Saimaan ammattikorkeakoulu
Tekniikka Lappeenranta
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

Kimmo Tahvanainen

PROSESSIN ARVIOINTIJÄRJESTELMÄN KEHITTÄMINEN

Opinnäytetyö 2011

TIIVISTELMÄ

Kimmo Tahvanainen
Prosessin arviointijärjestelmän kehittäminen, 67 sivua
Saimaan ammattikorkeakoulu, Lappeenranta
Tekniikka, Tietotekniikan koulutusohjelma
Ohjelmistotekniikan suuntautumisvaihtoehto
Opinnäytetyö 2011
Ohjaajat: lehtori Martti Ylä-Jussila, Saimaan ammattikorkeakoulu
laatupäällikkö Varpu Savolainen, Stora Enso Oyj, Metsä

Opinnäytetyön tarkoituksena on kehittää asiakasyrityksen pääprosessien laadun arviointiin ja tarkasteluun soveltuva helppokäyttöinen työkalu. Prosessien laadun arviointimallin perustana oli Philipsin kehittämä Process Survey Tools (PST), josta oli tarkoitus kehittää asiakkaan omaan organisaatioon räätälöity prosessien arviointityökalu, PAT-tietojärjestelmä. Työkalussa jokaiselle pääprosessille määriteltiin yksitoistatasoinen asteikko, joka kertoi prosessien laadukkuudesta ja kehitysasteesta. Pääprosessien tasojen määrittelemiseksi jokainen prosessi jaettiin aliprosesseihin, joiden tasot määriteltiin erikseen. Tasojen määrittelyssä käytettiin tasokysymyksiä, jotka piti jokaista tasoa kohti täyttää saavuttaakseen kyseisen tason. Lisäksi kaikki edelliset tasot tuli olla täytettyinä.

Työkalu toteutettiin Microsoft Visual Studio -sovelluskehittimellä, sivujen toteutus toimii Microsoft IIS -palvelimen päällä olevalla, .NET Frameworkia käyttävällä ASP.NET:llä. Taustalla olevan logiikan ohjelmointikielenä on Visual Basic .NET ja tietojen tallennus tapahtuu Microsoft SQL Server -tietokantaan.

Työkalu suunniteltiin ja toteutettiin Web-sovelluksena, jotta se olisi mahdollisimman helposti saatavilla ja ylläpidettävissä yrityksen intranetissä. Prosessien tietojen katselu sallittiin kaikille, mutta tietojen ylläpito sallittiin vain tietyille henkilöille.

Web-käyttöliittymän kieleksi valittiin asiakkaan yrityksessään käyttämän kielen mukaisesti englanti, mutta sovellukseen myös määriteltiin tuki useammalle kielelle ja mahdollisuus vaihtaa kieltä selaamisen aikana.

Työn tuloksena syntynyt PAT-tietojärjestelmä otettiin aluksi käyttöön asiakasyrityksen pilottiprosessissa, jonka jälkeen järjestelmä on tarkoitus ottaa laajemmin käyttöön yrityksen kaikissa pääprosesseissa.

Avainsanat: prosessi, arviointimalli, prosessin arviointi, arviointityökalu, tietojärjestelmä, kehittäminen, laatu, EFQM

ABSTRACT

Kimmo Tahvanainen
Developing of Process Assessment System, 67 pages
Saimaa University of Applied Sciences, Lappeenranta
Technology, Degree Programme in Information Technology
Software Engineering
Bachelor's Thesis, 2011
Instructors: Lecturer Martti Ylä-Jussila, Saimaa University of Applied Sciences
Quality Manager Varpu Savolainen, Stora Enso Wood Supply Finland

The purpose of this Bachelor's Thesis is to develop an easy to use tool for quality assessment and observation of the client enterprise's main processes. The basis of the process quality assessment model was Process Survey Tools (PST), developed by Philips. The tool to be evolved from this model and tailored to the needs of the client enterprise is called PAT, Process Assessment Tool. In this tool, every main process has eleven steps, which describe the quality and maturity of each process. To specify the level of each main process they must be divided into sub processes and specify then the level of each sub process. For level specification, each level has level questions, whose requirements must all be met to reach that particular level. In addition to that, all levels below that had to be already reached.

The tool was carried out with Microsoft Visual Studio and the system's web pages are displayed with ASP.NET, using .NET Framework and ran over Microsoft IIS server. The background logics are coded with Visual Basic .NET and the system data is recorded into a Microsoft SQL Server -database.

The tool was designed and developed as a Web application to make it as easy as possible to be put available and maintain in the enterprise's intranet. Viewing of the processes' data was allowed for everyone, but data administration was allowed only for certain individuals.

The language of the Web interface, English, was selected according to the language used in the client enterprise, but the application was also specified with multilingual support and an option to change language during browsing.

The resulting data system, PAT, was initially taken into use by an enterprise's pilot process. After that, the system is meant to be taken into full use by all the enterprise's main processes.

Keywords: Process, Assessment Model, Process Assessment, Assessment Tool, Data System, Developing, Quality, EFQM

SISÄLLYS

KÄSITTEET, TERMIT JA LYHENTEET

Active Directory	Microsoftin kehittämä hakemistopalvelu, joka sisältää tietoa käyttäjistä, tietokoneista sekä verkon resursseista. Se mahdollistaa keskitetyn resurssien jakamisen käyttäjille ja sovelluksille sekä tarjoaa tavan nimetä, kuvata, paikallistaa, hallita ja suojata käytössä olevia verkon resursseja.
EFQM	European Foundation for Quality Management. Voittoa tuottamaton järjestö, jonka tarkoituksena on auttaa organisaatioita kehittämään liiketoimintansa laadukkuutta.
EFQM Excellence Award	Arvostettu palkinto, jonka EFQM vuosittain myöntää Euroopan laadukkaimmille organisaatioille.
EFQM Excellence Model	EFQM:n kehittämä laadukkuuden malli, jonka tarkoituksena on helpottaa organisaatioita kehittämään toimintaansa laadukkaammaksi.
LDAP	Lightweight Directory Access Protocol. Verkkoprotokolla, jota käytetään pääasiassa käyttäjäntunnistukseen tai erilaisten resurssien käyttöoikeuksien tarkistuksiin, mm. Microsoftin Active Directory käyttää tätä.
PAT	Asiakkaalle räätälöity prosessien arviointityökalu, tietojärjestelmä, joka on tämän opinnäytetyön lopputulos.
PST	Philipsin kehittämä Process Survey Tools -työkalu, joka on pohjana asiakkaalle tehtävän PAT-työkalun kehitystyössä.
RUP	IBM Rational Unified Process. Alun perin Rational Software Corporation -yrityksen kehittämä iteratiivisen ja inkrementaalisen ohjelmistokehityksen prosessikehys. Tämän työn teossa käytetyn systeemityömallin perusta.
VPN	Virtual Private Network. Tapa, jolla yrityksen verkkoja yhdistetään toisiinsa tai yksittäisiä työasemia liitetään yrityksen verkkoon julkisen verkon yli. Yksityisyys turvataan joko fyysisen yhteyden tai yhteyden salauksen avulla.
WYSIWYG	What You See Is What You Get. Termiä käytetään viitattaessa ohjelmistoihin (mm. tekstinkäsittelyohjelmat ja erilaiset editorit), joissa sisältö näyttää muokattaessa hyvin samalta kuin lopputulos.

1 JOHDANTO

Tämän opinnäytetyön viitekehyksenä on yrityksen liiketoiminnan ja prosessien laadun tarkastelua ja mittaamista helpottavan työkalun kehityshanke. Asiakkaana ja työn tilaajana on Suomessa mm. Stora Enson puunhankinnasta vastaava Stora Enso Metsä.

Asiakas haluaa parantaa liiketoimintansa laatua ja kilpailukykyä ottamalla organisaatiossaan käyttöön EFQM Excellence Model:in ja testata sen jälkeen organisaationsa laadukkuutta osallistumalla Suomen laatupalkinto -kilpailuun. Heidän organisaationsa pääprosessien laadun arviointia varten tarvitaan kuitenkin työkalu, jonka kehityspohjana ja mallina käytetään Philipsin kehittämää työkalua. Tämä työkalu ei kuitenkaan sellaisenaan sovellu Stora Enso Metsän käyttöön, joten asiakkaalle tarvitsee kehittää heidän organisaationsa tarpeisiin räätälöity prosessien arviointityökalu.

Tässä dokumentissa kuvataan asiakkaan käyttöön tehtävän prosessien arviointityökalu -tietojärjestelmän kehittämis- ja toteuttamisprojektin vaiheet ja systeemityössä käytetyt työkalut ja tekniikat. Lisäksi esitellään toteutuneen järjestelmän päätoiminnot sekä pohditaan työn onnistumista, työssä kohdattuja ongelmia ja tehdään koko ohjelmistokehitysprojektista yhteenveto.

2 ASIAKAS

Stora Enso on kansainvälinen konserni, jonka päätoimialaa ovat pakkaus-, paperi- ja puutuoteteollisuus. Stora Enso Metsä on taas osa Stora Enson konsernia Suomessa. Stora Enso kertoo Internet-sivuillaan koko konsernista ja Stora Enso Metsän toiminnasta seuraavaa:

Konsernin palveluksessa on noin 26 000 henkilöä ja sillä on 85 tuotantoyksikköä ympäri maailmaa. Stora Enson osakkeet noteerataan Helsingin ja Tukholman arvopaperipörsseissä. Asiakkaita ovat kustantamot, painotalot ja paperitukkurit sekä pakkaus-, puusepän- ja rakennusteollisuus.

Stora Enson vuosittainen tuotantokapasiteetti on 11,8 miljoonaa tonnia paperia ja kartonkia, 1,3 miljardia neliömetriä aaltopahvia ja 6,4 miljoonaa kuutiometriä puutuotteita, josta 3,2 miljoonaa kuutiometriä on jatkojalosteita. Konsernin liike-

vaihto vuonna 2010 oli 10,3 miljardia euroa ja liiketulos ilman kertaluonteisia eriä ja käyvän arvon muutoksia 754,1 miljoonaa euroa.

Stora Enso Metsä vastaa Stora Enson Suomen tuotantolaitosten puuhuollosta, niin kotimaan hankinnan kuin tuontipuun osalta. Stora Enso Metsä kuuluu Stora Enson puunhankintaorganisaatioon yhdessä Ruotsin, Baltian, Venäjän, Manner-Euroopan ja Aasian puunhankintayksiköiden kanssa.

Stora Enso on maailman johtava vastuullinen metsäteollisuusyritys. Tarjoamme asiakkaillemme uusiutuviin raaka-aineisiin perustuvia ratkaisuja. Tuotteemme tarjoavat ilmastoystävällisen sekä hiilijalanjäljeltään pienemmän vaihtoehdon monille kilpaileville tuotteille, jotka on valmistettu uusiutumattomista materiaaleista.

Stora Enso Metsä huolehtii luotettavasti ja joustavasti Stora Enson Suomen tuotantolaitosten puuhuollosta. Lisäksi hankimme bioenergiaa, teemme metsänhoitotöitä ja tarjoamme metsänomistajille neuvontapalveluita.

Stora Enso Metsä hoitaa puuntoimituksensa ympäristötietoisesti ja yhteiskuntavastuullisesti kestäviä puunhankintaperiaatteita noudattaen. Tunnemme toimintamme ympäristöön ja yhteiskuntaan kohdistuvat vaikutukset ja työskentelemme ympäristörasituksen vähentämiseksi.

Toiminnan laadusta kertovat laatupalkinnot ja -sertifikaatit.

Puunhankintamme painopiste on Suomessa. Puuvirrastamme yli puolet tulee suomalaisista yksityismetsistä ja vajaa 10 prosenttia valtion metsistä. Lisäksi hankimme Suomen tehtaille tuontipuuta, lähinnä Venäjältä ja Baltiasta.

Metsä on hallinnollisesti jaettu neljään hankinta-alueeseen, joiden operatiivisina perusyksikköinä ovat hankintatiimit. Hankintatiimien päätehtävät painottuvat puuraaka-aineen ostoon ja korjuuseen sekä kuljetusoperaatioiden valmisteluun.

Stora Enso Metsän palveluksessa on noin 630 henkilöä ja noin 400 kone- ja autoyrittäjää alihankkijoina.

3 PROSESSIEN LAATU JA ARVIOINTI

Yleensä yrityksistä ja prosesseista puhuttaessa ymmärretään tarkoitettavan yrityksen tuotantoprosesseja, mutta myös yrityksen kaikki muut toiminnot voidaan käsittää prosesseina. Tällä voidaan tarkoittaa sitä, että yrityksen toiminnot eivät missään tapauksessa ole tietyn vaiheen jälkeen niin sanotusti valmiita ja staattisia, joihin ei tule enää muutoksia, vaan toiminnot nähdään alati muuttuvina ja kehittyvinä prosesseina. Lisäksi Tuurala (2011) on todennut prosesseista seuraavaa:

Prosessi on toisiinsa kytkeytyvien tapahtumien sarja ja niiden toteuttamiseen tarvittavat resurssit, joiden avulla saadaan aikaan toiminnan tulokset. Palveluprosessi on toimintaa asiakkaan palvelemiseksi. Asiakasprosessi on asiakkaan näkökulma palvelutapahtumaan. Palveluprosessin osat ovat heräte (impulse), toiminta (process) ja lopputulos (output). Prosessiin tuodaan syötteinä (input) osaamista, energiaa, raaka-aineita ja muita prosessin tarvitsemia panoksia. Osaprosessit koostuvat työvaiheista ja tehtävistä. Ydinprosessit ovat organisaation ydintehtävän mukaisia prosesseja. Laajimpia ydinprosesseja voidaan kutsua pääprosesseiksi. Organisaation menestykselle tärkeimmät prosessit ovat avainprosesseja. Tukiprosessit ovat organisaation ydinprosesseja tukevia sisäisiä prosesseja, esimerkiksi hallinnon tehtävät tai laitostaloustehtävät.

Edellä mainituista prosesseista tehdään prosessikuvaukset, jotka ovat toiminnan ymmärtämiseksi ja ohjeistamiseksi tehtyjä sanallisia ja usein myös graafisia kuvauksia toiminnasta (Tuurala 2010). Stora Enson toimintojen laatu ja arviointi perustuu hyvin pitkälti juuri näihin näkemyksiin ja siksi toimintojen laatua ja kehitystä arvioidaan käsittelemällä niitä prosesseina. Tällä tavoin toimimalla yrityksen toimintojen jatkuvaa kehittämistä ja parantamista voidaan johtaa ja hallita helpommin sekä tuoda se konkreettisemmin esille yrityksen päivittäisessä toiminnassa.

Yrityksen prosessien laadun arviointi on tärkeä osa kehitettäessä yrityksen liiketoimintaa ja sen laatua. Mikäli laatua ei tarkkailla, yrityksen liiketoiminta on hallitsematonta, mikä heikentää sen mahdollisuuksia pärjätä kilpailussa muita saman alan yrityksiä vastaan. Tämän takia yritykset kehittävät itselleen laatujärjestelmän, jossa kerrotaan mm. suuntaviivat ja strategiat yrityksen laaduntarkkailulle. Lisäksi laatujärjestelmä pitää sisällään laatukäsikirjan, prosessikuvaukset, työtapakuvaukset ja viiteaineiston (Tuurala 2010). Tämä koostuu siis sekä tehtäväkohtaisista, pikkutarkoista ohjeista että yleispätevistä ohjeistuksista, joiden pohjalta voidaan tehdä tarkempia ohjeita ja parantaa edelleen tehtävien suorittamista, laadunvalvontaa ja -kehitystä. Laatujärjestelmässä kerrotaan lisäksi tavat ja kriteerit, miten laatua arvioidaan. Pelkkä laatujärjestelmän tekeminen ei vielä riitä, vaan sitä on lisäksi päivitettävä ja kehitettävä koko ajan. Vain näin tekemällä yrityksen liiketoiminnan laatu voi parantua ja yritys kehittyä.

Prosessien laatua voidaan arvioida monilla eri tavoilla, kuten auditoinneilla, asiakaspalautteella, benchmarkingilla, itsearvioinneilla, katselmoinneilla sekä erilaisilla laatumittareilla ja tuloskorteilla. Lisäksi voidaan käyttää tilastollista prosessinohjausta, joka on menetelmä prosessien suoritustason parantamiseksi

ja prosesseissa syntyvien häiriöiden ja virheiden ennaltaehkäisyyn. Prosessista systemaattisesti kerättävää tilastollista dataa analysoimalla saadaan tietoa prosessin tilasta ja näin saatua tietoa hyväksikäytetään prosessin ohjaamisessa tavoitteena nollavirhetuotanto. Tilastollisuus tarkoittaa sitä, että prosessia ei ohjata yksittäisten tapahtumien tai mittatulosten perusteella vaan keskiarvojen avulla. Tilastolliset menetelmät antavat mahdollisuuden hallita riskejä ja ennustaa tulevaisuutta. Laadunarviointiin liittyvät myös standardit, joita ovat standardisoinnista huolehtivan viranomaisen, järjestön tai muun tunnustetun elimen hyväksymät kirjallisesti dokumentoidut ohjeelliset määritelmät, mallit, tyypit tai normit; esimerkiksi ISO-standardit. Standardisointi on yhteisten suositusluonteisten sääntöjen laatimista helpottamaan viranomaisvalvontaa, kauppaa ja muuta elinkeinoelämän kanssakäymistä ja kuluttajien elämää. Standardeilla lisätään tuotteiden yhteensopivuutta ja turvallisuutta, helpotetaan kotimaista ja kansainvälistä kauppaa ja suojellaan ympäristöä. Standardeja on erilaisiin käyttötarkoituksiin, mutta laadun arvioinnissa ja laadunhallinnassa keskitytään lähinnä ISO 9000 -standardiperheen standardeihin. Lisäksi laadusta voidaan myöntää erilaisia sertifikaatteja, kun täytetään tietyt ennalta määritetyt kriteerit. Niitä voi antaa yrityksen johto, asiakas ja ulkopuolinen akkreditoitu sertifioija. (Tuurala 2010.)

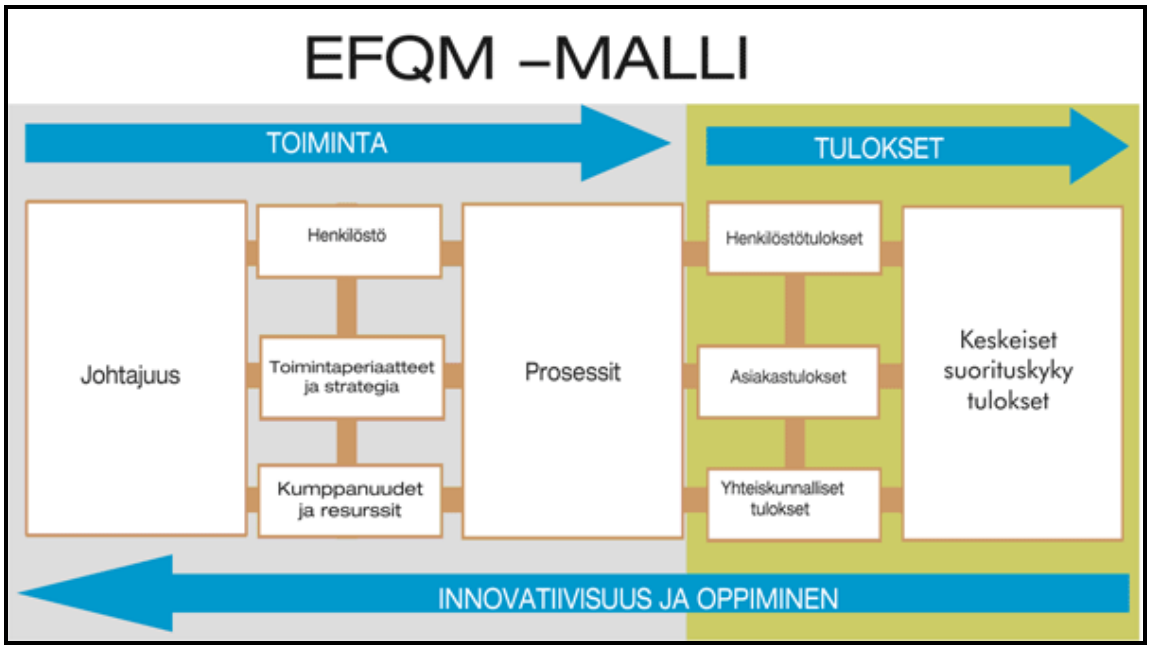
Auditointi tarkoittaa itsearviointia tai ulkopuolisen tahon suorittamaa riippumatonta systemaattista ja dokumentoitua arviointia sen selvittämiseksi, onko arvioinnin kohde asetettujen tavoitteiden ja kriteerien mukainen ja tarkoitukseen sopiva. Asiakaspalaute on asiakkaan kokemuksen selvittämistä hänen saamastaan palvelusta ja sen hyödyistä. Asiakaspalautetta voidaan hankkia suunnitelmallisesti ja kohdennetusti esimerkiksi kerran vuodessa toteutettavien kyselyjen tai haastattelujen avulla. Asiakas voi antaa palautetta myös satunnaisesti tai esimerkiksi ilmaisemalla tyytymättömyytensä muun muassa valitusten kautta. Benchmarking on taas organisaatioiden keskinäistä vertaisarviointia ja kehittämistä molempien osapuolien toiminnan parantamiseksi. Itsearvioinnissa omaa toimintaa, toimintatapoja ja kokemuksia tarkastellaan järjestelmällisesti tiettyjä arviointikriteerejä vasten (THL 2011). Katselmoinnit ovat tarkasteltavien tekijöiden soveltuvuuden, asiaankuuluvuuden ja tehokkuuden määrittämisiä johtopäätöksineen. Laatumittari on mitattavan kohteen ominaisuuden ilmaisin valitulla

suureella, kuten esimerkiksi virheellisten tuotteiden lukumäärä tietyssä valmiste-erässä. Lisäksi se on arviointitapa, joka auttaa ymmärtämään kohdetta yksiselitteisellä tavalla. Lukemalla on aina tietty epätarkkuus suhteessa mitattavan suureen todelliseen arvoon. Tuloskortti on kehityksen mittaamisen ja seurannan väline prosessin ohjausta varten. Tuloskortti on mittaristo, joka koostuu mm. erilaisista laatumittareista. (Tuurala 2010.)

Prosessien laadun arvioinnin apuna käytettävistä erilaisista työkaluista yksi on tässä opinnäytetyössä suunniteltava ja toteutettava prosessin arviointityökalu Process Assessment Tool eli PAT. Työkalu pohjautuu Philipsin rakentamaan Process Survey Tools eli PST-järjestelmään. PAT-työkalusta kerrotaan tarkemmin luvussa 6 Prosessin arviointimalliin perustuva tietojärjestelmä.

3.1 EFQM-malli

EFQM eli European Foundation for Quality Management on voittoa tuottamaton järjestö, jonka tarkoituksena on auttaa organisaatioita kehittämään liiketoimintansa laadukkuutta. Järjestö sai alkunsa 1988, kun 14 eurooppalaista suuryritystä päätti kehittää hallintatyökalun parantaakseen eurooppalaisten organisaatioiden kilpailukykyä (EFQM). Järjestö jakaa vuosittain EFQM Excellence Award -laatupalkinnon eli Euroopan laatupalkinnon eri kategorioiden laadukkaimmille organisaatioille Euroopassa. Suomessa EFQM:n partneriorganisaationa toimii Laatukeskus, joka jakaa Suomen laatupalkinnon suomalaisille organisaatioille neljässä eri kilpailusarjassa (Laatukeskus 2011). Jokaisessa sarjassa jaetaan yksi laatupalkinto, mikäli sarjassa löytyy palkinnon arvoinen organisaatio (Laatukeskus 2011). Lisäksi tuomaristo voi myöntää kunniamainintoja ja kaikki yli 400 pistettä saavuttaneet ovat oikeutettuja hankkimaan EFQM Recognised for Excellence -tunnustuksen ja käyttämään sitä markkinoinnissaan (Opetushallitus 2011). Laatupalkintojen arviointikriteerinä on EFQM:n kehittämä EFQM Excellence Model -laadukkuusmalli, joka on kehitetty organisaatioiden oman toiminnan arviointi- ja kehittämistyökaluksi (Laatukeskus 2011). EFQM-malli on tarkoitettu itsearvioinnin viitekehykseksi, eikä sitä ole pakko noudattaa orjallisesti ja se voidaan ottaa osissa tai osittain käyttöön. Tosin laatukilpailuun osallistuttaessa organisaatiot arvioidaan koko viitekehyksen mukaisesti. Kuva 3.1 ilmentää EFQM-mallin rakennetta.



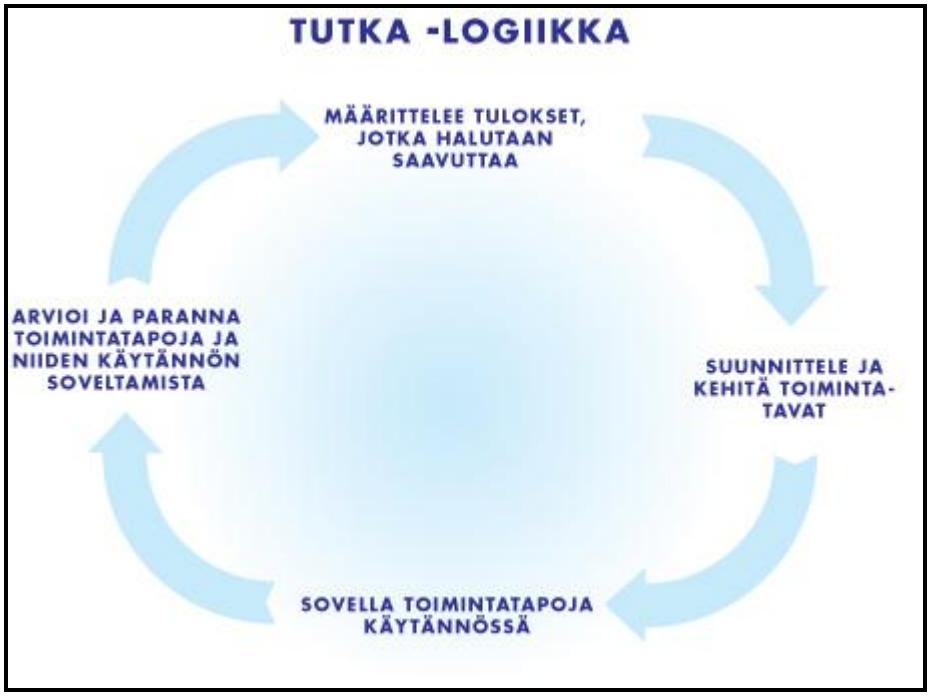
Kuva 3.1 EFQM-malli (Keto & Malinen 2007, The EFQM Excellence Model 2003 mukaan)

EFQM-malli perustuu yhdeksään arviointialueeseen, joita ovat johtajuus, henkilöstö, toimintaperiaatteet ja strategia, kumppanuudet ja resurssit, prosessit, henkilöstötulokset, asiakastulokset, yhteiskunnalliset tulokset ja keskeiset suorituskykytulokset. EFQM-mallin viisi ensimmäistä arviointialuetta tarkastelevat toimintatapojen kuvaamista. Toisin sanoen mitä tehdään ja millä tavalla? Lopuissa neljässä arviointialueessa keskitytään tulosten arvioimiseen. Tällöin joudutaan ottamaan kantaa, osoittavatko tulokset myönteistä kehitystä tai hyvää suorituskykyä, ovatko tulokset hyviä muihin verrattuna, ovatko tulokset seurausta toimintatavoista ja miten kattavia tulokset ovat. (Opetushallitus 2011, Euroopan Laatupalkintomalli julkisella sektorilla 2001 mukaan.)

Laatupalkintomallien täydellisen soveltamisen arvo kehittämistyökaluina perustuu siihen, että arviointi tuottaa yrityksen toiminnan laatutasosta pisteytyksen. Tämä puolestaan mahdollistaa vertailut eri organisaatioiden ja eri ajankohtien välillä. EFQM-mallissa arviointi tapahtuu TUTKA-logiikalla. Kun itsearviointimallia halutaan keventää, ilman pisteytystäkin saadaan esille organisaation vahvuudet ja parantamisalueet, kehittämiskohteet. Itsearviointien tavoitteena ei tarvitse olla laatupalkintokilpailuun valmistautuminen tai muu vastaava, vaan ennen kaikkea saada niin johto kuin henkilöstö omaksumaan organisaation toi-

minnan tavoitteet, havaitsemaan ja levittämään parhaita toimintamalleja ja tunnistamaan kehittämishankkeita. (Opetushallitus 2011.)

Kuten edellisessä kappaleessa todetaan, EFQM Excellence Model:n tarkoituksena on helpottaa organisaatioita itsearvioinnin avulla kehittämään toimintaansa laadukkaammaksi, myös ilman pakkoa tai tarvetta osallistua laatukilpailuun. Tämän kehittämisen tukena ja mallin keskeisenä periaatteena on organisaation eriomaisuuden saavuttaminen TUTKA-logiikan avulla (Opetushallitus 2011). Kuva 3.2 havainnollistaa tätä logiikkaa.

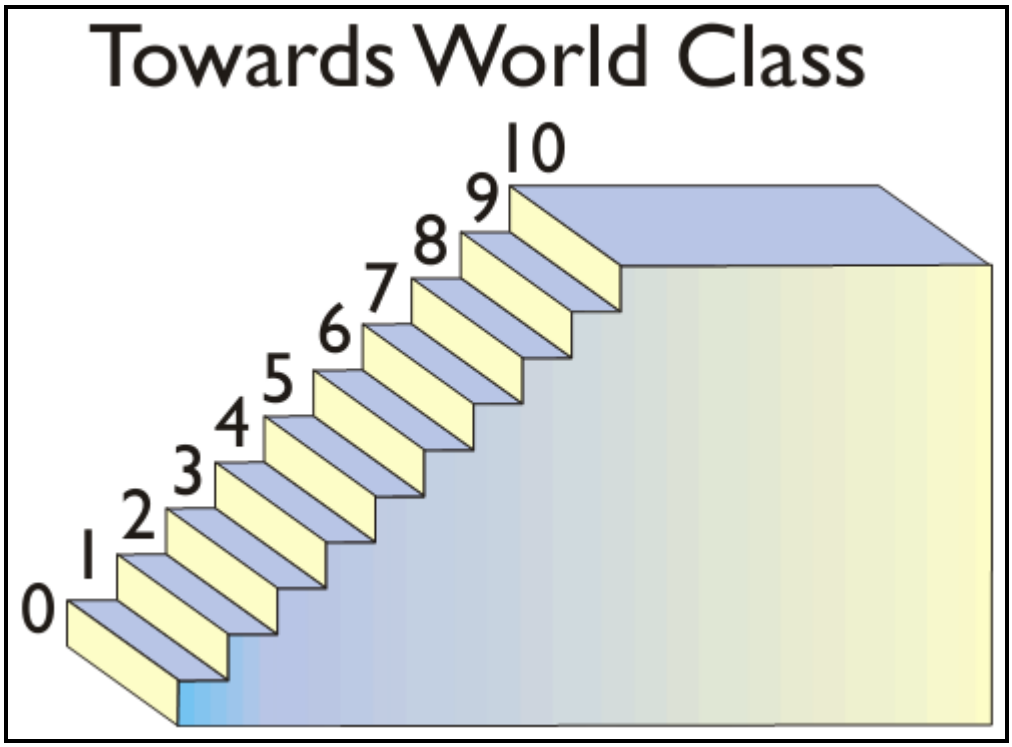


Kuva 3.2 TUTKA-logiikka (Opetushallitus 2011)

TUTKA-lyhenne tulee sanoista tulokset (TU), toimintatapa (T), käytännön soveltaminen (K) ja arviointi ja parantaminen (A). Eli aluksi määritellään saavutettavat tulokset, jonka jälkeen suunnitellaan ja kehitetään toimintatavat tavoitteiden saavuttamiseksi. Seuraavaksi sovelletaan näitä toimintatapoja käytäntöön ja lopuksi arvioidaan sekä parannetaan toimintatapoja ja niiden käytännön soveltamista. Kun aikaisemmin määritellyt tulokset on saavutettu, voidaan sykli aloittaa alusta uusien saavutettavien tuloksien määrittämisellä. Kuten kaaviosta käy ilmi, logiikalla tähdätään jatkuvan kehittämisen ja parantamisen periaatteeseen, mikä on EFQM:n toiminnan kulmakiviä.

3.2 Philipsin malli

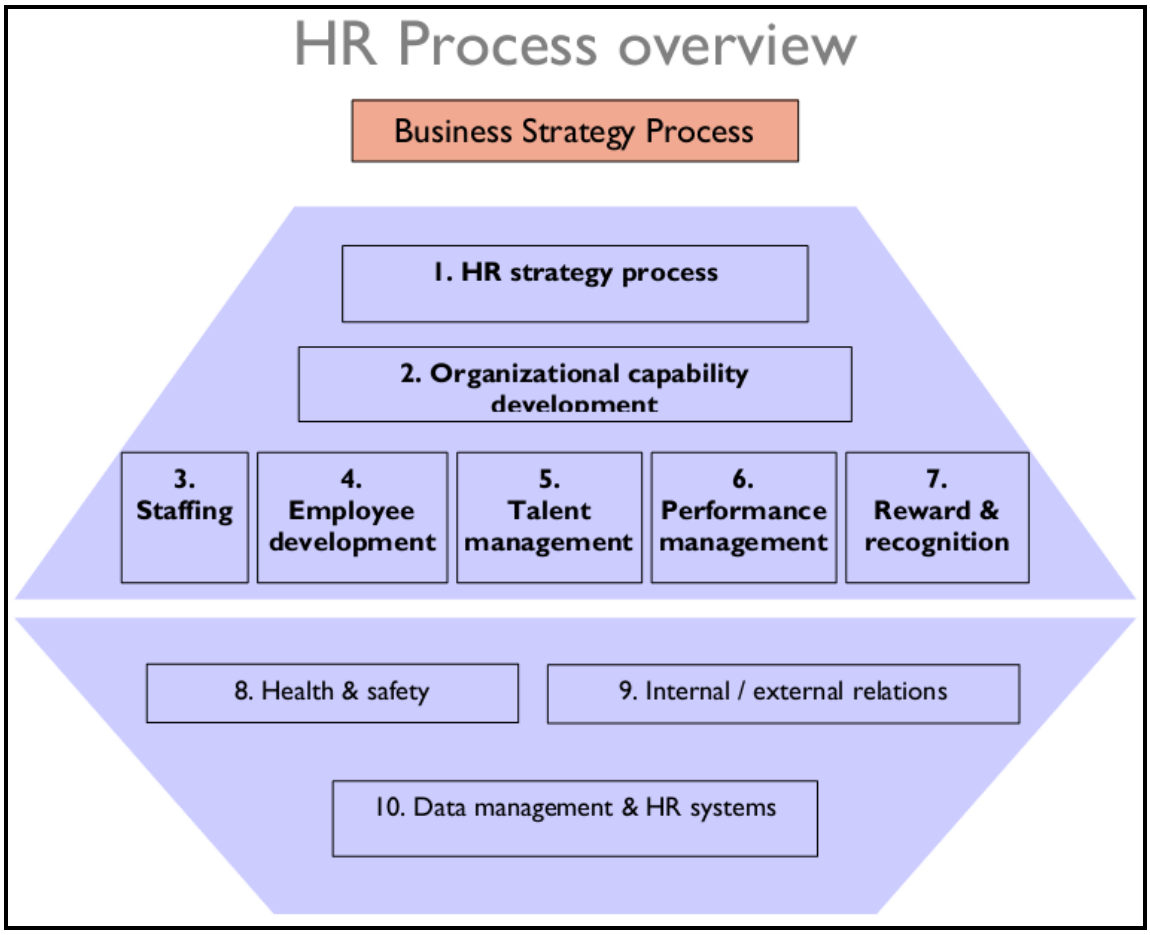
Philips-yhtiö on EFQM:n perustajajäseniä ja se on edelläkävijöitä liiketoiminnan laadukkuuden kehittämisessä (EFQM). Philips itse on perustettu alun perin Hollannissa 1891 ja nykyään se on Euroopan suurin elektroniikka-alan yhtiö, jonka liikevaihto vuonna 2010 oli 25,4 miljardia euroa ja vuoden 2010 lopussa sillä oli yli 119 000 työntekijää (Philips 2011). Philips käyttää luonnollisesti liiketoimintansa laadukkuuden kehittämiseen EFQM-mallia, mutta tämän konseptin lisäksi se on kehittänyt laadun arviointia ja tarkastelua varten konkreettisen Process Survey Tools -työkalun. Työkalu sisältää esimerkkiohjelman lisäksi dokumentaatiota prosessien arvioinnista ja EFQM-mallin käyttöönotosta organisaatiossa. Työkalun periaatteista oli lisäksi kuvallisia havainnollistuksia, joista tärkeimpiä ja olennaisimpia esitellään seuraavana.



Kuva 3.3 PST-elementtien kypsyysasteet (EFQM - Philips 2004a)

Prosessien laadukkuutta arvioidaan jakamalla prosessit elementteihin, joita jokaista arvioidaan kypsyysasteen eli tason mukaan. Tasot kulkevasta nollasta kymmeneen kymmenennen tason edustaessa maailman huippuluokkaa edustavaa elementtiä, kuten kuva 3.3 osoittaa. Elementit jaetaan prosessin pää- ja tukielementteihin, aivan kuten itse prosessienkin jako tehdään sekä tässä että EFQM-mallissa (EFQM - Philips 2004a). Prosessin jakamisesta elementteihin

toimii hyvänä esimerkkinä Philipsin mallissa oleva Human Resources (HR) eli henkilöstöprosessi (EFQM - Philips 2004b). Tästä kertoo tarkemmin kuva 3.4:



Kuva 3.4 Human Resources -prosessin yhteenveto (EFQM - Philips 2004b)

Kuten kuvasta näkyy, itse henkilöstöprosessi luokitellaan businessstrategiaprosessien joukkoon ja se koostuu kymmenestä eri elementistä, jotka jakaantuvat seitsemään pää- ja kolmeen tukielementtiin. Pääelementtejä ovat HR strategy process eli HR-strategiaprosessi, Organizational capability development eli organisaation pystyvyyskehitys, Staffing eli henkilöstönhallinta, Employee development eli työntekijäkehitys, Talent management eli kykyjenhallinta, Performance management eli suorituskyvyn hallinta sekä Reward & recognition eli palkitseminen & tunnustukset ja tukielementtejä ovat Health & safety eli terveys & turvallisuus, Internal / external relations eli sisäiset / ulkoiset suhteet sekä Data management & HR systems eli tiedon hallinta & HR-järjestelmät. Prosessien jako elementteihin helpottaa kokonaisuuden hahmottamista ja prosessien arviointia, kun prosessit pilkotaan pienemmiksi ja käsitellään osissa. Näille elemen-

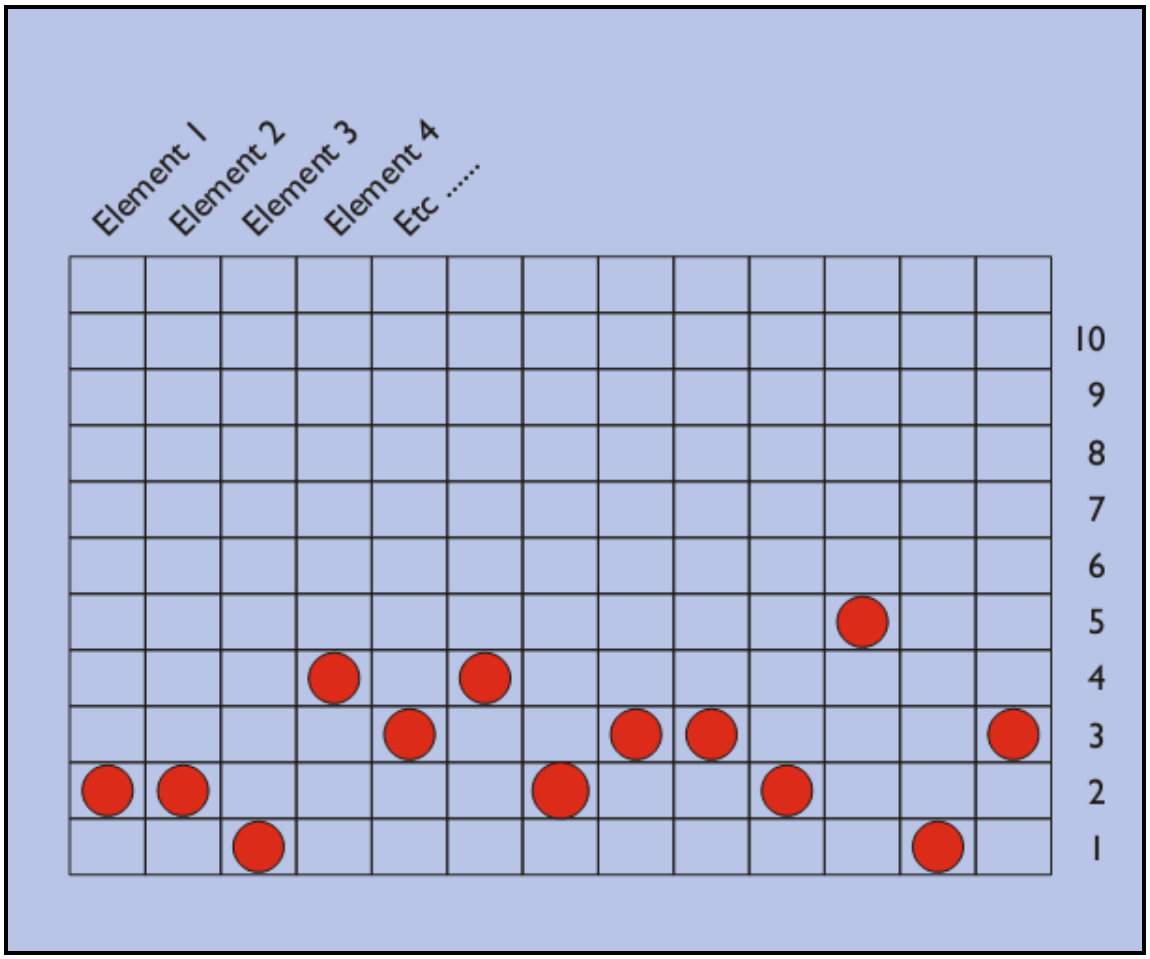
teille määritellään kypsyysasteet eli tasot karkeasti ottaen perustasoihin (0-3), keskitasoihin (4-7) ja huipputasoihin (8-10). Tasojen tarkemmat määrittelyt kulkevat taulukko 3.1:n mukaisesti. (EFQM - Philips 2004a, 2004b)

Taulukko 3.1 PST-elementtien kypsyysasteiden yleismäärittelyt

Taso(t)	Määrittelyt
0	Lähtötaso
1	Ad hoc -lähestymistapa, vähäinen dokumentointi.
2 ja 3	Jonkin verran toteutusta, dokumentointia ja käyttöönottoa.
4 ja 5	Muodollinen dokumentointi ja jonkin verran tarkastelua, benchmarking-tutkimuksia ja tulosparannusta.
6 ja 7	Järjestelmällinen toteutus, mittaaminen ja tavoitteet, selkeä tarkasteluohjelma sekä jatkuva ja selkeä parantaminen, kestävät tulokset.
8, 9 ja 10	Täysi toteutus, kattava tarkastelu, suunnitellut parannukset & tavoitteet saavutettu, johto seuraa toteutumattomia tuloksia, eteneminen kohti maailmanluokkaa.

Yllä olevat tasojen määrittelyt ovat yleisluontoiset ja toimivat määrittelypohjina kaikille prosesseille. Määrittelyissä painotetaan dokumentaation määrää ja laatua, tarkastelujen ja arvioinnin suunnitelmallisuutta ja systemaattisuutta, tavoitteiden asettamista ja saavuttamista, tulosten jatkuvaa ja kestävää parantamista sekä prosessimallin käyttöönoton eli toteutuksen kokonaisvaltaisuutta. Mitä paremmalla tasolla nämä edellä mainitut asiat ovat, sitä korkeampia ovat myös prosessien ja niiden elementtien tasot eli kypsyysasteet. Määrittelyjä voidaan tarvittaessa muuttaa ja tarkentaa prosessi- ja elementtikohtaisesti, kun halutaan ottaa prosessien erityispiirteet huomioon. Näin käytännössä tehdäänkin, että prosessien arviointi olisi mahdollisimman tarkkaa ja paikkansa pitävää. Esimerkiksi äsken mainitussa HR-prosessissa elementtien kriteereissä on hieman poikkeavat painotukset, muun muassa tasot nollasta kolmeen luokitellaan perustasoihin, tasot neljästä kuuteen luokitellaan keskitasoihin ja tasot seitsemästä kymmeneen kuuluvat huipputasoihin (EFQM - Philips 2004b). Näistä prosessikohtaisista tasomäärittelyistä johdetaan sitten jokaisen tason tarkemmat kriteerit eli tasokysymykset. Tasokysymyksiä on tavallisemmin noin kahdesta

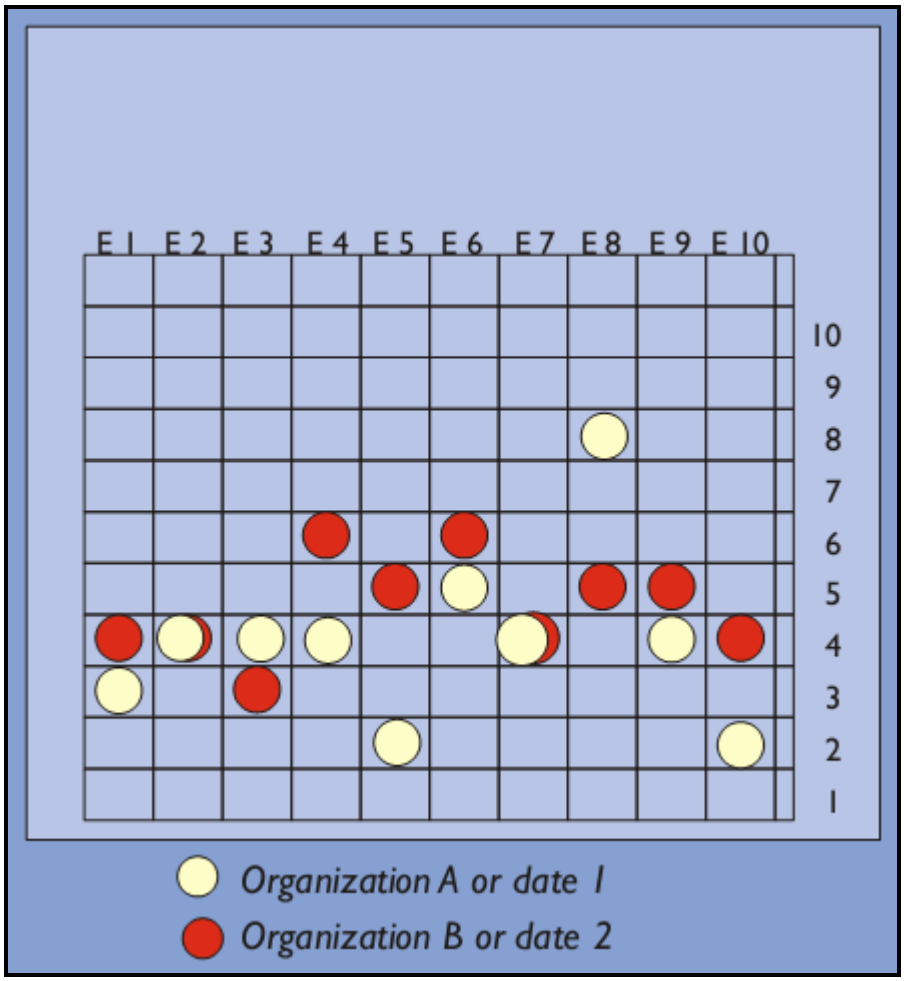
kymmeneen kappaletta tasoa kohti ja ne tulee kaikki täyttää saavuttaakseen kyseisen tason. Lisäksi kaikki aiemmat tasot tulee olla täytettyinä, jotta ylempi taso voidaan saavuttaa. Philipsin mukaan tasot neljästä viiteen ovat alimmat tyydyttävät tasot prosessien hallinnassa ja mallin käyttöönotossa. Alapuolella oleva kuva 3.5 näyttää työkalun tasojen ja kypsyysasteiden seuraamista ja arviointia varten tehdyn kypsyysasteprofiilin. (EFQM - Philips 2004a)



Kuva 3.5 PST-prosessin eri elementtien kypsyysasteprofiili (EFQM - Philips 2004a)

Kypsyysasteprofiilista nähdään pikaisella silmäyksellä prosessin eri elementtien nykytasot, jolloin prosessin laadun seuraaminen ja arviointi helpottuu. Hyvällä ja huonolla tasolla olevat elementit erottuvat joukosta, jolloin voidaan painottaa kehitysresursseja näihin huonommalla tasolla oleviin elementteihin ottaen samalla mallia parhaimmalla tasolla olevista elementeistä. Arvioinnin tueksi voidaan PST-työkalussa ottaa myös benchmarking-arviointitekniikka, jolloin prosessin elementtien tasoja verrataan joko aikaisemman päivämäärän tasoihin tai

toisen organisaation vastaavaan prosessin tasoihin, josta on esimerkkinä seuraava kuva 3.6.



Kuva 3.6 Elementtien kypsyysasteiden vertailu (EFQM - Philips 2004a)

Kypsyysasteiden erot käyvät kuvasta hyvin ilmi, jolloin aikaperusteisessa vertailussa nähdään helposti elementtien kehittyminen, paikallaan pysyminen tai jopa taantuminen ja organisaatioperusteisessa vertailussa taas nähdään eri organisaatioissa toisiaan vastaavien prosessien ja niiden elementtien väliset erot. Benchmarking koituu tällöin molempien organisaatioiden hyödyksi, kun kumpikin voi verrata ja parantaa huonommalla tasolla olevia prosessejaan ja elementtejään toisen organisaation vastaaviin ottaen oppia toisen organisaation prosesseista.

4 MENETELMÄT JA VÄLINEET

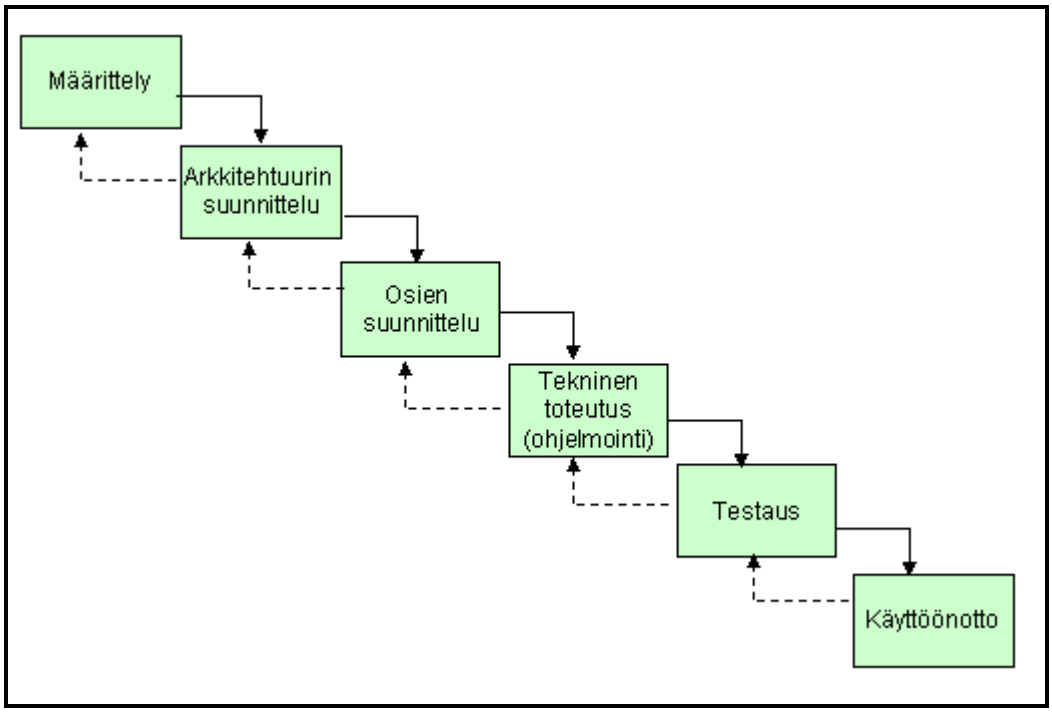
Tässä luvussa esitellään opinnäytetyössä käytetyt tietotekniset menetelmät ja välineet, mukaan lukien systeemityömalli, suunnittelumenetelmät sekä ohjelmointityökalut ja -tekniikat.

4.1 Systeemityömallit

Tietotekniikan käytön alkuaikoina tietojärjestelmät olivat verrattain pieniä ja rakenteeltaan yksinkertaisia, ja tästä johtuen systeemityökin oli melko suoraviivaista ja yksinkertaista. Ajan saatossa tietojärjestelmien koot kuitenkin koko ajan kasvoivat ja samalla monimutkaistuvat, jolloin myös niiden kehittäminen monimutkaistui ja vaikeutui, mikä aiheutti systeemityölle uusia haasteita. Tähän haettiin ratkaisuja muun muassa kehittämällä erilaisia systeemityömalleja, joiden avulla systeemityötä yritettiin helpottaa ja järkeistää sekä parantaa kehitysprojektin hallittavuutta. Eri mallit kehitettiin eri tarpeisiin, joten jokaisella mallilla on omat hyvät ja huonot puolensa. Siksi ei olekaan sellaista systeemityömallia, joka soveltuisi parhaiten kaikkiin kehitysprojekteihin, vaan jokaista projektia varten pitää valita juuri siihen tilanteeseen ja ympäristöön soveltuva systeemityömalli.

4.1.1 Vesiputousmalli eli vaihejakomalli

Vesiputousmalli on systeemityömalleista vanhin ja rakenteeltaan yksinkertaisin. Systeemityömalli perustuu siihen, että seuraavaan vaiheeseen siirrytään vasta kun edellinen vaihe on valmis. Käytännön opettamana vesiputousmallia kehitettiin edelleen lisäämällä siihen iteraatio, jossa palattiin takaisin edelliseen vaiheeseen tarkentamaan ja korjaamaan jo tehtyä työtä (Kalliala 1999). Vesiputousmalli toimii perustana myöhemmin kehitetyille, edistyneemmille systeemityömalleille. Alla olevassa kuvassa näkyy kaavio vesiputousmallista ja siitä ilmenee, miksi mallia kutsutaan kyseisellä nimellä. Systeemityöprojekti etenee mallissa ikään kuin vesi joessa pudoten vaiheesta seuraavaan.



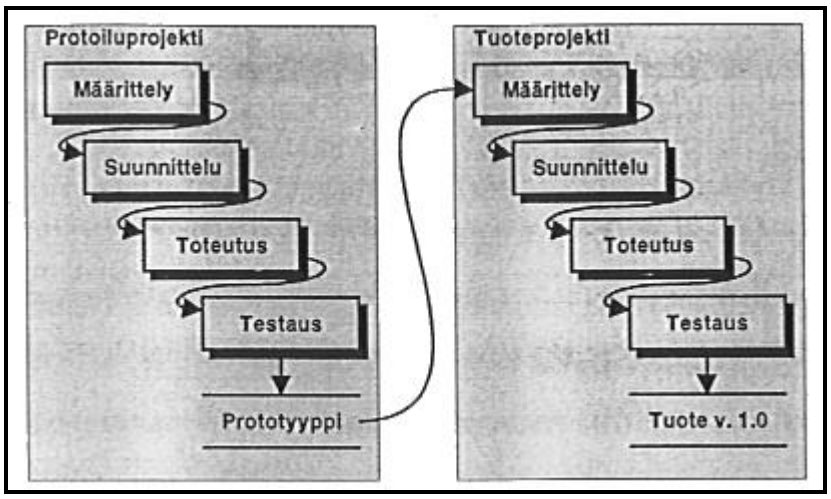
Kuva 4.1 Vesiputousmalli (Holvikivi 2005)

Kuten kuva 4.1 osoittaa, vesiputousmallissa projekti etenee varsin suoraviivaisesti vaiheesta toiseen. Mikäli seuraavassa vaiheessa huomataan virheitä tai puutteista aiemmissa vaiheissa, niin aiempaan vaiheeseen voidaan palata. Projektin seuraamisen kannalta tämä on hyvä asia, sillä selkeästä mallista nähdään melko hyvin se, missä mennään. Mallissa kuitenkin on ongelmia, mikäli alkupään vaiheessa oleva virhe tai puute havaitaan vasta loppupään vaiheessa. Tällöin korjaaminen aiheuttaa ison työmäärän jo valmiiksi tehtyyn työhön ja aikataulun viivästymisen, kun esimerkiksi muutos määrittelyssä aiheuttaa muutoksia jokaiseen sen jälkeen tulevaan vaiheeseen. Vesiputousmallia käytetäänkin lähinnä vain melko pienissä ja selkeissä projekteissa, systeemityön opettelussa sekä muiden systeemityömallien pohjana.

4.1.2 Prototyyppimalli

Prototyyppimallissa toteutettavasti järjestelmästä tehdään ensin prototyyppi, jonka perusteella tehdään vasta varsinainen järjestelmä. Uusi järjestelmä kirjoitetaan tällöin täysin uudelleen, eikä prototyyppiä käytetä muuten kuin referenssinä ja määrittelyn apuna. Määrittelyä varten tehty prototyyppi suunnitellaan kuitenkin täyttämään mahdollisimman tarkasti valmiin järjestelmän vaatimukset. Tällä tavalla toimimalla saadaan valmiista järjestelmästä karsittua pois proto-

tyyppiin jääneet viat ja puutteet, erityisesti määrittelyn ja suunnittelun osalta, joissa olevia virheitä olisi hankalinta korjata jälkikäteen. Prototyyppimallin eri vaiheet tulevat ilmi kuva 4.2:n kaaviokuvasta.



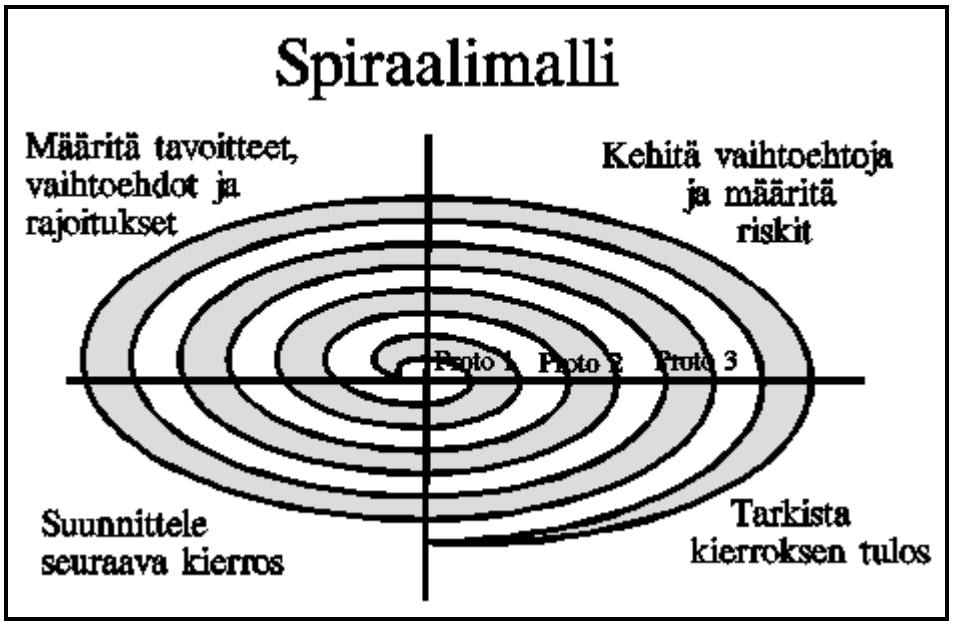
Kuva 4.2 Prototyyppimalli (Haikala & Märijärvi 2006)

Kaaviokuvasta näkyy, miten protoilumallin mukainen systeemityö etenee. Yksinkertaistettuna voidaan sanoa, että prototyyppimallissa luodaan ensin vesiputousmallia käyttäen ensin järjestelmästä prototyyppi, jonka perusteella määritellään valmis järjestelmä, joka taas tehdään loppuun vesiputousmallin avulla. Tämä prosessi muodostaa sitten kokonaisuudessaan prototyyppimallin. Mallia voitaisiinkin hyvin kuvata kaksinkertaisena vesiputousmallina. Prototyyppimallin hyviä puolia verrattuna vesiputousmalliin on virheiden ja puutteiden havaitseminen, erityisesti määrittely- ja suunnitteluvaiheissa, joissa tehdyt virheet on kalleinta ja aikaa vievintä korjata.

4.1.3 Spiraalimalli

Spiraalimallin kehittäminen sai alkunsa protoilumallista. Protoilumallissa olevat puutteet, muun muassa riskianalyysin puuttuminen olivat suurin syy spiraalimallin kehittämiseen. Spiraalimallin pääperiaatteisiin kuuluu järjestelmän kehittäminen kierroksissa. Kierros muodostuu neljästä vaiheesta: ensin määritellään kierroksen tavoitteet, vaihtoehdot ja rajoitukset. Seuraavassa vaiheessa kehitetään vaihtoehdot ja määritetään riskit. Tämän vaiheen tuloksena syntyy kyseisen kierroksen prototyyppi kehitettävästä järjestelmästä. Kolmannessa vaiheessa tarkistetaan kierroksen tulos (eli prototyyppi) ja neljännessä vaiheessa suunnitellaan seuraava kierros. Kierroksia toistetaan, kunnes prototyypeistä saadaan

kehitettyä vaatimukset täyttävä valmis järjestelmä. Spiraalimallin rakennetta ja toimintaa havainnollistaa hyvin kuva 4.3.



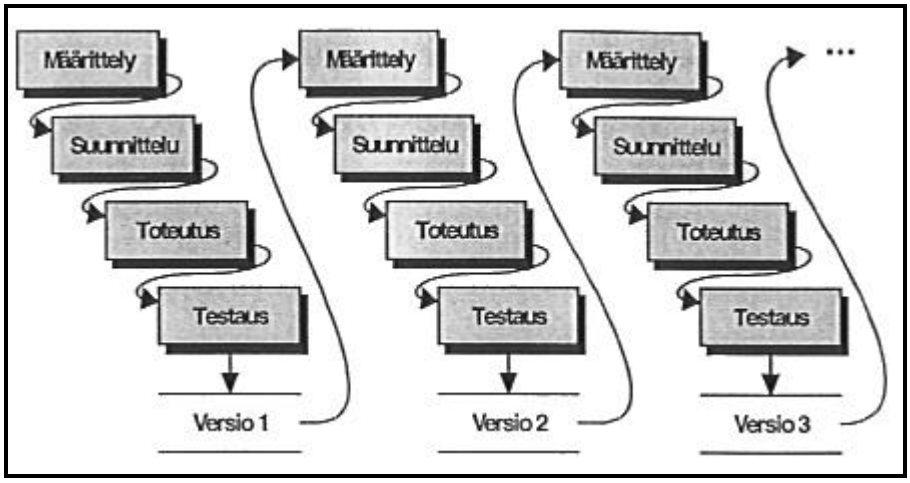
Kuva 4.3 Spiraalimalli (Kalliala 1999)

Spiraalimallin kaaviokuvassa näkyvä ristikko kuvaa eri vaiheiden välisiä rajoja ja itse spiraali eli järjestelmän kehitystyö alkaa keskeltä. Tästä ilmenee, että spiraalimallissa järjestelmän kehittäminen lähtee liikkeelle määrittelyvaiheesta. Spiraali kiertää itseään ulospäin myötäpäivään, jolloin seuraavana vaiheena tulee kehittämis- ja riskien määrittämisvaihe, jonka tuloksena syntyy ensimmäinen prototyyppi järjestelmästä. Tehty työ tarkistetaan tarkistusvaiheessa, jonka jälkeen suunnitellaan seuraava kierros suunnitteluvaiheessa. Tämän jälkeen alkaa uusi kierros, joita toistetaan kaaviossa vielä kolme kertaa. Viimeisellä kierroksella kehittämisvaiheen jälkeen ei synny enää pelkkää prototyyppiä, vaan valmis järjestelmä, joka lopuksi vielä tarkistetaan. Tällöin uuden kierroksen suunnitteluvaihettakaan ei enää tarvita, sillä kehittämistyö loppuu tähän.

4.1.4 Evoluutiomalli

Evoluutiomalli perustuu järjestelmän rakentamiseen kehitysversioina. Versioiden määrä ja niissä tehtävät ominaisuudet määritellään etukäteen projektisuunnitelmassa, mutta ominaisuuksia voidaan tarvittaessa lisätä, poistaa tai muuttaa projektin aikanakin yhteisellä sopimuksella. Evoluutio tapahtuu siten, että aluksi suunnitellaan ja toteutetaan keskeisimmät ydintoiminnot ja toiminnot, jotka ym-

märretään parhaiten. Sitten lisätään ja suunnitellaan näiden perusteella seuraavat osat seuraavaan versioon. Koska systeemityömallissa järjestelmä ”elää” koko ajan uusien versioiden myötä, alkuperäiset määrittelytkin muuttuvat koko ajan. Mallin vahvuuksia ovat mahdollisuus muuttaa järjestelmää uudessa versiossa ja kehittää järjestelmää edelleen jo rakennusvaiheessa. Heikkouksina ovat epäselvyys siitä, milloin projekti on lopulta valmis sekä se, että koska seuraava versio pohjautuu edelliseen, niin määrittely- ja suunnitteluvaiheiden virheitä ja puutteita ei voida täysin poistaa myöhemmissäkään versioissa. Alla oleva kuva 4.4 näyttää kaavion kehitystyön etenemisestä evoluutiomallin mukaisesti.

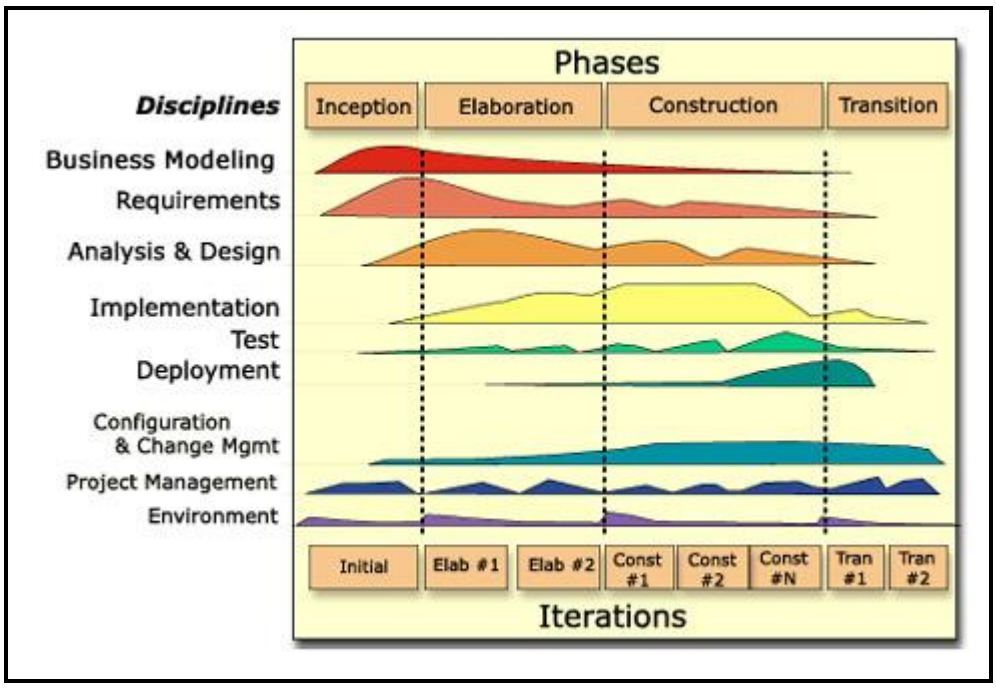


Kuva 4.4 Evoluutiomalli (Haikala & Märijärvi 2006)

Kuvan kaavio muistuttaa prototyyppimallin kaaviota, mutta siinä on yksi oleellinen ero: jokainen iteraatiokierros tuottaa prototyypin sijasta evoluutioversion järjestelmästä, johon on lisätty edellisen version jälkeisessä iteraatiossa kehitettyjä lisäominaisuuksia ja korjauksia järjestelmään. Evoluutiomalli luokitellaankin tämän takia inkrementaalisiin eli osissa toteutettaviin ja toimitettaviin malleihin. Evoluutioversioita tekemällä jokainen versio voidaan ottaa rajattuine ominaisuuksineen suoraan käyttöön ja jatkaa järjestelmän jatkokehittämistä uudessa iteraatiokierroksessa. Tällä tavalla toimimalla voidaan kehittää ja ottaa käyttöön järjestelmän keskeisimmät ydintoiminnot ja kehittää tarvittavia lisäominaisuuksia sitten hallitusti myöhemmissä versioissa. Jatkokehittäminen kuitenkin hieman hankaloituu, koska tehtävät muutokset täytyy saada yhteensopiviksi nykyisen, käytössä olevan evoluutioversion kanssa. Lisäksi evoluutiomallissa on vaara kehittää järjestelmää loputtomasti, joten valmiille järjestelmälle tulee asettaa tarkat kriteerit projektin alkuvaiheessa.

4.1.5 RUP-malli

Projektin systeemityömallin pohjana käytettiin RUP-mallia, joka on alun perin vuonna 1981 perustetun Rational Software Corporation -yrityksen kehittämä iteratiivisen ja inkrementaalisen ohjelmistokehityksen prosessikehys, koko nimeltään Rational Unified Process. IBM:n ostettua yrityksen vuonna 2003 RUP-mallia kehitettiin eteenpäin, jolloin sen koko nimikin muutettiin muotoon IBM Rational Unified Process. Koska RUP-malli ei ole itsenäinen prosessi, vaan laajennettava kehys, ei sitä voida suoraan soveltaa jokaiseen systeemityöprojektiin, vaan sitä tulee muokata kulloisenkin organisaation ja projektin erityispiirteisiin sopivaksi. RUP-mallissa toteutettavan järjestelmän toimintoja julkaistaan osissa, joten se kuuluu evoluutiomallin tapaisesti inkrementaalisiin malleihin. RUP-mallin prosessikehyksen koko rakennetta havainnollistaa kuva 4.5.



Kuva 4.5 RUP-mallin prosessikehys (Aked 2003)

Kaaviokuvan vasemmassa reunassa näkyvät projektin eri tehtävät: Business Modeling eli mallintaminen, Requirements eli vaatimusmäärittely, Analysis & Design eli analysointi ja määrittely, Implementation eli toteutus, Test eli testaus, Deployment eli käyttöönotto, Configuration & Change Management eli rakenteen ja muutosten hallinta, Project Management eli projektinhallinta ja Environment eli ympäristö. Kaavion yläreunassa näkyvät taas projektin eri vaiheet: Inception eli aloittaminen, Elaboration eli tarkentuminen, Construction eli raken-

taminen ja Transition eli siirtyminen. Alareunassa taas näkyvät eri vaiheiden iteraatiot ja niiden lukumäärät. Aika kulkee kaaviossa vasemmalta oikealle ja katkoviivat merkitsevät eri vaiheiden välistä rajaa.

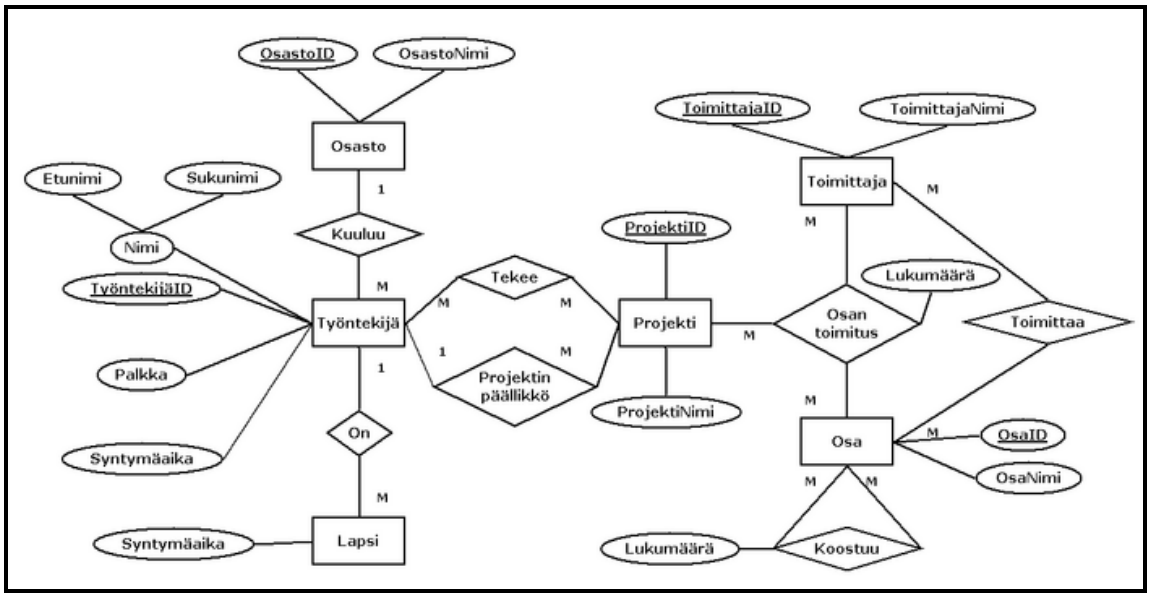
Eri värein merkatut aluekuvaajat kertovat jokaisen eri tehtävän vaatimasta työmäärästä projektin eri vaiheissa ja ajankohdissa. Kuvaajista näkyvät selkeästi eri vaiheiden ja vaiheiden sisäisten iteraatioiden väliset rajat sekä erot projektin työmäärissä. Esimerkiksi mallintaminen ja vaatimusmäärittely vaativat paljon työtä projektin aloittamis- ja tarkentumisvaiheissa ja vastaavasti käyttöönotto vaatii paljon työtä rakentamisvaiheen viimeisissä iteraatioissa ja siirtymisvaiheen ensimmäisessä iteraatiossa. Nämä seikat eivät sinänsä ole uutta tietoa, mutta näiden asioiden kokoaminen ja jäsentäminen RUP-mallin prosessikehykseen luovat systeemikehitystyöprosessille raamit ja auttavat muun muassa projektin hallinnassa, resurssien käytössä, aikataulutuksessa ja projektin ajoituksessa.

4.2 Suunnittelumenetelmät

Tietojärjestelmien suunnittelu ja toteuttaminen on luonteeltaan varsin systemaattista työtä. Suunnittelun tueksi on kehitetty erilaisia menetelmiä, työkaluja ja tekniikoita helpottamaan ja jäsentämään tätä työtä. Seuraavaksi luetellaan tässä työssä käytetyt suunnittelumenetelmät.

4.2.1 Tietokannan suunnittelu

Tavallisin tapa suunnitella relaatiotietokanta on käyttää käsitteellistä mallintamista. Siinä tietokannasta laaditaan käsitekaavio, jolla kuvataan tietokannan sisältö ja rakenne. Tunnetuin ja käytetyin on ER-malli, joka on lyhenne sanoista Entity-Relationship model. ER-mallista on esimerkkinä kuva 4.6. (Ekonoja ym 2004).



Kuva 4.6 Esimerkki ER-mallista (Ekonoja ym 2004)

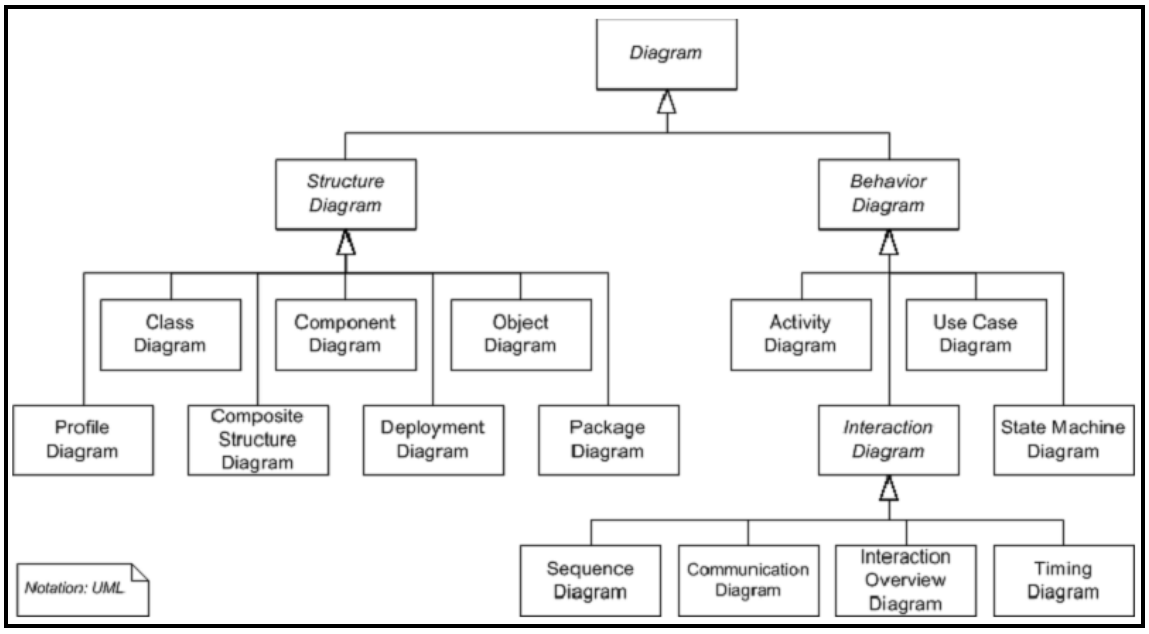
Esimerkissä suorakaiteen muotoiset kuviot ovat kohteita tai entiteettejä, jotka kuvaavat tunnistettavissa olevia asioita ja tapahtumia. Soikeat kuviot kuvaavat kohteiden ominaisuuksia. Alleviivatut ominaisuudet ovat kohteen yksilöiviä uniikkeja eli ainutkertaisia avaimia. Salmiakkikuviot kuvaavat kahden kohteen välistä suhdetta eli relaatiota. Suhteita on kolmea tyyppiä eli kardinaalisuutta: yhden suhde yhteen, yhden suhde moneen ja monen suhde moneen. Esimerkkikuvassa Osaston ja Työntekijän välillä on yhden suhde moneen -suhde eli Osastolle kuuluu monta Työntekijää ja Työntekijä kuuluu yhdelle Osastolle. Mallin mukaisten kaavioiden piirtäminen aloitetaan kohteista, joille piirretään seuraavaksi suhteet. Lopuksi suhteille merkitään kardinaalisuudet ja kohteille ominaisuudet, alleviivaten avainominaisuudet. (Ekonoja ym 2004).

Tietokannan normalisointi on tietokannan tietojen järjestämistä. Järjestämisellä poistetaan toistuvan tiedon tallennusta tietokannasta, varmistetaan tietokannan eheys, mahdollistetaan tiedon dynaaminen tallennus tietokantaan ja tehdään tietokannasta skaalautuvampi. Normalisointia on viittä eri muotoa, joista käytännössä käytetään vain kolmea ensimmäistä muotoa. Ensimmäisessä normaalimuodossa tauluilla on uniikit perusavaimet ja tietokannan taulujen moniarvoisten sarakkeiden arvot siirretään omiin tauluihinsa, esimerkiksi opiskelijataulussa olevasta kurssit-sarakkeesta tehtäisiin kurssi-taulu, ja taulut liitettäisiin perusavain-viiteavainsuhteilla toisiinsa. Toisessa normaalimuodossa tietokannan tulee olla ensimmäisessä normaalimuodossa sekä taulun sarakkeiden pe-

rus- ja viiteavaimiin kuulumattomat sarakkeet ovat täysin riippuvaisia taulun perusavaimesta. Äskeisen opiskelija-tauluesimerkin tapauksessa opiskelijan osoite riippuu täysin opiskelijan perusavaimesta, jolloin osoite jätettäisiin opiskelija-tauluun. Kolmannessa normaalimuodossa tietokannan tulee olla toisessa normaalimuodossa sekä mikään taulun avaimiin kuulumaton sarake ei saa olla riippuvainen toisesta avaimiin kuulumattomasta sarakkeesta. Esimerkkinä opiskelija-taulun osoitetiedoista postitoimipaikka riippuu postinumerosta. Tällöin pitäisi luoda postitoimipaikka-taulu, jonka perusavain postinumero olisi. Seuraavien normaalimuotojen käyttäminen hidastaisi yleensä liikaa tietokannasta tehtäviä hakuja relaatiomäärän kasvaessa ja sekä saattaisi ylittää tietokantapalvelimen muistin kapasiteetit. Joskus tietokannan kolmannesta normaalimuodostakin joudutaan denormalisoimaan alempien muotojen koostetauluja suorituskyvyn takia. Yleensä kuitenkin kolmas normaalimuoto takaa riittävän hyvän tietokannan rakenteen, varsinkin jos tietokannassa on tarve tietojen jatkuvalle päivittämiselle. (Ekonoja ym 2004).

4.2.2 UML

Unified Modeling Language on oliosuuntautunut mallinnusmenetelmä, jonka kehittivät 1990-luvun loppupuolella usean eri mallinnuskielten kehittäjät yhteistyönään. UML:n perustana ovat BOOCH-, OOSE- ja OMT-menetelmät. Tavoitteena oli tehdä yksi mallinnuskieli, joka olisi toisaalta riittävän laaja vastaamaan mahdollisimman monien kehittäjien tarpeisiin, mutta riittävän yksinkertainen ollakseen ymmärrettävä. UML pohjautuu vahvasti kaavioihin, joita on UML:n versiossa 2.2 yhteensä neljätoista erilaista, tietyn tarkoituksen mukaista tyyppiä. Kuva 4.7 havainnollistaa kaavioiden luokkien rakenteen. (Pitkänen 2011, Booch, Jacobson & Rumbaugh 1999 mukaan).



Kuva 4.7 Luokkakaavio UML-kaavioista (Pitkänen 2011, Rissanen 2010 mukaan)

Kaaviotyyppejä ovat rakennekaaviot, käyttäytymiskaaviot ja käyttäytymiskaavioihin kuuluvat vuorovaikutuskaaviot. Rakennekaavioihin kuuluvat luokkakaaviot, komponenttikaaviot, oliokaaviot, koostekaaviot, pakkauskaaviot, sijoittelukaaviot ja profiilikaaviot. Vuorovaikutuskaavioita ovat ajoituskaaviot, kokoavat vuorovaikutuskaaviot, kommunikointikaaviot sekä sekvenssikaaviot. Käyttäytymiskaavioihin kuuluvat taas aktiviteettikaaviot, käyttötapauskaaviot ja tilakaaviot. (Pitkänen 2011, Fowler & Scott 2002; Erikkson & Penker 2000 mukaan).

4.3 Ohjelmointityökalut ja -tekniikat

Tässä luetellaan työssä käytetyt ohjelmointitekniikat ja -työkalut.

4.3.1 .NET Framework 2.0

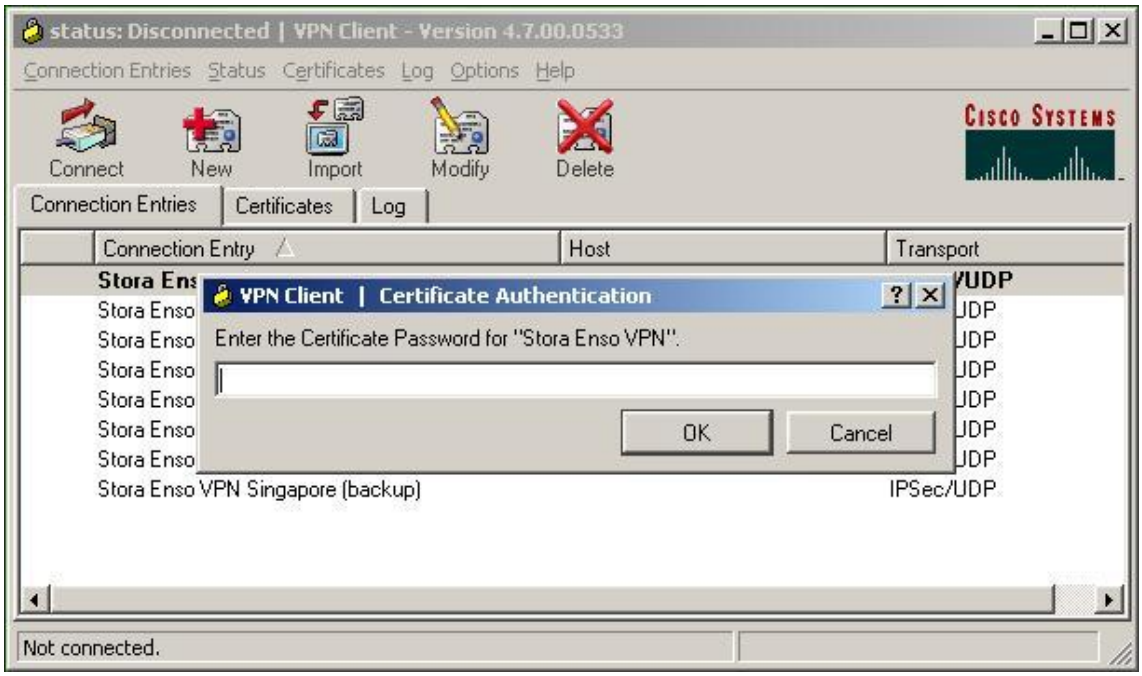
.NET Framework 2.0 on Microsoftin kehittämä ohjelmistokomponenttikirjasto, jota Microsoftin Visual Studio .NET -ympäristössä kehitetyt ohjelmistot käyttävät. Komponenttikirjasto sisältää sovellusten skaalattavuutta ja suorituskykyä parantavia ominaisuuksia, joita ovat tehostettu välimuistin käyttö, sovellusten käyttöönotto ja päivittäminen ClickOncen avulla ja laaja erilaisten selainten sekä laitteiden tuki ASP.NET 2.0 -ohjausobjektien ja -palvelujen avulla. (Microsoft 2011).

4.3.2 ASP.NET

ASP (Active Server Pages) on Microsoftin kehittämä dynaamisten Web-sivujen luomiseen tarkoitettu palvelinpuolen ohjelmointimenetelmä. Alkuperäisen ASP-menetelmän, joka julkaistiin IIS-palvelimen lisäosana, korvasi myöhemmin julkaistu ASP.NET. Uusittu ASP.NET tuli .NET Framework 2.0:n mukana ja se sisältää useita parannuksia vanhaan ASP-malliin verrattuna, muun muassa paremman suorituskyvyn ja skaalattavuuden, tuen palvelinfarmeille, Web-lomakkeet, uusitun infrastruktuurin sekä se tukee seuraavia ohjelmointikieliä: Jscript.NET, Visual Basic .NET ja C# (KK Mediat 2011). Sovelluskehittimistä muun muassa Visual Studio 2005 käyttää tätä tekniikkaa ja mainittuja ohjelmointikieliä. ASP.NET:in Web-sovellusten käyttämiseen tarvitaan palvelinohjelmisto, kuten Microsoftin IIS-palvelin. Sivujen dynaamisuus toimii siten, että asiakaskoneelta palvelimelle tulevat sivunlatauspyynnöt käsitellään palvelimella, joka laittaa .NET Frameworkin esiprosessoimaan pyydetyn sivun. Sivusta muodostetaan HTML-koodattu sivu, joka lähetetään asiakaskoneelle. Kaikki ASP-sivujen tiedon prosessointi tapahtuu siis palvelimella, ellei sivu sisällä lisäksi asiakaskoneella suoritettavaa JavaScriptiä.

4.3.3 Cisco Systems VPN Client

Cisco Systems VPN Client on Cisco Systems -yhtiön VPN-yhteyksien muodostamista varten kehittämä ohjelma. Ohjelman avulla voidaan liittää yksittäisiä työasemia yrityksen verkkoon julkisen verkon yli. Yksityisyys ja tietoturva turvataan yhteyden salauksen avulla, joka toteutetaan IPsec-tunnelointiprotokollalla. Yrityksen verkossa on vastaava Ciscon VPN-palvelinohjelmisto, joka tunnistaa VPN-käyttäjät Client-ohjelmaan tallennetun sertifikaatin ja yhteyttä muodostettaessa annettavan salasanan avulla. Kuva 4.8 on kuvakaappaus Client-ohjelman pääikkunasta.



Kuva 4.8 Cisco Systems VPN Client -ohjelman pääikkuna

Ohjelmaan on esisyötetty Host-osoitteet, joita tarvitaan VPN-yhteyden muodostamiseen. Kuvassa ollaan muodostamassa yhteyttä Stora Enso VPN -palvelimeen ja kysytään sitä varten sertifikaatin salasanaa.

4.3.4 CSS

CSS (Cascading Style Sheets) on W3C:n kehittämä ja ylläpitämä tyyliohjejärjestelmien laji, joka sisältää periaatteen useiden eri lähteistä peräisin olevien tyyliohjeiden soveltamisen samanaikaisesti tarkoin määriteltyjen preferenssisääntöjen mukaan. CSS1 (Cascading Style Sheet Level 1) on W3C:n kehittämän tyyliohjejärjestelmän ensimmäinen kehitysversio, joka julkaistiin vuonna 1996 HTML:ää varten. CSS2 (Cascading Style Sheet Level 2) on vuonna 1998 julkaistu CSS1 version tyyliohje määritelmiä laajentava toinen kehitysversio. CSS:ää käytetään muun muassa Web-sivujen esitystavan ja ulkoasun muokkaamiseen. (Paajanen 2011).

4.3.5 HTML

HTML (HyperText Markup Language) on dokumentin muodon määrittelykieli. HTML on yksinkertainen ja helposti opittava strukturoidun eli rakenteisen tekstin merkkauskieli. Dokumentin kirjoittaja voi määritellä tekstin korostukset, luetteloinnit, taulukot, linkit eli siirtymät muihin dokumentteihin tai dokumentin osiin.

HTML-kieli mahdollistaa viittaukset muihin verkon resursseihin kuten uutisryhmiin ja FTP-palvelimiin. Dokumentteihin voidaan lisätä kuvia ja ääntä. Huomioitavaa HTML-kielessä on se, ettei se ole varsinainen ohjelmointikieli. (Paajanen 2011, Perustietoa HTML:stä mukaan).

4.3.6 HTTP-protokolla

HTTP-protokollaa (Hypertext Transfer Protocol) käytetään tiedonsiirtoon esimerkiksi WWW-selaimen ja WWW-palvelimen välillä. HTTP-protokolla sijaitsee OSI-mallin sovelluskerroksella. HTTP on geneerinen tilaton protokolla, jota voidaan käyttää moniin muihinkin käyttötarkoituksiin kuin hypertekstin jakeluun, kuten nimipalvelimiin ja hajautettujen olioiden hallintajärjestelmiin. Kyseisiä toimintoja voidaan toteuttaa HTTP:n tarjoamien metodien, vikakoodien ja otsikkotietojen avulla. HTTP:n tehtävänä on tyypittää ja neuvotella tiedon esitystapa sekä mahdollistaa päälle rakennettavien itsenäisten järjestelmien tiedonsiirto ja toiminta. HTTP on ollut laajalti käytössä 1990-luvulta lähtien, ja sen uusin standardi HTTP/1.1 on peräisin vuodelta 1999. (Paajanen 2011).

4.3.7 Internet Explorer 6.0 & 7.0

Internet Explorer on Microsoftin kehittämä Web-selain, joka on ollut jo vuosia maailman käytetyin selain, varsinkin yritysmaailmassa Microsoft-tekniikoiden kytköksien takia. Selaimella voi muun muassa katsella erilaisia verkkosivuja, kuten HTML- ja ASP-sivuja. Koska toteutettava tietojärjestelmä tehdään ASP.NET:iä hyödyntävänä Web-sovelluksena, ja sitä katsellaan Stora Enso Metsän portaalista kyseisen selaimen 6.0-versiolla, oli tärkeää testata järjestelmän toimivuus tällä selaimella. Lisäksi järjestelmä rakennettiin yhteensopivaksi projektin aloitusajan tienoilla julkaistun Internet Explorerin 7.0-versioon, tulevaisuuden käyttöönottoa varten.

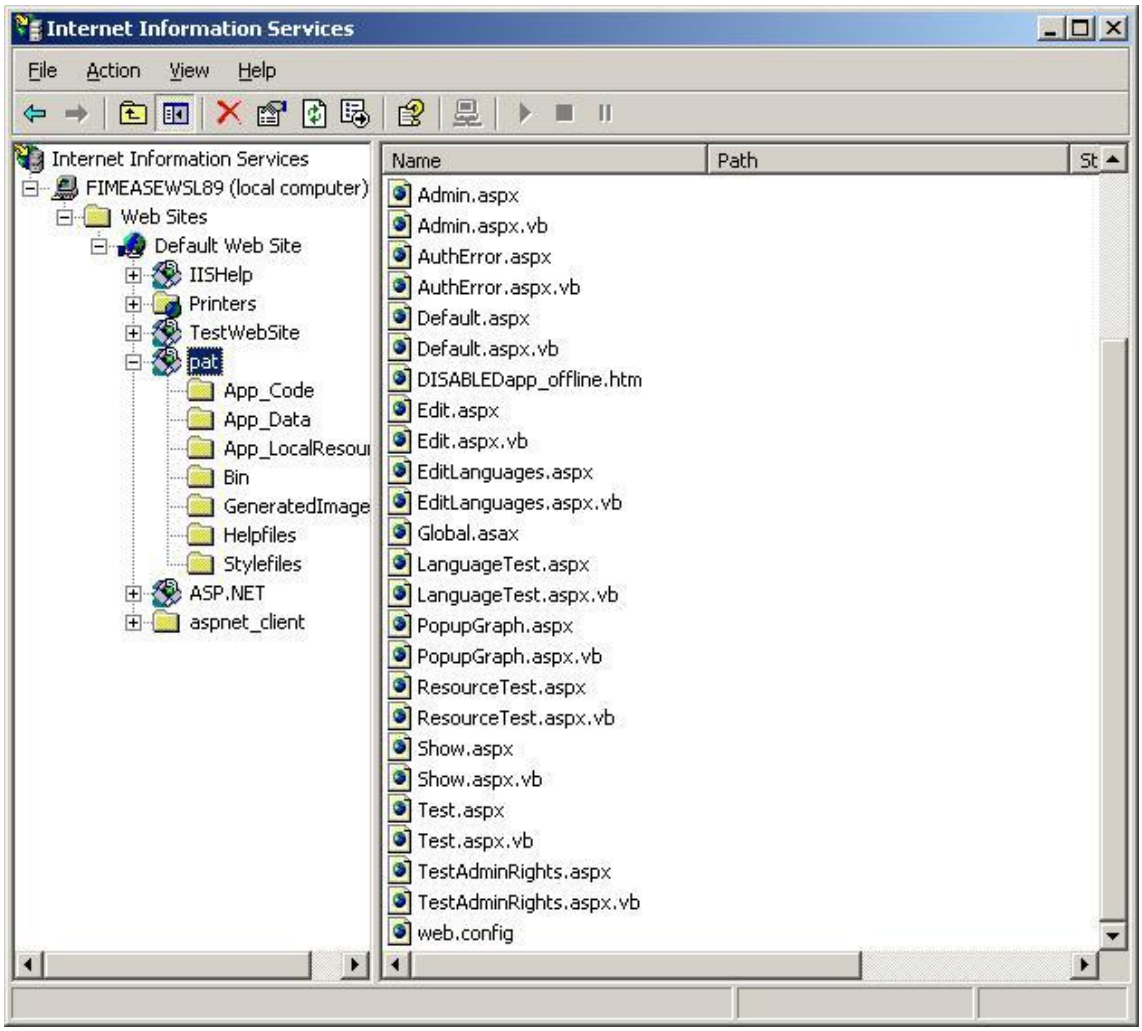
4.3.8 JavaScript

JavaScript, jonka virallinen nimi on ECMAScript, on Netscape Communications Corporationin kehittämä, suoraan selaimessa suoritettava komentosarjakieli, jonka pääasiallinen tarkoitus on lisätä Web-sivujen dynaamista toiminnallisuutta (Pitkänen 2011). JavaScriptin avulla voidaan esimerkiksi toteuttaa asiakasym-

päristössä tapahtuvat lomakkeen tietojen esitarkastaminen, hiireen reagoivat linkkipainikkeet ja popup-ikkunat.

4.3.9 Microsoft IIS 6.0

IIS (Internet Information Services) Microsoft-yhtiön Web-sivujen ja -sovellusten hallinnointia varten kehittämä palvelinohjelmisto Windows-pohjaisia palvelimia varten. IIS tukee useita verkkoprotokollia, mutta HTTP on käytössä yleisin. IIS-palvelimelle voidaan asettaa Web-sovelluskohtaisia sivuohjauksia esimerkiksi käyttäjähallinnointia ja virheilmoituksia varten. IIS-palvelimelle voidaan myös asentaa .NET Framework, jolloin se tukee myös ASP-sovellusten hallinnointia ja näyttämistä. .NET Framework tulee kuitenkin ottaa IIS-palvelimen hallinta-paneelin kautta erikseen käyttöön jokaiselle sitä tarvitsevalle Web-sovellukselle. IIS-palvelimen versio 6.0 tukee lisäksi usean Web-sovelluksen hallinnointia sovelluskohtaisissa pooleissa, erilaisia käyttäjän todennusmetodeja, tietoliikenneyhteyden SSL-salausta ja se tarjoaa paremman tietoturvan palvelimen edellisiin versioihin verrattuna. Kuva 4.9 on ruutukaappaus IIS-palvelimen hallinta-paneelista.



Kuva 4.9 IIS 6.0 -palvelimen hallintapaneeli

Ruutukaappauksesta näkyy, miten eri Web-sovellukset sijaitsevat palvelimen tiedostohierarkiassa ja millainen on ASP.NET:iä hyödyntävän Web-sovelluksen kansiohierarkia.

4.3.10 Microsoft SQL Server 2005

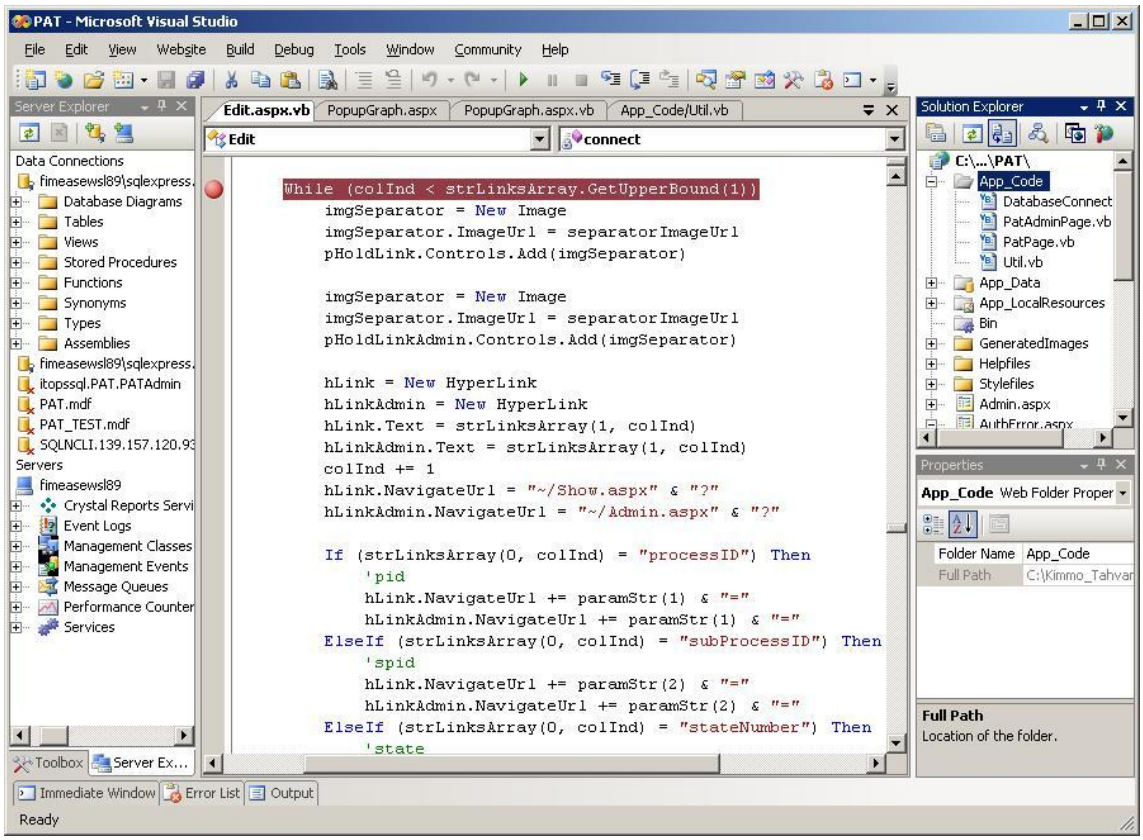
SQL Server 2005 on Microsoftin kehittämä suljetun lähdekoodin relaatiotietokantapalvelinohjelmisto. Ohjelmiston päätarkoitus on tiedon tallennus- ja hakupalveluiden tarjoaminen joko samassa fyysisessä tietokoneessa oleville tai toisaalla verkossa oleville sovelluksille. Palvelin tukee nimensä mukaisesti standardoituja T-SQL (Transact-SQL) ja SQL (Structured Query Language) -kyselykieliä sekä XML-muotoista dataa. Palvelin tukee myös transaktioita ja virheiden hallintaa kyselyissä ja käskyissä.

4.3.11 Microsoft SQL Server Express

SQL Server Express on myös Microsoftin kehittämä relaatiotietokantapalvelinohjelmisto. Vaikka sekin on julkaistu suljetun lähdekoodin ohjelmistona, se on kuitenkin vapaasti kaikkien saatavilla. Express-versio on kevyempi ja karsitumpi kuin varsinainen SQL Server sekä siihen on asetettu muun muassa tietokannan koko- ja laitteistorajoitteita, mutta päätoiminnoiltaan se on varsin samanlainen kuin raskaampaan yrityskäyttöön tarkoitettu isoveliohjelmistonsa. SQL Server Express soveltuukin hyvin tietokantojen ja palvelimen hallinnan opetteluun ja kevyisiin käyttöympäristöihin. Lisäksi se on helppo asentaa ja käyttää vähemmilläkin resursseilla varustettuihin sovelluskehitystietokoneisiin esimerkiksi sovellusten paikallista kehittämistä ja testausta varten.

4.3.12 Microsoft Visual Studio 2005 Professional

Visual Studio 2005 Professional on Microsoft-yhtiön sovelluskehitinympäristö, joka tukee .NET Framework 2.0:aa ja sen ohjelmointikieliä, kuten Visual Basic .NET, C++, C# ja J# ja siten myös CLR-virtuaalikoneen tekemää ajonaikaista koodin kääntämistä. Lisäksi se tukee paljon erilaisia tekniikoita, muun muassa Windows-ohjelmistojen ja ASP.NET-sovellusten kehittämistä. Visual Studion avulla graafisten käyttöliittymien luonti on erittäin helppoa WYSIWYG-editoriluonteen ja sovelluskehittimestä löytyvien lukuisten valmiiden komponenttien ansiosta, mikä nopeuttaa Visual Studion käytön opettelemista sekä sovellusten kehittämistä. Visual Studio 2005 -versiolla luotujen ohjelmien suorittamiseen tarvitaan vähintään 2.0-version ajonaikainen .NET Framework -kirjasto. Visual Studion myöhemmissä versioissa tuetaan .NET Frameworkia versionumeroon 4.5 asti, mutta niillä voi tuottaa myös sitä aiempien versioita käyttäviä sovelluksia. Kuva 4.10 näyttää ruutukaappauksen Visual Studio 2005 -sovelluskehittimen pääikkunanäkymästä.



Kuva 4.10 Visual Studio 2005 Professional -sovelluskehittimen pääikkuna

Ikkunan yläreunassa on ohjelman valikot ja heti niiden alapuolella tiedostojen muokkaus- ja koodin kääntämistyökalut. Ikkunan vasemmassa reunassa näkyy tietokantojen sekä niiden yhteyksien hallintatyökalut ja oikeassa reunassa ylhäältä kehitettävän sovelluksen kansiohierarkia sekä sen alapuolella käsittelyssä olevan tiedoston ominaisuudet. Käsiteltävä tiedosto, tässä tapauksessa ASP-sivun kooditiedosto, näkyy ikkunan keskellä ja aivan ikkunan alareunassa näkyvät koodin kääntämisen ja sovelluksen ajonaikaisten viestien ilmoitusikkunat.

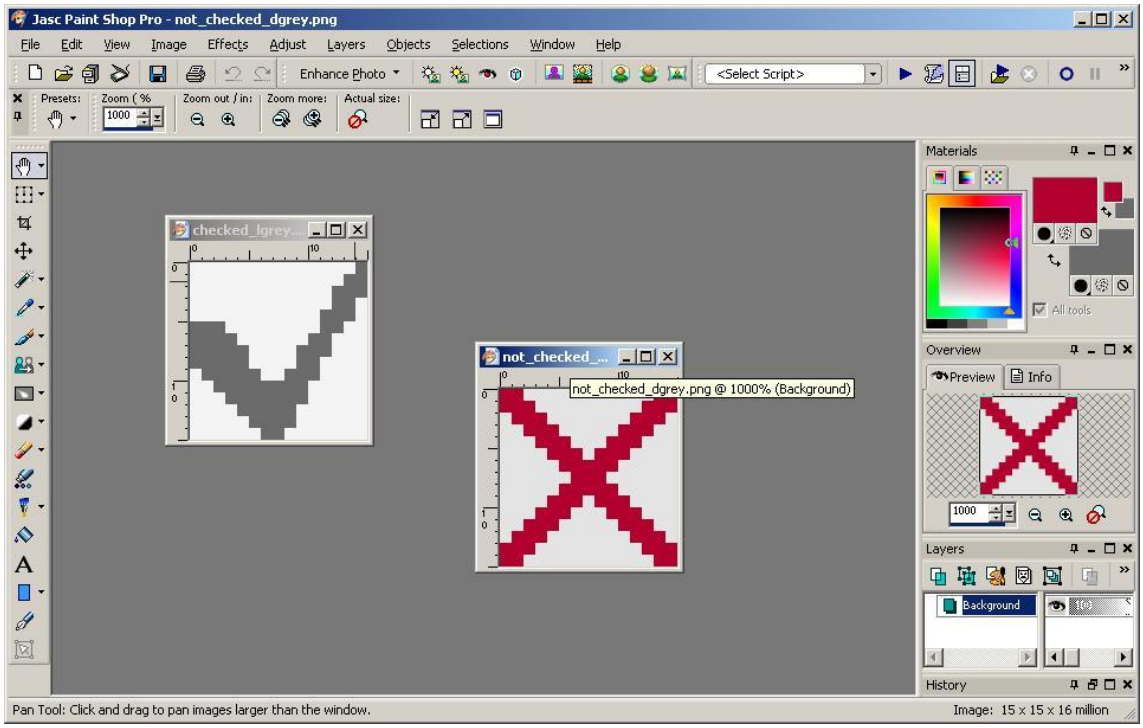
4.3.13 Mozilla Firefox 2.0

Mozilla Firefox on Mozilla-projektin tekemä vapaa, avoimen lähdekoodin Web-selain. Selain oli projektin aloitusajan tienoilla saavuttanut Suomessa verrattain paljon suosiota, joten toteutettava Web-sovellus rakennettiin yhteensopivaksi myös tämän selaimen kanssa. Tämä ei olisi ollut välttämätöntä tietojärjestelmän ja loppukäyttäjien takia, mutta koska Internet Explorer -selaimen eri versioita varten piti jo rakentaa yhteensopivuus, ei yhteensopivuuden rakentaminen Fire-

foxia varten ollut enää suuri lisätyö. Lisäksi se toimi Web-developerin työhön liittyvänä harjoituksena.

4.3.14 Paint Shop Pro 9

Paint Shop Pro on alun perin Jasc Software -yhtiön julkaisema kuvankäsittely-ohjelma. Sillä voitiin aluksi tehdä ja muokata pelkästään rasterikuvia, mutta myöhemmin ohjelmaan tuli tuki myös vektorigrafiikalle. Ohjelmalla on kätevä luoda ja muokata Web-sovellukseen ja sen ulkoasuun olennaisesti kuuluvia kuvia. Seuraavassa kuva 4.11, jossa sisältää ruutukaappauksen ohjelman pääikkunasta.

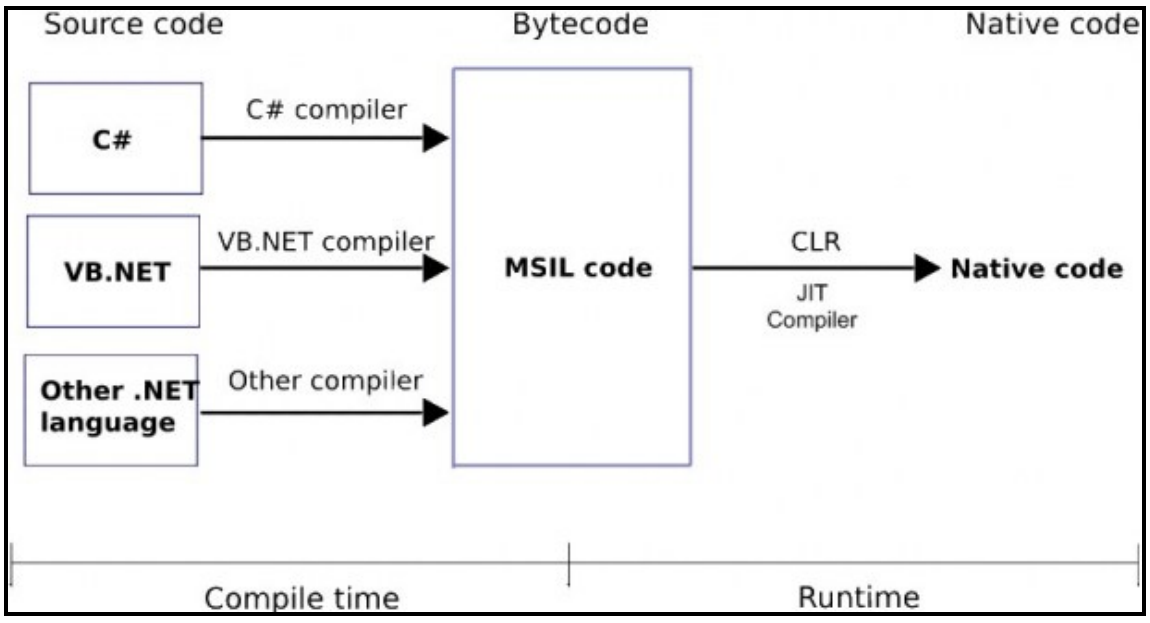


Kuva 4.11 Paint Shop Pro 9 -kuvankäsittelyohjelman pääikkuna

Aivan ohjelmaikkunan yläreunassa ovat ohjelman valikot, joiden alapuolella ja ikkunan vasemmassa reunassa on kuvien tarkastelu-, piirto- ja muokkaustyökalut. Ikkunan oikeassa reunassa on ylhäältä alaspäin lueteltuina värien ja kuvioiden hallintatyökalu, kuvien esikatselutyökalu sekä kuvatasojen hallintatyökalu ja ikkunan keskeltä löytyvät käsittelyn alla olevat kuvatiedostot.

4.3.15 Visual Basic .NET

Visual Basic .NET (VB.NET) on yksi Visual Studio .NET -sovelluskehittimen tukemista ohjelmointikielistä, joka laajensi aiemmin käytetyn Visual Basic -ohjelmointikielen käytön tukemaan CLR-virtuaalikonetta (Common Language Runtime). Vaikka kielen syntaksi pysyi samankaltaisena kuin aiemminkin, tämä tekniikka mahdollisti VB.NET-kielisen lähdekoodin esikääntämisen CLR-virtuaalikoneen ymmärtämälle CIL-välikielelle (Common Intermediate Language), josta virtuaalikone sitten kääntää ajonaikaisella JIT-käännöksellä (just-in-time) välikielisen koodin alkuperäiskoodiksi, joka taas Windows-ympäristön tapauksessa on C/C++-koodia. Vaikka tämä monimutkaistaa koodin kääntämisoperaatioita, se mahdollistaa lähdekoodille paremman muistinhallinnan, tietotyyppien turvallisen käsittelyn sekä poikkeuksien hallinnan. Lisäksi CIL-välikielen ansiosta .NET-sovellukset voidaan periaatteessa koodata millä tahansa .NET Frameworkin tukemalla ohjelmointikielellä. Kuva 4.12 auttaa hahmottamaan kääntämisoperaatioita.



Kuva 4.12 .NET-lähdekoodin kääntäminen CLR-virtuaalikoneen avulla (Taggart 2008)

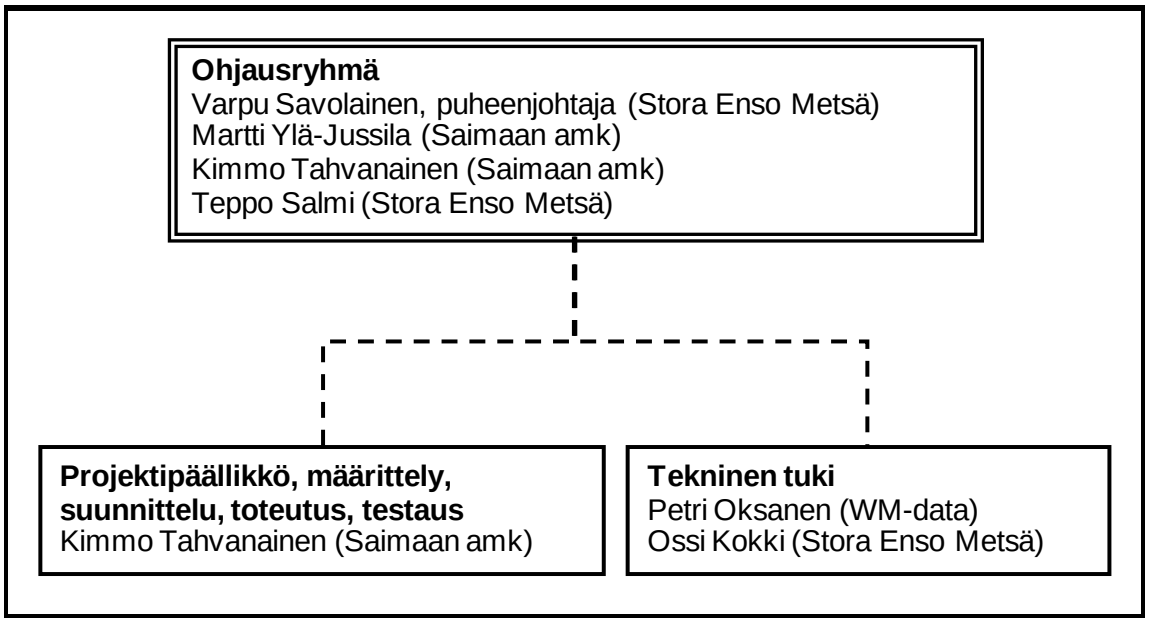
Kuvassa nähdään, miten lähdekoodit käännetään ensin kunkin .NET-kielen omalla kääntäjällä CIL-välikielelle, josta CLR-virtuaalikone kääntää sen ajonaikaisesti alkuperäiskoodiksi.

5 KEHITYSPROJEKTIN KULKU

Tässä luvussa on kerrottu Prosessin arviointityökalu -projektin organisaatio ja eri työvaiheiden kulku.

5.1 Projektiorganisaatio ja projektin ohjaus

Kuva 5.1 sisältää Prosessin arviointityökalu -projektin organisaation ja henkilöstön.



Kuva 5.1 Projektin organisaatiokaavio

Projektia ohjattiin laaditun projektisuunnitelman mukaisesti kuukausittain pidetyissä ohjausryhmän kokouksissa Stora Enson Metsän Imatran metsäkonttorilla. Lisäksi tarvittaessa pidettiin lisäpalavereja, kun projekti sitä vaati, esimerkiksi jotakin ongelmaa ratkottaessa tai projektiin liittyvistä asioista keskusteltaessa. Kokouksissa tarkasteltiin työn edistymistä ja suunniteltiin tulevia töitä sekä tarkastettiin projektin aikataulua. Projektipäällikkö teki kokouksista muistion ja työn edistymisestä kuukausittaisia raportteja. Lisäksi merkittävien työvaiheiden valmistuessa pidettiin katselmuksia. Katselmukset ajoitettiin yleensä ohjausryhmän kokousten yhteyteen ja siellä saattoi olla läsnä muitakin kuin ohjausryhmän jäseniä, muun muassa teknistä tukea tai järjestelmän testikäyttäjiä. Katselmuksissa tarkasteltiin ja hyväksyttiin siihen asti tehty työ ja tehtiin mahdolliset korjausehdotukset.

5.2 Esitutkimus ja määrittely

Asiakas, Stora Enso Metsä, tarvitsi toimintansa pääprosessien laadun arviointiin ja tarkasteluun helppokäyttöisen työkalun. Työkalu oli ensisijaisesti tarkoitettu heidän Business Excellence -tiiminsä käyttöön, mutta käyttäjäkuntaa oli tarkoitus laajentaa koskemaan kaikkia työntekijöitä, kunhan työkalu oli ensin otettu käyttöön ja sinne syötetty tarvittava data. Työkalua varten oli ajateltu ja etsitty sopiva aihio, Philipsin oma Process Survey Tools, mutta asiakkaalla ei sillä hetkellä ollut riittävästi henkilöresursseja oman työkalun räätälöintiä ja kehittämistä varten. Tämän takia Business Excellence -tiimin johtaja, Varpu Savolainen, otti-vat yhteyttä Saimaan ammattikorkeakouluun ja haki opinnäytetyöntekijää kehittämään heille tämän työkalun. Opinnäytetöistä pääasiallisesti vastuussa oleva opettajani ja työn tuleva ohjaaja, Martti Ylä-Jussila, otti vastaavasti minuun yhteyttä heti saatuaan toimeksiannon ja sovimme tapaamisen Varpu Savolaisen kanssa.

Kehitysprojektin ensimmäinen kokous pidettiin 8.3.2006 Stora Enson metsäkonttorilla Imatralla, läsnä olivat Kimmo Tahvanainen, Varpu Savolainen ja Martti Ylä-Jussila. Kokouksessa tehtiin opinnäytetyösopimus, sovittiin projektin alustava aikataulu ja aloitettiin määrittely. Kaikki sovitut asiat kirjattiin ylös ja sisällytettiin seuraavaksi tehtyyn projektisuunnitelmaan. Projektisuunnitelmassa sovittiin, että projektiin kuuluvat Prosessin arviointityökalu -tietojärjestelmän määrittely, suunnittelu, toteutus, testaus ja käyttöönotto. Projektisuunnitelmassa otettiin lisäksi huomioon projektin aikataulu, resurssit, kulut ja mahdolliset uhat. Systeemityömalliksi valittiin RUP-malli, jota sovellettiin projektin erityispiirteisiin, lähinnä uusien tekniikoiden opettelun ja projektiryhmän koon (1 henkilö) takia. Määrittelyä jatkettiin Stora Enson Imatran metsäkonttorilla noin 2 - 3 päivänä viikossa.

5.3 Suunnittelu, toteutus ja testaus

Työkalun tärkeimmiksi ulkoisiksi ominaisuuksiksi tulivat käytettävyys, skaalautuvuus ja visuaalinen ilme. Tämän takia työkalu päätettiin tehdä Web-sovelluksena, jotta se olisi helppo lisätä, ylläpitää ja käyttää Stora Enso Metsän portaalin kautta. Tämän takia toteutustekniikoiksi valittiin ASP.NET sekä Visual Basic .NET ja toteutustyökaluksi Visual Studio 2005 Professional. Lisäksi Web-

sovellusten hallinnointiin tarvittiin IIS 6.0 -palvelin, joka käyttää .NET Framework 2.0:aa ASP.NET:in dynaamisten Web-sivujen prosessointiin ja SQL Server 2005 -palvelin ja tietokanta datan tallennukseen. Suunnittelua, toteutusta ja esitestausta varten tämä kokonaisuus asennettiin Stora Enso Metsältä saatuun ja järjestelmän kehittämiseen varattuun laptop-tietokoneeseen, jotta nämä työvaiheet voitiin suorittaa. Koska ASP.NET ja IIS olivat uusia työkaluja, piti projektissa varata aikaa näiden tekniikoiden opettelua varten. Määrittelyn jälkeen alkoikin tekniikoiden opiskelu- ja protoilu, jonka jälkeen vasta voitiin siirtyä varsinaiseen suunnitteluvaiheeseen, kun työkalut olivat tulleet tutummiksi ja niiden ominaisuuksia osasi hyödyntää ja käyttää. Suunnittelu-, toteutus- ja testausvaiheiden alussa jatkettiin määrittelyvaiheesta tuttua noin 2 - 3 kolmen työpäivän viikkorytmiä Imatran metsäkonttorilla, jonka jälkeen työtä tehtiin enemmän Lappeenrannassa ja Kotkassa, joissa työn apuna käytettiin VPN-yhteyttä Stora Enson intranetiin. Tällä tavalla tiedostojen siirto ja varmuuskopiointi paikalliselta ohjelmistokehitykseen käytettävältä laptop-tietokoneelta Stora Enson palvelimille sekä järjestelmän käyttö testiympäristössä saatiin suoritettua miltei yhtä helposti kuin paikan päällä Imatralla. Yhteydenpito piti vain hoitaa tällöin pelkästään puhelimitse ja sähköpostitse, mutta projektin ollessa hyvin käynnissä ei yhteydenpidon tarvinnutkaan enää olla niin tiivistä kuin projektin alkuvaiheessa. Ohjausryhmän kuukausittaisia kokouksia ja järjestelmän katselmuksia varten sovittiin kuitenkin kuukausittaiset tapaamiset Imatralle.

Suunnittelu-, toteutus- ja testausvaiheita varten Stora Enson palvelimille perustettiin testiympäristö, jossa jo olemassa olevan kokoonpanon IIS-palvelimelle asennettiin .NET Framework 2.0 ja luotiin tarvittava SQL Server -tietokanta. Kun järjestelmästä saatiin tehtyä ensimmäinen demoversio, se asennettiin testiympäristöön Web-sovellukseksi, jotta testausta voitiin suorittaa lopullisen kaltaisessa ympäristössä. Tämä oli erittäin tärkeää, sillä jotkin paikallisella kehityskoneella tehdyt ratkaisut ja Visual Studion valmiit komponentit eivät toimineet, kuten paikallisella koneella. Muun muassa tietokantayhteyttä ei saatu muodostettua näillä valmiilla komponenteilla, vaan oli rakennettava erillinen logiikka tietokantayhteyksiä ja SQL-kyselyitä varten. Ylläpitokäyttäjien käyttöoikeuksia varten Stora Enson AD-hakemistoon tehtiin lisäksi erillinen PATAdmin-ryhmä, johon lisättiin PAT-järjestelmän ylläpitäjät. Testiympäristön Web-sovellukseen tehtiin

lopulliselle järjestelmälle tarpeettomia, mutta testausvaiheelle tärkeitä testaussivuja, muun muassa tietokantayhteyksien ja ylläpito-oikeuksien testausta varten.

Ympäristön perustamisen jälkeen ensimmäisiä tehtäviä oli määritellä Prosessin arviointityökalun rakenne ja päätoiminnot, joiden jälkeen voitiin suunnitella sen tietokannan rakenne ja aloittaa järjestelmän muu suunnittelu. Tämän jälkeen testiympäristön tietokantaan perustettiin paikallisella koneella tehdyn ja testatun tietokannan taulut ja relaatiot. Varsinaiset järjestelmän toiminnot lisättiin tehtyjen määrittelyjen mukaisesti RUP-mallista sovelletulla suunnittelu – toteutus – testaus -syklillä. Projektin edetessä ja uusien toimintojen valmistuessa ne siirrettiin testiympäristöön inkrementaalisissa osissa Web-sovelluksen hakemistoon ja testattiin siellä.

5.4 Käyttöönotto ja koulutus

Kun riittävä määrä toimintoja oli saatu valmiiksi ja testauksen jälkeiset tarvittavat korjaukset oli tehty, kopioitiin toimiva kokonaisuus myös varsinaiselle tuotantopalvelimelle integraatiotestiä varten. Osana testiä syötettiin yhden pilottiprosessin data järjestelmän avulla tietokantaan. Testikäyttäjät testasivat järjestelmää ja raportoivat järjestelmän toiminnasta ja havaituista virheistä. Kun integraatiotesti oli suoritettu onnistuneesti, Prosessin arviointityökalun ensimmäinen versio oli otettu käyttöön. Tämä vaihe sujui melko kivuttomasti, sillä aikaisempi testaus oli ollut riittävää ja testiympäristön hyväksikäyttö oli ollut hyvä ratkaisu.

Lopuksi tehtiin vielä loppukäyttäjien koulutus, jota varten tehtiin ohjemateriaali ja opastettiin järjestelmän pilottiprosessin ylläpitokäyttäjät parin tunnin koulutuksessa. Koulutusvaihe rajattiin projektin suurien työtuntien takia projektin ulkopuolelle ja suoritettiin siihen liittyvänä työharjoitteluna.

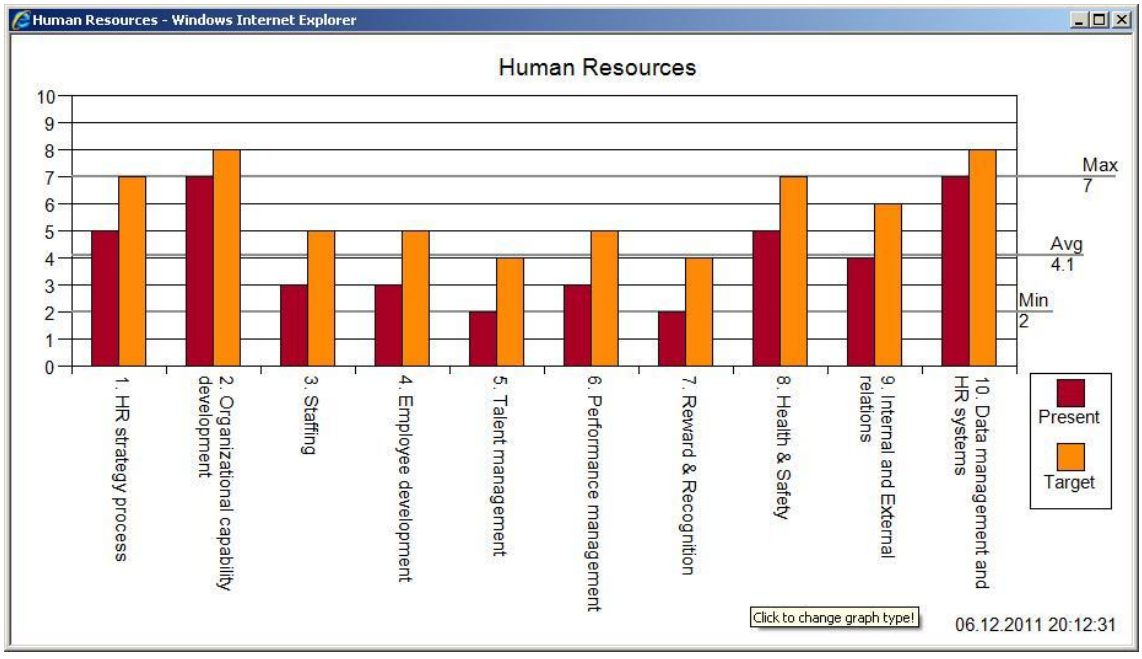
6 PROSESSIN ARVIOINTIMALLIIN PERUSTUVA TIETOJÄRJESTELMÄ

Tarkoituksena oli kehittää asiakasyrityksen pääprosessien laadun arviointiin ja tarkasteluun soveltuva helppokäyttöinen työkalu. Prosessien laadun arviointimallin perustana oli Philipsin kehittämä Process Survey Tools (PST), josta oli tarkoitus kehittää asiakkaan omaan organisaatioon räätälöity prosessien arvi-

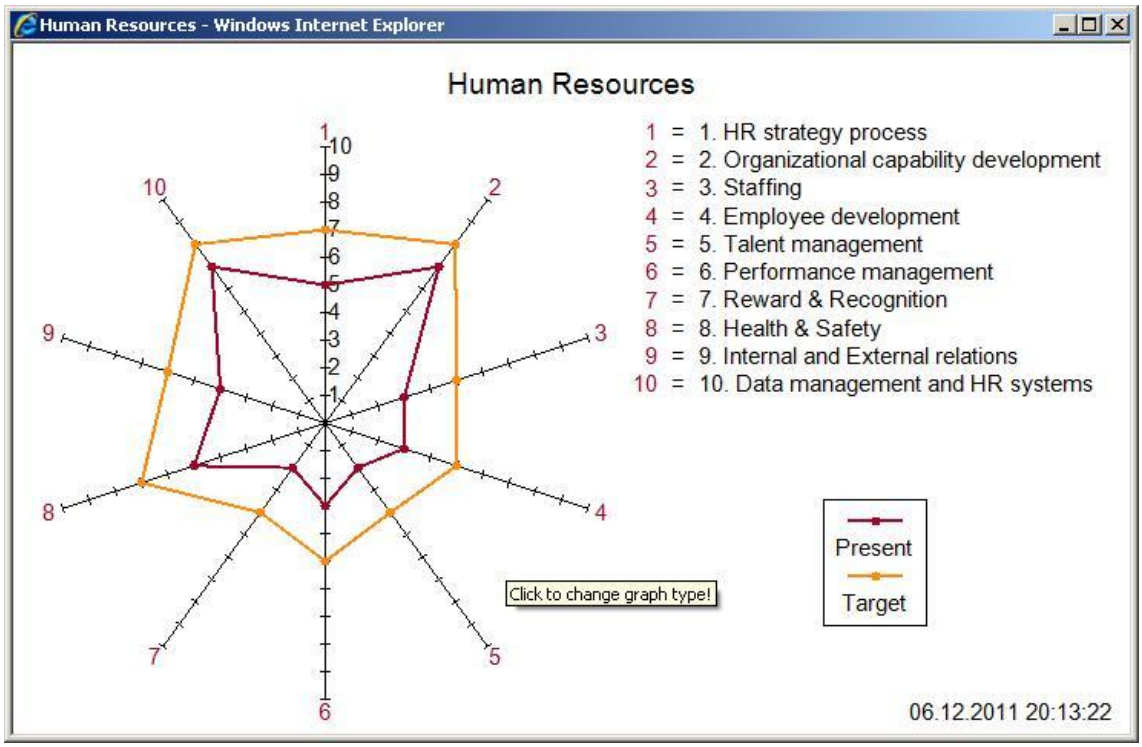
ointityökalu, PAT-tietojärjestelmä. Työkalussa jokaiselle pääprosessille määriteltiin yksitoistaportainen taso, joka kertoi prosessien laadukkuudesta ja kehitysasteesta. Pääprosessien tasojen määrittelemiseksi jokainen prosessi jaettiin aliprosesseihin, joiden tasot määriteltiin erikseen. Tasojen määrittelyssä käytettiin tasokysymyksiä, jotka piti jokaista tasoa kohti täyttää saavuttaakseen kyseisen tason. Lisäksi kaikki edelliset tasot tuli olla täytettyinä. Alimman tason saavuttamiseksi ei tarvinnut täyttää tasokysymyksiä, mutta seuraavan tason saavuttamiseksi ne tuli olla täytettyinä.

6.1 Graafiset kuvaajat

Pää- ja aliprosessien tasojen tarkastelun helpottamiseksi työkaluun tuli lisätä graafiset kuvaajat. Kuvaajien tuli pystyä kertomaan prosessien nykytilat selkeästi pikaisilla silmäilyillä ja siksi tähän tarkoitukseen valittiinkin kaksi kuvaajatyyppiä; normaali pylväskuvaaja sekä säteittäinen kuvaaja. Kuvaajat tuli lisätä käyttöliittymään kuvina, jolloin ne olivat helposti kopioitavissa ja siirrettävissä myös muihin käyttötarkoituksiin, kuten dia-esityksiin tai raportteihin. Kuvaajien värit valittiin Stora Enson värikartasta, jotta kuvaajat olisivat yhtenäiset yhtiön intra-portaalin värimaailman kanssa sekä mahdollisissa muissa käyttötarkoituksissa. Seuraavista kuvista kuva 6.1 ja kuva 6.2 näkyvät esimerkit järjestelmään tehdyistä kuvaajatyypeistä.



Kuva 6.1 Prosessin palkkikuvaaja



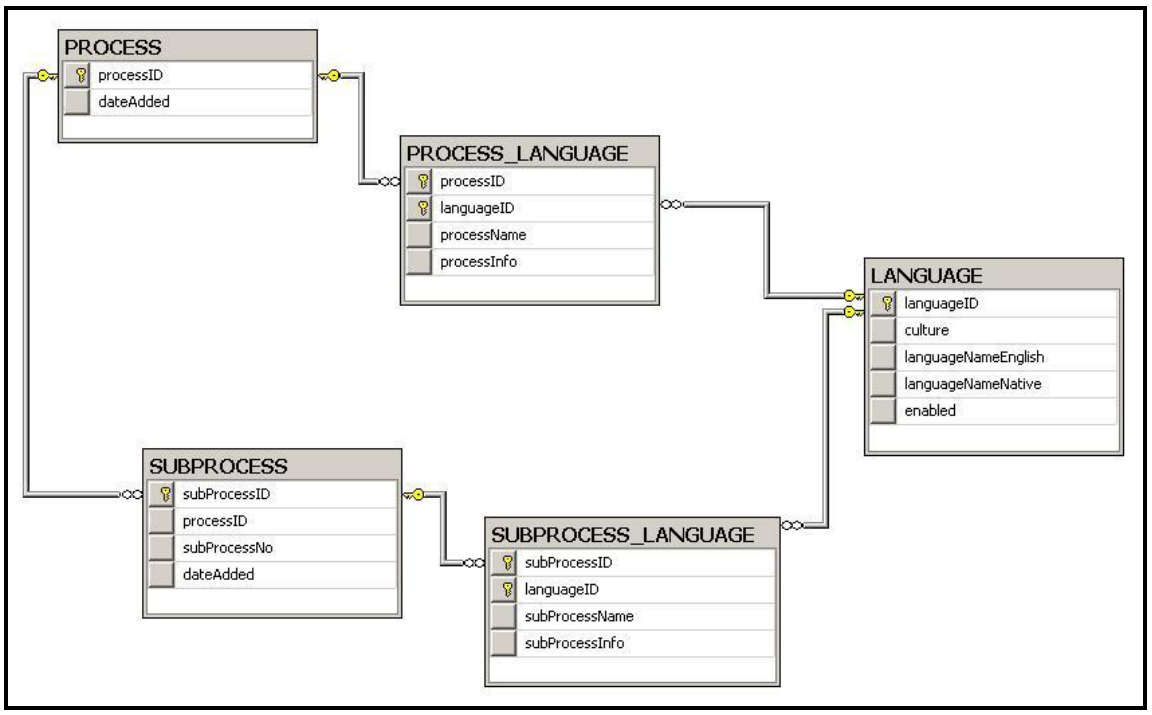
Kuva 6.2 Prosessin säteittäinen kuvaaja

Kuvaajatyyppejä on kaksi, pylväskuvaaja ja säteittäinen kuvaaja. Kuvaajien väreinä ovat musta, valkoinen ja Stora Enson määrittelemät punainen, oranssi ja harmaa. Kuvaajien ajonaikaiseen piirtämiseen etsittiin sopivaa Microsoft Visual Studio -lisäosaa, mutta sopivan puutteessa kuvaajat päätettiin piirtää jokainen viiva ja komponentti kerrallaan käyttäen Visual Studion omia piirtokirjastoja. Kuvaajien piirtäminen tuli tapahtua dynaamisesti siten, että tarvittava data haettiin tietokannasta juuri ennen kuvan piirtämistä ja sivun näyttämistä. Tällöin kuvaaja oli katseluhetkellä aina ajan tasalla ja voitiin olla varmoja siitä, että kuvaaja kertoi pää- ja aliprosessien nykytilan. Kuvaajien ajantasaisuuden varmistaminen on tärkeää, jos niitä halutaan kopioida ja siirtää toisiin sovelluksiin.

Kuvien piirtämiseksi kuva jaettiin useampaan loogiseen piirtoalueeseen, jotka käsiteltiin ja piirrettiin erikseen, muokaten kuitenkin koko ajan yhtä ja samaa kuvatiedostoa. Piirtoalueiden jakaminen tapahtui jaottelemalla kuva suorakaiteen muotoisiin osa-alueisiin, jotka kukin muodostivat jonkin selkeän, erillisen kokonaisuutensa, kuten kuvaajien koordinaatisto, kuvaajien nimet, selitteet ja niin edelleen. Tällä tavalla eri piirto-objektien mittaaminen ja piirtäminen helpottui kuvien layoutia suunnitellessa ja kooditasolle siirryttäessä sekä eri piirtoalueita voitiin käsitellä erikseen toisistaan riippumatta.

6.2 Tietokanta

Järjestelmän tietojen tallennus tapahtuu Microsoft SQL Server 2005 -alustalla olevaan relaatiotietokantaan, johon rakennettiin myös monikielisyyden tuki. Tämä tehtiin suunnittelemalla ja rakentamalla ensin tietokanta järjestelmää varten ja sitten lisäämällä tähän tietokantaan eri kielet kertova LANGUAGE-taulu, joka linkitettiin monta-moneen-liitoksilla jokaiseen käännettävää tekstiä sisältävään tietokannan tauluun. Monta-moneen-liitokset purettiin auki luomalla taulujen välille kolmas liittämistaulu ja tallennettiin sitten näin luotuihin liittämistauluihin tekstien käännökset kaikilla tarvittavilla kielillä. Alla oleva kuva 6.3 on tästä esimerkki.

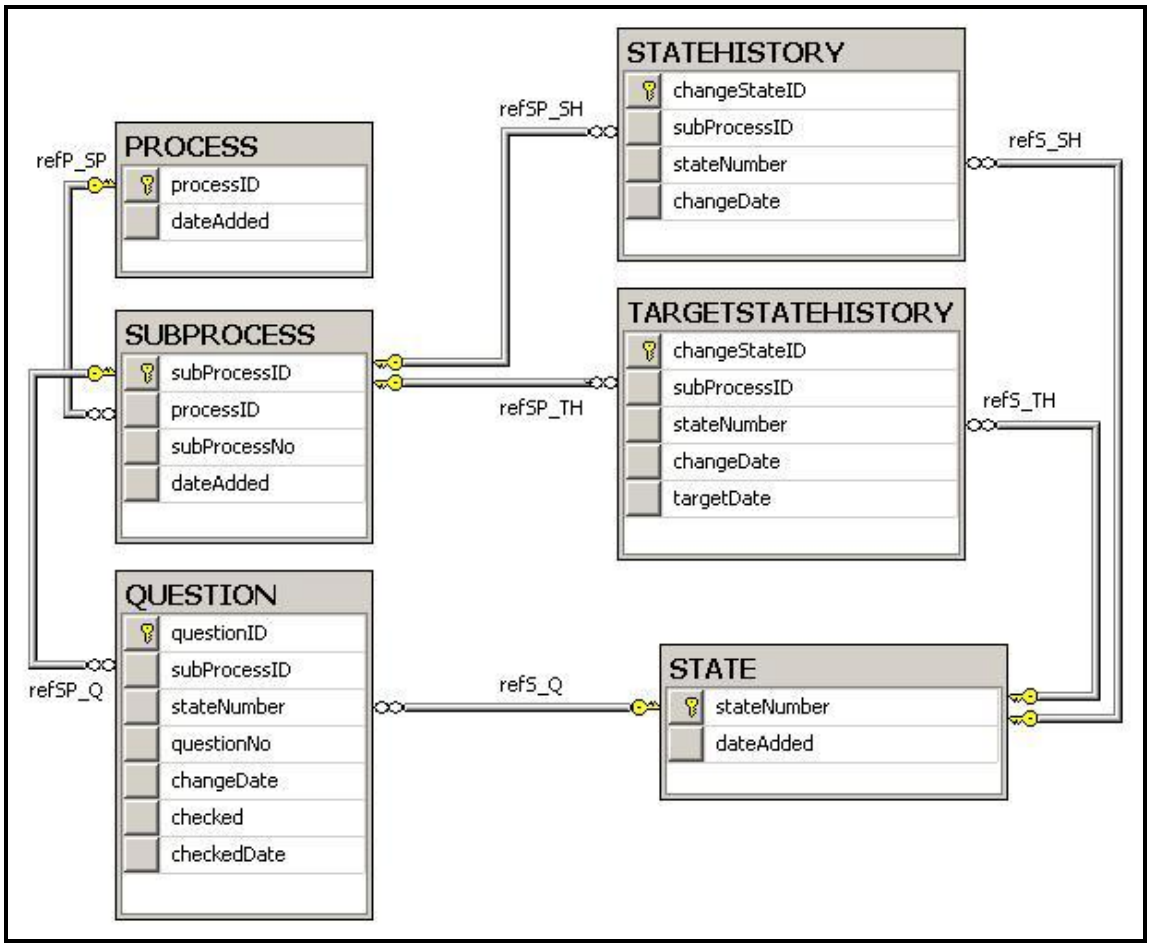


Kuva 6.3 Kielitaulujen liitosten purkuesimerkki

Kuten kuvasta näkyy, sekä prosessilla että aliprosessilla voi olla useita kieliä ja kielillä voi olla useita prosesseja ja aliprosesseja. Nämä monta-moneen-yhteydet puretaan auki kuvassa olevien PROCESS_LANGUAGE- ja PROCESS_LANGUAGE-aputaulujen avulla, joihin sitten tallennetaan eri kielisyyksiä tarvitsevat tietorivit.

Tietokannan suunnittelu aloitettiin analysoimalla järjestelmän vaatimukset ja piirtämällä kaavio vaatimukset täyttävästä tietokannasta. Kaavion pohjalta rakennettiin prototietokanta, jolla testattiin ja hienosäädettiin tietokannan toimi-

vuutta lopullista tuotantotietokantaa varten. Tälle tietokannalle määriteltiin lisäksi viite-eheys antamalla jokaisella tietokannan taululle perusavain ja siihen liitetyille tauluille vastaava viiteavain. Kuva 6.4 on kaaviokuva lopullisen tietokannan rakenteesta.



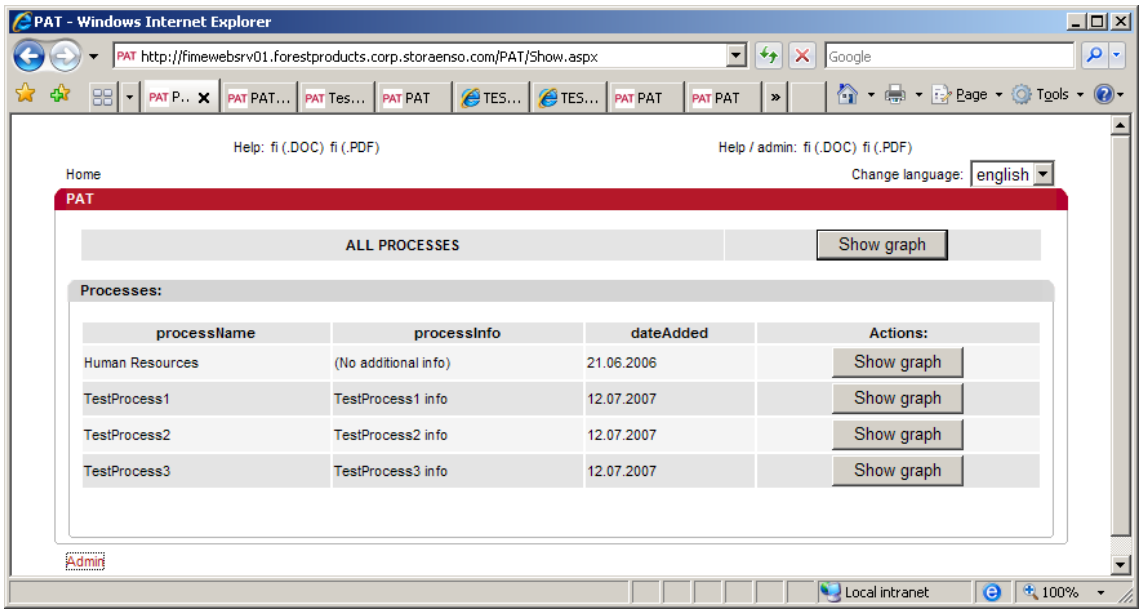
Kuva 6.4 Tietokannan kaaviokuva ilman kieli- ja aputauluja

Tietokannan perusrakenne ilmenee kaaviosta, josta on poistettu kieli- ja aputaulut kaavion tulkitsemisen selkeyttämiseksi. Kaaviosta nähdään, miten prosessilla on useita aliprosesseja (yksi-moneen-yhteys) ja aliprosessilla on taas monta tasoa ja tasolla useita aliprosesseja (monta-moneen-yhteys), joka on purettu kysymystaulun avulla kahdeksi yksi-moneen-yhteydeksi. Sama monta-moneen-yhteyksien purkaminen pätee myös tasohistoria- ja tavoitetasohistoria-taulujen kohdalla. Kun lisätään LANGUAGE-taulu ja tarvittavat aputaulut, tietokantaan tuli yhteensä 13 taulua.

6.3 Käyttöliittymä

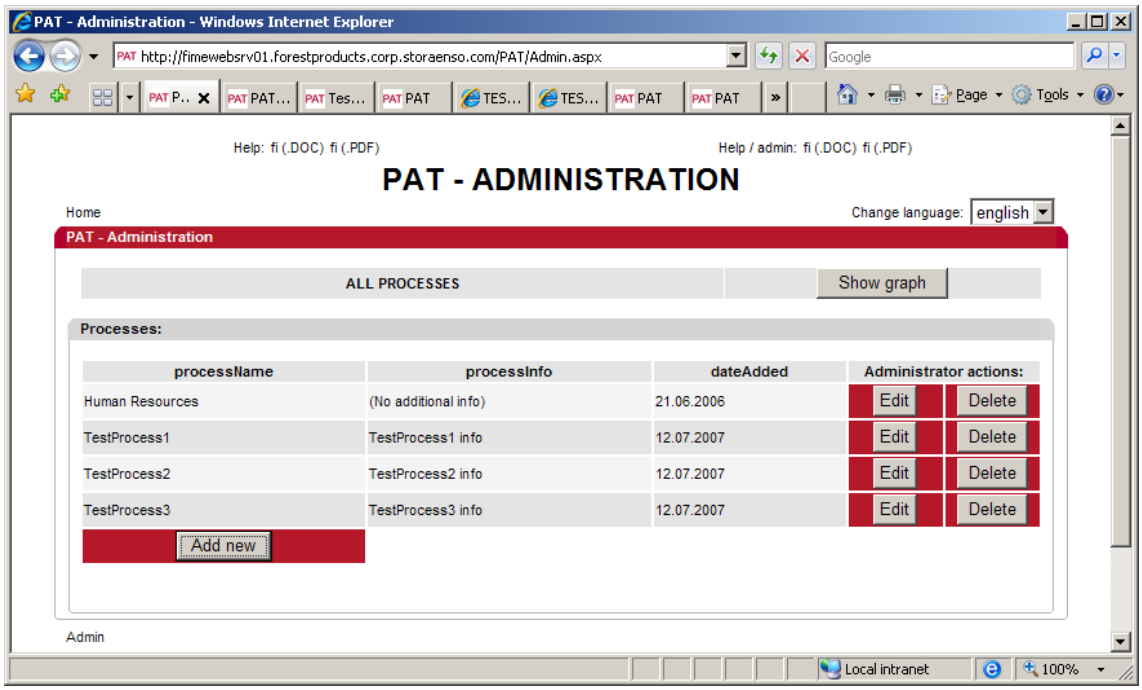
Käyttöliittymä suunniteltiin mahdollisimman yksinkertaiseksi käytettävyyden ja havainnollisuuden takia, mutta siihenkin rakennettiin monikielisyyden tuki, jolloin käyttäjä pystyy valitsemaan siihen haluamansa kielen sivujen yläreunassa sijaitsevasta pudotusvalikosta. Sivut näytettiin kuitenkin aluksi käyttäjän selaimeensa määrittämän kielivalinnan mukaan, jotta kieltä ei tarvitsisi välttämättä erikseen muuttaa. Käyttöliittymän tekstit eri kielille tallennettiin käyttöliittymään siten, että jokaiselle järjestelmän ASP-sivulle erilliselle kielelle tallennettiin Visual Studion WYSIWYG-editorilla erillinen, kullakin ISO 639-1 -standardin mukaisella kielikoodilla merkattu ASP-sivu. Käyttöliittymän tekstejä ei tallennettu samaan tietokantaan kuin järjestelmän varsinainen data kolmesta erillisestä syystä. Ensimmäinen syy oli varmistaa käyttöliittymän tekstien ja selitteiden näkyminen, mikäli tietokantaan ei saataisi jostain syystä muodostettua yhteyttä. Toinen syy oli se, että käyttöliittymän tekstit ovat melko yksinkertaiset ja niitä on melko vähän, kuten järjestelmän ASP-sivujakin, joten ne on helppo kääntää kieleltä toiselle. Tällöin pystyttiin näyttämään jo käyttöliittymän tekstit halutuilla kielillä, vaikka tietokannassa olevan suuren datamäärän käännöstyö olisi vielä kesken. Tässä tapauksessa käyttöliittymän tekstit näytetään valitulla kielellä ja tietokannan data järjestelmän oletuskielellä eli englannilla. Kolmas syy oli varmistaa tekstien oikeanlainen asemointi ASP-sivuilla Visual Studion WYSIWYG-editorilla, jolla voitiin nähdä käännöstekstien sijoittuminen sivujen layoutiin välittömästi muokkauksen jälkeen.

Käyttöliittymän värimaailma ja layout noudattavat Stora Enson värikarttaa ja heidän intranetissään olevan portaalin layoutia. Käyttöliittymä jaettiin lisäksi vielä kahteen erilliseen osioon, perus- ja ylläpitokäyttöliittymiin. Tämä sen takia, että järjestelmän ylläpitäjille olisi oma, enemmän toimintoja sisältävä käyttöliittymänsä, josta voi helposti lisätä, poistaa ja muokata järjestelmän tietoja ja peruskäyttäjille taas olisi oma, yksinkertaisempi ja selkeämpi käyttöliittymänsä, jolla voi vain katsella tietoja eikä tehdä muutoksia järjestelmään. Peruskäyttöliittymän alareunassa on linkki, jota klikkaamalla pääsee kirjautumaan ylläpitokäyttöliittymään. Kuva 6.5 näyttää ruutukaappauksen järjestelmän päänäkymästä eli peruskäyttöliittymästä.



Kuva 6.5 Järjestelmän peruskäyttöliittymä

Peruskäyttöliittymän pelkistetty ulkoasu ja portaalin värimaailma ilmenevät hyvin kuvasta, huomaa sivun alalaidassa oleva linkki ylläpitokäyttöliittymään. Kirjautuminen tapahtuu automaattisesti järjestelmän tekemällä LDAP-kyselyllä, joka vertaa asiakaskoneelle kirjautuneen käyttäjän käyttäjänimeä Stora Enson palvelinkoneen Active Directory -hakemistossa olevaan PAT-järjestelmän ylläpitäjäryhmään. Käyttäjälle avataan ylläpitokäyttöliittymä, mikäli käyttäjänimi löytyy ryhmästä, muuten näytetään oikeuksien puuttumisesta kertova virhesivu. Kuva 6.6 on ruutukaappaus järjestelmän ylläpitokäyttöliittymästä.



Kuva 6.6 Järjestelmän ylläpitokäyttöliittymä

Ylläpitokäyttöliittymän layout on pääpirteissään melko samankaltainen perusylläpitokäyttöliittymän kanssa, jotta sen käyttöön olisi helppo siirtyä peruskäyttöliittymän käytön jälkeen, mutta siitä tehtiin tarkoituksella hieman erilainen, että käyttäjä huomaisi siirtyneensä ylläpitopuolelle. Ylläpitokäyttöliittymässä on enemmän kontrolleja tiedon muokkausta varten ja niiden tausta on punainen, jotta käyttäjä huomaisi eron käyttöliittymissä, eikä tekisi vahingossa peruuttamattomia muutoksia järjestelmän tietoihin. Molemmissa käyttöliittymissä on sivujen ylä- ja alareunoissa linkki toiseen käyttöliittymään, joita klikkaamalla toinen käyttöliittymä avautuu täsmälleen samasta kohtaa. Tällä tavoin järjestelmän ylläpitäjille mahdollistetaan nopea siirtyminen liittymästä toiseen kesken sivuston selaamisen mm. pikaista tiedon korjaamista tai lisäystä varten.

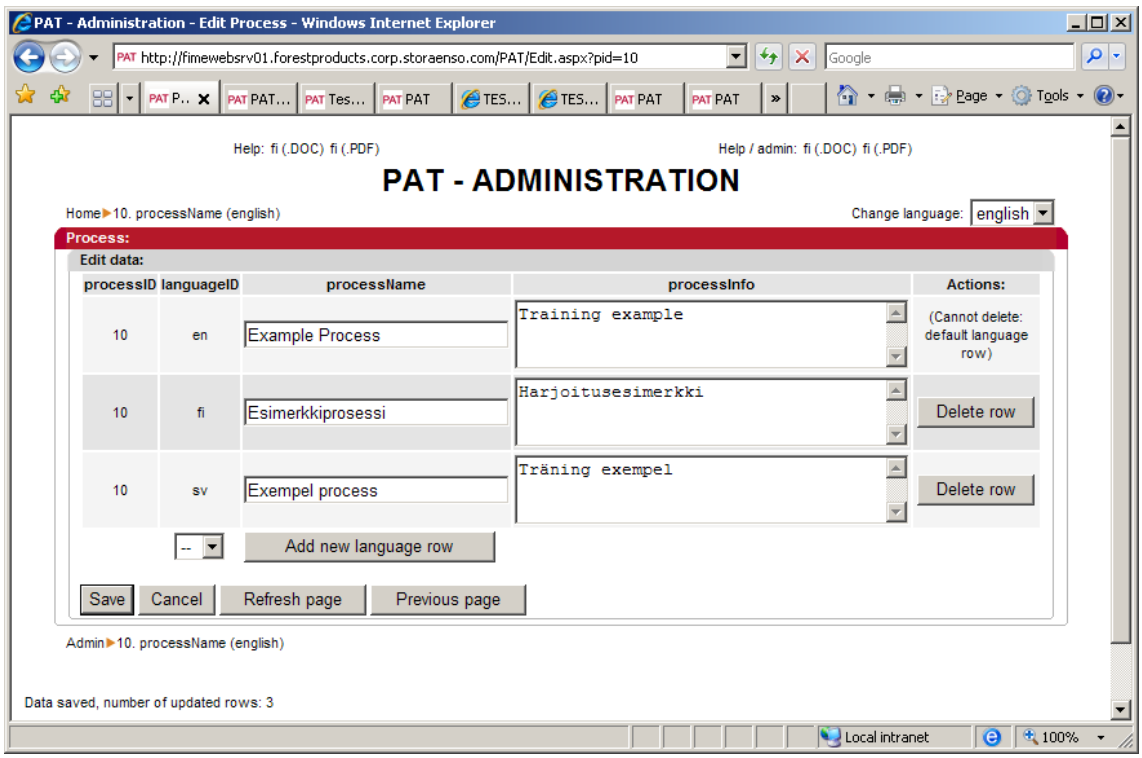
6.4 Järjestelmä

Tässä esitellään Prosessin arviointityökalu -tietojärjestelmän päätoiminnot. Esittelyssä keskitytään pelkästään ylläpitokäyttöliittymään ja jätetään peruskäyttöliittymän esittely pois, sillä ylläpitokäyttöliittymä sisältää ylläpitotoimintojen lisäksi myös kaikki peruskäyttöliittymän toiminnot. Näin esittely on mahdollisimman tiivis ja vältymme turhalta toistolta. Tietojärjestelmän Web-sovelluksen kansiohierarkia ja juurikansiossa olevat ASP-sivut koodisivuineen ovat nähtävillä aiemmin esitellyssä, IIS-palvelimen hallintapaneelia esittelevässä ruutukaappa-

uksessa kuva 4.9. Lisäksi järjestelmän perus- ja ylläpitokäyttöliittymien ruutukaappaukset näkyivät jo kuvissa kuva 6.5 ja kuva 6.6.

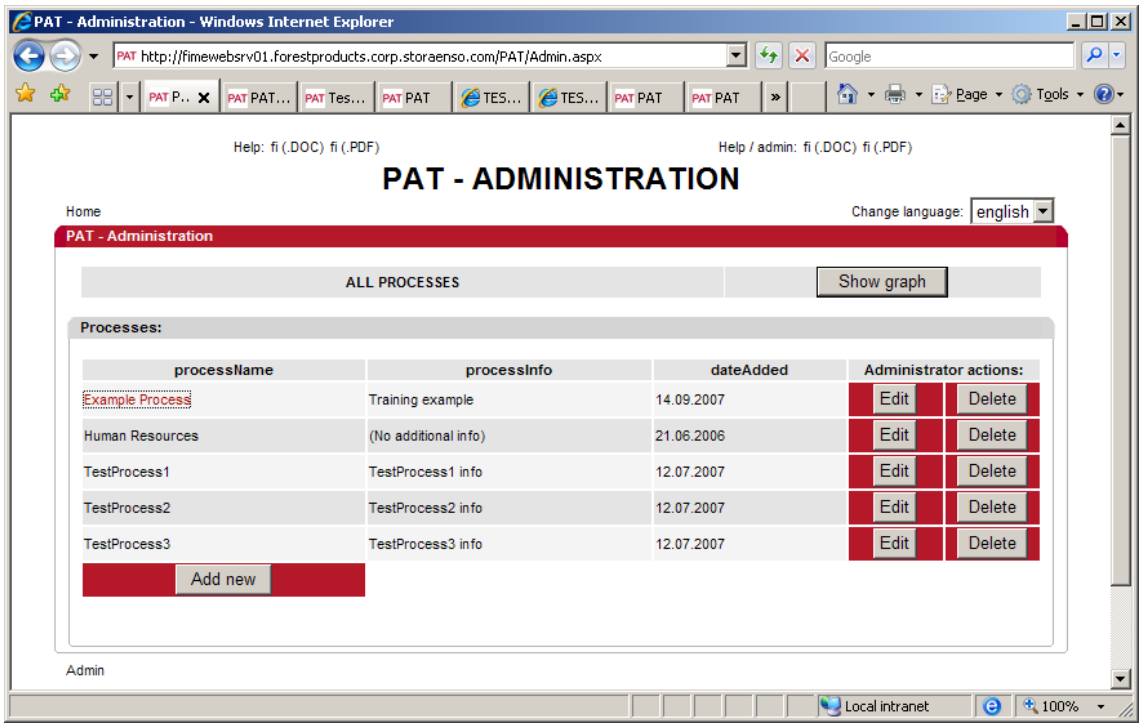
6.4.1 Prosessien katselu, lisääminen, muokkaus ja poisto

Aikaisemmassa kuvassa Kuva 6.6 on prosessin lisäys-, muokkaus- ja poistonapit sekä kaikkien prosessien kuvaajien katselunappi. Lisäysnappia painamalla pääsee prosessin muokkaussivulle lisäämään prosessin tiedot. Muokkausnappi avaa saman sivun, mutta jo aikaisemmin tallennetuin tiedoin. Alla oleva kuva 6.7 näyttää prosessin muokkaussivun.



Kuva 6.7 Uuden prosessin luonti ja prosessin muokkaus

Kuvassa näkyy, miten prosessille voidaan lisätä tietoja eri kieliriveille. Tallennusnapin painallus tallentaa syötetyt tiedot, tästä ilmoitetaan sivun alalaidassa olevalla inforivillä. Myös virheellisistä syötteistä tai pakollisten tietojen puuttumisesta ilmoitetaan samalla inforivillä. Edelliseen näkymään pääsee Previous page -napilla. Seuraava kuva 6.8 näyttää napin painalluksen jälkeisen siirtymisen edelliseen näkymään.

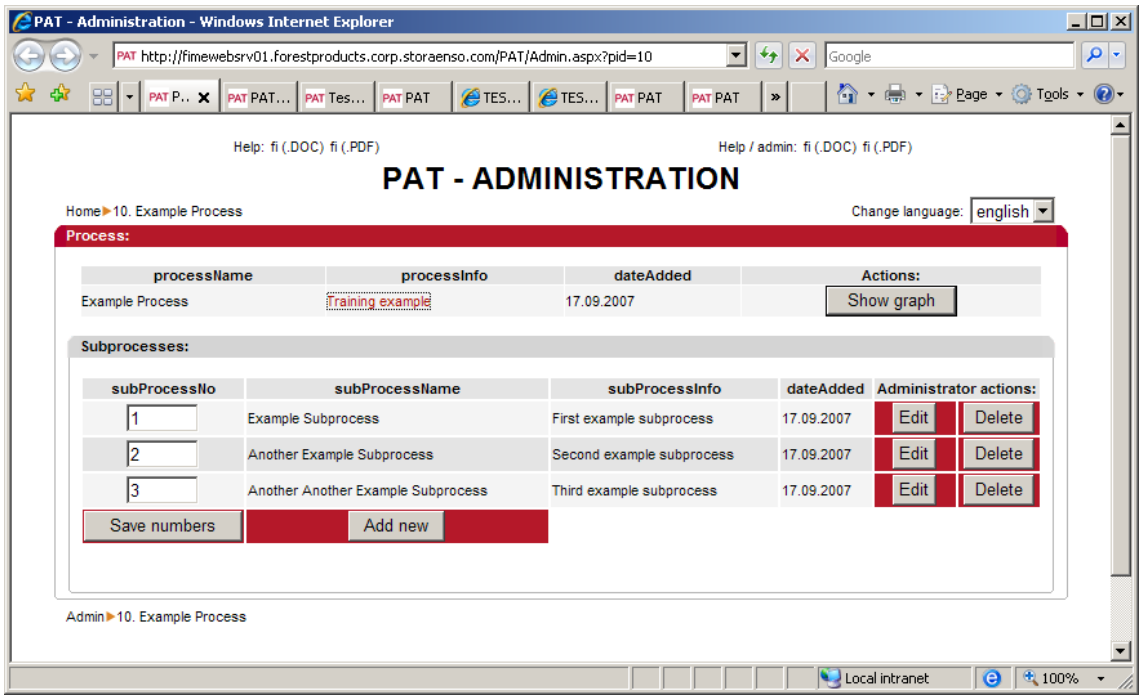


Kuva 6.8 Prosessinäkymä uuden prosessin luonnin jälkeen

Nyt Example Process -prosessi on luotu ja se näkyy prosessinäkymässä. Muokkausnappia painamalla tästä pääsisi nyt äskeiseen muokkaustilaan ja poistonapista poistamaan prosessin. Prosessin lisäys- ja poistonappia painettaessa käyttäjältä kysytään varmennus, haluaako hän varmasti tehdä lisäyksen tai poiston. Lisäksi tietokannan viite-eheyden takia sellaisen prosessin poistaminen, jolla on alaprosesseja, ei onnistu, vaan alaprosessit tulee poistaa ensin. Näin estetään suuren tietomäärän vahinkopoistot. Juuri luodun prosessin nimeä klikkaamalla pääsee syvemmälle prosessin tietoihin katselemaan sen alaprosesseja.

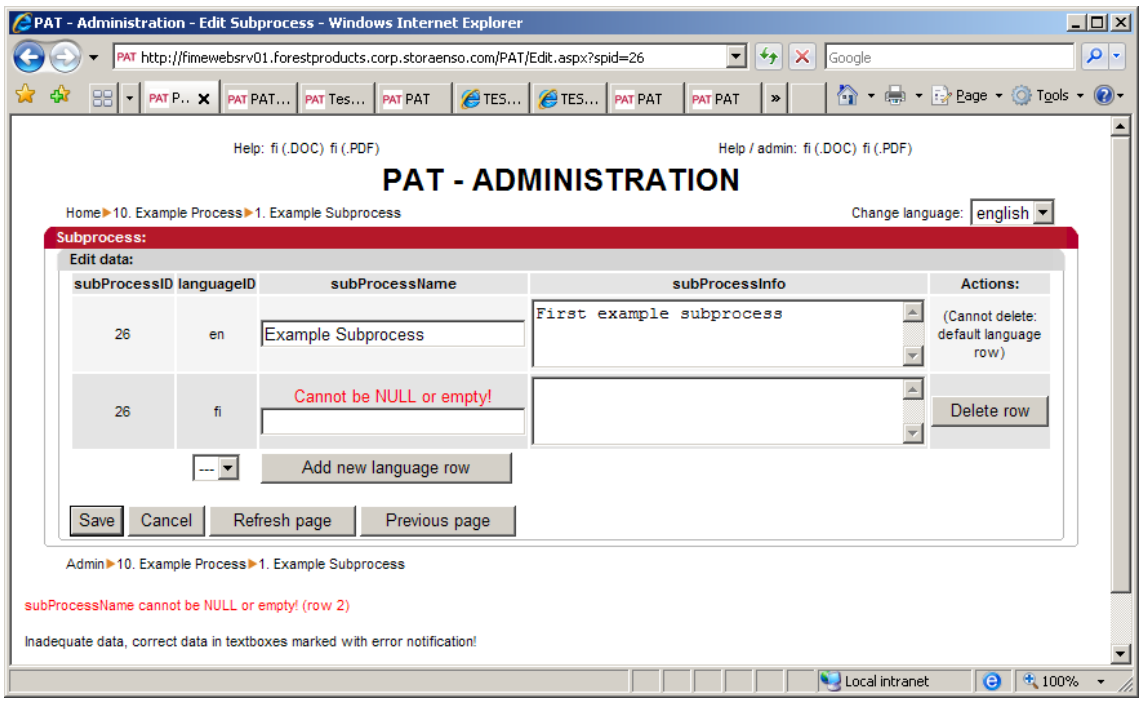
6.4.2 Alaprosessien katselu, lisääminen, muokkaus ja poisto

Alaprosessinäkymässä listataan prosessin alaprosessit ja niitä voi täältä lisätä, muokata tai poistaa. Navigointilogiikka on samanlainen kuin prosessinäkymässäkin, kuten kuva 6.9 ilmaisee.



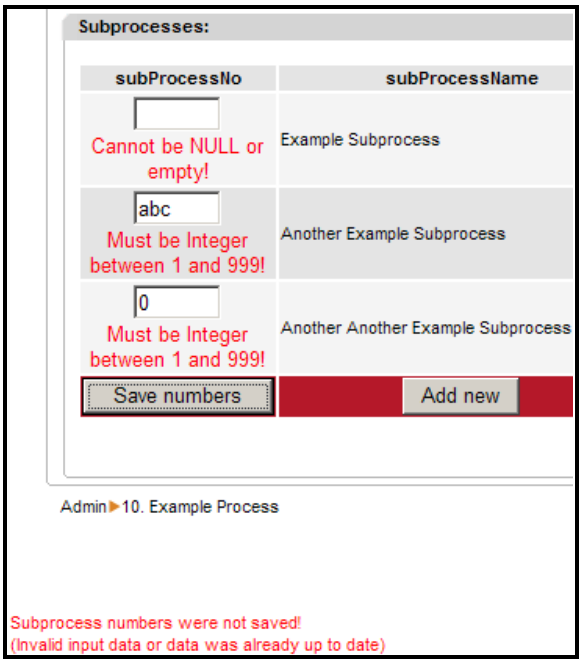
Kuva 6.9 Alaprosessinäkymä

Tämän näkymän erot prosessinäkymään ovat isäntäprosessin näkyminen alaprosessien yläpuolella, prosessin kuvaajan näyttönapin toiminta, joka avaa vain kyseisen prosessin tilojen kuvaajan ja alaprosessin järjestysnumeroiden vaihtonappi. Muuten toiminnot ovat samankaltaiset kuin prosessinäkymässä. Seuraava kuva 6.10 näyttää alaprosessin muokkausnäkymän.



Kuva 6.10 Uuden alaprosessin luonti ja muokkaus

Kuten nähdään, muokkausnäkymä on miltei täsmälleen samanlainen kuin prosessin muokkausnäkymä, ainoana erona näkyvät vain sivun ylä- ja alalaidassa olevien navigointilinkkien erot, joissa yläreunassa on linkit peruskäyttöliittymään ja alalaidassa linkit ylläpitokäyttöliittymään. Home- ja Admin-linkit vievät käyttöliittymien alkuun ja isäntäprosessin nimen klikkaaminen vie prosessinäkymään. Kuvassa näkyy, miten JavaScriptiä hyödyntävillä validointityökaluilla voidaan informoida käyttäjää virheellisistä tai puutteellisista syötteistä. Tarkempana esimerkkinä tästä kuva 6.11.



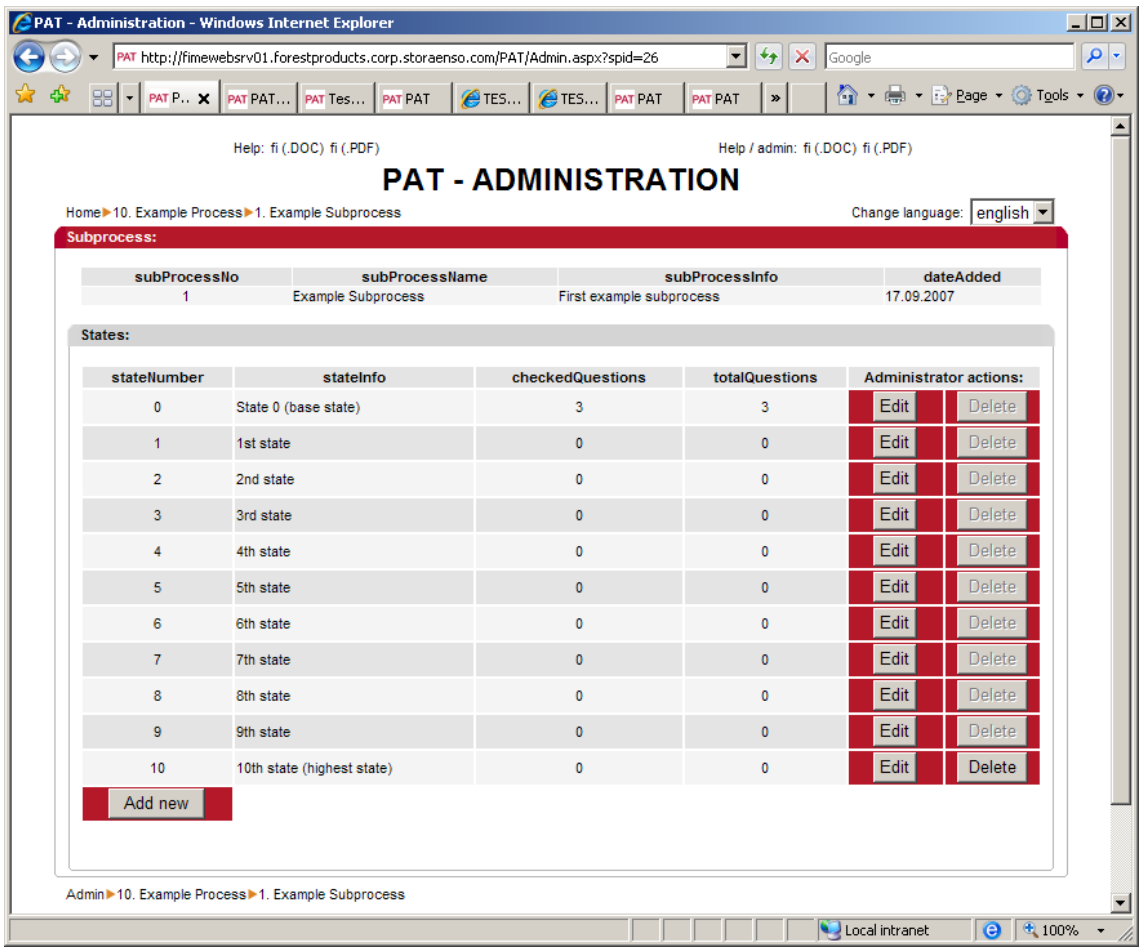
Kuva 6.11 Esimerkki syötteentarkistuksesta

Kuvassa on esimerkki kolmenlaisesta syötteentarkistuksesta: kenttään vaaditaan joku arvo, kentän syötteen tulee olla kokonaisluku ja kokonaisluvun tulee olla tietyltä väliltä.

6.4.3 Alaprosessin tasojen katselu, lisääminen, muokkaus ja poisto

Tässä näkymässä, johon pääsee klikkaamalla halutun alaprosessin nimeä, nähdään alaprosessin tasojen tilat sekä tasokysymysten lukumäärät. Vaikka tämäkin näkymä on hyvin samankaltainen, kuin edellä olleet prosessi- ja alaprosessinäkymät, on siinä yksi merkittävä ero. Tasojen lisäys ja poisto vaikuttavat kaikkiin alaprosesseihin, esimerkiksi jos luodaan uusi taso, silloin kaikkien alaprosessien tasojen lukumäärä kasvaa yhdellä ja sama toisinpäin. Poistossa

tosin viite-eheys tulee jälleen kuvaan ja estää sellaisten tasojen poiston, joilla on tasokysymyksiä. Tasojen lukumäärää tuskin kuitenkaan tarvitsee muuttaa, sillä nuo tasot nollasta kymmeneen ovat EFQM Excellence Model:in mukaiset, eikä tuon mallin tasojen lukumäärään tulle tulevaisuudessa muutoksia. Toiminnallisuus rakennettiin kuitenkin kaiken varalta, kun se ei ollut iso lisätyö ja tukee järjestelmän skaalautuvuutta, mikäli noihin tasoihin tulisi muutoksia. Tasonäkymästä ruutukaappauksena kuva 6.12.



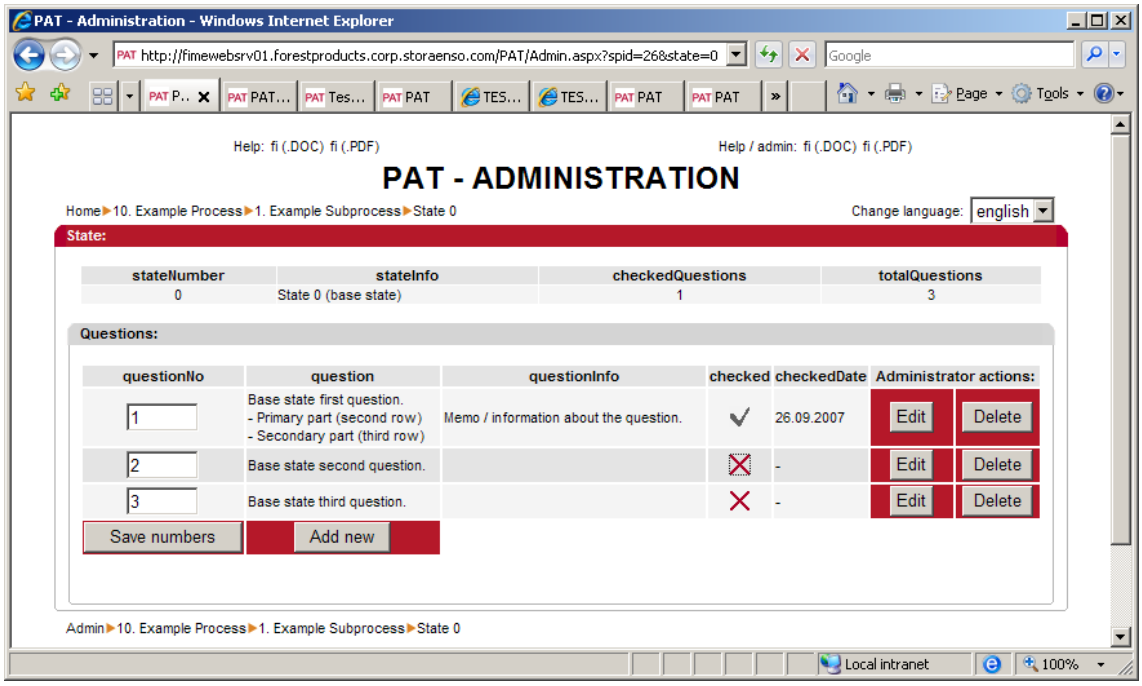
Kuva 6.12 Alaprosessin tasonäkymä

Tässä näkyvät alaprosessin tasojen tasokysymysten lukumäärät. Siis sekä jokaisen tason kaikkien kysymysten lukumäärät että jokaisen tason täytettyjen kysymysten lukumäärät. Alaprosessin nimi näkyy tasojen yllä.

6.4.4 Tasokysymyksien katselu, lisääminen, muokkaus ja poisto

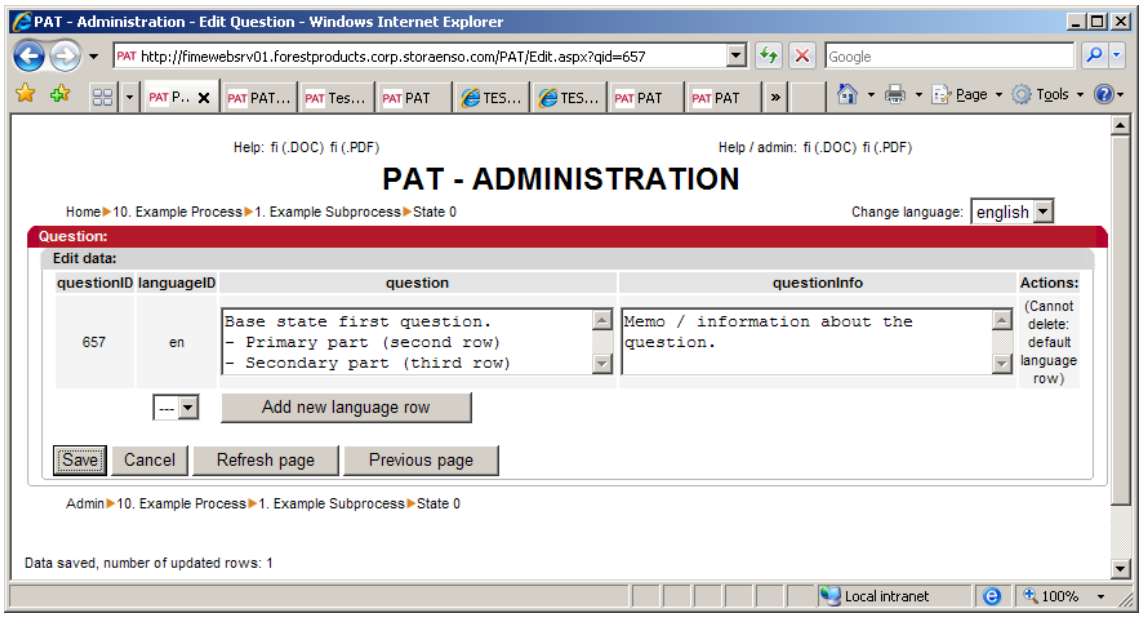
Tason nimi- tai infolinkkiä klikkaamalla pääsee katselemaan tasokysymyksiä. Tasokysymysnäkymä on toiminnoiltaan taas kuten alaprosessinäkymä, kysy-

mysnumeroiden muokkausnappia myöten, sillä erolla että kysymyksiä voi merkata täytetyiksi tai täyttämättömiksi klikkaamalla niiden tilaa osoittavia kuvakkeita. Muuten kysymyksien lisäys, muokkaus ja poisto noudattavat tutuksi tullutta kaavaa. Tämän havainnollistaa kuva 6.13.



Kuva 6.13 Tasokysymysnäkymä

Harmaa väkänen merkitsee, että kysymys on täytetty ja punainen ruksi taas merkitsee täyttämättömän kysymyksen. Kuva 6.14 näyttää kysymysten luonti- ja muokkausnäkymän.

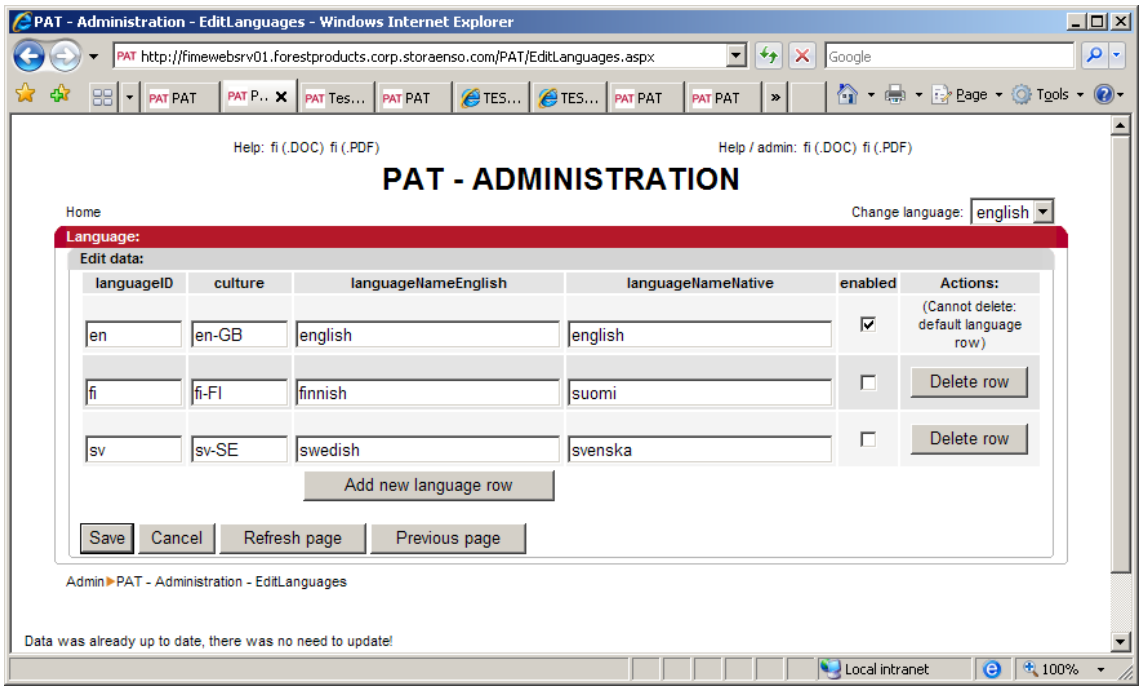


Kuva 6.14 Uuden tasokysymyksen luonti ja muokkaus

Tässäkin näkymässä noudatetaan samaa kaavaa, muokkaus tapahtuu kuten aiemmissa prosessin / alaprosessin luonti- ja muokkausnäkymissä.

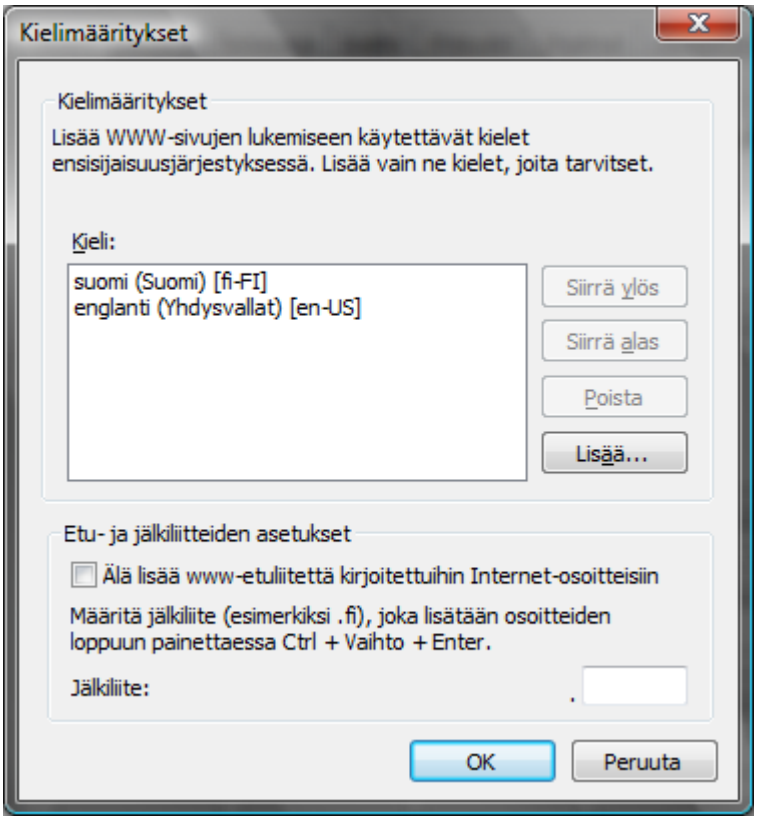
6.4.5 Kielien katselu, lisääminen, muokkaus ja poisto

Edellä olevien päätoimintojen lisäksi järjestelmässä on mahdollisuus lisätä kieliä suoraan käyttöliittymän avulla. Tähän muokkausnäkymään pääsee kielenvaihtopudotusvalikon vasemmalla puolella olevasta Change language -linkistä, mikä on linkki vain ylläpitokäyttöliittymässä piilossa peruskäyttäjiltä. Tässä näkymässä kielten lisääminen, muokkaus ja poisto tapahtuvat edellä esiteltyjen periaatteiden mukaisesti sillä erotuksella, että kielten ID-kenttiin tulee lisätä ISO 639-1 -standardin mukaiset, kaksikirjaimiset kielikoodit. Lisäksi culture-kenttään voidaan laittaa kielen tarkempi määrite, kuten kuva 6.15 havainnollistaa viittauksia brittienglantiin, Suomessa käytettävään suomeen ja riikinruotsiin.



Kuva 6.15 Kielten luonti- ja muokkausnäkymä

Kuvassa näkyy, että englanninkielistä kieliriviä ei voi poistaa. Tämä johtuu siitä, että englanti on asetettu järjestelmän oletuskieleksi, jolloin sen kielinen data on pakko olla. Enabled-kentän avulla voidaan keskeneräiset kielet piilottaa perus-käyttäjiltä ottaen ne sitten käyttöön, kun kielen kääntötyö ja tietojen syöttö on valmis. Edellä mainitut culture-tarkenteet on tehty sitä varten, että jokainen käyttäjä voi asettaa selaimensa asetuksista halutut kielivalinnat, jolloin sivun kieli näytetään oletusarvoisesti käyttäjän määrittämän listan mukaisesti. Kuva 6.16 on esimerkki Internet Explorer -selaimen asetusten kielimäärityksistä, joissa suomi on asetettu ensisijaiseksi ja amerikanenglanti toissijaiseksi kieleksi.



Kuva 6.16 Internet Explorer -selaimen kielimääritykset

7 POHDINTA

Työn kohteena olevan laatuajattelun ymmärtäminen ja siitä johdettu järjestelmän esitutkimus ja määrittely olivat melko selkeitä ja nopeasti hahmotettavia asioita, koska Stora Enso Metsän puolelta tulevan opinnäytetyön ohjaajan ja työn tilaajan Varpu Savolaisen käsitys työn pohjana olevasta EFQM:n laatukonseptista oli kattava. Lisäksi Varpu Savolaisen antamat materiaalit EFQM:n, Philipsin mallin ja Stora Enson omista tiedoista olivat erinomaisia ja havainnollisia järjestelmää määriteltäessä. Ja mikäli jokin asia jäi epäselväksi tai vaati muuten vaan tarkennusta, se oli helppo selvittää joko Varpu Savolaisen tai Teppo Salmen kanssa. Teppo Salmen panos oli myös sikäli tärkeä, sillä hän oli tehnyt noin vuosi ennen tätä työtä vastaavankaltaisen projektin Stora Enso Metsälle opinnäytetyönään, joten häneltä sai hyvää tietoa Stora Enso Metsän toimintatavoista eri projektin vaiheissa ja tilanteissa. Tätä projektia olikin alustavasti ajateltu hänen tehtäväkseen, mutta hänellä ei ollut muilta töiltään aikaa keskittyä täysipainoisesti näin laajaan projektiin. Kommunikointi ja yhteistyö projektin muiden sidosryhmien kanssa sujuivat muutenkin kauttaaltaan hyvin, joten sen

suhteen ei ollut ongelmia. Kommunikoinnin osalta omaksi tehtäväkseni tuli toimia eräänlaisena teknisenä tulkkina, sillä vaikka Varpu Savolaisella oli hyvät tiedot ja taidot projektin laatuasioista ja koko konseptin ymmärtämisestä, niin systeemityön tekeminen ja siihen liittyvät tekniset asiat ja termit olivat verrattain vieraita. Mielestäni onnistuin tämän suhteen hyvin ja osasin selittää tekniset yksityiskohdat niin sanotusti maalaisjärjellä hahmotettaviksi Varpu Savolaiselle ja muille ei-teknisesti orientoituneille loppukäyttäjille sekä vastaavasti keskustella Teppo Salmen, Martti Ylä-Jussilan ja teknisen tuen kanssa projektista ja järjestelmästä oikeilla teknisillä termeillä.

Projektin alustava aikataulu, joka oli viimeistään vuoden 2006 loppuun mennessä, venyi lopulta alkusyksyyn 2007. Syinä tähän olivat pääosin uusien tekniikoiden ja asioiden opettelun viemä aika, kun ASP.NET-tekniikan ja IIS-palvelimen Web-sovellusten hallinnoinnin opettelu oli tehtävä projektin alussa ja osittain sen aikana. Myös liika innokkuus tekniikan sisäistämisen aikana kostautui, sillä samalla kun tekniikasta oppi lisää uusia ominaisuuksia, näitä uusia ominaisuuksia hyödyntäviä toimintoja tuli kasattua järjestelmän määrittelyyn ja suunnitteluun liikaa. Systeemityön ja projektin edetessä näitä ylimääräisiä toimintoja ja ominaisuuksia karsittiinkin melko kovalla kädellä, mikä oli lopulta järjestelmän ulkoasun ja toiminnallisuuden kannalta hyvä asia, sillä järjestelmän käytön helppous ja selkeys olivat avainasioita järjestelmän tuleville perus- ja ylläpitokäyttäjille. Lisäksi oma kuntoutuminen vuoden 2004 kevättalvella tapahtuneesta selkäydinvammasta vaati veronsa, sillä samanaikainen itsensä kuntouttaminen ja opinnäytetyön tekeminen olivat haasteellisia yhdistää toimivaksi kokonaisuudeksi. Projektin ajanhallintaa oli muutenkin hieman hankala hahmottaa, sillä tämä projekti oli laajuudessaan ensimmäinen projektipäällikkönä ja kaikki langat piti pitää omissa käsissä, eikä kuitenkaan tarkkoja arvioita eri vaiheiden vaatimista ajoista ja työmääristä osannut arvioida oikein. Tämä korostui varsinkin vasta opeteltujen uusien tekniikoiden kohdalla, sillä vaikka hahmotti, miten joku tietty ongelma ratkaistaan, siihen vaatimaa aikaa ei silti osannut arvioida oikein. Yleensä tuo aika-arvio menikin selkeästi alakanttiin ja työtä piti tehdä enemmän ja kauemmin, jolloin projektin aikataulu jälleen venyi.

Työn tekoa häiritsi myös työkaluna olleen laptop-tietokoneen satunnainen ilmoitus tapahtuneesta virheestä ja pakollisesta tietokoneen uudelleenkäynnistämi-

sestä muutaman minuutin sisällä. Virheen syytä yritettiin metsästää teknisen tuen kanssa muun muassa testaamalla muistit ja vaihtamalla laitteistoa. Vian syy ei kuitenkaan koskaan selvinnyt, todennäköisesti syy lienee joidenkin tietokoneeseen asennettujen ohjelmien yhteensopimattomuudessa, mutta perusteelliselle asian tutkimiselle ei projektin aikaa voinut uhrata. Muutaman kerran selkänsä koneelta käännettyään oppi ainakin jatkuvan tallentamisen tärkeyden, sillä kone saattoi käynnistyä minä hetkenä tahansa uudelleen ja hävittää siihen asti tehdyn työn. Vaikka ongelman tiedostamisen jälkeen tiedon häviämistä ei juuri enää päässyt tapahtumaan, niin ajatus ainakin keskeytyi, kun piti odottaa koneen käynnistymistä ja kaikkien raskaiden kehityssovellusten uudelleenlataamista. Seuraavassa alaluvussa kerrotaan tarkemmin varsinaisessa kehitystyössä kohdatuista ongelmista ja vaikeasti ratkaistavista kohdista.

7.1 Vaikeiden kohtien ratkaisuja

Työtä tehdessä tuli vastaan muutama ongelmallinen kohta, jotka vaativat hieman pohtimista, kokeilua ja tutkimista. Yleensä nämä ongelmat liittyivät jollakin lailla ASP.NET-sivujen muotoilemiseen ja näyttämiseen. Tämä johtui pääasiassa siitä, että vaikka valittu ohjelmointityökalu Visual Studio ja valittu kieli Visual Basic olivat tuttuja, niin ASP.NET-ohjelmointitekniikka piti opetella työn tekemisen ohessa. Yleensä kyseessä olleeseen ongelmaan löytyikin melko suoraan ratkaisu ohjelmointityökalun sisään rakennettuja komponentteja hyödyntämällä, mutta joskus piti keksiä uusi tapa ongelman ratkaisemiseksi tai kiertämiseksi. Monesti apua löytyi myös Internetistä löytyviltä opassivuilta sekä erilaisilta kehittäjien keskustelupalstoilta, joissa joku oli jo aiemmin törmännyt vastaavanlaiseen ongelmaan. Hyvinä esimerkkeinä tästä olivat esimerkiksi luvussa 6.1 esitellyt graafiset kuvaajat ja niiden piirtäminen sekä luvuissa 6.2 ja 6.3 esitelty sivujen monikielisyyden rakentaminen, joista kerrotaan seuraavaksi tarkemmin.

7.1.1 Graafisten kuvaajien piirtäminen

Ongelman ratkaisua lähestyttiin ensin etsimällä valmiita komponentteja liitettäväksi Visual Studio -sovelluskehittimeen, mutta löydetyt ratkaisut olivat joko tarkoitukseen sopimattomia, riittämättömästi muunneltavissa tai liian kalliita saatuun hyötyyn nähden. Tästä johtuen päädyttiinkin piirtämään kuvaajat kokonaan itse Visual Studion piirtokirjastoja sekä Web-sovelluksen ominaisuuksia ja toi-

mintoja hyväksi käyttäen. Aluksi päätettiin tehdä neljänlaisia kuvaajia: palkki- ja säteittäiset kuvaajat prosessin tasojen esittämistä varten sekä viivakaaviot kertomaan tasojen ja tavoitteiden tilahistoriasta sekä viivakaaviot niiden vertailusta. Viivakaavioiden tekemisestä luovuttiin kuitenkin liian työläänä ja aikaa vievänä, kun nähtiin, etteivät ne kuitenkaan olisi olennaisen tärkeitä järjestelmän kannalta.

Palkki- ja säteittäisistä kuvaajista hahmoteltiin luonnokset ulkoasun muodostamiseksi ja valittiin eri komponenteille mahdollisimman kuvaavat värit Stora Enson värikartasta. Tarkoitus oli tehdä kuvaajista mahdollisimman selkeät ja pelkistetyt sekä niiden hahmottamisen helpottamiseksi että vaadittavan työmäärän minimoimiseksi. Kun kuvaajien layout oli luonnosteltu ja päätetty, kuvaaja jaettiin komponenttien mukaisiin loogisiin piirtoalueisiin. Suunnittelu- ja toteutusvaiheiden aikana nämä piirtoalueet eroteltiin vihreillä rajausviivoilla, että alueiden hahmottaminen olisi vaivattomampaa. Alueiden kokoja ei kuitenkaan toteutuksessa lyöty täysin lukkoon, vaan ne laskettiin tiettyjen ennalta määritettyjen parametrien, kuten viivakokojen ja fonttikokojen mukaan. Parametrit annettiin piirtämisestä varten tehdyn luokan oliolle sitä kutsuttaessa. Tällä saavutettiin joustavuutta ja skaalautuvuutta kuvaajien piirtoon, kun esimerkiksi prosessien ja alaprosessien nimien vaatima tila laskettiin lennossa. Tällöin ei päässyt käymään niin, että kuvan reunoihin olisi joko jäänyt liikaa ylimääräistä tilaa tai tekstit eivät olisi mahtuneet kuvaajaan kokonaan. Valitun ratkaisun johdosta kuvaajat saatiin räätälöityä täysin asiakkaan haluamiksi, eikä tarvinnut yrittää kiertää jonkin toisen kuvaajakomponentin rajoitteita ja puutteita.

Koska kuvaajien ajantasaisuus koettiin tärkeäksi, kuvien piirto päätettiin tehdä reaaliaikaisesti hakemalla ensin tarvittava tieto SQL Serverin tietokannasta tehtävällä kyselyllä ja piirtämällä kuvaajat näiden tietojen ja äskeisessä kappaleessa mainittujen parametrien pohjalta. Kun kuvaaja oli piirretty, se tallennettiin IIS-palvelimelle Web-sovelluksen sessiokohtaiseen kuva-alikansioon. Kuva haettiin sitten tuosta alikansiosta Web-sivulla käyttäjälle näyttämistä varten. Tiedon haku ja kuvaajien piirto tapahtuivat sekunnin murto-osissa, mikä oli mahdollista nopeiden tietokantayhteyksien ja riittävien palvelinresurssien ansiosta, eikä tämä kuvaajien reaaliaikainen piirtäminen ja näyttäminen haitannut yhtään järjestelmän käyttöä. Kuvien tyhjennys alikansioista tapahtui hyödyntämällä Web-

sovelluksen sessionhallintaa, jolloin session aikakatkaisun ja päättämisen yhteydessä tyhjennettiin kyseisen session kuva-alikansio, eivätkä kuvat jääneet täyttämään palvelimen levytilaa.

Vaikka graafisten kuvaajien piirto oli kokonaisuudessaan yksi projektin työläimmistä ja loogista ajattelua eniten vaativista osa-alueista, oli niiden tekeminen erittäin palkitsevaa. Toteutus onnistui mielestäni erittäin hyvin ja onkin siksi mielestäni paras yksittäinen osa-alue koko järjestelmässä. Lisäksi kuvaajien piirtoalueiden hahmottamisessa ja jaottelemisessa sai käyttää kunnolla luovuutta ja loogista päättelykykyä, että ratkaisusta saataisiin mahdollisimman selkeä, havainnollinen ja teknisesti toimiva. Tämä toimi lisäksi erittäin hyvänä harjoituksena ja opetteluna Visual Studion piirtotyökalujen, IIS-palvelimen Web-sovellusten hallinnoinnin ja olio-ohjelmoinnin osaamisen osalta. Jatkokehitysideoitakin tästä jäi, muun muassa nuo hylätyt viivakuvaajat ja aiemmin esitellyn Philips-mallin tasovertailun kaltaiset kuvaajat (ks. kuva 3.6) olisi kätevä lisätä järjestelmään.

7.1.2 Sivujen monikielisyys

Idea sivujen monikielisyydestä lähti siitä ajatuksesta, että monikansallisena yhtiönä Stora Enson talon kielenä on englanti, mutta päivittäinen yhteydenpito paikallisesti Suomessa tapahtuu kuitenkin suomen kielellä. Dokumentaatiot tehtiin englanniksi, mutta esimerkiksi Stora Enso Metsän portaalissa oli suomenkielisiä artikkeleita. Näin ajateltiinkin, että järjestelmä voitaisiin toteuttaa useampikielisenä, käyttäjän suosimien kielien mukaan ja järjestelmään saataisiin lisäarvoa mahdollisuudella näyttää järjestelmän tiedot erikielisille käyttäjille heidän parhaiten osaamansa kielen mukaan. Alustaviksi kieliksi valittiin englannin lisäksi suomi ja ruotsi, mutta järjestelmän tuli tukea aivan minkä tahansa muun kielen lisäämistä.

Ensimmäinen haaste oli tietokannan rakenteen muokkaaminen tukemaan skaalautuvasti monikielisyyksiä, mikä ratkaistiin lisäämällä valmiiseen tietokantaan kielet luetteleva LANGUAGE-taulu ja purkamalla tämän taulun monta-moneen-yhteydet muihin tietokannan tauluihin aputaulujen avulla. Muiden taulujen eri kieliä vaativat kentät siirrettiin näihin aputauluihin, jolloin tietokannan tarvitsemien taulujen lukumäärä yli kaksinkertaistui. Tämä ei kuitenkaan ollut suuri on-

gelma, sillä vaikka monikielisyys monimutkaisti tietokannan rakennetta, ei tietokannan koko lopulta ollut kuin 13 taulua.

Seuraavaksi monikielisyys piti ulottaa järjestelmän käyttöliittymään, joka tehtiin käyttämällä Visual Studion monikielisyysominaisuuksia hyväksi. ASP-sivujen monikielisyys toteutettiin kääntämällä järjestelmään luotujen sivujen käyttöliittymät toiselle kielelle ja tallentamalla ne Visual Studion käyttämän nimeämistavan mukaisella kielitunnisteella. Tämä tehtiin sen takia näin, että tietokantayhteyden puuttuessa sivuilla näkyisi edes jotain tekstiä. Jälkikäteen ajatellen myös käyttöliittymän tekstien lisääminen tietokantaan olisi ehkä kuitenkin järjestelmän ylläpidon kannalta ollut parempi ratkaisu. Kun kuitenkin järjestelmän data tallennetaan jo valmiiksi tietokantaan, eikä näin ollen ole saatavissa mahdollisten tietokantayhteyskatkoksien aikana. Tämä ajatus kuitenkin hylättiin, sillä tuon muutoksen teko olisi vaatinut muokkausta tietokannan rakenteeseen, eikä siihen ollut enää aikaa projektin tuossa vaiheessa. Tämä johtui myös riittämättömästä tekniikan tutkimisesta, sillä tajuttaessa ASP.NET:in kielisyystuki, lähdettiin suoraan rakentamaan äsken kuvattua ratkaisua, eikä tutkittu riittävästi mahdollisuutta myös käyttöliittymän tietojen tallentamiseen ja hakemiseen tietokannasta, johon olisi kaiken lisäksi löytynyt suoraan tuki Visual Studiosta.

Kielen näyttäminen ja vaihto tehtiin siten, että käyttöliittymän oikeaan yläreunaan lisättiin pudotusvalikko järjestelmän eri kielille. Lisäksi kielet näytettiin selaimen kielimääritysten mukaisesti, kuten kuva 6.16 havainnollisti, ja käyttäjän haluaman kielen puuttuessa näytettiin järjestelmän oletuskieli eli englanti. Tämän toiminnallisuudessa käyttöönotossa ajanpuute tuli ilmi ja vaikka tuki monikielisyydelle periaatteessa on, sivujen kääntötyö ja täysi monikielisyys jätettiin tuomatta järjestelmän tuotantoversioon. Mikäli järjestelmän monikielisyysvalinta pitäisi tehdä nyt, se olisi todennäköisesti hylätty kokonaan liian työläänä ja aikaa vievänä toteuttaa. Mahdollisena jatkokehitysideana monikielisyyden toteuttaminen olisi tosin ihan toimiva.

8 YHTEENVETO

Opinnäytetyönä tehdystä Prosessin arviointityökalu -tietojärjestelmän määritys- ja toteutusprojektista voidaan kootusti sanoa, että huolimatta työssä havaituista

ongelmista ja työn aikataulun venymisestä, alussa asetetut päätavoitteet saavutettiin. Asiakas sai haluamansa laadun arviointityökalun vuoden 2007 Suomen laatupalkinto -kilpailuun mennessä, vaikka joitakin järjestelmään määriteltyjä työkalun ominaisuuksia ja toimintoja piti karsia. Itse laatupalkintokilpailukin meni asiakkaalta erinomaisesti, sillä se voitti suurten yritysten ja liiketoimintayksiköiden sarjan ollen kyseisen vuoden ainoa voittaja kaikki sarjat huomioiden (Laatukeskus 2011). Myös yhteistyö ja kommunikointi asiakkaan kanssa onnistuivat hyvin sekä asiakkaan että omien sosiaalisten taitojen ansiosta.

Opinnäytetyössä huonommin meni projektin hallinta ja aikataulutus, mikä oli osiltaan ennakoitavissakin, sillä tämä oli ensimmäinen kerta tämän kokoisen projektin päällikkönä. Työtunnit ylittivät reilusti opinnäytetyön vaatimat tuntimäärät ja riittävän ajoissa toimien projekti olisi luultavasti saatu nopeammin valmiiksi ottamalla projektiin esimerkiksi toisen opiskelijan lisäresurssiksi. Onnistuneempi riskienhallintakin olisi auttanut asiaa. Jälkikäteen ajatellen projekti olisi varmaan saatu normaaliin opinnäytetyön työmäärän puitteissa ja alkuperäiseen aikatauluun mennessä valmiiksi, mikäli käytetyt tekniikat ja menetelmät olisivat olleet jo ennestään tuttuja. Tällöin niiden opetteluun ei olisi tarvinnut tuhlata aikaa ja työssä tehdyt ratkaisut olisi voitu rakentaa nopeammin, kun olisi tiedetty miten vastaavanlaista projektia ja tietojärjestelmää rakennetaan annetuilla työkaluilla. Toisaalta taas opinnäytetyössä pystyttiin opettelemaan sellaisia asioita, tekniikoita ja projektinhallintaa, joita ei todennäköisesti olisi muuten pystynyt oppimaan. Työstä saatu kokemus ja opitut asiat auttavat varmasti tulevaisuudessa, joten näin ajatellen tehdyt työmäärä ja virheet eivät olleet turhia.

Vastaavanlaisen projektin sattuessa kohdalle osaa jatkossa varautua aikataulutuksen ja riskienhallinnan tärkeyteen, sekä varata riittävästi aikaa uusien asioiden ja tekniikoiden opetteluun. Myös systeemityömallin valintaa ja soveltamista osaa miettiä paremmin, mikä osaltaan auttaa projektin hallintaa ja sen eri vaiheiden läpikäyntiä merkittävästi. Myös projektin rajaukseen tulee jatkossa kiinnitettyä paremmin huomiota, että osaa suhteuttaa toteutettavat ominaisuudet ja toiminnot käytettävään aikamäärään ja muihin resursseihin. Lisäksi toteutetusta tietojärjestelmästä jäi kehitysideoita, muun muassa aliluvuissa 7.1.1 ja 7.1.2 mainitut alaprosessien taso- ja tavoitehistorioiden viiva- ja vertailukuvaajat sekä

järjestelmän ASP-sivujen monikielisyyden rakentaminen kaikki tiedot suoraan tietokannasta hakevaksi ja tallentavaksi kokonaisuudeksi.

KUVAT

TAULUKOT

LÄHTEET

Aked, M. 2003. Risk reduction with the RUP phase plan. IBM Technical library.
http://www.ibm.com/developerworks/rational/library/1826.html
(Luettu 9.8.2011)

EFQM - Philips. 2004. Process Survey Tool - A guide to using and applying PSTs.
(Luettu 24.11.2011)

EFQM - Philips. 2004. Process Survey Tool for Human Resources Management.
(Luettu 24.11.2011)

Ekonoja A., Lahtonen T. & Mäntylä J. 2011. Henkilökohtaisen tiedonhallinnan perusteet.
http://appro.mit.jyu.fi/doc/tiedonhallinta
(Luettu 28.11.2011)

European Foundation for Quality Management. EFQM > Home.
http://www.efqm.org
(Luettu 29.8.2011)

Haikala I. & Märijärvi J. 2006. Ohjelmistotuotanto. 11. painos. Helsinki: Talentum.

Holvikivi, J. 2005. Sovelluskehitys: systeemityö.
http://users.evtek.fi/~jaanah/SovKehTuta/systeemityo.htm
(Luettu 11.9.2011)

Immonen, J. 2003. Johdatus ohjelmistotuotantoon - luentomoniste.
http://cs.joensuu.fi/~jimmonen/jot_moniste/jot_moniste_121.html
(Luettu 21.9.2011)

Jyväskylän yliopisto. Opetuksen laatuprosessi.
http://www.jyu.fi/hallinto/oplaapro/laatuprosessi
(Luettu 29.8.2011)

KK Mediat. 2011. ASP.NET opas: Johdatus ASP.NET:n maailmaan.
http://www.2kmediat.com/dotnet/aspnet1.asp
(Luettu 3.12.2011)

Kalliala, E. 1999. Miksi olioita systeemityöhön.
http://myy.helia.fi/~kalei/oliomats/oliot_systyossa.html
(Luettu 11.9.2011)

Kankaanpää, T. 2004. Ohjelmiston määrittely.
http://www.cc.puv.fi/~tka/kurssit/Ohjelmiston_maarittely/luennot.htm
(Luettu 21.9.2011)

Keto, U. & Malinen H. 2007. Itsearviointi laatujärjestelmän osana. KeVer 6.
http://ojs.seamk.fi/index.php/kever/article/viewArticle/1015/864
(Luettu 7.9.2011)

Laatukeskus. 2011. Laatukeskuksen verkkosivut.
http://www.laatukeskus.fi
(Luettu 22.11.2011)

Microsoft. 2011. Lisätietoja Microsoft .NET Framework Version 2.0 Redistributable Package (x86).
http://www.microsoft.com/downloads/fi-fi/details.aspx?FamilyID=0856eacb-4362-4b0d-8edd-aab15c5e04f5
(Luettu 3.12.2011)

Opetushallitus. 2011. Laadunhallinnan tuki.
http://www.oph.fi/saadokset_ja_ohjeet/laadunhallinnan_tuki
(Luettu 22.11.2011)

Paajanen, T. 2011. Innonet-toiminnanohjausjärjestelmä. Opinnäytetyö. Saimaan ammattikorkeakoulu.

Philips. 2011. Philipsin verkkosivut.
http://www.philips.fi
(Luettu 22.11.2011)

Pitkänen, T. 2011. Innoweb-toiminnanohjausjärjestelmä. Opinnäytetyö. Saimaan ammattikorkeakoulu.

Saimaan ammattikorkeakoulu, julkaisuryhmä. 2010. Opinnäytetyöraportin kirjoitusohje. Lappeenranta - Imatra

Stora Enso. Stora Enso Metsä.
http://www.storaenso.com/wood-forest/stora-enso-metsa
(Luettu 11.5.2011)

Taggart, G. 2008. Open Source Scripting in Second Life: Getting Ready for LSL-Mono. The Seventh Sun.
http://www.theseventhsun.com/0208_gigsCorner008.htm
(Luettu 3.12.2011)

Terveyden ja hyvinvoinnin laitos (THL). 2011. Hyvä käytäntö.
http://www.sosiaaliportti.fi/hyvakaytanto
(Luettu 2.9.2011)

Tuurala, T. 2010. Laatuakatemia.
http://www.kotiposti.net/tuurala
(Luettu 2.9.2011)