä

yhteen. Toisin kuin Intecrossa, Alma-tiedonhallintaratkaisussa kaikki projektit ovat samassa tietokannassa, ja tätä tietokantaa hallitaan keskitetysti.



Kuva 8 - ALMA-tiedonhallintaratkaisu

3.2.1 Vaihtoehtoiset järjestelmät

Uutta Intecron korvaavaa tiedonhallintaratkaisua valitessa vaihtoehtoina oli kahdentyyppisiä ratkaisuja. Ensimmäinen vaihtoehto oli hankkia pelkkä tietokannanhallintajärjestelmä, kuten Microsoftin SQL Server, ilman mitään toimialakohtaisia lisäsovellutuksia, ja liittää CTS:n sisäiset sovellukset käyttämään tätä tietokannanhallintajärjestelmää. Toinen vaihtoehto oli hankkia järjestelmä, joka sisälsi tietokannan lisäksi myös toimialakohtaisen sovelluksen tai sovelluksia. Vaihtoehtoisia järjestelmiä olivat Alman lisäksi mm. (Kiminki, 2009)

- Microsoft SQL Server
- Oracle
- Comos
- Intergraph SmartPlant

Näistä kaksi ensin mainittua, Microsoft SQL Server ja Oracle, olivat pelkkiä tietokannanhallintajärjestelmiä eikä niiden mukana olisi tullut mitään suunnittelutyötä helpottavia sovelluksia. Näiden järjestelmien kohtaloksi koitui korkea hinta, sekä käyttöön ja ylläpitoon vaadittavat henkilöstöresurssit. Uuden tiedonhallintaratkaisun käyttöönoton haluttiin myös olevan mahdollisimman sujuva, ja näiden tuotteiden kohdalla käyttöönottoa haittasi se, ettei niiden mukana toimitettu mitään valmiita insinöörityössä tarvittavia sovelluksia. Tästä seurasi se, että CTS:n omien sovellusten kehittäminen ja liittäminen osaksi näitä järjestelmiä olisi hidastanut käyttöönottoprosessia liiaksi. Lisäksi projektissa käytettävä tietokanta on yleensä myytävä myös asiakkaille, ja näiden ratkaisuiden myyminen projektissa käytettävänä tietokantana koettiin hankalaksi.

Comosin ja Intergraphin järjestelmät olivat puolestaan hyvin pitkälti valmiita tiedonhallintajärjestelmiä, sillä ne pitivät sisällään myös useita eri suunnitteluosastojen tarvitsemia sovelluksia. Tällöin uuden tiedonhallintaratkaisun käyttöönotto olisi ollut sujuvampaa kuin SQL Serverin ja Oraclen tapauksessa, koska ainakin osa tarvittavista sovelluksista olisi ollut jo valmiina, eikä CTS:n omien sovellusten liittäminen järjestelmään olisi hidastanut käyttöönottoa yhtä merkittävästi. Lisäksi näiden järjestelmien etuna oli keskitetty ja valvottu tietokanta, jonka ympärille eri osastojen, kuten automaatio- tai sähköosaston tarvitsemat sovellukset oli valvotusti liitetty. Tällöin pystytään paremmin välttämään tilanteita, joissa suunnittelijoilla on käytössä erilaista, vanhentunutta, tai väärää tietoa. Esimerkiksi prosessisuunnittelija saattaa määrittää PI-kaaviossa olevaan linjaan putkiposition, josta käy ilmi että käytettävän putken tulee olla haponkestävää. Putkistosuunnittelijalla saattaa puolestaan olla käytössä jokin esisuunnitteluvaiheessa käytetty materiaalilistaus, jossa sama putki on alustavasti määritelty tavalliseksi happoa kestämättömäksi putkeksi. Mikäli putkistosuunnittelija ei huomaa tarkastaa putken tietoja tietokannasta, tehtaalle saattaa päätyä vääräntyyppistä putkea, joka puolestaan aiheuttaa ylimääräisiä kustannuksia. Tällaisten tilanteiden välttämiseksi olisi optimaalista, jos käytettävä tiedonhallintaratkaisu valvoisi ja pakottaisi suunnittelijat käyttämään vain yhtä tietoa yhdessä paikassa.

Comos ja SmartPlant koettiin kuitenkin turhan massiivisiksi järjestelmiksi sekä SQL Serverin ja Oraclen tapaan liian kalliiksi. Comos ja SmartPlant ovat modulaarisia

järjestelmiä, eli vaihtoehtona olisi ollut myös ostaa keskustietokanta ja vain osa tarvittavista suunnittelusovelluksista, jolloin kustannukset olisivat olleet huokeammat. Tällöin olisi kuitenkin osittain menetetty edellä mainittu keskitetyn tiedonhallintajärjestelmän etu, jossa järjestelmä valvoo sitä että yksi tieto on vain yhdessä paikassa. Lisäksi CTS:n omien sovellusten liittäminen Comosiin ja SmartPlantiin miellettiin hankalaksi. Ne olisivat myös nostaneet asiakastyön keskituntihintaa liiaksi. Tiedonhallintajärjestelmäksi haluttiin tuote, jolla projekteja olisi mahdollisuus myydä kaikenkokoisille yrityksille pienistä suuriin. Massiivinen ja kallis tiedonhallintajärjestelmä olisi ollut ongelmallinen varsinkin pienien asiakasyritysten kohdalla, joilla ei välttämättä ole resursseja sijoittaa kalliiseen projektitietokantaan.

3.2.2 Valintaperusteet

Uuden tiedonhallintajärjestelmän valinnassa päädyttiin lopulta Kokkolalaisen Alma Softwaren ALMA-järjestelmään. Alman valintaan johtivat mm. seuraavat tekijät: (Kiminki, 2009)

- Valmiita toiminnallisuuksia eri osastoille
- Ohjelmointirajapinta
- Keskitetty ratkaisu
- Mahdollisuus käyttää Internetin ylitse
- Mahdollisuus dokumenttien hallintaan
- Kotimaisuus
- Hinta / laatusuhde

Comosin ja SmartPlantin tapaan Alma ei ollut pelkästään tietokannanhallintajärjestelmä vaan tietokannanhallintajärjestelmän päällä toimiva sovellus suunnitteluprojektien tiedonhallintaan. Tämän vuoksi Alman käyttöönoton katsottiin sujuvan sulavammin kuin pelkkien puhtaiden tietokannanhallintajärjestelmien, koska pohjalla oli jo jotain minkä päälle rakentaa ja liittää omia sovelluksia. Intecro-tiedonhallintajärjestelmän elinaikana CTS:llä oli luotu useita suunnittelutyön kannalta tärkeitä sovelluksia, jotka olivat integroitu

toimimaan Intecro-järjestelmän kanssa. Näitä sovelluksia, kuten esimerkiksi CTS PI-CADia, haluttiin käyttää vastaisuudessakin, ja tärkeää oli, että uusi tiedonhallintajärjestelmä sisältäisi jonkinlaisen rajapinnan, jonka kautta järjestelmässä olevaan tietoon pääsisi ohjelmallisesti käsiksi. Alma ei alun perin sisältänyt ohjelmointirajapintaa, mutta jo ensimmäisissä neuvotteluissa Alma Software lupautui toteuttamaan sellaisen.

Alma-tiedonhallintajärjestelmä tarjosi lisäksi keskitetyn tiedonhallintaratkaisun, toisin kuin aikaisemmin käytössä ollut Intecro. Keskitetyssä ratkaisussa tiedon hallinta ja valvominen on helpompaa kuin hajautetussa ratkaisussa, ja esimerkiksi tiedon kopiointi projektista toiseen onnistuu vaivattomammin. Myös sovellusohjelmien, kuten PI-CADin mahdollisten uusien ominaisuuksien vaatimien muutosten tekeminen tietokantaan on helpompaa keskitetyn hallinnan johdosta. Jos esimerkiksi jollekin objektille pitää lisätä uusi attribuutti, voidaan lisäys suorittaa Almassa massa-ajona kaikille projekteille, kun taas Intecrossa attribuutti pitää lisätä jokaiseen projektiin erikseen, mikä on hidasta, virhealtista ja aikaavievää.

Suunnittelutyön luonteen takia oli myös tärkeää, että ajantasaiseen suunnittelutietoon pääsee käsiksi myös toimiston ulkopuolelta. Alma-tiedonhallintajärjestelmä sisältää sisäänrakennetun WWW-palvelimen (World Wide Web), jonka avulla projektitietokantojen sisältämään informaatioon voidaan päästä käsiksi Internetin välityksellä. WWW-palvelimen avulla myös asiakas pystyy tarkastelemaan projektin tietoja reaaliajassa siltä osin, mitä asiakkaan kanssa on sovittu. Aikaisemmin käytäntönä oli tehdä tietokannasta määräajoin kopio asiakkaalle, jolloin asiakkaan käytössä ei välttämättä ollut aivan viimeisin tieto. Käyttöoikeuksien avulla pystytään varmistamaan, ettei asiakas näe muihin kuin omaan projektiinsa liittyvää tietoa.

Alma sisältää myös dokumenttien hallinnassa tarvittavia työkaluja, jotka katsottiin myös eduksi järjestelmää valittaessa. Toistaiseksi dokumenttienhallinta on käytössä projektitasolla siten, että tietokantaobjekteihin on liitetty niihin liittyviä dokumentteja. Esimerkiksi automaatiopiirin rakennetta kuvaavan AutoCADin dxf-piirrustuksen voi linkittää automaatiopositio-objektiin tietokannassa. Laajemmalti Alman tarjoama

dokumenttienhallintajärjestelmä ei kuitenkaan ole CTS:llä käytössä, vaan dokumenttien hallintaan käytetään muita järjestelmiä.

Alman eduksi laskettiin myös järjestelmän kotimaisuus. Jo valinnan alkuvaiheissa tiedettiin, että tuotteen valmistajaan oli saatava toimiva keskusteluyhteys, jotta CTS:n omien sovellusten liittäminen osaksi järjestelmää olisi mahdollisimman kitkatonta. Alma Softwaren koko tuki- ja kehitysryhmä sijaitsi tuotteen kotimaisuuden johdosta inhimillisen välimatkan päässä ja toimi lisäksi CTS:n kanssa saman yrityskulttuurin piirissä.

Alman suurimpia valtteja oli kuitenkin hinta/laatusuhde. Jos hinnasta ei olisi ollut väliä, valinta olisi osunut johonkin toiseen tuotteeseen. Ominaisuuksien puolesta esim. SmartPlant ja Comos olivat askeleen edellä. Ne esimerkiksi valvovat tarkemmin tiedon käyttöä, raportoivat virheistä, eivätkä anna julkaista projektia ennen kuin virheet on korjattu. Tiedonhallintajärjestelmän piti kuitenkin olla myös inhimillisen hintainen, jotta projekteissa myös asiakas investoisi siihen. Näin ollen Alma koettiin parhaana ratkaisuna uudeksi projektien tiedonhallintajärjestelmäksi.

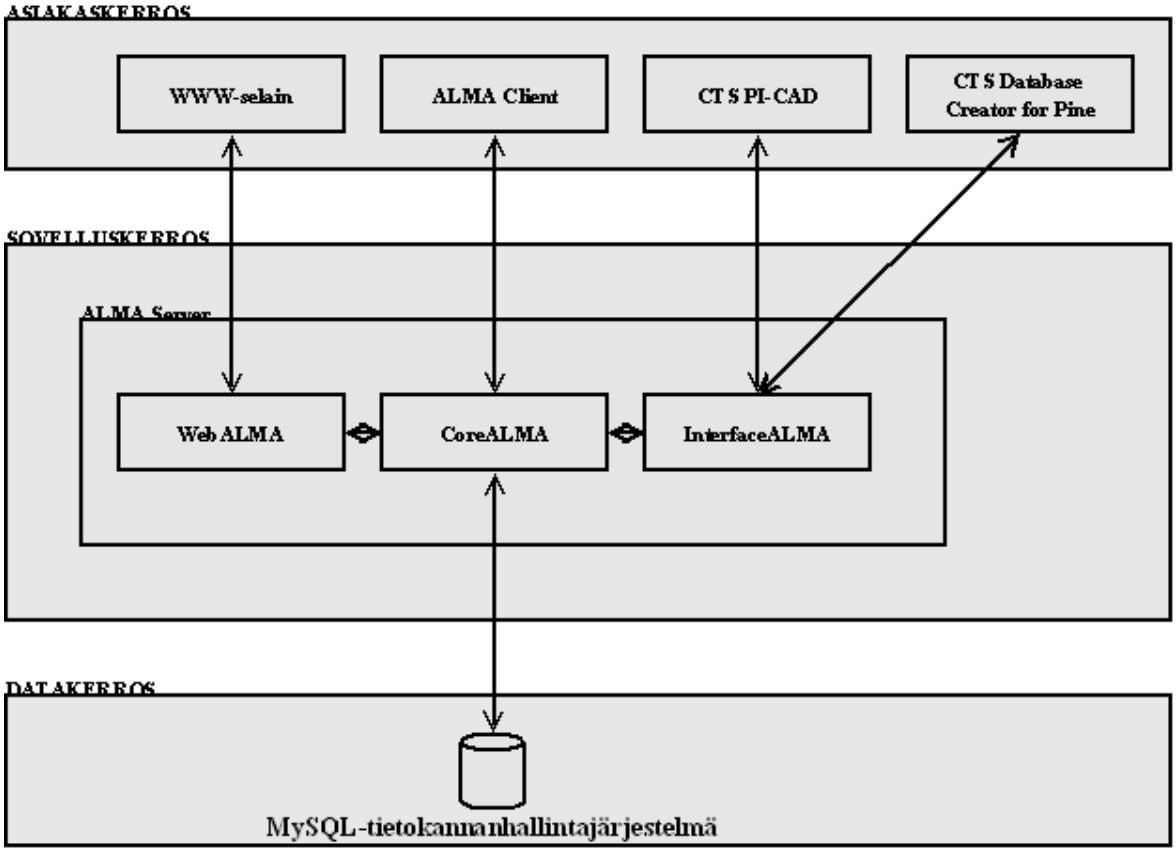
3.2.3 Rakenne ja toiminta

Alma-järjestelmä perustuu niin sanottuun kolmikerrosarkkitehtuuriin, jonka voidaan katsoa olevan laajennettu versio asiakas/palvelin -arkkitehtuurista. Asiakas/palvelin -arkkitehtuurin mukainen järjestelmä koostuu yleensä kolmesta peruskomponentista, jotka ovat asiakas, palvelin, ja nämä toisiinsa yhdistävä verkko (Savino & Queroli, 1996). Asiakaskomponenttia kutsutaan asiakasohjelmaksi, ja sillä tarkoitetaan ohjelmistoa, joka on asennettu järjestelmän loppukäyttäjien koneille. Tämän ohjelmiston avulla käyttäjät voivat lähettää pyyntöjä palvelinohjelmistolle. Pyyntöjen lähettäminen, ja niihin liittyvien vastauksien esittäminen hoidetaan usein graafista käyttöliittymää apuna käyttäen. Varsinainen sovelluslogiikka sijaitsee usein palvelinohjelmistossa, mutta myös asiakasohjelma voi sisältää mm. tarkistuksia lähetettävän pyynnön järkevyyden suhteen. Palvelinohjelmistoa ajetaan yleensä tarkoitukseen pyhitetyllä järeän laskentakapasiteetin

palvelinlaitteistolla. Asiakas- ja palvelinohjelmisto yhdistetään toisiinsa verkon avulla. Tämä verkko on yleensä joko yrityksen lähiverkko tai Internet.

Kolmikerrosarkkitehtuuri koostuu nimensä mukaisesti kolmesta kerroksesta, jotka ovat asiakaskerros, sovelluskerros ja datakerros (MSDN, 2010b; Ortiz, 2000). Asiakaskerrosta kutsutaan usein myös esityskerrokseksi ja sovelluskerrosta logiikkakerrokseksi. Asiakasohjelma sijaitsee asiakaskerroksella, palvelinohjelma sovelluskerroksella ja tietokanta datakerroksella. Ohjelmiston hajauttamisella eri kerroksille pyritään siihen, että kunkin kerroksen muokkaaminen tai korvaaminen onnistuisi ilman radikaalien muutoksien tekemistä muihin kerroksiin. Esimerkiksi sovelluslogiikkaan voidaan näin ollen tehdä muutoksia ilman kaikkien käytössä olevien asiakasohjelmien päivittämistä. Lisäksi käyttöliittymän, sovelluslogiikan, ja tietokannan eriyttäminen mahdollistaa kuorman tasaamisen, kun tietokanta ja sovellus eivät välttämättä sijaitse samalla koneella.

Alma-tiedonhallintajärjestelmän kolmikerrosarkkitehtuuri, ja sen CTS Engtecin sekä tämän diplomityön kannalta oleelliset komponentit on esitetty kuvassa 9. Asiakaskerroksella toimivat graafisella käyttöliittymällä varustettu Alma-asiakasohjelma (ALMA Client) sekä WWW-selain, joilla järjestelmään voi ottaa yhteyden. Myös Alman ohjelmointirajapintaa hyödyntävät sovellukset, kuten CTS PI-CAD ja CTS Database Creator for Pine voidaan katsoa kuuluvan tälle kerrokselle. Alma-palvelinohjelmisto (ALMA Server) sijaitsee sovelluskerroksella, ja se koostuu muun muassa CoreALMA, WebALMA ja InterfaceALMA -komponenteista. CoreALMA on ALMA-järjestelmän ydin, ja se sisältää suuren osan järjestelmän sovelluslogiikasta. WebALMA on Alman päälle rakennettu WWW-palvelinohjelmisto, joka mahdollistaa suunnittelutietojen katselemisen WWW-selaimella. InterfaceALMA puolestaan mahdollistaa ulkopuolisten sovellusten liittämisen Alma-järjestelmään ohjelmointirajapinnan avulla. Alman käyttämä MySQL- tai Oracle-tietokanta sijaitsee datakerroksella.



Kuva 9 - CTS Engtecillä käytössä olevan ALMA-tiedonhallintaratkaisun tärkeimmät komponentit

3.2.3.1 ALMA Client

Alma-asiakasohjelma on suunnittelutyökalu, jonka avulla suunnittelijat voivat tarkastella ja syöttää tietoa Alma-järjestelmään. Asiakasohjelma on toteutettu Java-ohjelmointikielellä, ja sen suoritus tapahtuu Java-virtuaalikoneessa. Java-toteutus on käyttäjille siinä mielessä näkymätön, että ohjelmasta on luotu normaali suoritettava exe-tiedosto, jota kaksoisnapauttamalla se voidaan normaaliin tapaan käynnistää. Käynnistyessään asiakasohjelma kysyy käyttäjältä käyttäjätunnuksen, salasanan sekä Alma-palvelimen IP-osoitteen (Internet Protocol) ja portin. Kun kirjautumistiedot on syötetty, asiakasohjelma ottaa yhteyden Alma-palvelimeen, josta se tarpeen mukaan pyytää tietoja näytettäväksi asiakasohjelmassa. Vaihtoehtoisesti asiakasohjelma voi lähettää palvelimelle asiakasohjelmaan syötettyjä tietoja. Tietojen siirtäminen tapahtuu yleensä CTS:n lähiverkon yli, ja toimiston ulkopuolella asiakasohjelman käyttö on mahdollista VPN-yhteyttä (Virtual Private Networking) hyödyntämällä. Asiakasohjelmalla on mahdollista

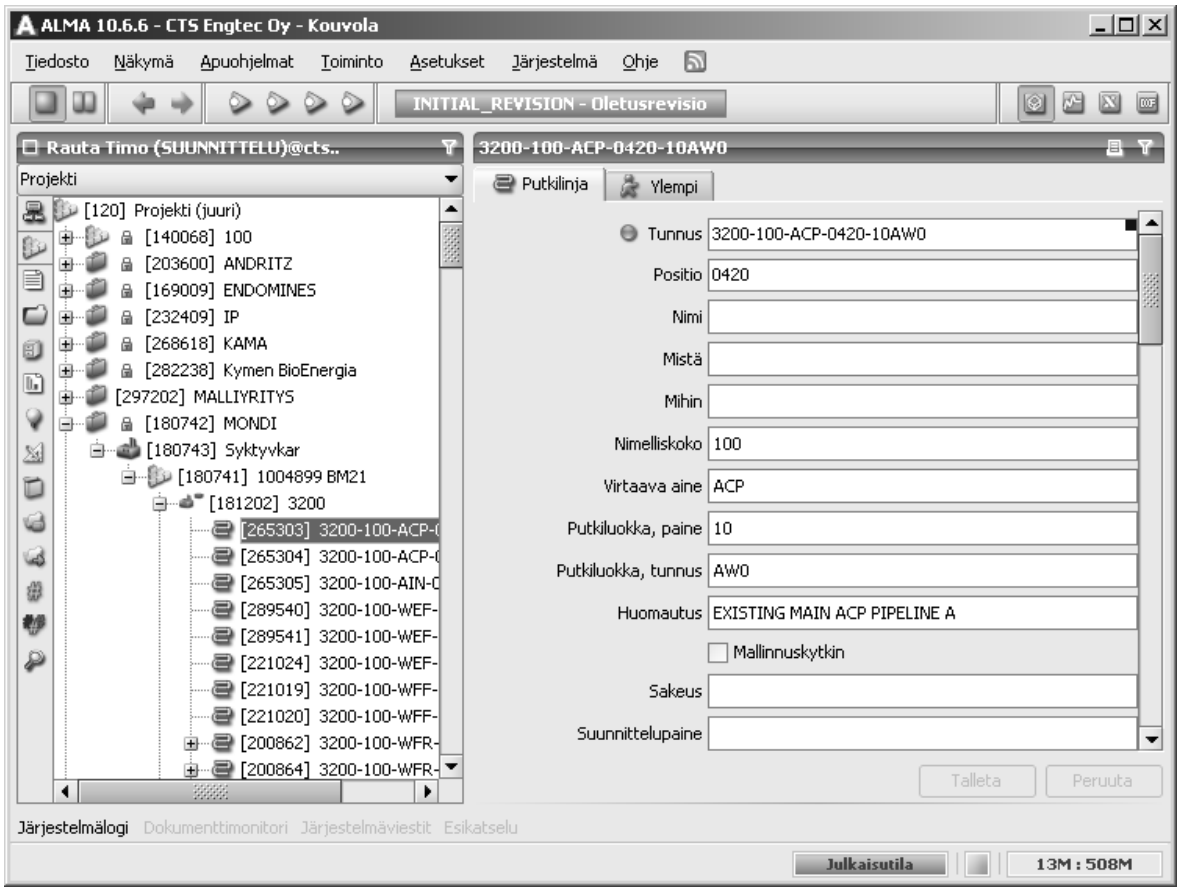
ottaa myös suora yhteys Alma-palvelimeen Internetin yli, mutta tämä vaatisi portin avaamista CTS:n palomuuriin. Suora yhteys asiakas- ja palvelinohjelmiston välillä voidaan tarvittaessa suojata SSL-protokollan (Secure Sockets Layer) avulla. Mikäli Alma-palvelinohjelmisto päivitetään, asiakasohjelmisto päivittää itsensä automaattisesti seuraavan käynnistyksen yhteydessä.

Alma-asiakasohjelmassa tiedot esitetään ja jäsennellään pääsääntöisesti hyödyntämällä hierarkkisia puurakenteita. Puut sisältävät erityyppisiä objekteja, joista kukin pyrkii esittämään jotain reaalimaailman ilmiötä, kuten yritystä, projektia, tehdasta, osastoa, putkilinjaa tai käyttäjää. Objektien tyyppiä kutsutaan usein myös perustyypiksi. Perustyypit ovat ALMA Softwaren määrittämiä, eivätkä käyttäjät voi luoda uusia perustyyppejä. Jokaisella objektilla on perustyypin lisäksi objektin yksilöivä tunniste (ID) sekä joukko muita attribuutteja, jotka sisältävät objektiin liittyvää tietoa.

Käyttäjät Alma-järjestelmässä kuuluvat aina joko yhteen tai useampaan käyttäjäryhmään. Jokaisen hierarkiassa olevan objektin näkyvyys voidaan määritellä käyttäjäryhmäkohtaisesti, jolloin kukin objekti on näkyvillä vain niille käyttäjäryhmille, joille on määritelty pääsyoikeudet objektiin. Muokkausoikeuksia ei voi määritellä objektikohtaisiksi, vaan ne voidaan määritellä ainoastaan kullekin perustyypille erikseen. Tämä tarkoittaa sitä, että käyttäjäryhmällä voi olla muokkausoikeudet esimerkiksi joko kaikkiin yritystyyppisiin objekteihin, tai ei mihinkään niistä. Myös attribuuteille voidaan määritellä näkyvyys- ja muokkausoikeudet käyttäjäryhmittäin. Yksittäisen objektin yksittäiselle attribuutille ei kuitenkaan voida määrittää erillisiä näkyvyys- tai muokkausoikeuksia, vaan attribuuttien oikeuksien muutokset koskevat aina kaikkia samanlaisia attribuutteja.

Kuvassa 10 on esitetty Alma-asiakasohjelman tyypillinen näkymä. Ohjelman ikkuna on jaettu kahteen osaan, jossa vasemmalla on selain, jolla puurakennetta voidaan selata, ja oikealla tavallisesti objektieditori. Muokattava objekti tai objekteja valitaan puurakenteen selainta hyödyntämällä. Kunkin objektin edessä on kuvake, joka kertoo objektin perustyypin. Kuvaketta seuraa hakasuluissa objektin ID sekä tunnus. Objekteja voidaan myös lisätä sekä poistaa puurakenneselaimen avulla. Almaan määritettävien sääntöjen

perusteella voidaan määritellä, minkä tyyppisiä objekteja kunkin objektityypin alle voidaan hierarkiassa luoda. Esimerkiksi putkilinja-objektin alle ei voi luoda yritys-objektia, mutta käsiventtiiliobjektin voi.



Kuva 10 - ALMA-asiakasohjelma

Puuselaimesta valitun objektin attribuutteja voidaan muokata objektieditoria käyttäen. Kuvassa 10 objektieditoriin on aukaistu putkilinja-objekti, jolla on erilaisia attribuutteja kuten ”positio”, ”nimelliskoko”, ”virtaava aine” jne. Objektieditorin avulla on mahdollista lisätä tai poistaa objektiin liittyviä attribuutteja sekä muokata niiden arvoja. Jokaisella attribuutilla on tyyppi, joka määrittää minkä tyylistä tietoa attribuuttiin voidaan tallentaa. Attribuuttityyppejä ovat esimerkiksi koodi, teksti, numero ja aika. Attribuutin tyyppi määrittää osaksi myös miten attribuutissa olevaa tietoa voidaan käsitellä - esimerkiksi kahden numerotyyppisen attribuutin arvot on mahdollista summata keskenään. Koodi-tyyppisille attribuuteille voidaan määrittää ns. laskentakaava, joka on Java-ohjelmointikielellä määritetty funktio, jonka suorittaminen palauttaa attribuutille arvon.

Tämän avulla on mahdollista mm. koostaa attribuutin arvo muitten attribuuttien arvoja hyödyntämällä. Esimerkiksi kuvassa 10 olevan putkilinjan ”Tunnus”-attribuutti on koostettu hierarkiassa ylemmällä tasolla olevan osasto-objektin tunnuksesta (3200), sekä putkilinjan nimelliskoosta (100), virtaavasta aineesta (ACP), positiosta (0420) ja putkiluokkatunnuksesta (AW0). Laskentakaavan avulla laskettua attribuutin arvoa kutsutaan laskentavastaukseksi.

3.2.3.2 ALMA Server

Alma-palvelinohjelmisto on Alma-järjestelmän komponentti, johon asiakaskerroksen sovellukset ottavat yhteyttä käsitelläkseen tietoa. Se voidaan nähdä eräänlaisena Alma-ohjelmiston ytimenä sekä viestintärajapintana tietojen tuontiin ja vientiin Alman ja asiakaskerroksen sovellusten välillä, mahdollistaen näin osaltaan järjestelmän hajautetun käytön. Suurin osa Alma-järjestelmän sovelluslogiikasta sijaitsee Alma-palvelimella. Palvelinohjelmisto on asiakasohjelmiston tapaan toteutettu Java-ohjelmointikielellä. Se ei kuitenkaan sisällä graafista käyttöliittymää asetuksien säätämiseen ja palvelimen valvontaan, vaan sen käyttö tapahtuu komentorivipohjaisen sovelluksen avulla. Järjestelmän pääkäyttäjäryhmään kuuluvat käyttäjät voivat tosin tarkastella palvelimen tilaa ja asetuksia myös Alma-asiakasohjelmasta käsin. Normaaliolosuhteissa palvelimen toimintaan ei juuri tarvitse puuttua, vaan se pyörii itsenäisesti taustalla. Palvelinohjelmisto on mahdollista asettaa myös Windowsin palveluksi, jolloin se käynnistyy automaattisesti käyttöjärjestelmän käynnistyksen yhteydessä. CTS:llä palvelinohjelmistoa ajetaan VMwaren virtualisointiohjelmistolla luodussa virtuaalikoneessa, jossa on Windows 2000 -käyttöjärjestelmä. Isäntäkoneen suorittimena toimii Intelin palvelinkäyttöön suunnattu Xeon X5355 -prosessori. Muistia virtuaalikoneelle on varattu gigatavun verran, joka on Alma Softwaren mukaan riittävästi järjestelmälle, jonka tietokanta koostuu noin miljoonasta objektista (ALMA käyttöohje, 2010). Työn kirjoitushetkellä CTS:n Alma-tietokanta sisältää n. 140 000 objektia.

Palvelinohjelmisto kommunikoi tietokannanhallintajärjestelmän kanssa JDBC-ajurin (Java Database Connectivity) välityksellä. Alman yhteydessä yleisesti käytettyjä

tietokannanhallintajärjestelmiä ovat SQL-92 standardin mukaiset MySQL ja Oracle. Tietokannanhallintajärjestelmä voi olla asennettuna samalle tai erilliselle koneelle Alma-palvelinohjelmiston kanssa. Erikseen asennettuna palvelinohjelmisto ja tietokannanhallintajärjestelmä eivät kamppaile samoista resursseista, kuten suoritinajasta ja muistista, mikä saattaa joissakin tilanteissa parantaa tiedonhallintajärjestelmän suorituskykyä. Toisaalta tällöin tulee myös huomioida verkon kaistanleveys ja latenssi, jotta verkon suorituskyky ei muodostaisi pullonkaulaa. CTS:llä Alma-palvelinohjelmisto ja MySQL-tietokanta sijaitsevat samalla virtuaalikoneella.

Alma-asiakasohjelman lisäksi palvelinohjelmiston kanssa voidaan kommunikoida joko web- tai ohjelmointirajapintojen kautta. WebALMA on Alma-palvelinohjelmiston sisään rakennettu WWW-palvelinohjelmisto, joka mahdollistaa Alman tietokannan selaamisen Internetin ylitse WWW-selainta käyttäen. Asiakkaalle WebALMA mahdollistaa ajan tasalla olevan projektimateriaalin tarkastelun ilman Alma-asiakasohjelmaa. Lisäksi se mahdollistaa Alman tietokannan selaamisen kaikilla laitteilla, jotka ovat varustettu Internet-yhteydellä ja WWW-selaimella. Tietoturvasyistä CTS:n Alma-palvelimeen ei kuitenkaan saa yhteyttä Internetistä ilman palomuuriin tehtävää tietoliikenteen sallivaa sääntöä. Tämä sääntö voidaan tarvittaessa luoda palomuuriin, mikäli asiakas haluaa selata Alman tietokantaa Internetin ylitse.

InterfaceALMA on kokoelma ohjelmointirajapintoja, jonka avulla Almasta voidaan lukea, ja jonka avulla sinne voidaan kirjoittaa tietoja ohjelmallisesti. CoreALMAan yhteydessä oleva Java-rajapinta mahdollistaa Java-pohjaisten ohjelmien yhdistämisen Almaan. CTS:llä ohjelmistokehityksessä on yleensä käytetty Visual Basic -ohjelmointikieltä, ja sen eri sovelluksiin integroitua VBA-varianttia (Visual Basic for Applications). Visual Basicilla toteutetut sovellukset voivat kommunikoida Alman kanssa ActiveX-rajapinnan kautta. Tämä ActiveX-rajapinta on toteutettu Java-rajapinnan päälle, ja sen tarkoituksena on välittää ulkoapäin tulevat pyynnöt Java-rajapinnalle. Java-rajapinta puolestaan palauttaa mahdolliset vastaukset pyyntöihin ActiveX-rajapinnalle. Sovelluksista ActiveX-rajapintaan päästään käsiksi hyödyntämällä ALMAX.dll-kirjastoa (Dynamic Link Library).

Alma ei alun perin sisältänyt CTS:n tarpeisiin soveltuvaa ohjelmointirajapintaa, mutta jo ensimmäisissä neuvotteluissa CTS:n ja Alma Softwaren välillä sellainen luvattiin toteuttaa. Rajapinnan ja sen funktioiden kehityksessä ja toteutuksessa on näin ollen otettu huomioon sekä CTS:n tarpeet että Alma Softwarelle annettu palaute. Tämän diplomityön puitteissa ohjelmoitujen ohjelmien (CTS Database Creator for Pine, CTS XML Creator for PMMATE) toteutuksen aikana ActiveX-rajapinta oli vielä kehitysvaiheessa, mistä seurasi se, että funktiokutsujen parametrit ja palautusarvot eivät olleet vielä tarkasti määriteltyjä (lukkoon löytyjä), ja niihin voitiin tarvittaessa tehdä muutoksia. Lisäksi joitakin mahdollisesti hyödyllisiä toimintoja, kuten esimerkiksi ilman käyttäjän interaktiota tapahtuvan kirjautumisen mahdollistavaa ns. SilentLogin-funktiota ei ollut vielä toteutettu.

Osa Alma-palvelinohjelmiston asetuksista on tallennettuna ”ALMA.ini”-tiedostoon, ja niitä voidaan tarvittaessa muokata tätä tiedostoa muokkaamalla. Tiedostossa on määritelty mm. asiakas- ja palvelinohjelmiston välisten yhteyksien ominaisuuksia, kuten portti jota palvelinohjelmisto kuuntelee, sekä SSL-salauksen käyttäminen. Lisäksi tiedostosta löytyy WebALMA:aan sekä tietokantayhteyteen liittyviä asetuksia, kuten tietokannan käyttäjätunnus ja salasana. Molemmat ovat tallennettuna selkokielisinä, joten kovin tietoturvallisena ratkaisua ei voida pitää. Mikäli palvelinohjelmisto ei löydä asetustiedostoa käynnistyessään, se kysyy käyttäjältä joitakin asetuksia käynnistyksen yhteydessä ja luo asetustiedoston uudelleen.

Alma-ohjelmiston käyttöoikeudet määrittelevä käyttölisenssi sijaitsee niin ikään tekstitiedostossa, jonka tiedostonimi on ”Alma.inf”. Lisenssitiedostossa määritellään mm. kuinka monta yhtäaikaista käyttäjää järjestelmässä voi olla, ja mitä Alma-ohjelmiston toiminnallisuuksia on käytettävissä. Lisenssin päivityksen yhteydessä Alma Software toimittaa uuden lisenssitiedoston, joka kopioidaan vanhan tiedoston päälle. Lisäksi Alma-palvelinohjelmisto on käynnistettävä uudelleen lisenssin päivityksen jälkeen. Vaikka lisenssiin liittyvät määritykset (kuten käyttäjien määrä) on luettavissa lisenssitiedostosta myös selkokielisenä, ei lisenssiä voida muokata manuaalisesti esimerkiksi tekstieditorilla.

3.2.3.3 MySQL-tietokannanhallintajärjestelmä

CTS:n Alma-ratkaisussa on käytössä MySQL-tietokannanhallintajärjestelmän versio 5.0, jonka kautta Alma-palvelinohjelmisto lukee ja kirjoittaa tietoja tietokantaan. MySQL on Oraclen tytäryhtiö Sun Microsystemsin omistama ja ylläpitämä laajalti käytössä oleva relaatiotietokannanhallintajärjestelmä. Sen vahvuuksiksi katsotaan mm. seuraavat ominaisuudet: (Gilmore, 2008; Kruckenberger & Pipes, 2005)

- Suorituskyky
- Joustavuus
- Helppokäyttöisyys
- Lisensointi

MySQL:n tavoitteena on alusta asti ollut tarjota suorituskykyinen tietokannanhallintajärjestelmä, aluksi jopa ominaisuuksien kustannuksella. MySQL mahdollistaa mm. usein suoritettavien SELECT-kyselyiden tallentamisen muistiin (engl. query caching), jolloin myöhemmin tehtävien samanlaisten kyselyiden suorittaminen nopeutuu. Myöhemmin MySQL:ään on lisätty myös monia kehittyneempiä tietokannanhallintajärjestelmissä käytettyjä ominaisuuksia, kuten tallennetut proseduurit, näkymät ja herättimet. Edellä mainitut ominaisuudet ovat sisältyneet MySQL:ään vuonna 2005 julkaistusta versiosta 5.0 lähtien.

Toinen MySQL:n vahvuuksista on joustavuus. Siitä on saatavilla versio useimmille käytössä oleville laitteistoalustoille ja käyttöjärjestelmille, kuten Microsoft Windowsille, Linuxille ja Oracle Solarikselle. Kilpailevista järjestelmistä esimerkiksi Microsoftin SQL Server toimii vain Microsoft Windows -pohjaisissa järjestelmissä (MSDN, 2010a). MySQL mahdollistaa myös käytettävän tietokantamoottorin vaihtamisen taulukohtaisesti, jolloin kussakin relaatiotietokannan taulussa voidaan käyttää kuhunkin tauluun ominaisuuksiltaan optimaalisesti sopivaa tietokantamoottoria. Tunnetuimpia MySQL:n hyödyntämiä tietokantamoottoreita ovat suorituskykyinen MyISAM, sekä tuen transaktioille sisältävä InnoDb. CTS:n Alma-tietokannassa kaikki taulut käyttävät InnoDb-tietokantamoottoria.

Kolmas MySQL:n vahvuus on helppokäyttöisyys. Sen hallintaan on olemassa useita työkaluja, kuten komentorivipohjainen MySQL Monitor, graafisella käyttöliittymällä varustetut MySQL Administrator, Query Browser ja Workbench sekä Internet-selaimella käytettävä, PHP:llä (PHP: Hypertext Preprocessor) toteutettu phpMyAdmin. Lisäksi MySQL sisältää ohjelmointirajapinnat useille käytössä oleville ohjelmointikielille, kuten Javalle, C++:lle sekä PHP:lle.

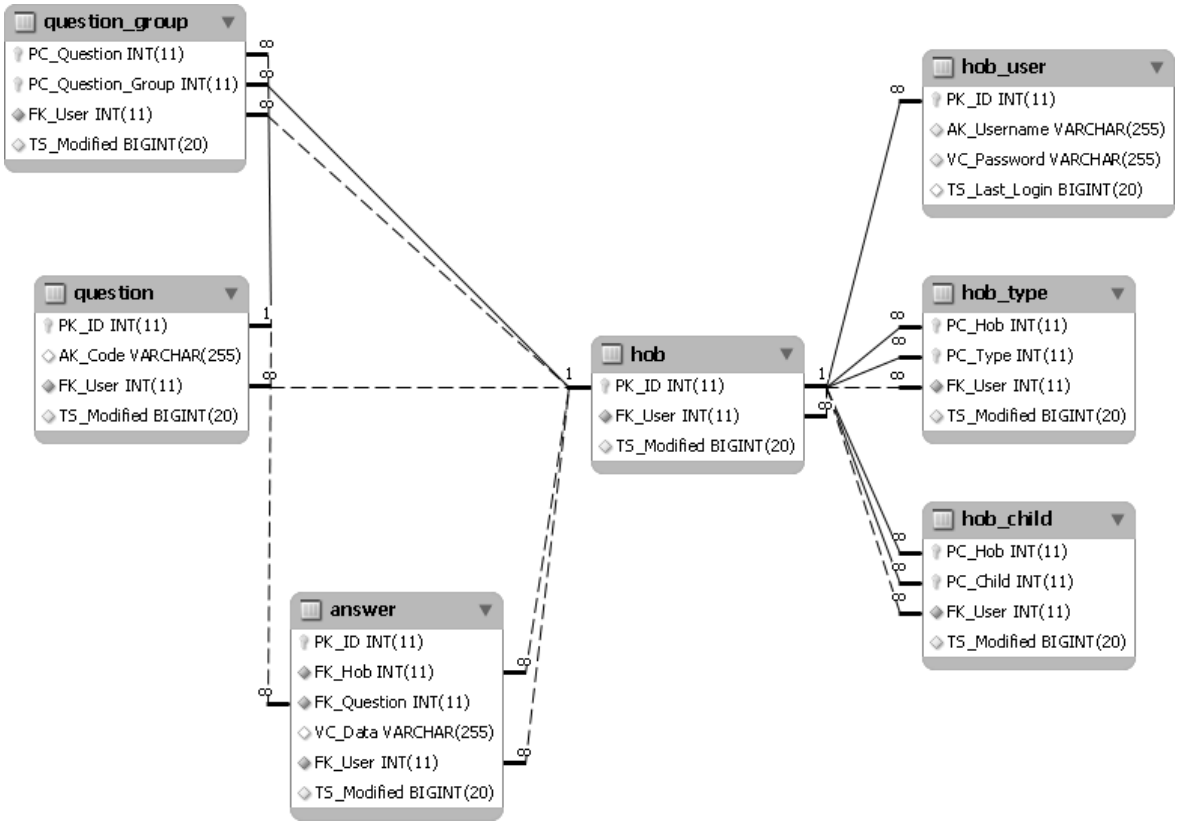
MySQL-ohjelmisto on saatavana kahdella vaihtoehtoisella lisenssillä. Siitä on saatavilla ilmainen GPL-lisenssiehtojen (GNU General Public Licence) mukainen lisenssi, sekä maksullinen kaupallinen lisenssi. Kaupallista lisenssiä tarvitaan esimerkiksi tilanteessa, jossa MySQL-ohjelmistoa halutaan levittää osana toista sellaista ohjelmistoa, jota ei haluta levittää GPL- tai GPL-yhteensopivan lisenssin alaisuudessa. GPL-lisenssiä voidaan puolestaan hyödyntää esimerkiksi tilanteessa, jossa MySQL-ohjelmistoa levitetään osana sellaista toista ohjelmistoa, jota levitetään myös GPL- tai GPL-yhteensopivan lisenssin alaisuudessa. Tällaisessa tapauksessa MySQL:n toimittamisesta osana toista ohjelmistoa ei aiheudu kustannuksia. Sen lisensointi palvelee siis sekä kaupallisten että vapaiden, avoimen lähdekoodin ohjelmistojen kehittäjiä.

3.2.3.4 Tietokanta

Alma-tiedonhallintajärjestelmään kuuluu olennaisena osana tietokanta, johon viitataan yleisesti käsitteellä Alma-tietotokanta. Alma-version 10.6.8 tietokanta koostuu 36 taulusta, ja sen rakenne pohjautuu pitkälti aikaisemmin esiteltyyn EAV-malliin (Raudasoja, 2009a). EAV-mallin käyttö Almassa mahdollistaa esimerkiksi sen, että käyttäjät pystyvät määrittelemään uusia attribuutteja objekteille, ilman että tietokannan skeemaan tarvitsee tehdä muutoksia. Koska Alma-tietokannan rakenne on suhteellisen monimutkainen, esitellään seuraavassa vain hyvin karkeasti, miten objektit ja niiden attribuutit Alma-tietokantaan tallentuvat.

Alma-tietokannan karkea rakenne keskeisimpine tauluineen on esitetty kuvassa 11. Huomioitavaa on, että kuvassa eivät näy kaikki Alma-tietokannan taulut, eivätkä niiden

väliset suhteet. Myös esitetyistä tauluista on karsittu joitakin attribuutteja selkeyden vuoksi.



Kuva 11 - Alma-tietokannan rakenne

Keskeisin taulu Alman tietokannassa on ”hob” (hierarchy object), joka sisältää kaikki tietokannan kiinteät sekä käyttäjien luomat objektit. Objekteja ovat mm. prosessilaitteet, putkilinjat, automaatiopositiot ja käyttäjät. Objektin tyyppi on määritettynä taulussa ”hob_type”. Jokaisella objektilla on yksilöllinen tunniste (ns. Alma ID), jonka avulla objekti yksilöidään tietokannassa. Tämä tunniste on tallennettuna ”hob”-taulun attribuuttiin ”PK_ID”. Lisäksi taulussa on mm. tieto siitä, kuka käyttäjistä objektin on luonut, sekä milloin sitä on viimeksi muokattu. Objektilla voi olla myös revisiotietoja, mutta CTS:llä nämä revisiointiominaisuudet eivät ole käytössä. EAV-mallissa Alman hierarkiaobjekti vastaa entiteettiä.

Suoraan kunkin objektin alla sijaitsevat objektit (lapsiobjektit) käyvät ilmi taulusta ”hob_child”. Tämän taulun avulla voidaan muodostaa Almassa käytössä oleva hierarkinen

puurakenne. Yhdellä objektilla voi olla useampi kuin yksi vanhempi, mikä tarkoittaa esimerkiksi sitä, että sama laitepositio voi olla sekä jonkun putkilinja-objektin että jonkun automaatiopositio-objektin alla.

Järjestelmän käyttäjien tiedot on tallennettuna tauluun ”hob_user”. Tästä taulusta löytyvät kunkin käyttäjän käyttäjätunnus sekä salasanasta laskettu tiiviste. Lisäksi taulusta ilmenee ajankohta, jolloin käyttäjä on viimeksi kirjautunut järjestelmään, ja jolloin käyttäjä on viimeksi vaihtanut salasanansa.

Tauluun ”question” on tallennettu Almassa käytettävät attribuuttimäärittelyt, joita kutsutaan Alma-tietokannassa myös kysymyksiksi. EAV-mallin attribuutti vastaa siis Alma-tietokannan kysymystä. Kullakin attribuutilla on yksilöllinen ID, sekä koodinimi, kuten ”ALMA_NAME” tai ”CPE_pine_position”. Attribuuteista voidaan tarvittaessa koostaa attribuuttiryhmiä, joiden tarkoituksena on tehostaa Alma-tiedonhallintajärjestelmän käyttöä. Jos useammalle objektille pitää lisätä samat attribuutit, on kokonaisen attribuuttiryhmän lisääminen objektille kätevämpää ja nopeampaa kuin attribuuttien lisääminen objektille yksi kerrallaan. Jokainen attribuutti voi kuulua useampaan attribuuttiryhmään, ja nämä attribuuttiryhmät ovat tallennettuna tauluun ”question_group”.

Alma-tietokannan taulu ”answer” yhdistää toisiinsa objektin ja siihen liittyvät attribuutit. Toisin sanoen se sisältää vierasavaimet ”hob” ja ”question” -tauluihin. Jokaiseen objektiin voi näin ollen liittyä teoriassa rajaton määrä joko valmiita tai käyttäjien määrittelemiä attribuutteja. Lisäksi tämä taulu sisältää kentän, johon on tallennettu kuhunkin attribuuttiin liittyvä arvo. Tätä arvoa kutsutaan Almassa myös vastaukseksi. EAV-mallissa Alma-tietokannan vastaus vastaa arvoa.

CTS:n Alma-tietokannasta otetaan ajastettu varmuuskopio kerran vuorokaudessa. Varmuuskopioinnissa hyödynnetään Alma Softwaren ohjeistuksen mukaisesti MySQL:n mukana toimitettavaa ”mysqldump”-ohjelmaa, joka luo sille parametrina annettavasta tietokannasta vedoksen, jota kutsutaan myös tietokantadumpiksi. Ohjelman luoma vedostiedosto sisältää SQL-käskyjä, joiden avulla tietokanta voidaan luoda kokonaan

uusiksi, sisältö mukaan lukien. Vedostiedosto nimetään viikonpäivän mukaan, ja se tallentuu automaattisesti aina edellisen viikon vastaavan viikonpäivän vedostiedoston päälle. Tämä on tarpeen, koska muussa tapauksessa suurten vedostiedostojen levytilan käyttö kasvaisi ajan mittaan liian suureksi, ellei vanhempia tiedostoja käydä poistamassa joko ohjelmallisesti tai käsin. Käytännössä käytössä on siis tietokannasta päivittäin otetut varmuuskopiot viimeisen viikon ajalta. Vaihtoehtoisesti varmuuskopioinnissa voitaisiin käyttää MySQL:n mahdollistamaa inkrementaalista varmuuskopiointia. Inkrementaalisessa varmuuskopioinnissa koko tietokantaa ei varmuuskopioida joka kerta, vaan se varmuuskopioidaan ainoastaan siltä osin, miltä se on muuttunut edelliseen varmuuskopioon nähden.

Varmuuskopioinnissa syntyneet vedostiedostot sijaitsevat samalla kiintolevyllä itse tietokannan kanssa, joten ne eivät suojaa tietokannassa olevaa tietoa kiintolevyrikon varalta. Vedostiedostoja tarvitaankin lähinnä silloin, mikäli tietokanta menee syystä tai toisesta sekaisin, tai sieltä poistetaan vahingossa tietoa. Kiintolevyjen rikkoutumisesta aiheutuvat haitat pyritään puolestaan välttämään käyttämällä RAID-levyjärjestelmää (Redundant Array of Inexpensive Disks) sekä nauhavarmistusta. RAID-levyjärjestelmässä vikasietoisuus saavutetaan tiedon redundantilla eli ylimääräisellä tallentamisella (Patterson et al., 1988). Käytännössä tämä tarkoittaa sitä, että sama tieto kopioidaan automaattisesti useammalle kiintolevylle, tai että tiedosta lasketaan ns. pariteetti, jonka avulla alkuperäinen tieto pystytään rekonstruktoimaan levyrikon sattuessa.

4 CTS PINE

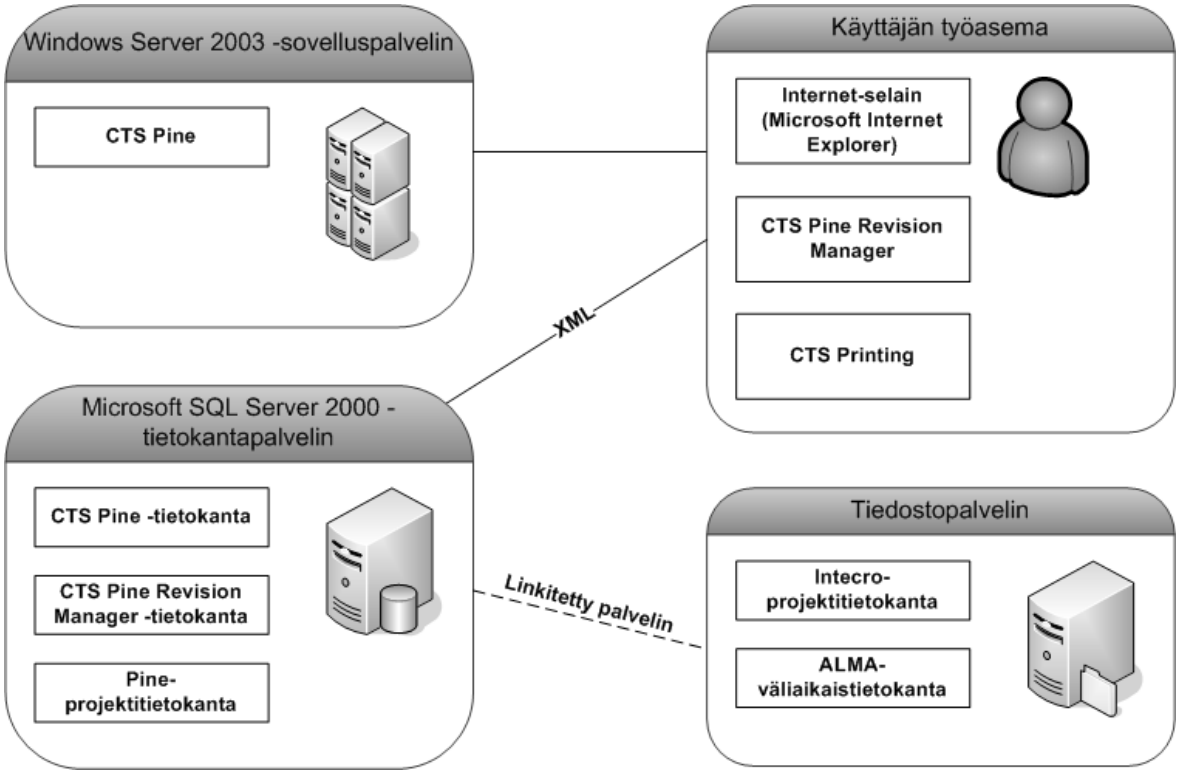
CTS Pine on CTS:llä kehitetty sovellus piirikohtaisten toimintakuvausdokumenttien luomiseen, hallintaan sekä tulostamiseen. Piirillä tarkoitetaan pienintä toiminnallista kokonaisuutta instrumentointijärjestelmässä, ja se voi sisältää yhden tai useampia instrumentteja (Venkula, 2003). Instrumentti puolestaan on yleisesti käytetty nimitys laitteesta, jota käytetään jonkin prosessin ohjaamiseen tai valvontaan. Instrumentteja ovat esimerkiksi pumput, venttiilit ja anturit. Yksittäinen piiri voi koostua näin ollen esimerkiksi pumpusta sekä siihen liittyvistä antureista ja säätimistä. Valmis prosessilinja voi sisältää tuhansia piirejä.

Kunkin piirin toiminta on määritelty piirikohtaisessa toimintakuvausdokumentissa, jossa esitellään, mitä piiri tekee sekä määritellään sellaiset säännöt, ehdot ja riippuvuussuhteet, joiden mukaan piiri toimii. Tärkeimpiä tällaisia määrittelyjä ovat ns. lukitukset ja käynnistysehdot. Lukituksissa määritellään ne olosuhteet, joissa piirin toiminta lukkiutuu. Lukkiutumisella tarkoitetaan esimerkiksi moottorin käynnistymistä tai pysähtymistä, tai venttiilin avautumista tai sulkeutumista. Käynnistysehdoissa puolestaan määritellään sellaiset olosuhteet, joiden toteutuessa piiri aloittaa toimintansa. Valmiita toimintakuvausdokumentteja voidaan käyttää mm. prosessilinjan ohjauksessa mahdollisesti tarvittavan tietokoneohjelmiston toteutuksessa sekä prosessilinjan operaattoreiden apuvälineenä.

Pinen tarkoituksena on tehostaa piirikohtaisten toimintakuvausdokumenttien luontia ja hallintaa. Ennen Pineä toimintakuvausdokumentit laadittiin käsin jotakin tekstinkäsittelyohjelmaa, kuten Microsoft Wordia apuna käyttäen. Dokumenttien laatiminen tällä tavalla oli kuitenkin työlästä ja virhealtista. Periaatteessa saman tiedon sisältäneet dokumentit piti luoda useampaan kertaan, jos asiakkaat ja alihankkijat halusivat ne eri ulkoasuissa. Myös muuttuneiden tietojen päivittäminen projektin kaikkiin toimintakuvausdokumentteihin käsin oli hankalaa. Pineä käytettäessä kirjoitusvirheet piirien tunnuksissa saadaan eliminoitua, koska piirien tiedot haetaan suoraan tietokannasta. Myös tietojen päivittäminen helpottuu, koska niitä ei tarvitse päivittää käsin jokaiseen

dokumenttiin erikseen. Kun muutokset on kerran tehty tietokantaan, toimintakuvausdokumentit ovat nopeasti generoitavissa ohjelmallisesti uudelleen. Toimintakuvausdokumenttien tieto on aluksi XML-muodossa (Extensible Markup Language), josta se muunnetaan tarvittaessa HTML-muotoon (Hypertext Markup Language). HTML-dokumenttien ulkoasua voidaan puolestaan muokata kutakin tarkoitusta varten sopivaksi XSLT- (Extensible Stylesheet Language Transformations) ja CSS (Cascading Style Sheets) -tyylisivuja käyttämällä.

Pinen lisäksi ohjelmistoon kuuluu kaksi muuta sovellusta, jotka ovat CTS Pine Revision Manager ja CTS Printing. Ohjelmiston sijoittelukaavio on esitetty kuvassa 12. Itse Pine-sovellus sijaitsee CTS:n sisäisellä IIS-pohjaisella (Internet Information Services) web-palvelimella, josta se voidaan käynnistää yksittäisten työasemien Internet-selaimissa suoritettavaksi. Suurin osa sovelluksen toiminnasta sijaitsee asiakkaan, eli käyttäjän päässä. Pinen käyttöliittymä asiakaspäässä on toteutettu HTML:n avulla, ja toiminnallisuus pääosin JavaScriptillä (Kauppinen, 2008). CTS Pine Revision Manager -sovellusta käytetään toimintakuvausdokumenttien revisioiden hallintaan. Se on Microsoft Visual Basic 6.0:lla toteutettu itsenäinen sovellus, joka tulee asentaa ennen käyttöä kullekin työasemalle erikseen. CTS Printing on puolestaan toimintakuvausdokumenttien jonotulostukseen käytetty ohjelma. Se on niin ikään Visual Basicilla toteutettu itsenäinen ohjelma, jonka toimintaan ei tämän diplomityön puitteissa kuitenkaan tarvinnut tutustua tarkemmin.



Kuva 12 - CTS Pine -ohjelmiston sijoittelukaavio

Pine-ohjelmiston käyttämät tietokannat sijaitsevat Microsoft SQL Server 2000 -pohjaisella tietokantapalvelimella, johon Pine-sovelluksen asiakaspää tekee hakuja SQLXML-rajapinnan kautta. Lähes kaikki tietokantapalvelimelle lähetettävä ja sieltä vastaanotettava tieto on XML-muodossa. Pineen liittyviä tietokantoja on kolme, joista Pine-tietokanta sisältää toimintakuvausdokumenttien tiedot, Revision Manager -tietokanta revisioajojen tiedot ja Pine-projektitietokanta erilaisia näkymiä linkitettyyn Intecro-projektitietokantaan. Pinen ja revisiomanagerin tietokannat ovat rakenteeltaan hyvin samankaltaisia, mutta revisiomanagerin tietokannassa säilytetään myös vanhempien toimintakuvausdokumenttien (revisioiden) tietoja, kun Pinen tietokannassa kustakin piiristä esitetään vain nykyisin toimintakuvausdokumentti.

Huomioitavaa on, että toimintakuvausdokumenteissa esiintyvien piirien tietoja ei haeta suoraan kunkin projektin projektitietokannasta, vaan jokaista Pine-projektia varten SQL Serverille luodaan oma Pine-projektitietokanta. Varsinainen projektitietokanta puolestaan liitetään tähän Pine-projektitietokantaan käyttämällä SQL Serverin tarjoamaa ”linkitetty palvelin” -toimintoa. Itse Pine-projektitietokanta sisältää ainoastaan näkymiä tähän

linkitettyyn projektitietokantaan. Näitä näkymiä muokkaamalla voidaan tietokantahakujen tuloksia muokata asiakkaan ja projektin tarpeita vastaaviksi. Esimerkki tästä on piirin positiotunnuksen koostaminen, joka voi vaihdella projektikohtaisesti. Näkymiä muokkaamalla positiotunnus voidaan koostaa piiriin liittyvien eri attribuuttien arvoista kuhunkin tilanteeseen sopivaksi.

Projekteissa käytössä olevan tiedonhallintajärjestelmän vaihtaminen Intecrosta Almaan johti Pinen tapauksessa siihen, että ohjelmisto ei enää toiminut sellaisenaan, koska Alma-tietokantaa ei enää pystytty linkittämään Pine-projektitietokantaan aikaisemman Intecro-tietokannan tapaan. Alma-järjestelmän käyttöönottoa suunniteltaessa tutkittiinkin, voisiko Pinen toiminnallisuuden jotenkin siirtää kokonaan Almaan, jolloin Pinen käytöstä voitaisiin luopua (CTS Engtec Oy, 2008). Tämä osoittautui kuitenkin liian hankalaksi, ja seuraava askel oli yrittää saada Alma ja Pine toimimaan keskenään. Vaihtoehtoina tässä vaiheessa oli joko Pine-sovelluksen muokkaaminen tai Intecro-tyylisen väliaikaistietokannan luominen Alma-tietokannan projektitiedoista. Koska Pinen sisäinen rakenne ja toiminta eivät olleet kenelläkään selvästi tiedossa, ja Alman käyttöönottamisella alkoi olla kiire, päätettiin Pinen integroiminen Almaan toteuttaa Intecro-tyylisen väliaikaistietokannan kautta. Tämä ratkaisu nähtiin luonteeltaan ainoastaan väliaikaisena, sillä Pinen tulevaisuutta oli tarkoitus miettiä myöhemmin uudelleen.

Tämän diplomityön puitteissa toteutettiin kaksi CTS Pineen liittyvää sovellusta: CTS Database Creator for Pine ja CTS Pine Management Utility. CTS Database Creator for Pine -sovelluksen tarkoituksena on luoda Intecro-projektitietokannan rakennetta vastaava tilannevedostietokanta Alma-tietokannan projektitiedoista, ja näin ollen mahdollistaa Pinen käytön jatkaminen. CTS Pine Management Utility -sovelluksen tarkoituksena on puolestaan helpottaa uuden Pine-projektin perustamista, sekä auttaa Pinen tietokannassa ongelmia aiheuttavien niin sanottujen duplikaattipositioiden paikantamisessa.

4.1 CTS Database Creator for Pine

CTS Database Creator for Pine -sovelluksen tarkoitus on luoda Alma-tietokannasta valitusta projektista väliaikainen Microsoft JET -pohjainen tietokanta. Tämä tietokanta vastaa rakenteeltaan aikaisemmin käytössä olleita Intecro-projektitietokantoja, jolloin sitä voidaan käyttää totuttuun tapaan CTS Pinen yhteydessä. Sovelluksen kehitys oli aloitettu jo ennen tämän diplomityön aloittamista, ja siitä puuttui lähinnä liityntä Alma-tietokantaan. Tämän diplomityön puitteissa sovellukseen toteutettiin tämä liityntä, joka mahdollistaa tietojen lukemisen ja kirjoittamisen Alma-tietokantaan. Lisäksi sovellusta paranneltiin ja siihen lisättiin käyttöä helpottavia toiminnallisuuksia. Seuraavissa kappaleissa käydään läpi sovelluksen toteutus, rakenne, toiminta ja siihen liittyviä ongelmia.

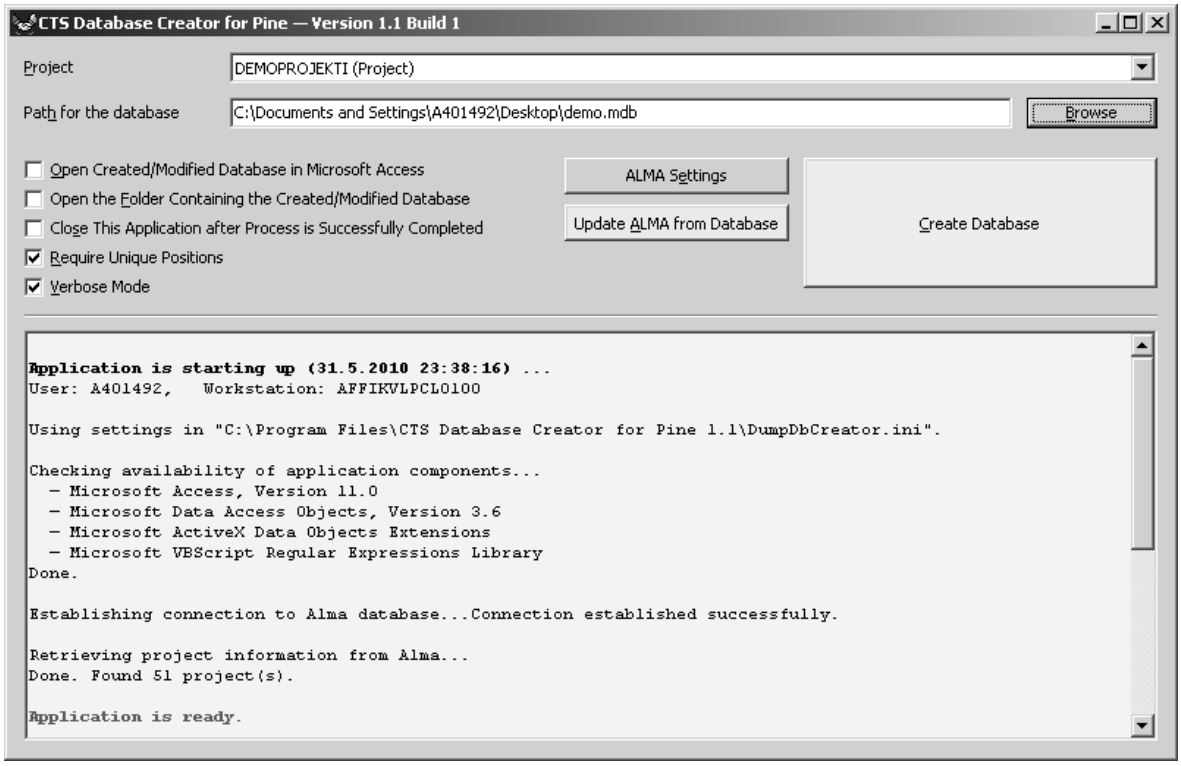
4.1.1 Toteutus

CTS Database Creator for Pine on toteutettu Microsoft Visual Basic 6.0:lla, joka on iästään huolimatta edelleen CTS:llä käytössä oleva ohjelmointityökalu. Uusimmista käyttöjärjestelmistä Microsoftin Windows 7 tukee edelleen Visual Basic 6.0:n ajonaikaista ympäristöä, mutta tulevissa käyttöjärjestelmissä sen tukemisesta ei ole enää varmuutta. Visual Basic 6.0:n sovelluskehitysympäristö ei puolestaan ole enää virallisesti tuettu (MSDN, 2010c).

Database Creator for Pinen toteutuksessa on hyödynnetty joukkoa Visual Basicin ulkopuolisia kirjastoja. Alma-järjestelmän kanssa sovellus kommunikoi käyttämällä Alman ActiveX-rajapinnan tarjoamia funktioita, jotka mahdollistavat mm. tiedon hakemisen sekä lukemisen Alma-tietokannasta että tiedon kirjoittamisen tietokantaan. Väliaikaistietokannan rakenteen luomisessa ja tiedon syöttämisessä sinne hyödynnetään puolestaan Microsoftin ADO- (ActiveX Data Objects) ja ADOX (ADO Extensions) -kirjastoja. Lisäksi käytössä on Microsoft VBScript Regular Expressions -kirjasto, jota hyödynnetään mm. luotavan väliaikaistietokannan tiedostonimen kelpoisuuden tarkistamisessa. Sovellus on toteutettu englanninkielisenä, jotta myös suomea hallitsemattomat suunnittelijat voisivat käyttää sitä.

4.1.2 Käyttöliittymä ja rakenne

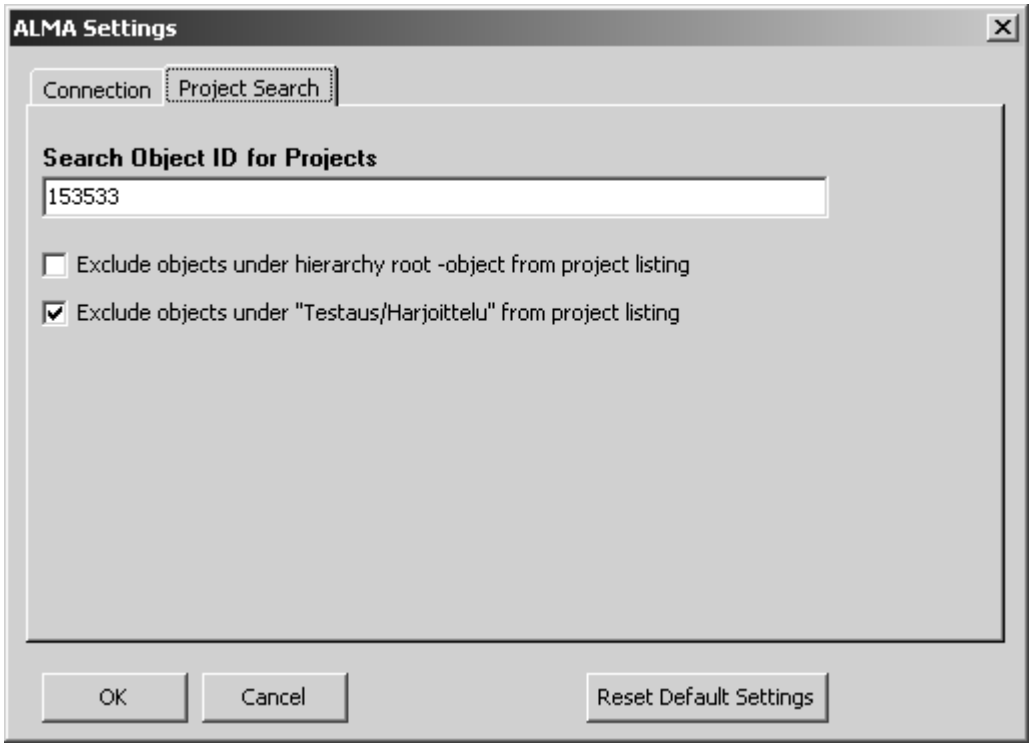
Sovelluksen käyttöliittymä koostuu kolmesta lomakkeesta. Ensimmäinen lomakkeista on sovelluksen päälomake, jonka avulla väliaikaistietokannan luominen pääsääntöisesti tapahtuu. Päälomakkeen ulkoasu ja toiminnot on nähtävissä kuvasta 13. Tällä lomakkeella voidaan määritellä mm. se Alma-tietokannan projekti, josta väliaikaistietokanta luodaan, sekä väliaikaistietokannan tiedostopolku ja -nimi. Lisäksi päälomakkeelta voidaan valita joitakin väliaikaistietokannan luomiseen liittyviä lisätoiminnallisuuksia. Tämä tapahtuu valitsemalla haluttuun toiminnallisuuteen liittyvä valintaruutu, kuten esimerkiksi ”Verbose Mode”. Kun kaikki asetukset on määritetty, väliaikaistietokannan luominen käynnistyy painiketta ”Create Database” napauttamalla.



Kuva 13 - CTS Database Creator for Pine -sovelluksen päälomake

Toisessa sovellukseen liittyvässä lomakkeessa voidaan määritellä erilaisia Almaan liittyviä asetuksia. Tämän lomakkeen ulkoasu on nähtävissä kuvassa 14. Se koostuu kahdesta välilehdestä, josta toisella voidaan määritellä Alma-yhteyteen liittyviä asetuksia, ja toisella projektihakuun liittyviä asetuksia. Yhteyteen liittyvissä asetuksissa voidaan mm. määritellä

sovelluksen vaatiman ”ALMALoader.jar” -tiedoston sijainti. Tätä tiedostoa tarvitaan käynnistämään instanssi Alma-asiakasohjelmasta, jota ilman Alma-rajapinnan tarjoamat toiminnot eivät toimi. Projektin hakuun liittyvien asetusten avulla voidaan puolestaan räätälöidä projektihakua paremmin henkilökohtaisia tarpeita vastaavaksi. Projektihakua käytetään Alma-tietokannassa olevien projektien lisäämiseen päälomakkeen projektinvalintalistaan. Haku tapahtuu hyödyntämällä Almaan luotua hakuobjektia. Jos projektinvalintalistauksessa ei haluta näyttää kaikkia Alma-tietokannan projekteja, voidaan Almaan käydä luomassa uusi hakuobjekti, joka palauttaa vain rajoitetun määrän tuloksia. Database Creator for Pine voidaan tämän jälkeen määrittää käyttämään tätä juuri luotua hakuobjektia syöttämällä sen ID kenttään ”Search Object ID for Projects”. Hakuasetuksista voidaan lisäksi määritellä, lisätäänkö hierarkiajuuren sekä ”Testaus/Harjoittelu” -haaran alla sijaitsevat projektit hakutuloksiin. Hierarkiajuuren alla sijaitsevat tavallisesti jo valmistuneet projektit, kun taas ”Testaus/Harjoittelu” -haaran alla sijaitsevat yleensä harjoittelutarkoitukseen luodut projektit. Näitten projektien näyttäminen projektinvalintalistauksessa on usein turhaa, sillä ne hankaloittavat ”oikeitten” projektin löytämistä ja valitsemista.



Kuva 14 - "ALMA Settings" -lomake

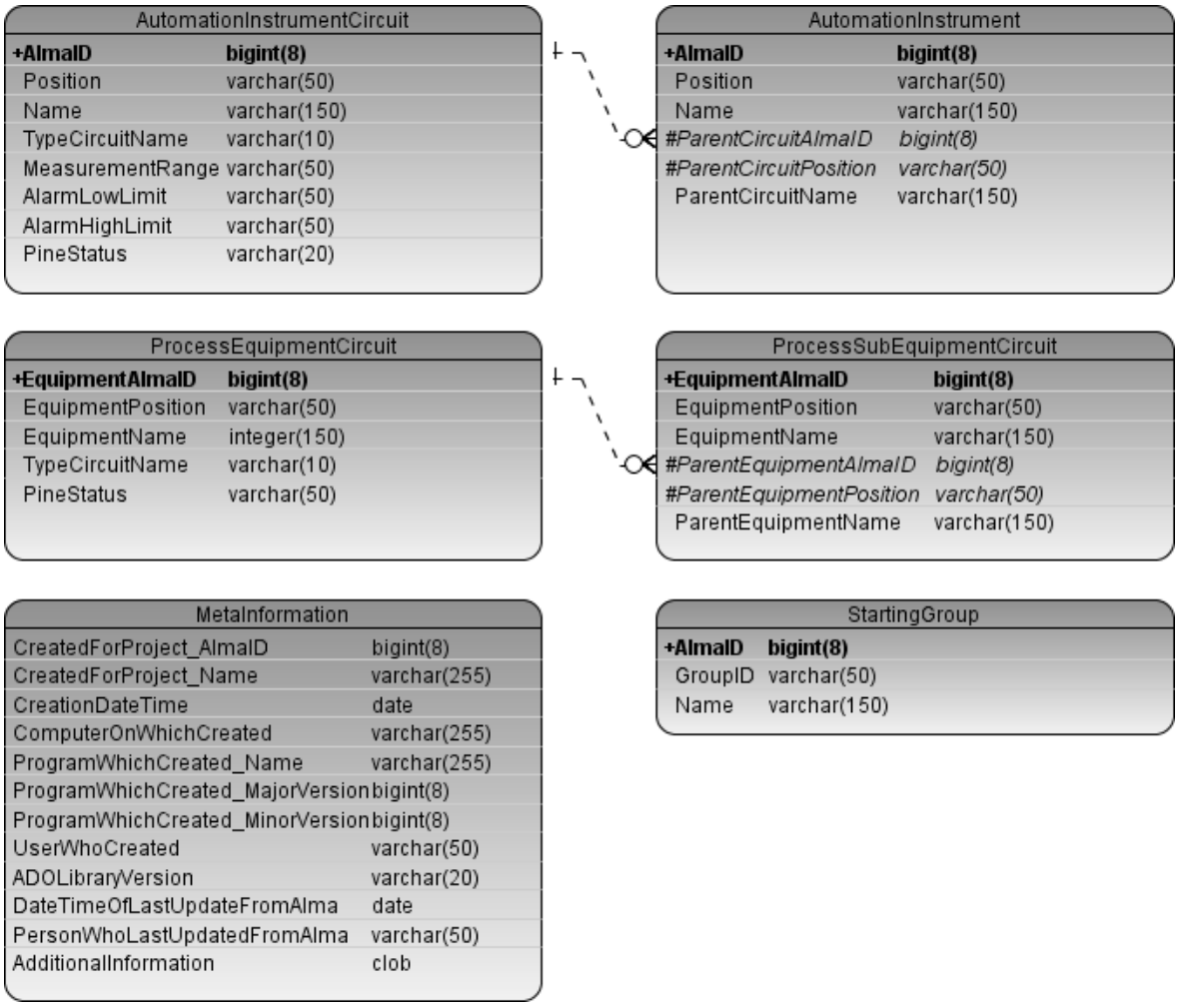
Kaikki sovelluksessa tehtävät asetukset tallennetaan sovelluksen kanssa samassa hakemistossa sijaitsevaan tiedostoon ”DumpDbCreator.ini”, josta niitä voi tarpeen vaatiessa käydä muokkaamassa myös manuaalisesti. Tämä tiedosto luetaan aina sovelluksen käynnistyessä, jolloin kukin sovelluksen asetuksista asetetaan tiedostosta luettuun tilaan. Näin ollen asetuksia ei tarvitse muokata mieleisekseen aina sovelluksen käynnistyksen yhteydessä, vaan ne ovat muistissa edellisiltä käyttökerroilta. Tiedostoon on edellä mainittujen sovelluksesta muutettavien asetuksien lisäksi tallennettu mm. sovellusikkunan viimeisin koko ja sijainti.

Sovelluksen kolmas käytössä oleva lomake sisältää yksinkertaisen tekstikentän, johon syötetyt lisätiedot tallennetaan väliaikaistietokannan piilotettuun metatietotauluun. Tähän tauluun tallentuu lisäksi mm. tieto siitä, milloin kyseessä oleva väliaikaistietokanta luotiin, kuka sen on luonut ja millä työasemalla, sekä millä Database Creator for Pinen versiolla väliaikaistietokanta on luotu. Metatietotaulua voidaan hyödyntää lähinnä mahdollisten ongelmatilanteiden selvittämisessä.

Osa lomakkeiden toiminnasta vastaavasta ohjelmakoodista löytyy suoraan lomaketiedostoista, mutta suurimmat toimintokokonaisuudet on sijoitettu erillisiin moduuleihin, joiden avulla koodin rakennetta pyritään selkeyttämään. Database Creator for Pine sisältää neljä moduulitiedostoa. Näistä ”AlmaX”-moduuli sisältää Almaan liittyvän toiminnallisuuden, kuten funktiot sisäänkirjautumiseen sekä piirien tietojen hakemiseen. ”API”-moduuli puolestaan sisältää sellaisia funktioita, joiden toiminnallisuuden toteuttamisessa on tarvittu Windowsin API:n (Application Programming Interface) tarjoamien toiminnallisuuksien hyödyntämistä. ”DBManagement”-moduuli koostuu tietokantojen hallintaan liittyvistä funktioista. Esimerkiksi väliaikaistietokannan luomisessa käytettävä ohjelmakoodi on määritetty täällä. Muu sekalainen toiminnallisuus on sijoitettu ”Miscellaneous”-moduuliin.

Sovelluksen avulla luotavan Alma-väliaikaistietokannan taulut ja niiden väliset suhteet ovat esitetty kuvassa 15. Väliaikaistietokanta koostuu kuudesta taulusta. Jokainen rivi kussakin taulussa, metatietotaulua lukuun ottamatta, vastaa tiettyä objektia Almassa. Nämä vastaavuudet on esitetty taulukossa 3. Esimerkiksi kaikkien valittuun projektiin kuuluvien

prosessilaitteiden tiedot tallentuvat väliaikaistietokannan tauluun ”ProcessEquipmentCircuit”. Aikaisemmin vastaava tieto oli puolestaan luettavissa Intecro-tietokannan taulusta ”PLAITE”. Vaikka väliaikaistietokannan ja Intecro-tietokannan taulujen nimet eivät vastaakaan toisiaan, ei tästä ole haittaa, sillä tietojen lukemisessa Pineen hyödynnetään Pine-projektitietokannassa määritettyjä näkymiä. Näihin näkymiin voi puolestaan määrittää myös taulun nimen, josta tieto haetaan. Jokaisessa väliaikaistietokannan taulussa, metatietotaulu pois lukien, rivien yksilöllisenä tunnisteena (perusavaimena) toimii vastaavan objektin Alma-ID.



Kuva 15 - Alma-väliaikaistietokannan taulut

Objektit Almassa	Taulu Alma-väliaikaiskannassa	Taulu Intecro-kannassa
Prosessilaiteet	ProcessEquipmentCircuit	PLAITE
Sähköpiirit	ProcessSubEquipmentCircuit	PLISALAITE
Automaatiopositiot	AutomationInstrumentCircuit	IPIIRI
Laitepositiot	AutomationInstrument	ILAITE
Käynnistysryhmät	StartingGroup	PKAYNRYHMA

Taulukko 3 - Tietokantojen taulujen vastaavuudet

Jokaisen väliaikaistietokantaan tuotavan laiteposition tulee kuulua Alman hierarkiassa suoraan jonkin automaatioposition alle. Vastaavasti jokaisen sähköpiirin täytyy kuulua suoraan jonkin prosessilaitteen alle. Mikäli näin ei ole, laitepositiota tai sähköpiiriä ei tuoda väliaikaistietokantaan. Väliaikaistietokannassa laitepositiot voidaan yhdistää isäntänä toimivaan automaatiopositioon ”ParentCircuitAlmaID” ja ”ParentCircuitPosition” -attribuuttien avulla, ja sähköpiirit isäntänä toimivaan prosessilaitteeseen puolestaan ”ParentEquipmentAlmaID” ja ”ParentEquipmentPosition” -attribuuttien avulla.

Kuten taulukosta 3 voidaan nähdä, CTS Pinen kannalta oleelliset Alma-tietokannan objektit ovat siis prosessilaitteet, sähköpiirit, automaatioposiot, laitepositiot sekä käynnistysryhmät. Suurin osa näitten objektien sisältämistä attribuuteista on Pinen kannalta hyödyttömiä, eikä niiden arvoja näin ollen tarvitse tuoda luotavaan väliaikaistietokantaan. Taulukossa 4 on esitetty, mistä Alma-tietokannan attribuutista kunkin väliaikaistietokannan attribuutin arvo noudetaan. Esimerkiksi väliaikaistietokannan kunkin piirin positiotunnus haetaan Alma-tietokannan attribuutista ”Pine-positio”. Kuten aiemmin on todettu, Intecron tapauksessa esimerkiksi tämä positiotunnus voidaan koostaa useista Intecro-tietokannan attribuuteista, hyödyntämällä Pine-projektitietokantaan luotavia näkymiä. Alma-väliaikaistietokannan tapauksessa nämä näkymät muuttuvat periaatteessa turhiksi, sillä esimerkiksi positiotunnus luetaan aina samasta väliaikaistietokannan attribuutista. Jos positiotunnus halutaan aikaisempaan tyyliin koostaa useista objektiin liittyvistä arvoista, voidaan tämä toteuttaa määrittelemällä halutunlainen laskentakaava Almassa attribuutille ”pine-positio”.

	Attribuutti väliaikaistietokannassa	Attribuutin nimi ALMA-tietokannassa	Attribuutin koodi ALMA-tietokannassa
ProcessEquipmentCircuit			
Perusavain	EquipmentAlmaID	Objektin ID	Objektin ID
Yksilöivä*	EquipmentPosition	Pine-positio	CPE_pine_position
	EquipmentName	Pine-nimi	CPE_pine_name
	TypeCircuitName	Pine-tyyppipiiri	CPE_pine_type_circuit
	PineStatus	Pine status	CPE_pine_status
ProcessSubEquipmentCircuit			
Perusavain	EquipmentAlmaID	Objektin ID	Objektin ID
Yksilöivä*	EquipmentPosition	Pine-positio	CEC_pine_position
	EquipmentName	Pine-nimi	CEC_pine_name
Vierasavain	ParentEquipmentAlmaID	Vanhemman ID	Vanhemman ID
Vierasavain	ParentEquipmentPosition	Vanhemman Pine-positio	Vanhemman CPE_pine_position
	ParentEquipmentName	Vanhemman Pine-nimi	Vanhemman CPE_pine_name
AutomationInstrumentCircuit			
Perusavain	AlmaID	Objektn ID	Objektin ID
Yksilöivä*	Position	Pine-positio	CAL_pine_position
	Name	Pine-nimi	CAL_pine_name
	TypeCircuitName	Pine-tyyppipiiri	CAL_pine_type_circuit
	MeasurementRange	Näyttö_ala - Näyttö_ylä Yksikkö	CAL_meas_low - CAL_meas_high CAL_unit
	AlarmLowLimit	Ala	CAL_low
	AlarmHighLimit	Ylä	CAL_high
	PineStatus	Pine status	CAL_pine_status
AutomationInstrument			
Perusavain	AlmaID	Objektin ID	Objektin ID
Yksilöivä*	Position	Pine-positio	CAE_pine_position
	Name	Pine-nimi	CAE_pine_name
Vierasavain	ParentCircuitAlmaID	Vanhemman ID	Vanhemman ID
Vierasavain	ParentCircuitPosition	Vanhemman Pine-positio	Vanhemman CAL_pine_position
	ParentCircuitName	Vanhemman Pine-nimi	Vanhemman CAL_pine_name
StartingGroup			
Perusavain	AlmaID	Objektin ID	Objektin ID
Yksilöivä*	GroupID	Tunnus	ALMA_CODE
	Name	Nimitys 1	ALMA_NAME

* Jos CTS Pine Database Creator for Pinessä on valittu valintaruutu ”Require Unique Positions”, tähän kenttään luettavien arvojen tulee olla yksilöllisiä tai tietokannan luominen epäonnistuu. Jos valintaruutua ei ole rastitettu, ”Yksilöivä” –määre ei ole käytössä.

Taulukko 4 - Alma-väliaikaistietokannan ja Alma-tietokannan attribuuttien vastaavuudet

4.1.3 Toiminta

Sovelluksen käynnistyksen yhteydessä luetaan aluksi käyttäjän määrittelemät asetukset asetustiedostosta, jonka jälkeen tarkistetaan, että kaikki sovelluksen tarvitsemat ulkoiset komponentit (kuten VBScript ja ADOX) ovat käytettävissä. Mikäli jokin komponentti ei ole käytettävissä, sen tarjoama toiminnallisuus joko poistetaan käytöstä, tai jos komponentti on ohjelman toiminnan kannalta kriittinen, lopetetaan ohjelman suoritus. Esimerkiksi mikäli tietokoneelle ei ole asennettu Microsoft Accessia, poistetaan käytöstä valintaruutu, jonka valitsemalla voidaan luotu väliaikaistietokanta avata Accessiin. Koska tämä toiminnallisuus ei ole ohjelman suorituksen kannalta kriittistä, ohjelman suorittamista ei tarvitse lopettaa.

Kun tarvittavat tarkistukset ja alustukset on tehty, Alma-rajapinnalle annetaan käsky käynnistää Alma, jolloin rajapinta käynnistää instanssin Alma-asiakasohjelmasta. Tätä asiakasohjelman instanssia ei saa sulkea Database Creator for Pinen ollessa käytössä, sillä muuten mitkään Alma-rajapinnan tarjoamat toiminnallisuudet eivät enää toimi. Asiakasohjelman käynnistyksen yhteydessä näytölle tulee Alman normaali kirjautumisikkuna, johon syötetään Alma-tiedonhallintajärjestelmän käyttäjätunnus ja salasana. Jos kirjautuminen epäonnistuu, Database Creator for Pine pyytää käyttäjää tarkistamaan Almaan liittyvät asetukset sekä yrittämään uudelleen.

Onnistuneen kirjautumisen jälkeen seuraava vaihe on hakea projektien nimet Almasta, sekä lisätä ne projektinvalintalistaan. Projektien hakeminen tapahtuu Almassa olevaa hakuobjektia hyödyntämällä, ja sitä muokkaamalla voidaan näin ollen muokata myös hakutuloksia. Kun haku on suoritettu ja projektit ovat lisättynä projektinvalintalistaan, jää sovellus odottamaan käyttäjän toimenpiteitä. Käyttäjän tulee nyt valita valintalistasta se projekti, josta hän haluaa luoda väliaikaistietokannan, sekä määrittää tiedostonimi ja sijainti johon väliaikaistietokanta luodaan. Lisäksi valittavana on erilaisia lisäoptioita, joita hyödyntämällä sovellus mm. avaa tiedostoselaimen luodun väliaikaistietokannan hakemistoon onnistuneen luomisen päätteeksi. Muita valittavia lisäoptioita ovat esimerkiksi ”Verbose Mode” -tila, jolloin sovellus raportoi toiminnastaan tarkemmin. Tämä on hyödyllistä etenkin ongelmatilanteita selvittäessä, mutta toisaalta se saattaa myös

hidastaa ohjelman suoritusta tapauksissa, joissa väliaikaistietokantaan lisättäviä objekteja on paljon. Tärkeä yksittäinen lisäoptio on ”Require Unique Positions”, joka määrittää tuleeko väliaikaistietokantaan lisättävien objektien positiotunnusten olla yksilöllisiä. Mikäli tämä valintaruutu ei ole valittuna, väliaikaistietokantaan ja sitä kautta myöhemmin myös Pinen tietokantaan saattaa päätyä kaksi positiota, joilla on sama positiotunnus. Tästä puolestaan aiheutuu myöhemmin ongelmia CTS Pine Revision Manager -sovelluksen kanssa. Tätä ns. duplikaattipositio-ongelmaa on käsitelty myöhemmin CTS Pine Management Utility -sovelluksen yhteydessä. Jotta ongelmalta vältyttäisiin, tulisi ”Require Unique Positions” -valintaruutu pitää aina valittuna.

Kun kaikki tarvittavat parametrit on määritelty, voidaan väliaikaistietokannan luominen aloittaa. Ennen tietokannan luomista sovellus kuitenkin tarkistaa, että syötetty tiedostonimi ja -polku ovat kelvollisia, ja mikäli näin ei ole, pysäyttää se tietokannan luomisen. Tämän jälkeen sovellus luo tietokannan rakenteen eli taulut ja niiden sarakkeet, rajoitukset, suhteet ja indeksit. Rakenteen luomisessa käytetään SQL-kieltä sekä ADO-kirjaston tarjoamia toiminnallisuuksia. Kun rakenne on luotu, jokaisen sarakkeen ominaisuuksiin lisätään vielä tekstimuotoinen kuvaus sarakkeen sisältämästä tiedosta, hyödyntämällä ADOX-kirjaston toiminnallisuuksia. Mikäli tietokannan rakenteen muodostamisessa tapahtuu jokin virhe, tietokannan luominen pysäytetään. Kun rakenne on onnistuneesti luotu, suoritetaan tietojen hakeminen Almasta.

Objektien tietojen hakeminen Almasta tapahtuu tyyppi kerrallaan, eli ensiksi haetaan prosessilaitteet ja niihin liittyvät sähköpiirit, sitten automaatiopositiot ja niihin liittyvät laitepositiot, ja lopuksi käynnistysryhmät. Tietoa ei tallenneta väliaikaistietokantaan suoraan hakemisen yhteydessä, vaan se talletetaan ensin Visual Basicissa määritettyihin tietorakenteisiin. Käytännössä tämä tarkoittaa sitä, että jokaista edellä mainittua objektityyppiä Almassa vastaa luokka Visual Basicissa, ja jokainen objekti Almassa esitetään vastaavan luokan oliona. Oliot puolestaan tallennetaan kokoelmaan, joka sisältää kaikki samantyyppiset oliot. Esimerkiksi prosessilaitekokoelmaan tallennetaan kaikki prosessilaite-oliot. Yksittäinen prosessilaite voi puolestaan sisältää kokoelman sähköpiirejä. Tiedon hakemisessa Almasta hyödynnetään Alma-rajapinnan tarjoamia haku- ja lukufunktioita. Mikäli tiedon hakemisessa tapahtuu jokin virhe, esimerkiksi yhteyden

katkeaminen Alma-tietokantaan, tietokannan luominen keskeytetään ja käyttäjälle annetaan virhettä kuvaava virheilmoitus.

Kun objektit ja niiden tarvittavat attribuutit on haettu, lisätään ne seuraavaksi väliaikaistietokantaan. Tämä tapahtuu käymällä kukin objektikokoelma läpi, ja lisäämällä siihen kuuluvien objektien attribuutit oikeaan tauluun tietokannassa. Jos lisäyksessä ei tapahdu virheitä, on väliaikaistietokanta luotu onnistuneesti, ja se on valmis käytettäväksi Pinen yhteydessä.

Pinestä käsin on myös mahdollista muuttaa joitakin väliaikaistietokannan attribuuttien arvoja, kuten esimerkiksi automaatiopositioiden hälytysrajoja. Mikäli näitä väliaikaistietokannan attribuuttien arvoja muokataan Pinestä käsin, tulisi samat muutokset päivittää myös Alma-tietokantaan, jotta tietokantojen välinen tiedon eheys säilyisi. Muussa tapauksessa tietokannat sisältävät ristiriitaista tietoa, ja usein on hankala sanoa, kummassa tietokannassa oleva tieto on tuoreempaa, voimassa olevaa tietoa. Väliaikaistietokannan tiedot pystytään soveltuvilta osin päivittämään takaisin Almaan Database Creator for Pine -sovellusta käyttämällä. Tietojen päivittäminen väliaikaistietokannasta Almaan aloitetaan syöttämällä sovellukselle olemassa olevan väliaikaistietokannan sijainti. Tämän jälkeen painetaan ”Update Alma from Database” -painiketta, jolloin tiedot päivittyvät Almaan.

4.1.4 Ongelmat

Database Creator for Pine -sovelluksen ongelmat voidaan jakaa kahteen ryhmään, jotka ovat toteutuksenaikaiset ongelmat sekä käytön aikana havaitut ongelmat. Toteutuksenaikaisia ongelmia aiheuttivat mm. Alma-rajapinta ja vaatimusten muuttuminen ohjelman ollessa jo lähes valmis. Suurin käytön aikana havaittu ongelma oli puolestaan eri tietokantojen pitäminen synkronissa toistensa kanssa.

Nykyisessä sovelluksessa käyttäjän on kirjauduttava sisään Almaan käynnistyksen yhteydessä. Alun perin tarkoituksena oli, ettei käyttäjän tarvitsisi hoitaa tätä sisäänkirjautumista itse, vaan että se tapahtuisi käyttäjälle näkymättömästi taustalla. Tässä

ei kuitenkaan onnistuttu, koska sovelluksen toteutushetkellä Alma-rajapinta ei vielä tarjonnut toiminnallisuutta, joka mahdollistaisi tämän ns. hiljaisen sisäänkirjautumisen eli kirjautumisen ilman käyttäjän interaktiota. Toinen toteutuksenaikainen ongelma aiheutui Alman käyttöoikeuksista. Alussa kaikilla suunnittelijoilla oli oikeudet kaikkien Alma-projektien tarkasteluun, mutta myöhemmin tilanne muuttui salassapitovelvoitteiden vuoksi siten, että osa projekteista näkyy vain tietylle käyttäjäryhmälle. Näitä käyttöoikeusrajoitteita ei puolestaan oltu huomioitu Database Creator for Pine -sovelluksen toteutuksessa lainkaan, mistä seurasi mm. virheilmoituksia sovelluksen käynnistyksessä tapahtuvan projektihaun yhteydessä. Lisäksi Alma-rajapinnan tarjoama toiminnallisuus käyttöoikeuksien tarkistamiseen on toistaiseksi puutteellista (Raudasoja, 2009b). Esimerkiksi rajapinnan tarjoamat hakufunktiot palauttavat myös sellaiset objektit, joihin käyttäjällä ei ole käyttöoikeuksia. Lisäksi rajapinta ei tarjoa sellaista toiminnallisuutta, jonka avulla pystyttäisiin yksikäsitteisesti tarkistamaan, onko käyttäjällä käyttöoikeus tiettyyn objektiin vai ei. Database Creator for Pineä muokattiin myöhemmin siten, että se tukee Almassa olevia käyttöoikeuksia, mutta tämä käyttöoikeuksien tarkistus pohjautuu monikäsitteiseen ”Object is static.” -virheilmoitukseen, joka saattaa kaiken lisäksi muuttua Alma-rajapinnan tulevissa versioissa.

Sovelluksen käyttöön liittyvä ongelma on puolestaan tiedon eheyden säilyttäminen eri tietokantojen välillä. Pineen suorasti tai epäsuorasti liittyviä tietokantoja on käytännössä neljä: Alma-tietokanta, Alma-väliaikaistietokanta, Pine-tietokanta sekä revisiomanagerin tietokanta. Pine-projektitietokanta on jätetty tarkoituksella huomioimatta, sillä sen tarkoituksena on ainoastaan sisältää näkymät Alma-väliaikaistietokantaan. Oletetaan esimerkiksi tilanne, jossa suunnittelija A päivittää Almassa jonkin prosessilaitteen positiotunnuksen, ja haluaa sen näkyviin myös Pineen. Samanaikaisesti suunnittelija B päivittää jonkin piirin hälytysrajatiedot Pinen kautta väliaikaistietokantaan. Jotta suunnittelija A saisi tekemänsä muutoksen näkyviin Pinessä, hänen täytyy luoda Alma-väliaikaistietokanta uudelleen, sillä Database Creator for Pine ei mahdollista olemassa olevan väliaikaistietokannan päivittämistä Almasta. Jos suunnittelija B ei ole ehtinyt siirtää tekemiään muutoksia Almaan ennen kuin suunnittelija A korvaa vanhan väliaikaistietokannan uudella, nämä muutokset häviävät. Tähän ongelmaan ratkaisuna oli

työtapojen muuttaminen siten, että Pinestä ei koskaan päivitetä mitään tietoja väliaikaistietokantaan, vaan kaikki muutokset tehdään suoraan Alma-tietokantaan.

Ongelmia aiheuttaa myös se, että Pinen tietokanta ei päivity vastaamaan Alma-väliaikaistietokannassa olevaa tietoa ilman, että kyseiselle projektille suoritetaan revisioajo revisiomanagerilla optio ”Update position, name and measurement range values from Intecro3 to CTSPine” valittuna. Tällöin Pinen tietokannasta poistuvat myös sellaisiin piireihin liittyvät toimintakuvausdokumentit, joita vastaavia piirejä ei löydy väliaikaistietokannasta. Poistaminen tapahtuu lisäksi ilman varoitusta. Mikäli väliaikaistietokantaa päivitetään usein, riski Pine-tietokannassa olevien toimintakuvausdokumenttien häviämiseen kasvaa. Riskialtis tilanne saattaa syntyä esimerkiksi siitä, että Alma-tietokantaan syötetään projektin alussa useita automaatiopositioita, joiden tarkemmat tiedot jätetään aluksi täyttämättä positiotunnusta lukuun ottamatta. Suunnittelija A luo näistä väliaikaistietokannan, ja käy työstämään piirikohtaisia toimintakuvauksia Pinessä. Suunnittelija B käy puolestaan täyttämään automaatiopositioiden tarkempia tietoja Almaan, ja huomaa että vastaavat tiedot sisältävä automaatiopositio on jo käytössä toisessa projektissa. Vaivan säästämiseksi hän saattaa poistaa alkuperäisen automaatioposition, ja kopioida tilalle valmiin automaatioposition toisesta projektista, muokaten sen tietoja sopivilta osilta. Tällöin automaatioposition Alma-ID vaihtuu. Kun Almasta nyt luodaan väliaikaistietokanta Pinen käytettäväksi, ja suoritetaan revisiomanageriajo Pine-tietokannan päivittämiseksi, poistuu suunnittelijan A luoma piirikuvausdokumentti Pine-tietokannasta, koska vastaavalla Alma-ID:llä olevaa automaatiopositiota ei enää löydy väliaikaistietokannasta. Vaikka tämä esimerkki onkin osaltaan keinotekoinen, eikä se välttämättä vastaa käytössä olevia työskentelytapoja, saa siitä kuitenkin kuvan, miksi väliaikaistietokannan jatkuva päivittäminen useiden suunnittelijoiden toimesta voi olla riskialtista. Tälläisten ongelmien ilmenemistä lisää se, että Pinen sisäinen rakenne ja toiminta eivät välttämättä ole selvillä kaikilla sen käyttäjillä, eivätkä väliaikaistietokannan päivityksen tai revisiomanageriajon (tietyillä parametreillä) suorittamisen seuraukset ole näin ollen kaikilta osin kaikkien tiedossa. Ongelmaan ei myöskään ole helppoa ratkaisua, vaan sen ratkaiseminen vaatisi joko suunnittelijoiden välisen koordinaation lisäämistä kohtuuttomasti, tai kokonaan uuden sovelluksen kehittämistä piirikuvausdokumenttien luomiseen.

4.2 CTS Pine Management Utility

CTS Pine -ohjelmiston toimintaan liittyi erinäisiä ongelmia jo ennen tiedonhallintajärjestelmän vaihtamista Intecrosta Almaan. Uuden Pine-projektin perustaminen miellettiin hankalaksi, ja siinä joudutaankin yleensä turvautumaan asiantuntija-apuun. Projektin perustaminen vaatii mm. uuden tietokannan luomista SQL-palvelimelle, Intecro- tai Alma-väliaikaistietokannan linkittämistä tähän luotuun tietokantaan sekä näkymien luomista. Tavallisella suunnittelijalla ei ole oikeuksia SQL-palvelimelle, joten operaation suorittaminen ei ole mahdollista ilman ATK-tuen avustusta. Myös ATK-tuelle uuden projektin luominen on työläs ja tarkkuutta vaativa toimenpide.

Toinen Pineen liittyvä ongelma on sen tietokantaan tietyissä tilanteissa syntyvät ns. duplikaattipositiot. Duplikaattipositiolla tarkoitetaan sellaista piirin positiotunnusta, joka esiintyy yhden projektin sisällä vähintään kahteen kertaan Pinen tietokannassa. Toisin sanoen tämä tarkoittaa sitä, että Pinen tietokannassa on vähintään kaksi sellaista piiriä, joilla on sama positiotunnus. Vaikka jokaisen piirin positiotunnuksen tulisikin olla projektin sisällä yksilöllinen, työtavoista johtuen sekä Alma- että Intecro-tietokantoihin päätyy silloin tällöin piirejä, joilla on sama positiotunnus. Intecro- ja Alma-väliaikaistietokannoista nämä piirit päätyvät lopulta myös Pinen tietokantaan. Pinen tietokannassa jokainen piiri yksilöidään sen tietokanta-ID:n mukaan (Intecro- tai Alma-ID), eikä positiotunnuksen mukaan.

Duplikaattipositioiden aiheuttama ongelma tulee esiin käytettäessä CTS Pine Revision Manageria. Revisiomanageri ottaa syötteekseen listan positioista, joista revisioajo halutaan suorittaa. Tämä lista on tekstitiedosto, johon on syötetty ajoon sisällytettävien positioiden positiotunnukset, positiot yksilöivien tietokanta-ID:itten sijaan. Koska positiotunnus ei ole yksilöllinen, ja useammalla positiolla voi näin ollen olla sama positiotunnus, revisiomanageri ei voi tietää mitä näistä positioista käyttäjä tarkoittaa. Tilanteessa jossa Pine-tietokannassa on useampi samalla positiotunnuksella varustettu positio, revisiomanageri valitsee niistä määrätyllä logiikalla yhden, josta se tuottaa toimintakuvausdokumentin. Mikään ei kuitenkaan takaa sitä, että tämä revisiomanagerin valitsema positio on se, mitä käyttäjä on tarkoittanut.

Duplikaattipositioiden kulkeutuminen Alma-väliaikaistietokantaan ja sitä kautta Pinen tietokantaan voidaan osittain estää käyttämällä Database Creator for Pinen ”Require Unique Positions” -optiota väliaikaistietokannan luomisessa. Tällöin väliaikaistietokannan luominen keskeytyy, mikäli sinne yritetään tallentaa kaksi samantyyppistä objektia, joilla on sama positiotunnus. Tämä ei ole kuitenkaan täysin varma tapa varmistaa, ettei duplikaatiopositioita päädy Pinen tietokantaan. Esimerkiksi jos automaatiopositiolla ja prosessilaitteella on jostain syystä sama positiotunnus, väliaikaistietokannan luonti onnistuu, koska kyseessä olevat objektit ovat erityyppisiä.

CTS Pine Management Utility on sovellus, jonka tarkoituksena on helpottaa uuden Pine-projektin perustamista sekä auttaa duplikaattipositioiden paikantamisessa ja poistamisessa Pinen tietokannasta. Lisäksi sovellus helpottaa olemassa olevien Pine-projektien poistamista. Sen toteutuksen yhteydessä jouduttiin perehtymään tarkemmin Pinen ja Pine Revision Managerin toimintaan sekä vastaaviin tietokantoihin, mistä oli myöhemmin hyötyä diplomityön kokonaisuutta hahmottaessa.

4.2.1 Toteutus

CTS Pine Management Utility on toteutettu käyttämällä kokeiluluontoisesti Microsoft Visual Basic 2008 Express Edition -sovelluskehitintä vanhentuneen Visual Basic 6.0:n sijasta. Express Edition on saatavilla ilmaiseksi, ja se on suunnattu lähinnä opiskelijoille, harrastelijoille ja aloitteleville sovelluskehittäjille. Siitä puuttuu joitakin maksullisien versioiden tukemia ominaisuuksia, joista tämän projektin puitteissa jäätiin kaipaamaan ainoastaan monipuolisempia julkaisutyökaluja. Express Editionin lisenssi ei rajoita sillä tehtyjen ohjelmien kaupallista käyttöä (Microsoft, 2010).

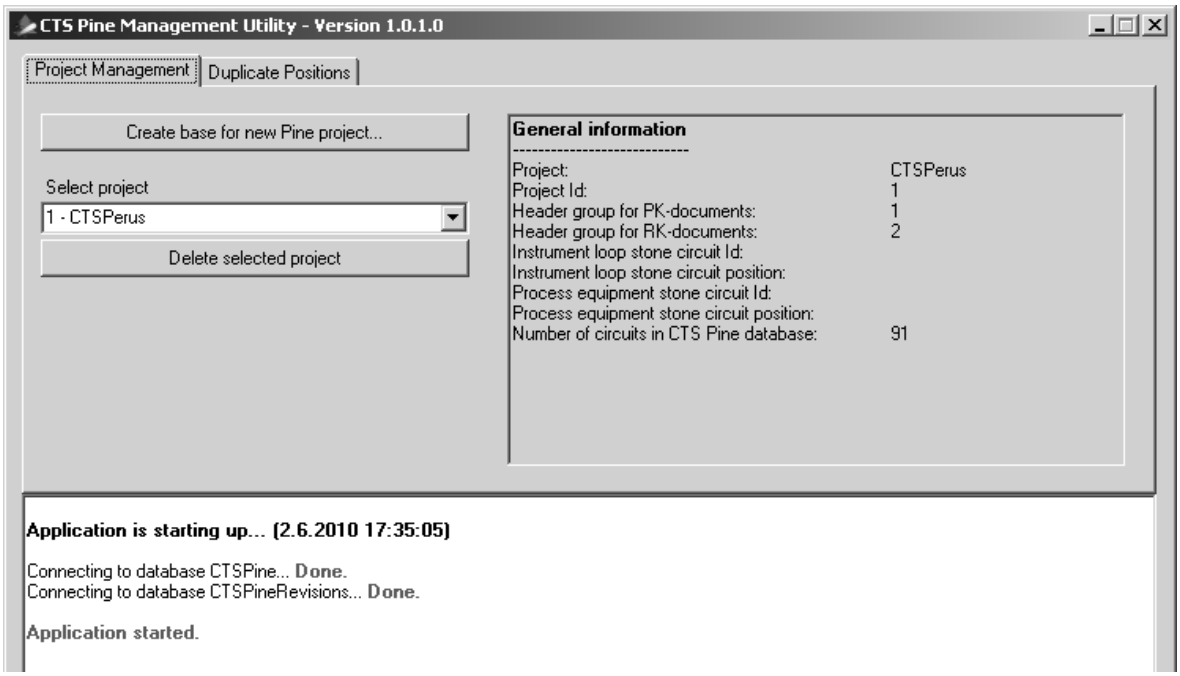
Pine Management Utility vaatii toimiakseen Microsoftin .NET Framework 3.5 SP1 -komponenttikirjaston asentamisen kullekin työasemalle, jossa sovellusta halutaan käyttää. Mikäli tätä komponenttikirjastoa ei ole valmiiksi asennettuna, sovelluksen asennusohjelma tarjoutuu asentamaan sen. NET Framerwork -komponenttikirjasto tarjoaa kaiken sovelluksessa tarvittavan toiminnallisuuden, kuten esimerkiksi SQL-tietokantaoperaatiot.

Sovellus on julkaistu Visual Basic Express Editionin ”ClickOnce” -julkaisutoimintoa käyttäen, mikä mahdollistaa mm. sovelluksen automaattisen päivittämisen kullekin työasemalle aina uuden version ilmestyessä.

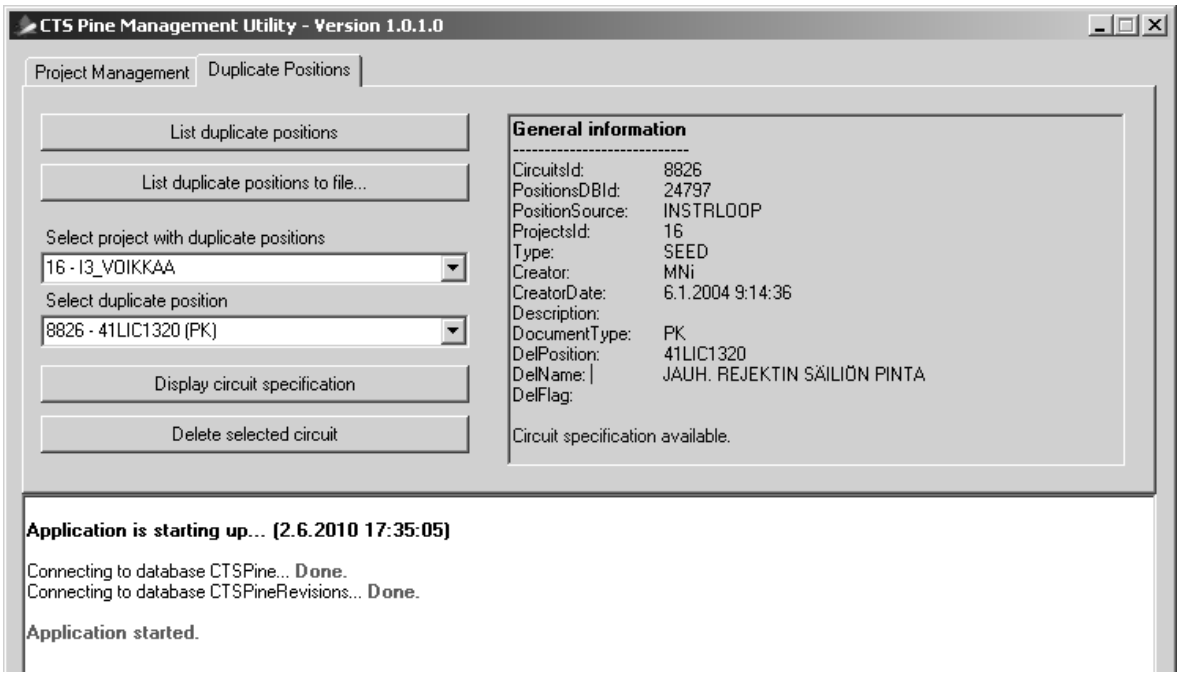
4.2.2 Käyttöliittymä ja rakenne

Sovelluksen käyttöliittymä koostuu yhdestä päälomakkeesta sekä neljästä tilanteen mukaan aukeutuvasta dialogista. Toiminnat on jaoteltu päälomakkeella kahteen välilehteen, jotka ovat nimiltään ”Project Management” ja ”Duplicate Positions”. Näistä ”Project Management” -välilehdellä ovat uuden projektin perustamiseen sekä vanhojen projektien poistamiseen liittyvät toiminnot. Välilehden ulkoasu on esitetty kuvassa 16. ”Duplicate Positions” -välilehdellä sijaitsee puolestaan duplikaattipositioiden listaamiseen, tarkastelemiseen ja poistamiseen liittyvät toiminnot. Tämän välilehden ulkoasu on esitetty puolestaan kuvassa 17. Päälomakkeen alareunassa on tilaikkuna, johon ilmestyy tietoa sovelluksen suorittamista toimista. Oikeassa yläreunassa on infoikkuna, josta näkee valittuun projektiin tai positioon liittyviä tietoja aktiivisesta välilehdestä riippuen. Uuden projektin pohjan luomista varten avautuu ”Create base for new Pine project…” -painikkeella kuvassa 18 esitetty ikkuna, jossa voidaan määritellä projektiin liittyviä tietoja, kuten nimi, käytettävät otsikkoryhmät, tyylisivut sekä revisioajojen hakemisto.

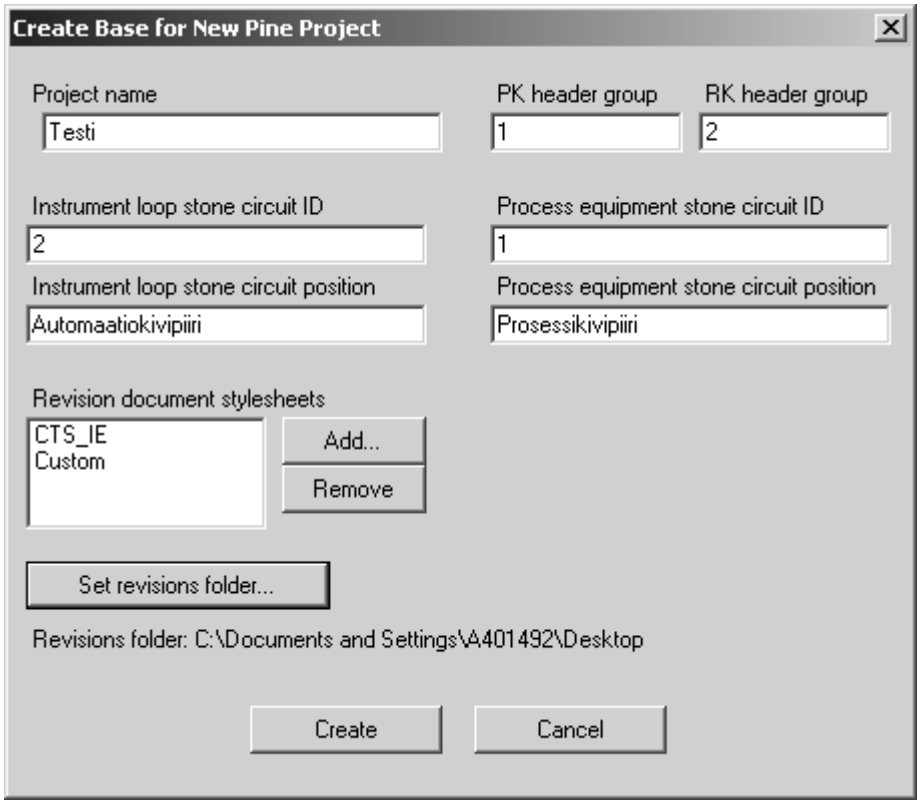
Kuten Database Creator for Pinessä, myös Pine Management Utilityssä osa ohjelmakoodista sijaitsee suoraan lomaketiedostoissa, mutta suurimpia toimintokokonaisuuksia varten on luotu erilliset moduulit, joiden avulla ohjelman rakennetta pyritään selkeyttämään. Näitä moduuleja on kolme, joista ensimmäinen sisältää projektien hallintaan liittyviä toimintoja, kuten uuden projektin luomisen ja vanhojen projektien poistamisen. Toinen moduuleista sisältää puolestaan duplikaattipositioihin liittyvän toiminnallisuuden. Tällaisia toimintoja ovat mm. duplikaattipositioiden paikantaminen tietokannasta, niiden listaaminen sekä niiden poistaminen. Kolmas moduuleista sisältää sekalaisia funktioita, kuten esimerkiksi tietokantaan yhdistämisen suorittavan funktion.



Kuva 16 - CTS Pine Management Utility -sovelluksen päälomake: Project Management -välilehti



Kuva 17 - CTS Pine Management Utility -sovelluksen päälomake: Duplicate Positions -välilehti



Kuva 18 - Uuden projektin luomisessa käytetty ikkuna

Sovelluksessa käytetään luokkia tietokannasta haetun tiedon varastoimiseen. Esimerkiksi projekteja varten sovelluksessa on projektiluokka, jonka jäsenmuuttujia ovat mm. projektin nimi, ID sekä lista projektiin liittyvistä duplikaattipositioista. Jokainen tietokannasta haettu projekti tallennetaan tällaisen projektiluokan ilmentymään eli olioon. Samoin jokainen tietokannasta haettu positio muodostaa positioluokan ilmentymän. Muita käytettyjä luokkia ovat mm. piirikuvausdokumenttien otsikoita kuvaava otsikko-luokka ja yksittäistä tekstimerkintää kuvaava tekstimerkintä-luokka. Kukin otsikkoluokan olio sisältää listan tekstimerkintä-olioita, eli toisin sanoen kunkin otsikon alla voi olla useampi tekstimerkintä. Lisäksi kullakin otsikko-oliolla on järjestysnumero, jonka avulla otsikot ja niihin liittyvät tekstit pystytään tarvittaessa tulostamaan oikeassa järjestyksessä. Pine Management Utility mahdollistaa näin ollen duplikaattipositioihin mahdollisesti liittyvien toimintakuvausten tulostamisen, joista on hyötyä päätellessä, mikä tai mitkä samannimisistä positioista ovat ylimääräisiä.

CTS:llä toteutetuissa Visual Basic -sovelluksissa on perinteisesti käytetty ns. ini-tiedostoja asetusten tallentamiseen. Pine Management Utility:ssä käytetään kuitenkin Visual Basic

Express Editionin tarjoamaa vaihtoehtoista ratkaisua asetusten tallentamiseen. Visual Basic -projektin asetuksissa voidaan määrittää asetusmuuttujia, joihin voidaan tallentaa, ja joista voidaan lukea tietoa. Nämä asetusmuuttujat arvoineen tallennetaan automaattisesti käyttäjäkohtaiseen XML-pohjaiseen tiedostoon nimeltään ”user.config”. Esimerkki sovelluksessa käytettävästä asetusmuuttujasta on ”WorkingDirectory”, jota käytetään hyväksi toiminnossa, jonka avulla duplikaattipositioiden tiedot voi listata tekstitiedostoon. Tähän asetusmuuttujaan tallentuu viimeisin käyttäjän määrittelemä hakemisto näiden tekstitiedostojen tallentamiseen, jolloin sovellus osaa asettaa tämän hakemiston oletukseksi myös seuraavalla kerralla, kun duplikaattipositioiden listaustoimintoa tiedostoon käytetään.

4.2.3 Toiminta

Käynnistyessään sovellus muodostaa aluksi yhteydet sekä Pinen että Pine-revisiomanagerin tietokantoihin. Mikäli yhdistäminen tietokantoihin epäonnistuu, sovellus antaa asiasta virheilmoituksen ja sulkeutuu. Kun tietokantayhteydet on muodostettu, sovellus hakee projektien tiedot Pinen tietokannasta ja lisää projektien nimet projektinvalintalistaan. Tämän jälkeen sovellus siirtyy käyttäjän ohjaukseen.

4.2.3.1 Uuden projektin luominen

Uusi Pine-projekti voidaan luoda käyttämällä ”Create base for new Pine project…” -painikkeesta avautuvaa toimintoa. Tämän toiminnon avulla voidaan lisätä uuden projektin tarpeelliset tiedot Pinen tietokantaan ilman, että niitä tarvitsee käydä syöttämässä SQL Serverille käsin. Tarvittavat tiedot syötetään Database Management Utilityn avaamaan lomakkeeseen. Kun tiedot on syötetty, voidaan uuden projektin luominen käynnistää lomakkeen alareunassa olevaa ”Create” -painiketta käyttämällä. Mikäli joihinkin kenttiin on syötetty epäkelpoja arvoja, sovellus antaa virheilmoituksen, jossa kerrotaan mitkä arvot tulee korjata. Tämän jälkeen sovellus suorittaa seuraavat toimenpiteet:

1. Etsitään ensimmäinen vapaa projekti-ID:n Pine-tietokannan taulusta ”ILockGroupSets”.

2. Lisätään projektin ID, nimi, otsikkoryhmien ID:t sekä kivipiirien ID:t ja positiotunnukset Pine-tietokannan tauluun ”ILockGroupSets”.
3. Lisätään revisionhallinnan hakemisto revisiomanagerin tietokannan tauluun ”RevisionsFolder”
4. Lisätään dokumenttien generoinnissa tarvittavat tyylisivut revisiomanagerin tietokannan tauluun ”XSLFiles”
5. Kopioidaan tyylisivumääritykset projektille määritettyyn revisionhallinnan hakemistoon, mikäli mahdollista.

Kun nämä toimenpiteet on saatu onnistuneesti suoritettua, on pohja uudelle Pine-projektille luotu. Pine-projektin perustamisessa on kuitenkin myös vaiheita, joita tämän ohjelman avulla ei kyetä suorittamaan, kuten esimerkiksi projektin lisääminen Pinen aloitussivulle. Nämä vaiheet on edelleen suoritettava manuaalisesti.

4.2.3.2 Vanhan projektin poistaminen

Olemassa olevan Pine-projektin poistaminen tapahtuu valitsemalla haluttu projekti projektilistauksesta ja napauttamalla ”Delete selected project” -painiketta. Tästä aukeaa varmistusikkuna, jossa käyttäjää vaaditaan syöttämään projektin nimi ikkunassa olevaan tekstikenttään. Näin varmistutaan siitä, ettei projektia poisteta vahingossa. Jos käyttäjä syöttää projektin nimen oikein, suorittaa sovellus projektiin liittyvien tietojen poistamisen sekä Pinen että revisiomanagerin tietokannoista. Projektin poistaminen etenee seuraavasti (kaikki tietokantaoperaatiot kohdistuvat oletusarvoisesti revisiomanagerin tietokantaan, ellei toisin ole mainittu):

1. Poistetaan kaikki valittuun projektiin liittyvät revisioajot taulusta ”RunSets”. Samalla kuhunkin revisioajoon liittyvät piirit sekä kaikki näihin piireihin liittyvä muu tieto poistuu myös ”Circuits”-taulusta.
2. Poistetaan tietokannasta kaikki ne projektiin kuuluvat piirit sekä niihin liittyvä muu tieto, jotka eivät ole poistuneet kohdan 1 yhteydessä.

3. Poistetaan taulusta ”DocumentRevisionHistory” sellaiset projektiin liittyvät tiedot, jotka eivät ole poistuneet kohdan 1 yhteydessä.
4. Poistetaan projektiin liittyvät tiedot taulusta ”RevisionTemp_Circuits”.
5. Poistetaan viittaus projektin revisionhallinnan hakemistoon taulusta ”RevisionsFolder”.
6. Poistetaan projektiin määritetyt tyylisivut taulusta ”XSLFiles”.
7. Poistetaan projektiin liittyvät tiedot taulusta ”ProjectsRevisionMarks”.
8. Poistetaan projektiin liittyvät, revisiomanagerissa käytössä olevat ns. projektisetit Pine-tietokannan taulusta ”SelectionSets”.
9. Poistetaan kaikki projektiin liittyvät piirit sekä kaikki näihin piireihin liittyvä muu tieto Pine-tietokannan taulusta ”Circuits”.
10. Poistetaan projektin tiedot Pine-tietokannan taulusta ”ILockGroupSets”.

Mikäli poistettava projekti sisältää runsaasti piirejä ja revisioajoja, saattaa sen poistamisessa kestää muutamia minuutteja. Mikäli projektin poistaminen keskeytyy esimerkiksi tietokoneen kaatumiseen, tulee projektin poistaminen suorittaa uudestaan, ettei tietokantaan jää ”leijumaan” turhaa tietoa. Projekti poistuu projektinvalintalistauksesta vasta sitten, kun edellisessä listauksessa esitetty kohta 10 on onnistuneesti suoritettu.

4.2.3.3 Duplikaattipositioiden tarkasteleminen ja poistaminen

Eri projektien sisältämiä duplikaattipositioita voidaan tarkistella ja poistaa päälomakkeen ”Duplicate Positions” -välilehdeltä löytyvien toimintojen avulla. Kun ”Duplicate Positions” -välilehteä napsautetaan, sovellus käy läpi kaikki Pine-tietokannassa olevat projektit sekä lisää projektilistaukseen sellaiset projektit, josta se löytää duplikaattipositioita. Duplikaattipositioiden etsintä tapahtuu SQL:n Count-funktiota hyödyntämällä. Esimerkiksi seuraava SQL-lause palauttaa taulun, joka sisältää ne positiotunnukset, jotka esiintyvät projektissa, jonka ID on 39, useammin kuin kerran:

SELECT DelPosition, COUNT(DelPosition) AS NumOccurrences FROM Circuits WHERE ProjectsId = 39 AND DocumentType = 'PK' GROUP BY DelPosition HAVING (COUNT(DELPosition) > 1)

Jokaisesta löydetystä duplikaattipositiosta kopioidaan perustiedot sekä mahdollinen toimintakuvausdokumentti sovelluksessa käytettäviin tietorakenteisiin myöhempää käyttöä varten. Kun kaikki duplikaattipositioita sisältävät projektit on etsitty ja lisätty projektin valintalistaan, asetetaan sovellus jälleen käyttäjän ohjaukseen. Käyttäjä voi nyt selailla duplikaattipositioita sisältäviä projekteja sekä itse löydettyjä duplikaattipositioita. Valitun duplikaattiposition perustiedot esitetään sovelluksen oikean yläreunan infoikkunassa. Jos valitusta duplikaattipositiosta on olemassa toimintakuvaus, se voidaan esittää sovelluksen alareunan tilaikkunassa painiketta ”Display circuit specification” napsauttamalla. Annettuja tietoja hyväksikäyttämällä käyttäjä voi yrittää päätellä, mikä tai mitkä listatuista positioista ovat turhia, ja poistaa ne. Lisäksi sovelluksessa on ominaisuus, jonka avulla duplikaattipositioita sisältävät projektit voidaan listata käyttäjän antamaan tekstitiedostoon.

4.2.4 Ongelmat

Toisin kuin Database Creator for Pinen kohdalla, Pine Management Utilityn toteutus sujui ilman mainittavia ongelmia. Ainoastaan Pinen sekä Pine-revisiomanagerin tietokantojen rakenteen ja toiminnan selvittäminen oli haasteellista, sillä niistä, kuten itse Pine-ohjelmistostakaan, ei ollut saatavilla kunnollista dokumentaatiota. Eräs Pine Management Utilityn ongelmista on, ettei se ole järin tietoturvallinen. Uuden projektin luomisen yhteydessä annettuja syötteitä ei tarkasteta, mikä mahdollistaa ns. SQL-injektiohyökkäyksen, jossa hyökkääjä pystyy suorittamaan tietokantapalvelimella sellaisia komentoja, joita hänen ei pitäisi pystyä suorittamaan. SQL-injektion estävien tarkistusten toteuttamisen katsottiin kuitenkin vaativan liikaa aikaa ja vaivaa saavutettuun hyötyyn nähden. Sovellus ja sen käsittelemät tietokannat ovat käytettävissä ainoastaan yrityksen sisäverkossa, ja mikäli tietojärjestelmään murtautuja pääsee käsiksi yrityksen sisäverkkoon, on tietokantojen väärinkäyttämiseen helpompiakin tapoja kuin Pine Management Utilityn käyttö.

5 PMMATE

PMMATE on CTS:llä kehitetty ohjelma putkimateriaalien hallintaan. Se mahdollistaa projektikohtaisen putkimateriaalistandardien määrittämisen (Hasu, 2010). Lisäksi sen avulla voidaan tuottaa erilaisia määräluetteloja, kuten linja- ja materiaalikohtaiset putkimateriaaliluettelot. Näitä määräluetteloja voidaan hyödyntää esimerkiksi liitteenä tarjouspyynnöissä sekä asennustyön suorittamisessa. Sovellus ja sen käyttämä tietokanta on toteutettu tietokantaohjelmistojen toteutukseen tarkoitetulla Microsoft Visual FoxPro:lla. PMMATEn tietokanta sisältää kirjoitushetkellä yli 25 000 erilaista putkimateriaalikomponenttia, joista projektissa käytettävä materiaali on valittavissa. Eri komponenttiryhmiin kuuluvia materiaaleja voidaan hakea sovelluksesta esim. mitta- ja materiaalistandardeja apuna käyttäen. Toiminnassaan PMMATE hyödyntää prosessiosaston projektitietokantaan tuottamaa putkilinjatietoa.

5.1 CTS XML Creator for PMMATE

Ennen tiedonhallintajärjestelmän muutosta, PMMATE-sovellus luki projektiin liittyvät putkilinjatiedot suoraan projektin Intecro-tietokannasta. Tiedonhallintajärjestelmän vaihtuessa Intecrosta Almaan, tietojen lukeminen suoraan projektitietokannasta ei ollut enää mahdollista ilman muutoksia PMMATE-ohjelmistoon. Ohjelmiston ylläpitäjä koki, että helpoin tapa tiedon siirtämiseksi Almasta PMMATE:een on luoda XML-muotoisia -tiedostoja Alman putkilinjatiedoista, sekä niiden yläpuolisesta objektihierarkiasta. CTS XML Creator for PMMATE on sovellus näiden XML-tiedostojen luomiseen.

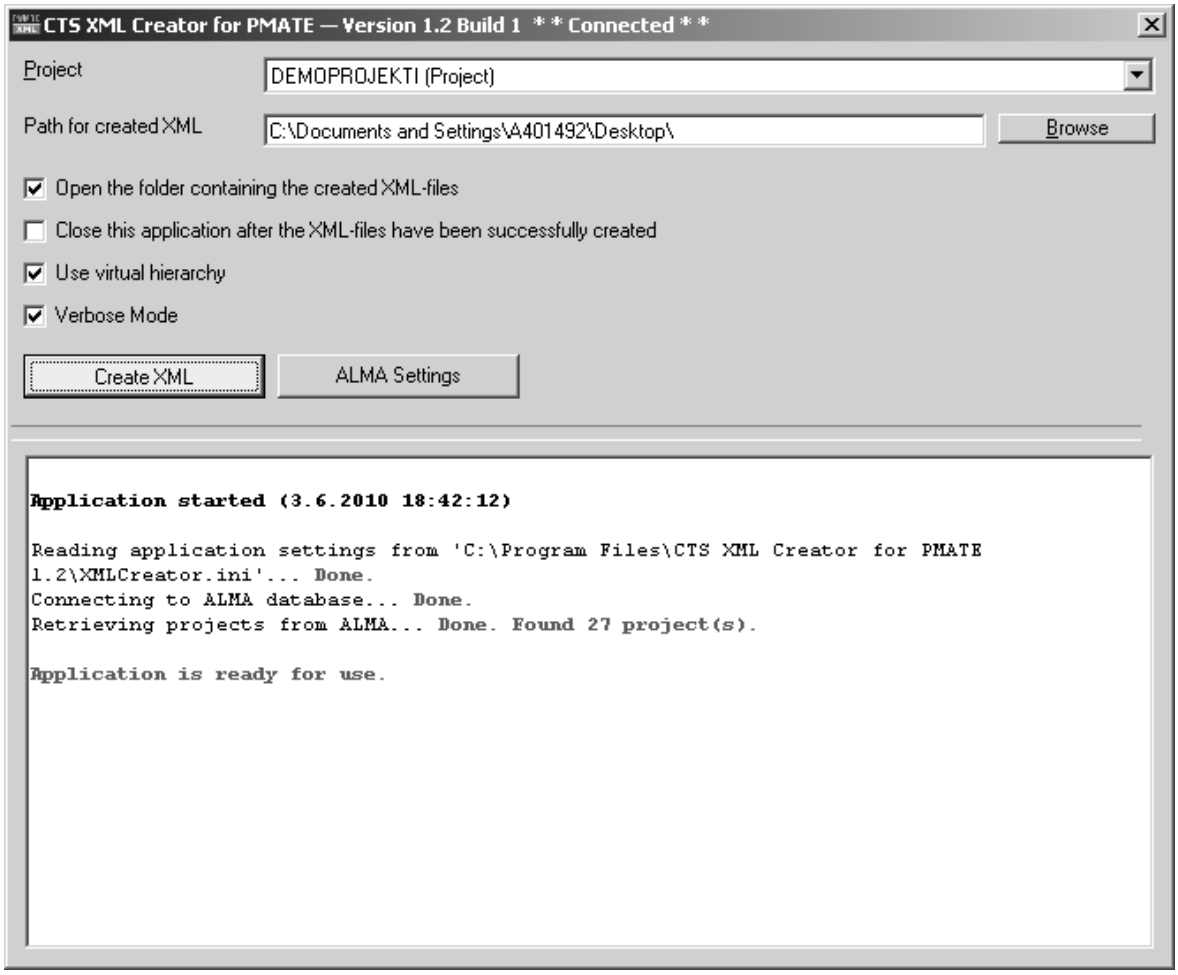
5.1.1 Toteutus

Kuten aikaisemmin kuvattu Database Creator for Pine -sovellus, myös XML Creator for PMMATE on toteutettu Microsoft Visual Basic 6.0 -sovelluskehittimellä. Nämä sovellukset sisältävät paljon yhteistä toiminnallisuutta, kuten esimerkiksi projektien hakemisen Almasta. Suuri osa Database Creator for Pinen koodista onkin käytetty

uudelleen XML Creator for PMMATE:ssa. XML Creator for PMMATE kommunikoi Alman kanssa käyttäen Alman ActiveX-rajapinnan tarjoamia toiminnallisuuksia. Vaikka sovelluksessa luotavien XML-dokumenttien rakenne onkin suhteellisen yksinkertainen, ja se olisi ollut yksinkertainen luoda myös ilman aputyökaluja, on sovelluksessa käytetty Microsoftin tarjoamaa XML-kirjastoa XML-dokumenttien rakenteen luomiseen. Lisäksi sovellus käyttää hyödyksi Microsoftin VBScript Regular Expressions -kirjastoa mm. tiedostonimien kelvollisuuden tarkistamiseen. Kuten muutkin tämän diplomityön yhteydessä toteutetut sovellukset, XML Creator for PMMATE on toteutettu englanninkielisenä, jotta se olisi mahdollisimman laajan käyttäjäryhmän käytettävissä. Sovelluksesta on lisäksi tuotettu kattava käyttö-opas, jonka tarkoituksena on selittää sovelluksen tarjoamat toiminnot mahdollisimman kattavasti, sekä opastaa sovelluksen käytössä.

5.1.2 Käyttöliittymä ja rakenne

Sovelluksen käyttöliittymä on pitkälti yhtenäinen Database Creator for Pinen kanssa. Se koostuu kahdesta lomakkeesta, jotka ovat päälomake ja asetuslomake. Sovelluksen päälomakkeen ulkoasu on esitetty kuvassa 19. Alma-tietokannan projekti, josta XML-tiedostot luodaan, valitaan sovelluksen yläreunassa olevasta alasvetovalikosta. Tämän alasvetovalikon alapuolella on tekstikenttä, johon syötetään hakemisto, mihin XML-tiedostot luodaan. Hakemiston voi määrittää joko manuaalisesti tai käyttämällä ”Browse”-painikkeesta aukeutuvaa dialogia. Hakemisto-tekstikentän alapuolella sijaitsevat 4 valintaruutua, joiden avulla voidaan määritellä joitakin lisä-asetuksia, kuten virtuaalisen hierarkian käyttäminen XML:ää luotaessa. Virtuaalinen hierarkia on selitetty tarkemmin kappaleessa 5.1.3. Sovelluksen alareunassa on tilaikkuna, jossa sovellus raportoi suoritetuista toimenpiteistä. ”ALMA Settings”-painikkeesta avautuva asetusvalikko sisältää täsmälleen samat ominaisuudet kuin Database Creator for Pinen vastaava valikko. Tämän asetusvalikon ulkoasu ja toiminnot on esitelty Database Creator for Pinen yhteydessä, kappaleessa 4.1.2.



Kuva 19 - CTS XML Creator for PMMATE -sovelluksen päälomake

Käyttöliittymän lomakkeiden lisäksi sovellus koostuu kolmesta moduulista, jotka sisältävät erilaisia tukitoimintoja. Moduuliin ”FileOperations” on sijoitettu tiedostojen käsittelyyn liittyvä toiminnallisuus, kuten asetustiedostosta lukeminen ja sinne kirjoittaminen. Käyttöliittymään syötettävän tiedostopolun kelvollisuuden tarkistuksessa käytettävät toiminnot ovat puolestaan sijoitettuna ”FileValidation” -moduuliin. Viimeinen moduuli on nimeltään ”Misc”, ja se sisältää mm. sovelluksessa käytettyjen julkisten muuttujien ja vakioiden määrittelyt.

Sovelluksen asetukset on tallennettu sovelluksen asennushakemistosta löytyvään tiedostoon ”XMLCreator.ini”. Tätä tiedostoa ei tarvitse muokata manuaalisesti, vaan kaikki tarvittavat muutokset pystytään tekemään myös itse ohjelmasta käsin. Asetuksiin tallentuu mm. päälomakkeen valintaruutujen tila sekä viimeksi käytetty hakemisto, johon XML-tiedostot on luotu.

5.1.3 Toiminta

Sovelluksen käynnistyessä tarkastetaan aluksi, että järjestelmästä löytyvät kaikki sen tarvitsemat kirjastot (kuten MSXML ja VBScript Regular Expressions). Mikäli jotakin tarvittavaa kirjastoa ei löydy, sovellus antaa käyttäjälle virheilmoituksen ja sulkeutuu. Käynnistyksen yhteydessä luetaan myös käyttäjän määrittelemät asetukset tiedostosta ”XMLCreator.ini”, sekä asetetaan sovellus niitä vastaavaan tilaan. Kun asetukset on luettu, on vuorossa yhteyden muodostaminen Alma-tietokantaan, joka tapahtuu samalla tavalla kuin Database Creator for Pinessä. Alma-rajapinta käynnistää instanssin Alma-asiakasohjelmasta, jota ei saa sulkea tai rajapinnan tarjoamat toiminnot lakkaavat toimimasta. Kun yhteys Alma-tietokantaan on onnistuneesti muodostettu, haetaan tietokannasta käytettävissä olevien projektien nimet ja lisätään ne projektinvalinnassa käytettävään alasvetovalikkoon. Tämän jälkeen sovellus asetetaan käyttäjän ohjaukseen.

Ennen XML-tiedostojen luomisen aloittamista käyttäjän tulee valita projekti, josta XML-tiedostot luodaan, sekä hakemisto, johon nämä tiedostot tallennetaan. XML-tiedostojen luonti aloitetaan käyttöliittymän painikkeella ”Create XML”. Sovellus luo tällöin neljä XML-tiedostoa, jotka ovat ”YTEHDAS.XML”, ”YOSASTO.XML”, ”YPIKAAV.XML” ja ”PLINJA.XML”. Jokainen näistä tiedostoista sisältää määrätyntyyppisten Alma-objektien, kuten projektien, osastojen tai putkilinjojen tietoja. PMMATE:ssa on käytössä seuraavanlainen hierarkia tallennuksessa ja raportoinnissa:

--- Tehdas
 --- Osasto
  --- Prosessi (PI-kaavio)
   --- Putkilinja

Tässä hierarkiassa kukin osasto kuuluu siis jonkin tehtaan alle, PI-kaavio jonkin osaston alle ja niin edelleen. Koska Alma-tietokannassa suunnittelijoita ei ole haluttu rajoittaa tietyn hierarkian käyttöön, on mahdollista, että esimerkiksi putkilinja-objekti on sijoitettu siellä suoraan osasto-objektin alle, tai että prosessi-objekti on sijoitettu suoraan tehdas-objektin alle. Alma-tietokannassa tehdas-objektin tilalla voi olla myös joko projekti- tai

yritys-objekti. Jotta PMMATE toimisi oikein myös tapauksissa, joissa Alman objektihierarkia ei vastaa sen omaa objektihierarkiaa, luodaan jokaiseen neljään XML-tiedostoon myös niin sanottu virtuaalinen objekti, jonka ID on etukäteen määritetty vakio. Esimerkiksi ”YTEHDAS.XML”-tiedostoon luodaan virtuaalinen tehdas. Nyt jos jonkin Alma-objektin vanhempana ei ole PMMATE-hierarkian mukainen objekti, määritetään kyseisen objektin vanhemmaksi ylemmän tason virtuaalinen objekti. Jos esimerkiksi Alma-tietokannassa olevan prosessi-objektin vanhempana on jotain muuta kun PMMATE-hierarkian vaatima osasto-objekti, määritetään kyseisen prosessiobjektin vanhemmaksi virtuaaliosasto. Kuten Alma-tietokannan rakenteen yhteydessä on esitetty, yhdellä objektilla voi Alma-tietokannassa olla useampi vanhempi, eli esimerkiksi sama putkilinja voi olla useamman prosessi-objektin alla. Tällaisessa tapauksessa objektin vanhemmaksi valikoituu se objekti, jonka ID:n Alma-rajapinnan hakutoiminto ensimmäisenä palauttaa.

Putkilinjojen yläpuolinen hierarkia on myös mahdollista määritellä pelkästään virtuaaliseksi valitsemalla sovelluksen päälomakkeesta valintaruutu ”Use virtual hierarchy”. Tällöin ”YTEHDAS.XML”, ”YOSASTO.XML” ja ”YPIKAAV.XML” – tiedostot sisältävät ainoastaan vastaavan virtuaalisen objektin. Jokaisen putkilinja-objektin vanhemman ID on tällöin siis virtuaalisen prosessi-objektin ID, prosessi-objektin vanhemman ID virtuaalisen osasto-objektin ID ja niin edelleen. Esimerkki sovelluksella luotujen XML-tiedostojen rakenteesta on esitetty liitteessä 1.

5.1.4 Ongelmat

XML Creator for PMMATE-sovellus kärsii pitkälti samoista ongelmista kuin sen pohjana toiminut Database Creator for Pinekin, pois lukien tietokantojen synkronointiin liittyvät ongelmat. Myös XML Creator for PMMATE-sovellukseen oli tarkoitus toteuttaa hiljainen sisäänkirjautuminen ilman käyttäjän interaktiota, mutta Alma-rajapinta ei tällaista toiminnallisuutta sovelluksen toteutushetkellä tarjonnut. Toinen ongelma on, että sovellus toteutettiin ennen kuin CTS:llä otettiin käyttöön rajoitetut käyttö-oikeudet Almassa, eikä se näin ollen toimi, ellei käyttäjällä ole pääkäyttäjän oikeuksia Alma-tietokantaan. Koska sovelluksen käyttäjäkunta on kuitenkin toistaiseksi harvalukuinen, ja tulee todennäköisesti

pysymään sellaisena myös tulevaisuudessa, tämän ongelman ei ole katsottu olevan niin häiritsevä, että se vaatisi korjausta. Kolmas ongelma toteutuksen yhteydessä oli se, että itse PMMATE:n toimintaa ei päästy kokeilemaan, vaan XML Creator for PMMATE toteutettiin pelkästään annettujen vaatimusten pohjalta. Tästä syystä kokonaiskuva PMMATE-ohjelmistosta jäi varsin suppeaksi, eikä vaihtoehtoisia ratkaisumalleja alkuperäisen ongelman (tiedonsiirto Alma-tietokannan ja PMMATE:n välillä) ratkaisemiseksi pystytty kehittelemään.

6 JOHTOPÄÄTÖKSET

Tämän diplomityön tavoitteena oli tutustua Alma-tiedonhallintajärjestelmään ja auttaa sen käyttöönotossa mm. integroimalla aikaisemmin käytössä olleita sovelluksia osaksi uutta järjestelmää. Työssä on käsitelty tiedonhallintaratkaisuiden teoriaa ja rakennetta, sekä määritelty yksinkertainen malli, jonka avulla tätä rakennetta voidaan kuvata. Lisäksi työssä toteutettiin kolme sovellusta, joiden tarkoituksena on mahdollistaa Alma-tiedonhallintajärjestelmän hyödyntäminen CTS Pine- ja PMMATE-sovellusten yhteydessä.

Työn aikana CTS Engtec Oy:ssä otettiin onnistuneesti käyttöön uusi Alma-tiedonhallintajärjestelmä vanhentuneen Intecro-järjestelmän tilalle. Alma tarjoaa suunnittelijoille useita Intecrosta puuttuneita toiminnallisuuksia, jotka alun totuttautumisen jälkeen helpottavat ja nopeuttavat suunnittelutyötä. Alma sisältää myös keskitetyn tiedonhallintajärjestelmän tarjoamat edut, joita ovat mm. tiedon helpompi hallinta sekä kopiointi projektista toiseen. Se on myös teknisesti ajantasainen, mahdollistaen esimerkiksi suunnittelutietojen tarkastelun Internetin ylitse pelkkää web-selainta hyödyntäen. Java-alusta takaa puolestaan sen, että järjestelmä toimii myös tulevaisuudessa, käytössä olevasta käyttöjärjestelmästä riippumatta.

Työssä toteutetut sovellukset täyttivät niille asetetut toiminnalliset vaatimukset, mahdollistaen näin CTS Pine- ja PMMATE-sovellusten käyttämisen Alma-tiedonhallintajärjestelmän yhteydessä. Tästä huolimatta varsinkaan Pine-järjestelmän käyttö ei ole sujunut kitkatta. Jo ennen liittämistä Almaan se sisälsi joitakin perustavanlaatuisia ongelmia. Projektin perustaminen koettiin hankalaksi, kuten myös tietojen kopiointi projektista toiseen. Lisäksi ongelmia aiheuttivat mm. tietokantaan päätyneet duplikaattipositiot sekä itse sovelluksen toiminnan monimutkaisuus. Alman liittämisessä Pineen hyödynnetty väliaikaistietokantaratkaisu puolestaan lisäsi sovelluksen kompleksisuutta edelleen, ja useat käyttäjät ovatkin kokeneet ohjelmiston hankalaksi käyttää (CTS Engtec Oy, 2009). Pinen ongelmia koetettiin osittain paikata CTS Pine Management Utility:llä, mutta kaiken kaikkiaan sekä Database Creator for Pine ja Pine Management Utility -sovellukset ovat jo lähtökohtaisesti olleet ainoastaan väliaikaisia

ratkaisuja, joiden avulla jo alun perin ongelmallista Pine-ohjelmistoa on yritetty tekohengittää. Pinen rakenteen katsotaan olevan sellainen, että sen korjaamista ei nähdä realistisena vaihtoehtona. Näin ollen CTS:llä onkin suunnitteilla uusi, helppokäyttöisempi ohjelmisto Pinen korvaamiseksi, jossa vanhaa Pineä vaivanneet ongelmat on korjattu, ja puutteellinen toiminnallisuus paikattu. Tämän kokonaan uuden ohjelmiston toteutusta olisi ollut hyvä harkita tarkemmin jo siinä vaiheessa, kun punnittiin mahdollisia tapoja Pinen liittämiseksi Almaan. Toisaalta Pine on kuitenkin jatkuvassa tuotantokäytössä oleva ohjelmisto, ja kokonaan uuden ohjelmiston kehittäminen Alma-tiedonhallintajärjestelmän käyttöönoton yhteydessä olisi mahdollisesti hidastunut käyttöönottoa liiaksi.

Uutta tiedonhallintajärjestelmää valittaessa perusteellinen vaatimusten määrittely on tärkeää. Lisäksi uuden järjestelmän toiminta pitää olla mahdollisimman tarkasti tiedossa, jotta sen vaikutukset sekä käyttäjiin että olemassa oleviin järjestelmiin pystytään mahdollisimman tarkasti ennakoimaan. Tärkeää on myös hahmotella toiminta tilanteissa, jossa uuden järjestelmän käyttöönotto ei sujukaan suunnitelmien mukaan. Voidaankin todeta, että keskeisimpänä tekijänä onnistuneessa uuden tiedonhallintajärjestelmän käyttöönotossa on käyttöönottoprojektin jokaisen vaiheen huolellinen suunnitteleminen.

LÄHTEET

ALMA käyttöohje. 2010. [online-käyttöohje] ALMA-asiakasohjelman sisäänrakennettu käyttöohje. [viitattu 10.6.2010].

ALMA Software Oy. 2009. [verkkojulkaisu] ALMA Knowledge Management - konseptiesite. [viitattu 21.10.2009]. Saatavissa: http://www.alma.fi/Link.aspx?id=1061029

Anhøj J. 2003. Generic Design of Web-Based Clinical Databases. Journal of Medical Internet Research, osa 5. Nro. 4. Saatavissa: http://www.jmir.org/2003/4/e27/

Bernstein P. A. 1990. Transaction Processing Monitors. Communications of the ACM, osa 33. Nro 11, marraskuu 1990. s. 75 - 86. ISSN: 0001-0782. Saatavissa: The ACM Portal.

Brodie M.L., Schmidt J. W. 1982. Final Report of the ANSI/X3/SPARC DBS-SG Relational Database Task Group. ACM SIGMOD Record, osa 12. Nro. 4, heinäkuu 1982. s. 1 - 62. ISSN 0163-5808. Saatavissa: The ACM Portal.

Codd E.F. 1970. A Relational Model of Data for Large Shared Databanks. Communications of the ACM, osa 13. Nro. 6, kesäkuu 1970. s. 377 - 387. ISSN: 0001-0782. Saatavissa: The ACM Portal.

CTS Engtec Oy. 2008. [CTS Engtec Oy:n sisäinen dokumentaatio] Alma-palaverimuistio 10 - 11.1.2008. Päivitetty 15.1.2008, [viitattu: 10.6.2010]. Saatavissa: Yrityksen lähiverkossa, vaatii käyttäjätunnukset.

CTS Engtec Oy. 2009. [CTS Engtec Oy:n sisäinen dokumentaatio] Pine-kehitys – palaverimuistio 10.12.2009. Päivitetty 19.1.2010, [viitattu 10.6.2010]. Saatavissa: Yrityksen lähiverkossa, vaatii käyttäjätunnukset.

CTS Engtec Oy. 2010. [yrityksen www-sivut] About us. [viitattu 4.6.2010]. Saatavissa: http://www.ctse.fi/about-us

Date C.J. 2004. An Introduction to Database Systems, Eight Edition. USA, Addison Wesley. 983 s. ISBN 0-321-18956-6

Dinu V., Nadkarni P. 2007. Guidelines for the Effective Use of Entity–Attribute–Value Modeling for Biomedical Databases. International Journal of Medical Informatics, osa 17. Nro. 11-12, marraskuu 2007. s. 769-779. Saatavissa: Elsevier ScienceDirect.

Gilmore, J. W. 2008. Beginning PHP and MySQL: From Novice to Professional, Third Edition. USA, Apress. 1080 s. ISBN: 1-59059-862-8

Haigh T. 2006. “A Veritable Bucket of Facts” Origins of the Data Base Management System. ACM SIGMOD Record, osa 35. Nro. 2, kesäkuu 2006. s. 33-49. ISSN: 0163-5808. Saatavissa: The ACM Portal.

Hasu P. 2010. [sähköposti] PMMATE-kuvaus, 24.5.2010. [viitattu 3.6.2010]. Saatavissa: Henkilökohtainen sähköpostiviesti.

Härder T., Reuter A. 1983. Principles of Transaction-Oriented Database Recovery. ACM Computing Surveys, osa 15. Nro 4, joulukuu 1983. s. 287 - 317. ISSN: 0360-0300. Saatavissa: The ACM Portal.

Kauppinen N. 2008. [CTS Engtec Oy:n sisäinen dokumentaatio] CTS Pine -kuvaus. Päivitetty 25.7.2008, [viitattu 2.6.2010]. Saatavissa: Yrityksen lähiverkossa, vaatii käyttäjätunnukset.

Kiminki, T. 2009. [sähköposti] Dtyö-rakenne, 22.1.2009. [viitattu 10.6.2010]. Saatavissa: Henkilökohtainen sähköpostiviesti.

Kruckenberg M., Pipes J. 2005. Pro MySQL. USA, Apress. 768 s. ISBN: 1-59059-505-X

McLeod D., Smith J. M. 1980. Abstraction in Databases. International Conference on Management of Data. Proceedings of the 1980 workshop on Data abstraction, databases and conceptual modelling. s. 19 - 25. ISBN: 0-89791-031-1. Saatavissa: The ACM Portal.

Microsoft. 2010. [verkkojulkaisu] Visual Studio Express - Frequently Asked Questions. [viitattu 2.6.2010]. Saatavissa: http://www.microsoft.com/express/support/support-faq.aspx

Morgenstern M. 1987. Security and Inference in Multilevel Database and Knowledge-Base Systems. ACM SIGMOD Record, osa 16. Nro. 3, joulukuu 1987. s. 357 - 373. ISSN: 0163-5808. Saatavissa: The ACM Portal.

MSDN. 2010a. [verkkojulkaisu] Hardware and Software Requirements for Installing SQL Server 2008 R2. [viitattu 27.5.2010]. Saatavissa: http://msdn.microsoft.com/en-us/library/ms143506.aspx

MSDN. 2010b. [verkkojulkaisu] Three-Tiered Distribution. [viitattu 5.2.2010]. Saatavissa: http://msdn.microsoft.com/en-us/library/ms978694.aspx

MSDN. 2010c. [verkkojulkaisu] Support Statement for Visual Basic 6.0 on Windows Vista, Windows Server 2008 and Windows 7. [Viitattu 31.5.2010]. Saatavissa: http://msdn.microsoft.com/en-us/vbasic/ms788708.aspx

MySQL. 2009. [verkkojulkaisu] MySQL Pluggable Storage Engine Architecture. [viitattu 24.7.2009]. Saatavissa: http://solutions.mysql.com/engines.html

MySQL. 2010. [verkkojulkaisu] MySQL 5.0 Reference Manual - The Maximum Number of Columns Per Table. [viitattu 29.5.2010]. Saatavissa: http://dev.mysql.com/doc/refman/5.0/en/column-count-limit.html

Nadkarni, P. M., Marenco L., Chen R., Skoufos E., Shepherd G., Miller P. 1999. Organization of Heterogeneous Scientific Data Using the EAV/CR Representation. Journal of the American Medical Informatics Association, osa 6. Nro. 6, marras-joulukuu 1999. s. 478-493.

Oppel A. J. 2004. Databases Demystified. USA, The McGraw-Hill Companies. 338 s. ISBN: 0-07-225364-9

Oracle. 2004. [verkkojulkaisu] What's the difference between Oracle Database Editions? [viitattu 9.6.2010]. Saatavissa: http://www.oracle.com/technology/products/lite/pdf/lite_and_bigoracle_diff.pdf

Ortiz A. 2000. Three-Tier Architecture. Linux Journal, Nro 75, heinäkuu 2000. ISSN: 1075-3583

Patterson D. A., Gibson G., Katz R. H. 1988. A Case for Redundant Arrays of Inexpensive Disks (RAID). ACM SIGMOD Record, osa 17. Nro 6, kesäkuu 1988. ISSN: 0163-5808. Saatavissa: The ACM Portal.

Raudasoja, H. 2009a. [sähköposti] ALMA-kysymyksiä, 4.3.2009. [viitattu 10.6.2010]. Saatavissa: Henkilökohtainen sähköpostiviesti.

Raudasoja, H. 2009b. [sähköposti] Objektin käyttöoikeuksien tarkistaminen ALMAX-rajapinnan avulla, 19.5.2009. [viitattu 10.6.2010]. Saatavissa: Henkilökohtainen sähköpostiviesti.

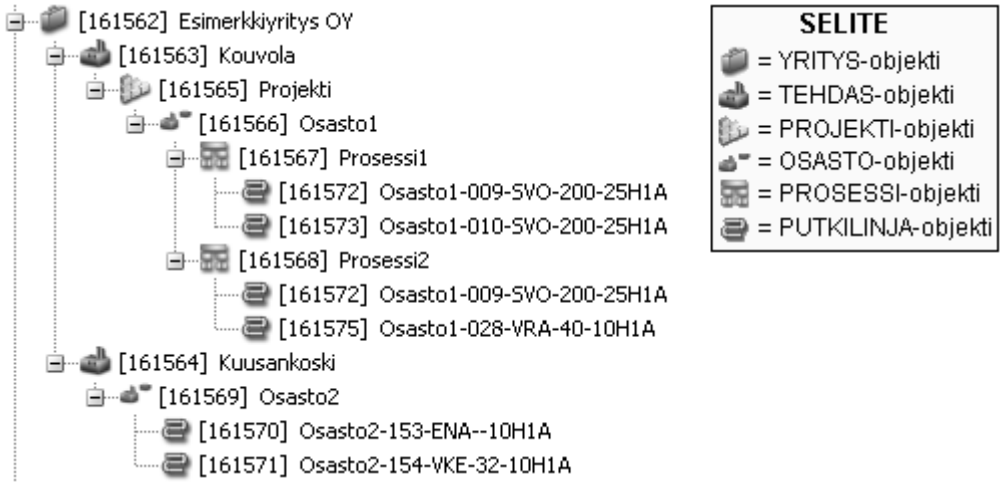
Savino S. P., Queroli S. M. 1996. Impact of Client/Server Computing on the Telecommunications Industry. International Conference on Engineering and Technology Management, IEMC 96. ISBN: 0-7803-3552-X

Venkula, J. 2003. [CTS Engtec Oy:n sisäinen dokumentaatio] CTS Pine - Piirikohtaisten toimintakuvausten laatimisen työkalu. Päivitetty 7.12.2004, [viitattu 1.6.2010]. Saatavissa: Yrityksen lähiverkossa, vaatii käyttäjätunnukset.

Yarger R.J., Reese G., King T. 1999. MySQL and mSQL, First Edition. USA, O'Reilly & Associates Inc.. 502 s. ISBN 1-56592-434-7

LIITE 1. ESIMERKKI XML-TIEDOSTOJEN RAKENTEESTA

Alla on esitetty esimerkki CTS XML Creator for PMMATE -ohjelmalla luoduista XML-tiedostoista, joita käytetään tiedon siirtämiseen Alma-tiedonhallintajärjestelmästä PMMATE-sovellukseen. Esimerkin avulla pyritään havainnollistamaan sovelluksella luotujen XML-tiedostojen rakennetta. Kuvassa 20 on esitetty Alma-tiedonhallintajärjestelmän projektihierarkia, jota on käytetty alla olevien XML-tiedostojen luomiseen. XML Creator for PMMATE:n projektivalintalistasta valittu projekti on tässä esimerkissä ”Esimerkkiyritys Oy”, eli XML-tiedostojen luominen aloitetaan tältä tasolta. Sovelluksen asetuksista valintaruutu ”Use virtual hierarchy” ei ollut valittuna tiedostoja luotaessa. Esimerkkihierarkiasta on syytä huomioida se, ettei se välttämättä edusta suositeltavaa rakennetta ja objektien nimeämiskäytäntöjä, vaan se toimii ainoastaan esimerkkinä XML Creator for PMMATE -sovelluksen toiminnallisuuden havainnollistamisessa. Huomioitavaa on myös, että putkilinja, jonka ID on 161572 (Tunnus: Osasto1-009-SVO-200-25H1A) on sekä Prosessi1:sen että Prosessi2:sen alla. Putkilinjojen 161570 ja 161571 vanhempana puolestaan ei ole prosessi-objekti vaan osasto.



Kuva 20 - Hierarkia, josta esimerkissä esiintyvät XML-tiedostot on luotu

(jatkuu)

YTEHDAS.XML

```
<?xml version="1.0" encoding="Windows-1252" standalone="yes"?>
<VFPDATA>
	<ytehdas>
		<pgid>-1</pgid>
		<pid>-1</pid>
		<gid>100</gid>
		<id>2</id>
		<tunnus>¤01</tunnus>
		<nimitys1>Virtuaalitehdas</nimitys1>
		<nimitys2></nimitys2>
	</ytehdas>
	<ytehdas>
		<pgid>-1</pgid>
		<pid>-1</pid>
		<gid>100</gid>
		<id>161563</id>
		<tunnus>¤Kouvola</tunnus>
		<nimitys1>Tehdas</nimitys1>
		<nimitys2></nimitys2>
	</ytehdas>
	<ytehdas>
		<pgid>-1</pgid>
		<pid>-1</pid>
		<gid>100</gid>
		<id>161564</id>
		<tunnus>¤Kuusankoski</tunnus>
		<nimitys1>Tehdas</nimitys1>
		<nimitys2></nimitys2>
	</ytehdas>
	<ytehdas>
		<pgid>-1</pgid>
		<pid>-1</pid>
		<gid>100</gid>
		<id>161565</id>
		<tunnus>¤Projekti</tunnus>
		<nimitys1>Projekti</nimitys1>
		<nimitys2></nimitys2>
	</ytehdas>
	<ytehdas>
		<pgid>-1</pgid>
		<pid>-1</pid>
		<gid>100</gid>
		<id>161562</id>
		<tunnus>¤Esimerkkiyritys OY</tunnus>
		<nimitys1>Yritys</nimitys1>
		<nimitys2></nimitys2>
	</ytehdas>
</VFPDATA>
```

(jatkuu)

Vakioina määritellyt arvot:

- PGID = -1
- PID = -1
- GID = 100
- Virtuaalisen tehtaan ID = 2
- Virtuaalisen tehtaan tunnus = ”¤01”
- Virtuaalisen tehtaan nimitys1 = ”Virtuaalitehdas”

Huomioitavaa: YTEHDAS.XML sisältää kaikki projektivalintalistauksesta valitun objektin alapuolella hierarkiassa sijaitsevat yritys-, projekti- ja tehdasobjektit, eikä pelkästään tehdasobjekteja. Tämä on toteutettu siksi, että Alma-tietokannan hierarkiassa ei välttämättä ole tehdasobjektia. Tällaisessa tapauksessa tiedot luetaan projekti- tai yritysobjektista. Ensimmäisenä tiedostossa on virtuaalitehdas, jota ei löydy Almasta. Jos osasto-tyyppiselle objektille ei löydy vanhempaa, jonka tyyppi on joko yritys, tehdas tai projekti, määritetään kyseessä olevan osaston vanhemmaksi tämän virtuaalitehtaan ID. Jos valintaruutu ”Use virtual hierarchy” on valittuna, tässä tiedostossa on määritetty ainoastaan virtuaalinen tehdas.

YOSASTO.XML

```
<?xml version="1.0" encoding="Windows-1252" standalone="yes"?>
<VFPDATA>
	<yosasto>
		<pgid>100</pgid>
		<pid>2</pid>
		<gid>101</gid>
		<id>3</id>
		<tunnus>¤01</tunnus>
		<nimitys1>Virtuaaliosasto</nimitys1>
	</yosasto>
	<yosasto>
		<pgid>100</pgid>
		<pid>161565</pid>
		<gid>101</gid>
		<id>161566</id>
		<tunnus>¤Osasto1</tunnus>
		<nimitys1>Osasto</nimitys1>
	</yosasto>
	<yosasto>
```

(jatkuu)

```
                    <pgid>100</pgid>
                    <pid>161564</pid>
                    <gid>101</gid>
                    <id>161569</id>
                    <tunnus>¤Osasto2</tunnus>
                    <nimitys1>Osasto</nimitys1>
        </yosasto>
</VFPDATA>
```

Vakioina määritellyt arvot:

- PGID = 100 (Tehtaan GID)
- GID = 101
- Virtuaalisen osaston ID = 3
- Virtuaalisen osaston tunnus = ”¤01”
- Virtuaalisen osaston nimitys1 = ”Virtuaaliosasto”

Huomioitavaa: YOSASTO.XML sisältää kaikki projektivalintalistauksesta valitun objektin alapuolella hierarkiassa sijaitsevat osasto-tyyppiset objektit. Osasto-objektin vanhempi voi olla tyypiltään yritys, tehdas tai projekti. Jos suoraan osasto-objektin yläpuolella oleva objekti ei ole tyypiltään mikään näistä, määritellään vanhemmaksi virtuaalinen tehdas (PID = 2). Jos valintaruutu ”Use virtual hierarchy” on valittuna, tässä tiedostossa on määritetty ainoastaan virtuaalinen osasto.

YPIKAAV.XML

```
<?xml version="1.0" encoding="Windows-1252" standalone="yes"?>
<VFPDATA>
        <ypikaav>
                    <pgid>101</pgid>
                    <pid>3</pid>
                    <gid>102</gid>
                    <id>4</id>
                    <tunnus>¤01</tunnus>
                    <nimitys1>Virtuaaliprosessi</nimitys1>
        </ypikaav>
        <ypikaav>
                    <pgid>101</pgid>
                    <pid>161566</pid>
                    <gid>102</gid>
                    <id>161567</id>
                    <tunnus>¤Prosessi1</tunnus>
```

(jatkuu)

```
                    <nimitys1>Prosessi</nimitys1>
        </ypikaav>
        <ypikaav>
                    <pgid>101</pgid>
                    <pid>161566</pid>
                    <gid>102</gid>
                    <id>161568</id>
                    <tunnus>¤Prosessi2</tunnus>
                    <nimitys1>Prosessi</nimitys1>
        </ypikaav>
</VFPDATA>
```

Vakioina määritellyt arvot:

- PGID = 101 (Osaston GID)
- GID = 102
- Virtuaalisen prosessin ID = 4
- Virtuaalisen prosessin tunnus = ”¤01”
- Virtuaalisen prosessin nimitys1 = ”Virtuaaliprosessi”

Huomioitavaa: YPIKAAV.XML sisältää kaikki projektivalintalistauksesta valitun objektin alapuolella hierarkiassa sijaitsevat prosessi-tyyppiset objektit. Prosessi-objektin vanhemman tulee olla tyypiltään osasto-objekti. Jos suoraan prosessi-objektin yläpuolella oleva objekti ei ole tyypiltään osasto, määritellään vanhemmaksi virtuaalinen osasto (PID = 3). Jos valintaruutu ”Use virtual hierarchy” on valittuna, tässä tiedostossa on määritetty ainoastaan virtuaalinen prosessi.

PLINJA.XML

```
<?xml version="1.0" encoding="Windows-1252" standalone="yes"?>
<VFPDATA>
        <plinja>
                    <pgid>102</pgid>
                    <pid>161567</pid>
                    <gid>105</gid>
                    <id>161572</id>
                    <tunnus>Osasto1-009-SVO-200-25H1A</tunnus>
                    <positio>¤009</positio>
                    <nimi></nimi>
                    <mista></mista>
                    <mihin></mihin>
                    <nimelliskoko>200</nimelliskoko>
```

```
                <virtaavaaine>SVO</virtaavaaine>
                <putkiluokkapaine>25</putkiluokkapaine>
                <putkiluokkatunnus>H1A</putkiluokkatunnus>
                <huomautus></huomautus>
                <mallinnuskytkin>0</mallinnuskytkin>
                <sakeus></sakeus>
                <suunnittelupaine>15</suunnittelupaine>
                <suunnittelulampotila>100</suunnittelulampotila>
                <maksimivirtaus></maksimivirtaus>
                <normaalivirtaus></normaalivirtaus>
                <minimivirtaus></minimivirtaus>
                <testiryhma></testiryhma>
                <kayttopaine></kayttopaine>
                <minimipaine></minimipaine>
                <kayttolampotila></kayttolampotila>
                <minimilampotila></minimilampotila>
                <pedluokka></pedluokka>
                <pedclassmodule></pedclassmodule>
                <pedclassannotiations></pedclassannotiations>
                <pedclassmessage></pedclassmessage>
                <eristystarkoitus></eristystarkoitus>
                <eristysmateriaali></eristysmateriaali>
                <eristyspaksuus></eristyspaksuus>
                <eristysstandardi></eristysstandardi>
                <toimittaja></toimittaja>
                <massavirta></massavirta>
                <viskositeetti></viskositeetti>
                <materiaali></materiaali>
                <saattolampotila></saattolampotila>
                <saattotyyppi></saattotyyppi>
                <tiheys></tiheys>
                <lisatunnus1></lisatunnus1>
                <lisatunnus2></lisatunnus2>
                <pharvo></pharvo>
                <koeponnistuspaine></koeponnistuspaine>
                <vaaditaankojannitysanalyysia></vaaditaankojannitysanalyysia>
                <systeemi></systeemi>
                <tyonumero></tyonumero>
                <projektinumero></projektinumero>
                <asennusvalmistusstatus></asennusvalmistusstatus>
                <asennusvastuu></asennusvastuu>
                <suunnittelustatus></suunnittelustatus>
                <suunnitellut></suunnitellut>
                <osasto>Osasto1</osasto>
                <pikaavio>Prosessi1</pikaavio>
                <muutostarv></muutostarv>
                <mvirtamax></mvirtamax>
                <mvirtanor></mvirtanor>
</plinja>
<plinja>
                <pgid>102</pgid>
                <pid>161567</pid>
                <gid>105</gid>
                <id>161573</id>
                <tunnus>Osasto1-010-SVO-200-25H1A</tunnus>
                <positio>¤010</positio>
                …
</plinja>
```

(jatkuu)

```
        <plinja>
                <pgid>102</pgid>
                <pid>161568</pid>
                <gid>105</gid>
                <id>161575</id>
                <tunnus>Osasto1-028-VRA-40-10H1A</tunnus>
                <positio>¤028</positio>
                …
        </plinja>
        <plinja>
                <pgid>102</pgid>
                <pid>4</pid>
                <gid>105</gid>
                <id>161570</id>
                <tunnus>Osasto2-153-ENA--10H1A</tunnus>
                <positio>¤153</positio>
                …
        </plinja>
        <plinja>
                <pgid>102</pgid>
                <pid>4</pid>
                <gid>105</gid>
                <id>161571</id>
                <tunnus>Osasto2-154-VKE-32-10H1A</tunnus>
                <positio>¤154</positio>
                …
        </plinja>
</VFPDATA>
```

Vakioina määritellyt arvot:

- PGID = 102 (Prosessin GID)
- GID = 105

Huomioitavaa: PLINJA.XML sisältää kaikki projektivalintalistauksesta valitun objektin alapuolella hierarkiassa sijaitsevat putkilinja-objektit. Putkilinja-objektin vanhemman tulee olla tyypiltään prosessi-objekti. Jos suoraan putkilinja-objektin yläpuolella oleva objekti ei ole prosessi-objekti, määritellään vanhemmaksi virtuaalinen prosessi (PID = 4). Vanhemmaksi määritellään virtuaalinen prosessi myös silloin, kun valintaruutu ”Use virtual hierarchy” on valittuna. Tässä esimerkissä kaksi viimeistä putkilinjaa sijaitsevat hierarkiassa suoraan osaston alla, jolloin niiden vanhemmaksi määritetään siis virtuaalinen prosessi. Putkilinja, jonka ID on ”161572”, sijaitsee hierarkiassa kahden eri prosessin alapuolella, jolloin sen vanhemmaksi valitaan prosessi-objekti, jonka Alma-rajapinnan hakutoiminto palauttaa ensimmäisenä. Tässä tapauksessa vanhemmaksi on valittu prosessi-

(jatkuu)

objekti, jonka ID on ”161567”. Viimeisestä neljästä putkilinjasta on tilan säästämiseksi esitelty vain ne attribuutit, jotka ovat esimerkin kannalta tärkeitä.