Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma
Anssi Lahtinen

Opinnäytetyö

HUOLLON YDINPROSESSIN TESTAUKSEN SUUNNITTELU OSANA ERP-JÄRJESTELMÄN KÄYTTÖÖNOTTOA

Työn ohjaaja FM, Lehtori Anne-Mari Sainio
Työn tilaaja John Deere Forestry Oy

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma

Tekijä	Anssi Lahtinen
Työn nimi	Huollon ydinprosessin testauksen suunnittelu osana ERP-järjestelmän käyttöönottoa
Sivumäärä	52
Valmistumisaika	31.12.2008
Työn ohjaaja	Anne-Mari Sainio
Työn tilaaja	John Deere Forestry Oy

Tiivistelmä

Tämän työn tarkoituksena on esitellä, mitä ohjelmistotestaus on ja mitä sen avulla pyritään saavuttamaan sekä miten toimeksiantajan tapauksessa toteutettiin testauksen suunnittelutyö. Pääpaino on ennen järjestelmän käyttöönottoa suoritettavassa toiminnallisessa testauksessa. Myös yrityksen huoltotoiminnan kehittämistä ja sen asettamia vaatimuksia järjestelmälle ja testaukselle on tarkasteltu.

Testaussuunnittelun tavoitteena on saada aikaan kattava testisuunnitelma, joka varmistaa että kaikki toiminnalliset vaatimukset tulee testattua. Suunnittelu keskittyy ohjelman käyttäytymiseen ja olen käyttänyt hyväkseni mustalaatikkotekniikkaa. Työskentelen toimeksiantajan palveluksessa huolto-osaston järjestelmäasiantuntijana.

Opinnäytetyön tuloksena on paitsi ohjelmistotestausta avaava teoreettinen osuus, niin myös kertomus yhden yrityksen toimenpiteistä matkalla kohti tuottavampaa liiketoimintaa. Työn tarkoituksena ei ole olla ohje, vaan pikemminkin valistava ja informatiivinen kokonaisuus siitä, miten testauksella vaikutetaan lopulliseen tulokseen ja mitä vaatimuksia toimeksiantaja asetti huollon moduulille. Tästä eteenpäin työ jatkuu huollon moduulin toiminnan testauksella, uusien ominaisuuksien opettelulla, koulutusmateriaalin ja manuaalin tekemisellä sekä tulevien käyttäjien kouluttamisella.

Avainsanat	ohjelmistotestaus, ydinprosessi, testaaminen, mustalaatikkotestaus, toiminnallinen testaus

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma

Tekijä	Anssi Lahtinen
Työn nimi	Planning the service core process testing as a part of ERP-system implementation
Sivumäärä	52
Valmistumisaika	31.12.2008
Työn ohjaaja	Anne-Mari Sainio
Työn tilaaja	John Deere Forestry Oy

Abstract

The purpose of this thesis is to show what a software testing includes, what its goals are and how the test planning was executed in the case of John Deere Forestry. The main focus is on the functional testing, which is performed prior to a system launch. The development of the company's service functions and the requirements that they bring to the system and the testing are also discussed.

The goal of this test planning is to create extensive test plan, which ensures that system's all functional requirements will be tested. Focus of this planning is on software's behaviour and I have utilized black-box testing technique. I work as a system expert in the service department of John Deere.

Results of this thesis are on one hand in the theoretical part, which opens the concept of software testing and on the other hand in a story of one company's actions on their way towards more productive business. This is not supposed to be a set of instructions, but rather an informative package about how testing can help to achieve the set goals and what kind of requirements the employer has set for a service module. As for now, the work continues by testing the service module in ERP, learning new features about it, making the training materials and manuals, and ultimately training new users.

Keywords: software testing, core process, testing, black-box testing, functional testing

Esipuhe

Tämän työn valmistumisen myötä saavutan yhden merkittävän tavoitteen elämässäni. Työn tekeminen on ollut opettavaista ja antanut paljon arvokasta kokemusta jatkoa ajatellen. Haluan kiittää omia vanhempiani heidän antamastaan tuesta. Haluan kiittää myös John Deere Forestry Oy:n Anssi Kumpulaista, Mikko-Pekka Happosta ja Pertti Rauvaa siitä, että antoivat minulle tämän hienon mahdollisuuden.

_____

Anssi Lahtinen

Sisällysluettelo

1 Johdanto

Tämä työ jakautuu kahteen eri osioon. Ensimmäinen osio esittelee ohjelmistotestauksen perusteita ja sen roolia ohjelmistokehityksen ja ohjelmiston käyttöönoton osana. Se kertoo myös, mitä testauksella viime kädessä saavutetaan ja mitä muuta se on virheiden löytämisen lisäksi. Erityisesti työ painottaa toiminnallista testausta, sekä sen eri tekniikoita mustalaatikkotestausta hyväksikäyttäen. Osana laajempaa ohjelmistotestauksen viitekehystä on lyhyesti tarkasteltu myös sitä, minkälainen on itse testaaja.

Toinen osio liittyy työhöni John Deere Forestry Oy:n palveluksessa. Olen ollut mukana kahdessa itsenäisessä, mutta toisiinsa tiukasti nivoutuvassa projektissa. Näistä toinen on yrityksen liiketoimintaa kehittävä Proserve-niminen projekti, ja toinen ERP-järjestelmän implementointiin liittyvä projekti.

Kerron myös siitä, miten John Deerellä on alettu kehittää metsäkonehuollon toimintaa ja mitkä syyt ovat johtaneet kehitystarpeisiin. Liiketoiminnan kehittämisen puolella on keskitytty ydinprosessin merkitykseen huoltotoiminnassa, ja huollon eri tehtäviin jälleenmyyntiorganisaatiossa.

Järjestelmäprojekti liittyy siihen, miten ERP voi auttaa näiden kehitystavoitteiden saavuttamisessa. ERP on lopulta hyvin merkittävässä roolissa, ja siksi myös sen testaaminen on avainasemassa matkalla kohti onnistunutta toimintatapamuutosta.

Esittelen, mitä tavoitteita ja vaatimuksia toiminnallisen vaiheen testauksen suunnitteluun liittyi, ja miten testaus on suunniteltu skenaarioiden ja testitapausten osalta. Varsinaiseen testaustyöhön en ole keskittynyt, sillä se ei ole vielä ehtinyt alkaa kunnolla.

2 Ohjelmistotestaus

Oman näkemykseni on, että ohjelmistoprojekti koostuu karkeasti kahdesta eri osa-alueesta: kehitys ja testaus. Ohjelmiston kehitys tarkoittaa vaatimusten määrittelyä, dokumentointia ja itse ohjelmakoodin kirjoittamista. Se on ollut yleensä se osa-alue johon ohjelman tuottajat sekä hankkijat ovat kiinnittäneet enemmän huomiota.

Testaus sen sijaan on jäänyt vähemmälle huomiolle. Ainakin tämän työn toimeksiantajan tapauksessa se on ollut asia, joka hoidetaan projektin lopussa jäljellä olevan ajan puitteissa. Itse asiassa testaaminen on vähintään yhtä tärkeä osa onnistunutta ohjelmistoprojektia kuin varsinaisen ohjelman luominen. Hyvin suunniteltu ja toteutettu testausprojekti voi vähentää huomattavasti ohjelman korjaamisesta myöhemmin koituvia taloudellisia tappioita.

2.1 Testauksen maalit

Mihin testaus sitten todella tähtää? Ohjelmistotestauksen, kuten kaiken muunkin testauksen, tavoitteena on taloudellisten riskien vähentäminen. Se tapahtuu käyttämällä oikeita tekniikoita ja luovuutta. Samalla ohjelmalle tuotetaan lisäarvoa havaitsemalla virheet jo ennen käyttöönottoa. (Myers 2004, 6.)

Tarkoituksena ei ole löytää kaikkia ohjelman virheitä, sillä niitä on aina, ja niiden löytämiseen menisi tuhottoman paljon aikaa. Sen sijaan on keskityttävä löytämään ne virheet, joilla on todellista merkitystä käyttäjän kannalta. (Loveland, Miller, Prewitt & Shannon 2005, 6.)

2.2 Testauksen vaatimuksia

Ensimmäisenä vaatimuksena on tutkia, sisältääkö uusi ominaisuus tai tuote riskejä, ja jos sisältää, niin minkälaisia. Tällöin puhutaan riskien tunnistamisesta. Riskien tunnis-

tamisella tarkoitetaan sitä, että riskien suuruus ja olinpaikka selvitetään mahdollisimman tarkkaan. Mikäli riski on suuri ja todennäköisyys sen toteutumiselle on suuri, on syytä miettiä testausta riskin vähentämiseksi. Hyvin suunniteltu ja läpiviety testaus vähentää riskejä, mutta vain, jos ne ovat tarkkaan tiedossa. (Everett & McLeod 2007, 10.)

Kuten todettua, kaikkien mahdollisten asioiden testaaminen ei ole käytännössä mahdollista, ja tästä syystä mikään riski ei poistu testauksella kokonaan. Siksi on tärkeää lähestyä testaamista oikealta kantilta. Itse testaaminen koostuu sekä positiivisesta että negatiivisesta lähestymistavasta. Positiivisella tarkoitetaan sitä, että testataan toimiiko ohjelma halutulla tavalla. Negatiivinen tapa puolestaan keskittyy virheiden ja murtumiskohtien löytämiseen. Suurin osa tehtävästä työstä pitää keskittää riskialueiden kanssa työskentelyyn, ja molemmat tavat ovat tarpeen sekä suunnittelussa että itse testausvaiheessa. (Everett yms. 2007, 10.)

Riskien ja niiden vähentämisen lisäksi on pohdittava, milloin testaus on valmis. Jokaisen testausprojektin pahin vihollinen on resurssien rajallisuus. Joko testausvaiheeseen varattu raha on loppumassa, tai yksinkertaisesti aika on käytetty loppuun. Silloin on oltava selvillä, mihin testaamisella halutaan vastauksia.

Tärkein asia on tietenkin, että ohjelma toimii halutulla tavalla, joten on suoritettava positiivinen testaus, jolla osoitetaan että ohjelma todella toimii siten kuin vaatimuksissa on määritelty. Seuraavana työlistalla on riskikohtien testaaminen, kuitenkin niin, ettei siihen mene liikaa resursseja. Hyvänä nyrkkisääntönä pidetään, että riskien testaus on hyödyllistä, mikäli siihen käytettävä rahamäärä on 10 - 20 % riskin arvioidusta suuruudesta. Seuraavaksi tärkein asia on negatiivinen testaus. Tällöin testataan kiellettyjä arvoja, tai yritetään ennakoida käyttäjien toimintaa sellaisissa tilanteissa joita ei mainita testisuunnitelmassa. (Everett yms. 2007, 10.)

Myös asenteilla on suuri merkitys testauksessa. Testausta ei kannata pitää vähäpätöisenä asiana, jota ei erikseen tarvitse suunnitella. Koska hyvä testaaminen tuo säästöä, on järkevää panostaa siihen kunnolla.

2.3 Testaaja

Testaajan rooli on tärkeä, mutta oman kokemukseni mukaan aliarvostettu. Ammattilaisten sijaan varsinaisen työn suorittajina kuulee usein käytettävän loppukäyttäjiä, joilla ei välttämättä ole silmää nähdä oikeita asioita. Myöskään ohjelman kehittäjä ei ole paras henkilö testaamaan tuotetta, koska hänellä on siihen liian läheinen suhde. (Loveland yms. 2005, 8-9.)

Huomionarvoinen seikka on myös se, ettei testaaminen ole pelkästään virheiden löytämistä ohjelmasta. Ison ohjelmiston parissa saattaa työskennellä useita eri henkilöitä, jotka kaikki eivät välttämättä ole loppukäyttäjiä. Joukossa on useita erilaisia ammattilaisia, jotka hoitavat esimerkiksi tietokantoja tai tietoturvaa. Koska kaikilla on omat vaatimuksensa ohjelmistolle, pitää testaajan olla teknisesti monipuolinen, ja pystyä ymmärtämään myös näiden vaatimukset. (Loveland yms. 2005, 8-9.)

Testaajan asenne saattaa olla jopa ratkaisevampi kuin yksittäinen testausprosessi. Ajattelu pitää suunnata siten, että ohjelman toimivuuden lisäksi siitä todella etsitään virheitä. Samalla tavalla asennetta pitää muokata siihen suuntaan, ettei kaikkia virheitä tarvitse löytää. (Myers 2007, 5-6.)

Testaajat ovatkin oma ammattikuntansa ja päinvastoin kuin ohjelmistojen kehittäjät, testaajat haluavat törmätä virheisiin. He haluavat murtaa oletuksia ja iskeä ohjelmistojen heikkouksiin. Testaajille hyödyllisiä ja tarpeellisia luonteenpiirteitä ovat muun muassa uteliaisuus, skeptisyys, levottomuus, positiivisuus ja diplomaattisuus. (Loveland yms. 2005, 10-11.)

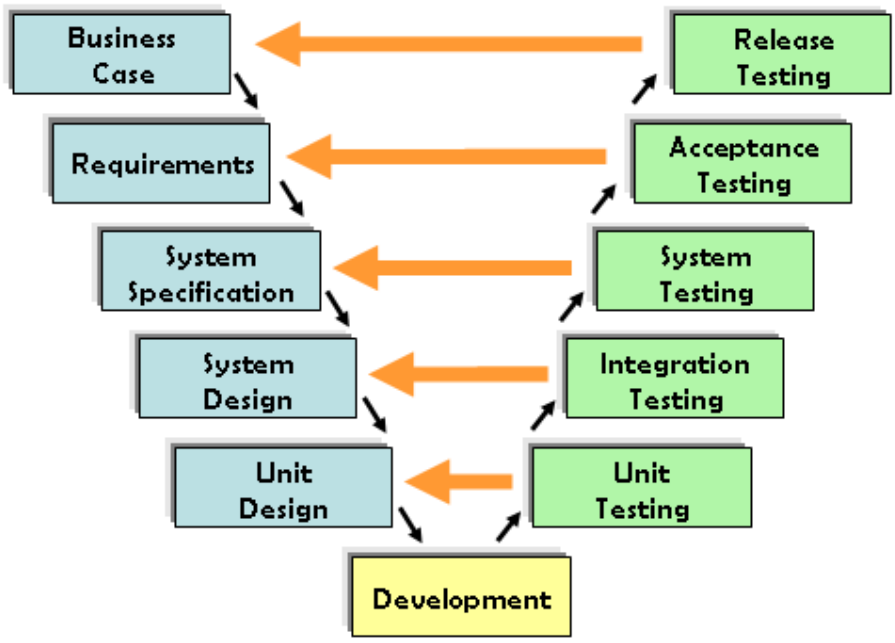
3 Testisuunnittelu

3.1 Testauksen elinkaari

Ohjelmiston tuottamista kuvataan monilla eri malleilla. Yleisimpien joukossa on tässä työssä esitelty V-malli (kuvio 1). Se kuvaa ohjelman kehityksen ja testauksen elinkaarta, ja sitä miten vaiheet sulautuvat toisiinsa.

Vasemmalla puolella kulkee ohjelmiston kehitys siten, että se alkaa ylhäältä vaatimusten määrittelystä ja päättyy V:n kärkeen koodin kirjoittamiseen. Oikealla puolella kuvataan testauksen vaiheita kehitysprosessin aikana. Poikittaisnuolet kuvaavat analyysi-, suunnittelu- ja implementaatiotehtäviä. (Black 2007, 24.)

Kunkin testivaiheen tuloksia verrataan kehitysvaiheen dokumentteihin. Esimerkiksi vaatimusmäärittelyä käytetään apuna järjestelmätestauksessa. Tämä määrittely yhdistyy ohjelmistokehitysprosessin alkuun, ja osoittaa että tämän testivaiheen virheitä voidaan vähentää jo ennen koko testausprosessin alkua. Samalla tavalla jokainen vaihe saa oman ”parinsa”. (Black 2007, 24.)



Kuvio 1. Esimerkki V-mallista. (Smoothlab 2007)

Testivaiheista on monta erilaista tulkintaa ja myös terminologia vaihtelee. Prosessi on kuitenkin aina sama. Seuraavissa kappaleissa on esitelty tärkeimmät vaiheet.

Yksikkötestaus

Yksikkötestaus on ensimmäinen vaihe testausprosessissa. Se keskittyy ohjelman koodin testaamiseen. Koodista testataan sen logiikkaa, syöttötietoja, tulosteita ja toiminnallisuutta. (Loveland yms. 2005, 29-30.)

Integraatiotestaus

Integraatiotestaus tarkoittaa ohjelman sisäisten kokonaisuuksien testaamista yhdessä. Ohjelmat koostuvat moduuleista, joiden tarkoitus on toimia vuorovaikutuksessa keskenään. Integraatiotestauksessa etsitään virheitä ohjelman toiminnassa. Lähdekoodi on edelleen käytössä, mutta ohjelmaa testataan korkeammalla tasolla kuin yksikkötestauksessa. (Loveland yms. 2005, 30-32.)

Järjestelmätestaus

Järjestelmätestaus tutkii kokonaista ohjelmistoa. Koodista on rakennettu moduuleita ja ohjelmia, ja nämä yhdistämällä on saatu aikaan kokonainen järjestelmä. Tässä vaiheessa testataan muun muassa järjestelmän kykyä kestää useita käyttäjiä ja kovaa kuormitusta. (Loveland yms. 2005, 32-34.)

Järjestelmän integraatiotestaus

Periaate on sama kuin aiemmin esitellyssä integraatiotestauksessa, mutta ohjelmistoa testataan yhdessä muiden järjestelmäkokonaisuuteen kuuluvien ohjelmien kanssa. Kyseessä on siis testaus, joka varmistaa ohjelman toimivuuden yhdessä asiakkaan muiden ohjelmien kanssa. (Loveland yms. (2005, 37-38.)

3.2 Testitekniikat

Testitekniikat voidaan karkeasti jakaa kolmeen pääryhmään: staattinen testaus, valkolaatikkotestaus (white-box testing) ja mustalaatikkotestaus (black-box testing). Testausprojektin aikana tulisi käyttää kaikkia edellä mainittuja tekniikoita. Tekniikat on esitelty seuraavissa kolmessa aliluvussa: (Black 2007, 45.)

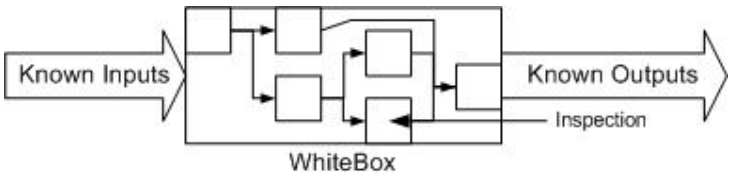
3.2.1 Staattinen testaus

Staattinen testaus tarkoittaa ohjelman dokumentaation testaamista. Sen tarkoituksena ei ole testata ohjelmakoodia, vaan varmistaa, ettei virheitä synny puutteellisen dokumentaation vuoksi. Yhden projektin aikana syntyy paljon dokumentteja, kuten ohjelman suunnitteludokumentit, koulutusmateriaali ja käyttöohjeet. Suuri määrä erilaisia dokumentteja sisältää helposti myös suuren määrän virheitä ja siksi testaaminen on suositeltavaa (Everett yms. 2007, 67). Tämä tekniikka tuo parhaimmillaan myös selvää säästöä projektin menoissa, koska virheet on huomattavasti halvempi korjata ennen varsinaisen koodaustyön aloittamista. (Black 2007, 46.)

3.2.2 Valkolaatikkotestaus

Valkolaatikkotestaus (kuvio 2) tutkii ohjelman sisäistä rakennetta ja koodia. Sen tarkoituksena on löytää virheet ohjelman toiminnoissa, tietorakenteissa ja käyttöliittymissä. Tämä menetelmä on käytössä testauksen elinkaaren alussa muun muassa yksikkötestaus- ja integraatiotestausvaiheessa. (Black 2007, 46.)

Testaajat ja ohjelmiston tekijät työskentelevät tässä vaiheessa yhdessä. Ohjelman koodin kirjoittaja tietää, miten sen tulisi toimia ja vastaavasti testaaja pyrkii löytämään ne kohdat joissa koodi ei toimi halutulla tavalla. Edellä mainituista syistä johtuen positiivista ja negatiivista testausta yhdistämällä saadaan valkolaatikkotestauksesta tehokkain tapa löytää virheitä ohjelmasta. (Everett yms. 2007, 68.)

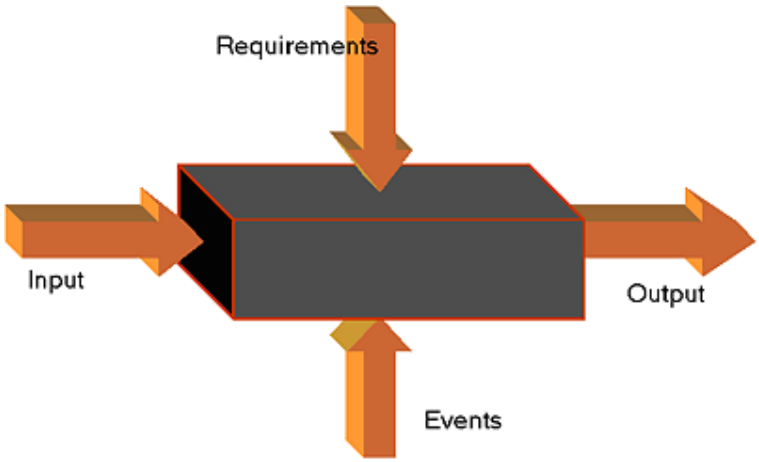


Kuvio 2. Valkolaatikkotestaus. (Clifton 2003)

3.2.3 Mustalaatikkotestaus

Mustalaatikkotestauksen tarkoituksena on testata ohjelman käyttäytymistä, tarkemmin sanoen sen prosessien toimivuutta. Jokaiselle prosessille on määritelty tietyt syötearvot, joilla se käynnistyy, sekä tulosteet joita se tuottaa. (kuvio 3) (Black 2007, 46.)

Tässä testauksessa ohjelman lähdekoodi ei ole käytössä lainkaan, eikä testi ota kantaa ohjelman sisäiseen rakenteeseen. Sen sijaan tulee tietää, mitä ohjelman odotetaan tekevän ja testata tekeekö se todella sen. Tätä tekniikkaa voidaan käyttää monessa eri testausprojektin vaiheessa eri tarkoituksiin. (Everett yms. 2007, 68.)



Kuvio 3. Mustalaatikkotestaus. (Elpida)

3.3 Toiminnallinen testaus

Toiminnallisen testauksen tavoitteena on testata ohjelman toimivuus ja käyttäytyminen suhteessa dokumentoituihin vaatimuksiin. Varsinkin yrityksen liiketoiminnan kannalta kriittiset toiminnot ovat suurennuslasin alla. (Nair 2008.)

Ohjelman toimivuutta testattaessa lähestymistapana käytetään mustalaatikkotestausta. Sen tekniikat ovat seuraavat: ekvivalenssiluokitus, raja-arvoanalyysi ja odotettuihin tuloksiin perustuva analyysi. (Everett yms. 2007, 112-117.)

Ekvivalenssiluokituksessa syöttötiedot jaetaan ryhmiin. Ryhmän sisällä oleva data aiheuttaa ohjelmassa aina samanlaisen tapahtuman, eli ohjelma käyttäytyy samalla tavalla. Tämän tekniikan etuna on, että sitä käytettäessä voi testien määrän pudottaa hyvin pieneksi. Parhaimmillaan riittää että kaikista mahdollisista syöttötiedosta käytetään vain kolme prosenttia. (Everett yms. 2007, 112-117.)

Raja-arvoanalyysi perustuu nimensä mukaisesti syöttötietojen ja tulostetietojen raja-arvoihin. Jokaisen ekvivalenssiluokan data omaa minimi- ja maksimiarvon. Näitä arvoja käytetään ei-toivottujen arvojen määrittelyssä. Jos ohjelmassa on esimerkiksi kenttä johon voidaan syöttää arvoja väliltä 1-99, ovat -1 ja 100 tällöin vääriä syöttöarvoja, joita ohjelman ei tulisi sallia. (Everett yms. 2007, 112-117.)

Kahden edellä mainitun tekniikan keskittyessä syöttötietoihin, odotettuihin tuloksiin perustuva tekniikka keskittyy ohjelman (tai prosessin) tulostetietoihin (output). Jotta datan määrittely onnistuu, pitää tuntea yrityksen liiketoiminnan tapahtumat, ja jokaisen tapahtuman odotettu tulos. Syöttötietoina käytetään sekä oikeita että vääriä arvoja, ja näiden odotettu tuloste kirjataan ylös. Odotettuja tuloksia verrataan sitten itse testin tulokseen. (Everett yms. 2007, 112-117.)

3.4 Testisuunnitelma

Testisuunnitelma kertoo, mitä ollaan testaamassa ja miksi se tehdään. Se toimii myös työkaluna monelle tekijälle hankkeen eri vaiheissa. Eri vaiheiden testaajat käyttävät sitä varmistaakseen, että kaikki olennainen tieto on otettu huomioon ja välttävät sen avulla turhia päällekkäisyyksiä. Uudet testaajat voivat käyttää sitä oppimismateriaalina, ja asiakas voi todeta siitä että kaikki hänen haluamansa asiat todella ovat suunnitelmassa.

Periaatteessa testisuunnitelman voi tehdä monella eri tavalla. Sen kuitenkin tulee olla riittävän kattava ja samalla riittävän ymmärrettävä, että siitä on hyötyä jokaiselle sitä tarvitsevalle. Yksinkertainen suunnitelma sisältää esim. yleiskatsauksen projektiin, testiympäristön ja testityökalujen kuvauksen, testauksen vastuuhenkilöt ja dokumentin levitykseen liittyvät säännöt. Nämä asiat pysyvät samana testistä toiseen, ja niistä voidaan tehdä valmis pohja, joka liitetään aina uuden vaiheen suunnitelmaan. (Loveland yms. 2005, 108-109.)

Testisuunnitelman vaihtuva osa ovat testiskenaariot, joissa kuvataan hieman tarkemmin testattavat asiat. Skenaarioita ei tarvitse kuvata kovin yksityiskohtaisesti, varsinkaan jos suunnitelma on tulossa käyttöön testaajalle itselleen. Usein riittää korkean tason kuvaus, niistä joita testataan ja mitä ne sisältävät. (Loveland yms. 2005, 108-109.)

3.5 Testicaset

Testicase, tai testitapaus, vastaa kysymykseen: miten jokin tehdään? Testitapaukset luodaan joko skenaarion tai käyttötapausten pohjalta siten, että ne kattavat kaiken oleellisen toiminnallisten vaatimusten saavuttamiseksi. Esimerkkinä skenaariosta voisi olla vaikkapa tuotteen myynti, ja tästä luotuina testitapauksina myynnin kirjaaminen järjestelmään ja laskun teko. (Everett yms. 2007, 84.)

Testitapausten rajaaminen voi olla hankalaa, koska niistä tulee helposti kovin laajoja. Onkin tärkeää keskittyä löytämään oikeasti merkitsevät toiminnot ja ymmärtää niiden käyttötarkoitus. Tällöin myös testitapausten luominen on helpompaa. (Everett yms. 2007, 84.)

4 Yritys

John Deere on vuonna 1837 perustettu raskaskoneiden valmistaja. Yhtiön toimialaan kuuluvat maanrakennus- ja metsäkoneet, ympäristökoneet sekä moottorit. Yhtiö työllistää maailmanlaajuisesti n. 47 000 henkilöä. John Deere on yksi USA:n vanhimmista teollisuusyrityksistä. (John Deere 2008)

John Deeren tuotanto on jakautunut neljään eri divisioonaan. Näitä ovat: maatalouskoneet, maanrakennus- ja metsäkoneet, kuluttajatuotteet ja voimajärjestelmät. Lisäksi yritykseen kuuluvat rahoituspalvelut ja varaosat. (John Deere 2008)

Kuvio 4. (John Deere 2008)

4.1 John Deere Forestry

John Deere Forestry on Deere & Companyn uusin toimiala. John Deere osti Timberjack-nimisen metsäkoneyrityksen Metso Oy:ltä vuonna 2000. Vuonna 2005 Timberjackista tuli John Deere Forestry. Aiemmin sama metsäkoneyhtiö on ollut Suomessa sekä Lokomon että UPM:n omistuksessa. (John Deere 2008)

John Deere Forestry kuuluu yrityksen raskaskonedivisioonaan ja toimii maailmanlaajuisesti. Se on tällä hetkellä maailman markkinajohtaja metsäkonealalla. Yritys tuottaa sekä tavaralajimenetelmän että kokopuun harvesterijärjestelmiä. (John Deere 2008)

John Deere Forestry Oy

Yrityksen tuotantolaitos sijaitsee Joensuussa. Tehtaassa on kaksi tuotantolinjaa, toinen kuormakoneille ja toinen harvestereille. Lisäksi tehtaassa toimii harvesteripäiden tuotantolinja. Yrityksellä on Euroopassa myyntiyhtiöt Suomen lisäksi, Ranskassa, Iso-Britanniassa, Irlannissa, Ruotsissa, Norjassa ja Venäjällä. (John Deere 2008)

Suomen pääkonttori sijaitsee Tampereella. Tampereen konttori työllistää 249 henkilöä, joista 45 jälleenmyynnin puolella. Koko Suomessa John Deere Forestry työllistää 720 henkilöä. Jälleenmyyntiorganisaation tehtävänä on metsäkoneiden myynti, huolto, tekninen tuki ja varaosien myynti. Jälleenmyyntiorganisaatio kattaa koko Suomen, ja toimipaikkoina ovat Tampere, Rovaniemi, Seinäjoki, Joensuu, Taavetti, Mikkeli, Suolahti ja Kajaani. Kaiken kaikkiaan jälleenmyynnin piirissä toimii Suomessa 80 henkilöä. (John Deere 2008)

4.2 Metsäkoneet

Suomessa valmistettavia ja käytettäviä metsäkoneita on kahta eri tyyppiä: kuormatraktorit ja harvesterit. Kuormatraktorit (kuvio 5) ovat pyöräalustaisia koneita, joiden tehtävänä on kuljettaa puut metsästä tien varteen, josta autot hakevat ne vieden ne edelleen sahoille. (John Deere 2008: Tuotespesifikaatio D-sarja)

Kuvio 5. John Deere 1410D-kuormakone (John Deere 2008)

Harvesterit (kuvio 6) ovat kehittyneitä puun kaatoon käytettäviä koneita. Niissä on puun kaatamiseen käytettävä harvesteripää, joka mittaa ja katkoo puut tietokoneen ohjeiden mukaisesti. (John Deere 2008: Tuotespesifikaatio D-sarja)

Kuvio 6. John Deere 1270D-harvesteri (John Deere 2008)

Metsäkoneiden kokoluokka on karkeasti seuraavanlainen:

	paino	pituus
harvesterit	11 500-19 500 kg	5900 mm-7715 mm
kuormakoneet	11 500-20 500 kg	8100 mm-10 900 mm

Metsäkoneet sisältävät raudan lisäksi paljon nykyaikaista automatiikkaa ja tietotekniikkaa, mm. mittalaitteissa ja ohjausjärjestelmissä. (John Deere 2008: Tuotespesifikaatio D-sarja)

5 ERP-projektin tausta

ERP tulee sanoista Enterprise Resource Planning. Lyhyesti ja ytimekkäästi ERPin tehtävänä on sisällyttää yrityksen eri osastot ja niiden toiminnot yhteen tietojärjestelmään. Tarkemmin sanottuna ERP on ohjelmisto, joka yhdistää eri osastojen ohjelmistot keskenään yhdeksi toimivaksi kokonaisuudeksi. Sen edut tulevat esiin ns. back-office -toiminnoissa, jotka tapahtuvat järjestelmän taustalla. (Wailgum 2008.)

Jotta ERP:tä voidaan hyödyntää parhaiten, yrityksen tarvitsee määritellä omat toimintamallinsa (Wallace & Kremzar 2001, 180). Siksi tässä työssä on esitelty myös huollon prosesseja, ja etenkin huollon ydinprosessia.

ERP parantaa yrityksen toimintaa, mahdollistamalla raportoinnin tehokkaamman käytön ja parantamalla myös yrityksen sisäistä tiedonkulkua ja logistiikan hallintaa. Jopa työssä jaksamista ja työssä viihtymistä voidaan parantaa, ei ehkä juuri ERPin avulla, mutta etenkin helpommin sujuvien tehtävien avulla.

ERPin yksi parhaista valteista on siinä, että kaikkien osastojen toiminta on nähtävissä kaikille. Huollon tapauksessa tällä on suuri merkitys. Aiemmin ongelmana saattoi olla, ettei huolto tiennyt mitään esimerkiksi työlle tilattujen osien kohtalosta. ERPin myötä tällaiset asiat poistuvat järjestelmän osalta.

Kyseiseen ERP-projektiin liittyy olennaisena osana yrityksessä aiemmin aloitettu projekti, joka on nimeltään Proserve. Proserven keskeisenä tavoitteena on liiketoiminnan kehittäminen, ja sen rooli on merkittävä, koska metsäkonealalla on odotettavissa taantumaa.

Proserve aloitettiin jo ennen kuin ERPin käyttöönotosta oli edes tietoa. Hyvin nopeasti kuitenkin selvisi, että yksi suurimmista kompastuskivistä ja tavoitteisiin pääsemisen esteistä oli yrityksen vanha toiminnanohjausjärjestelmä. Tämän vuoksi aloitettiin hetkeä myöhemmin myös tietojärjestelmän uusimisprojekti. Toimittajaksi valittiin ruotsalainen IFS.

Yrityksen vanha järjestelmä ei vastannut enää 2000-luvun tarpeita. Kyseinen MFG Pro -järjestelmä oli otettu käyttöön 1990-luvun alussa, ja oli pikemminkin tuotannon järjestelmä, eikä täten soveltunut jälleenmyyntiorganisaatioon kunnolla.

MFG Pro aiheutti ongelmia sillä, ettei se ollut kovinkaan käyttäjäystävällinen, vaan vaati monessa kohtaa suuren määrän työtä. Esimerkkinä voidaan mainita varaosatilaus, joka sisältää 60 riviä. Tämän tekeminen vanhalla järjestelmällä kesti noin tunnin. Tästä johtuen etenkin toimipisteet, joissa oli vain yksi työntekijä, olivat ongelmissa järjestelmän kanssa.

Toinen suuri ongelma oli riittävän raportoinnin saaminen. MFG toki tarjosi nämä mahdollisuudet, mutta raporttien tekeminen ja oikeiden ”näyttöjen” löytäminen oli vaikeaa. Osasyy tähän oli aikoinaan puutteelliseksi jäänyt räätälöinti, joka sekoitti paljon eri osastojen toimintaa. Käytännössä vain kaksi käyttäjää koko jälleenmyyntiorganisaatiossa osasivat näitä raportteja ajaa.

5.1 Proserve

Proserve on palveluliiketoiminnan kehitysprojekti, joka tähtää toiminnan parantamiseen Euroopan jälleenmyyntiorganisaatiossa. Projektin keskeisinä tavoitteina ovat yhtenäisen palvelupistemallin luominen sekä uusien palvelutuotteiden kehittäminen markkinoille.

Isona kuvana taustalla vaikutti ajatus siirtymisestä sopimustasoluokitukseen, joka määrittelisi erilaisia palveluja erihintaisille huoltosopimuksille. Palvelut käsittävät muun muassa vasteajat eri vikatilanteissa ja asiakkaan saaman koulutuksen määrän.

5.1.1 Yhtenäinen palvelupistemalli

Jokainen John Deere -lipun alla oleva toimipiste toimii samanlaisen prosessin mukaan. Asiakas tietää astuessaan sisään ovesta, minkälaista palvelua hän saa ja mitä voi odottaa. Aikaisemmin yhtenäistä mallia ei ollut, mikä johti siihen, että toimintatavoissa ja tehokkuudessa oli suuria eroja alueellisesti, jopa Suomen sisällä.

5.1.2 Uudet palvelutuotteet

Huollon käyttökapasiteetti oli yrityksen johdon mielestä liian alhainen. Tavoitteena projektille oli käyttöasteen nostaminen 80%:een sen hetkisestä 69%:sta. Käytännössä tämä siis tarkoittaa sitä, että korjaamoille oli saatava lisää toimeksiantoja. Koska samalla perusperiaate on, että koneita jouduttaisiin hälytyskorjaamaan mahdollisimman vähän, keskittyivät uudistukset ennakoivan huollon ja määräaikaishuoltojen kehittämiseen.

Esimerkkinä määräaikaishuolloista on metsäkoneelle suoritettava 500 tunnin määräaikaishuolto, joka on pakollinen kaikkiin koneisiin. Huolto käsittää moottoriöljyn vaihdon sekä neljän suodattimen vaihtamisen. Tuotteena tämä on niin pieni ja työnä niin vähäinen, että asiakas tekee huollon yleensä itse, ja maksaa näin ollen vain varaosista.

5.2 Proserve huollon näkökulmasta

Jälleenmyyntiorganisaatio käsittää kaikki ne tukitoimet, joita asiakas tarvitsee sen jälkeen kun hän on ostanut uuden metsäkoneen. Organisaatio sisältää seuraavat toiminnot: huolto, varaosamyynti, varaosaosto, tekninen asiakastuki, varasto ja jälkimarkkinointi. Tuloksen teon kannalta huolto on tärkein yksittäinen toimija ja tämän alan yrityksen tulisikin lähteä siitä että huolto toimii moitteettomasti.

Metsäkonealalla asiakkaiden ykköstarve on pitää metsäkone toiminnassa, jotta sillä voi tehdä rahaa. Tämän vuoksi huollon organisaatiolle kasautuu suuri vastuu ja paine siitä että se toimii moitteettomasti. Huolto onkin yksi keskeisimpiä, ellei tärkein, kehityskohde koko projektissa.

Huollon ympärillä toimivat sidosryhmät, joita ovat varaosamyynti, tekninen asiakastuki, varaosa-osto, varaosavarasto ja konemyyjät. Nämä kaikki vaikuttavat olennaisesti viime kädessä siihen, kuinka tyytyväinen asiakas on saamaansa palveluun. Suurin tuotto jälleenmyyntiorganisaatiolle tulee juuri korjaamotöistä, joten on selvää että sen prosessien pitää olla kunnossa joka suuntaan.

Proserve keskittyi paitsi luomaan uusia palvelutuotteita, niin myös hyvin voimakkaasti sisäisen logistiikan toimivuuteen ja sisäiseen tiedonkulkuun. Nämä asiat olivat tuotta-

neet ongelmia aikaisemmin ja vaikeuttivat toimintaa monella tasolla. Prosessien uudistamisen tavoitteena oli myös määritellä entistä tarkemmat tehtävänkuvaukset eri vastuuhenkilöille

5.3 Huollon prosessit

Kappaleessa 5 todettiin, että yrityksen on luotava omat toimintamallinsa, jotta ERPiä voidaan hyödyntää tehokkaasti. Niin ikään toiminnallisen testauksen näkökulmasta on tärkeää, että prosessit ovat mahdollisimman pitkälle selvillä ennen testausta. Proserven aikana luotiin uusi ydinprosessi, jota kaikki huollon perusprosessit noudattavat.

Huollon toimintaan kuuluvat seuraavat prosessit: hälytyskorjaus, suunniteltu korjaus, ennakkohuolto, vaihtokonekunnostus ja uuden koneen varustelu. Toiminnan tulisi ohjautua siten, että suurin osa tehtävistä huoltotöistä olisi ennalta suunniteltuja ja siten myös hyvin hallittavia. Proserven tehtävänä oli pureutua huollon osalta siihen että hälytyskorjausten määrä suhteessa kaikkiin toimeksiantoihin oli liian suuri.

5.3.1 Prosessien kuvaukset

Hälytyskorjaus

Hälytyskorjausta hyvin kuvaava termi on niin sanottu ambulanssikeikka. Hälytyskorjaus on kyseessä silloin, kun asiakkaan metsäkone on toimintakyvytön. Metsäkone tuottaa rahaa vain toimintakuntoisena, joten on ensiarvoisen tärkeää saada kone nopeasti takaisin tuottavaan työhön. Hälytyskorjauksen tunnuspiirteitä ovat niiden ennakoinnin mahdottomuus ja samalla erittäin haastava suunnittelu.

Hälytyskorjaukset suoritetaan joko maasto-olosuhteissa asiakkaan luona tai omalla korjaamolla. Kyseiset korjaukset ovat erittäin stressaavia ja aiheuttavat ongelmia esimerkiksi kapasiteetin osalta. Tämän vuoksi niiden määrän tulisi olla mahdollisimman pieni.

Keskeisiä ongelmia hälytyskorjauksissa:

1. Asiakkaalla monta kontaktia samalle asialle
2. Ongelmasta on usein väärää informaatiota
3. Pitkät vasteajat ilman ilmoituksia

Koska jälleenmyynnistä puuttui selkeä toimintamalli oli hälytyskorjauksissa usein mukana paljon ihmisiä. Asiakkaan lähin kontakti saattoi olla esimerkiksi konemyyjä tai huoltoneuvoja. Lisäksi hyvin tyypillinen tilanne on, että ongelmasta ilmoittaa ensimmäisenä metsässä oleva koneen kuljettaja. Kuljettajan isäntä eli asiakas otti sitten yhteyttä omaan lähimpään JDF- kontaktiin, josta asia jatkui eteenpäin suunnittelematta. Asiakkaat eivät välttämättä ottaneet ensikontaktia oikeaan henkilöön.

Selkeän ohjeistuksen puuttuminen aiheuttaa sen, että korjaamolle saapuva informaatio on väärää tai vajavaista. Selkeä ongelma on myös vian vaikea diagnosointi puhelimessa, sillä esim. kuljettajat eivät välttämättä osaa kuvailla todellista ongelmaa. Tämä aiheuttaa vaikeuksia varsinkin niissä korjauksissa, jotka tehdään maastossa, koska silloin varaosia ei saada nopeasti puutetilanteissa, tai mikäli vian diagnosointi osoittautuu vääräksi.

Vasteajat saattoivat olla pitkiä johtuen esimerkiksi osien saatavuudesta tai kapasiteetin puutteesta. Samalla asiakkaan informointi oli ollut vajavaista prosessin aikana. Kaiken kaikkiaan vanha malli tuotti paljon turhaa työtä useassa paikassa.

Suunniteltu korjaus

Suunniteltu korjaus tarkoittaa vian korjaamista suunnitellusti. Metsäkone on edelleen toimintakuntoinen, mutta se ei toimi parhaalla mahdollisella tavalla. Lisäksi suunnitelluksi korjaukseksi lasketaan mahdollisen myöhemmän vian ehkäiseminen vaihtamalla hajoamassa oleva osa uuteen. Suunniteltuun korjaukseen kuuluvat myös erilaiset niin sanotut päivitykset. Useasti kyseisiä korjauksia suoritetaan määräaikaishuoltojen yhteydessä. Tulevaisuudessa pyrkimyksenä on saada suunniteltujen korjausten osuus mahdollisimman isoksi, koska ne ovat paremmin hallittavissa kuin hälytyskorjaukset.

Havaittuja ongelmia suunnitellun korjauksen osalta:

1. Korjauksia joudutaan tekemään ennalta sopimatta
2. Asiakas ei tule vaikka on sopinut, jos kone toimii
3. Ei riittävästi markkinoitavia tuotteita eikä markkinointia

Asiakkaat toimivat yleensä niin sanotulla run to failure -periaatteella joka tarkoittaa, että koneita ajetaan niin kauan kuin ne toimivat. Suunniteltua korjausta tai päivitystä ei välttämättä tehdä kesken kiireistä aikaa, vaikka se olisikin suotavaa koneen suorituskyvyn ylläpidon kannalta.

Toisena isona ongelmana havaittiin, että markkinoitavia päivityksiä ei ole riittävästi. Samoin jo olemassa olevien päivitysten aktiivinen markkinointi oli heikkoa.

Ennakkohuolto

Ennakkohuolto tai ennakoiva kunnossapito tarkoittaa määräaikaishuoltoja. Määräaikaishuollot ovat helpoiten hallittava ja ennustettavissa oleva osa-alue huollon toimintaa. Myös ennakkohuollon osuus on keskeinen prosessiuudistuksessa, koska se parantaa sekä yrityksen toimintaa että metsäkoneen toimintakykyä. Koneille on määritelty tietyt intervallit, joiden välein kone on huollettava. Nämä ovat tuntimääräisiä ja koostuvat seuraavista ajoista: 250 tunnin ensihuolto, 500 tunnin huolto, 1000 tunnin huolto ja 2000 tunnin huolto.

Ennakoivan kunnossapidon ongelmia:

1. Kapasiteetin hallinta
2. Koneeseen ja asiakkaaseen liittyvän huoltohistorian kertyminen
3. Asiakasraportointi heikkoa
4. Tuote on liian suppea

Asiakkaat eivät välttämättä huolla koneitaan silloin, kun sen aika on. Määräaikaishuollot venyvät, koska koneita ajetaan silloin, kun on töitä, ja tämä aiheuttaa ongelmia jatkon kannalta. Huoltoja tehdään usein ns. seisokkiaikana, jolloin syntyy selkeitä kuormituspiikkejä, jotka tuottavat kapasiteettiongelmia.

Huoltohistorian kertyminen järjestelmään on vajavaista. Vaikka huoltoja olisi suoritettu samalle koneelle, ei tietojärjestelmä tarjoa riittävän hyvää mahdollisuutta huoltohistorian selaamiseen. Tämä tuottaa ongelmia varsinkin huollon ja teknisen tuen töissä, koska ei ole riittävää tietoa, mikäli joku osa on vaihdettu, tai onko määräaikaishuollot tehty ajallaan. Ylipäätään on erittäin vaikea tietää, mitä kullekin koneelle on tehty.

Asiakas ei saa suoritetusta huollosta muuta dokumenttia kuin laskun. Laskuun on mahdollista kirjoittaa tietoa myös tehdyistä töistä, mutta huollosta johtumattomista syistä laskun saapuminen voi viivästyä. Olisi tärkeää saada asiakkaalle heti työn jälkeen raportti suoritetuista töistä.

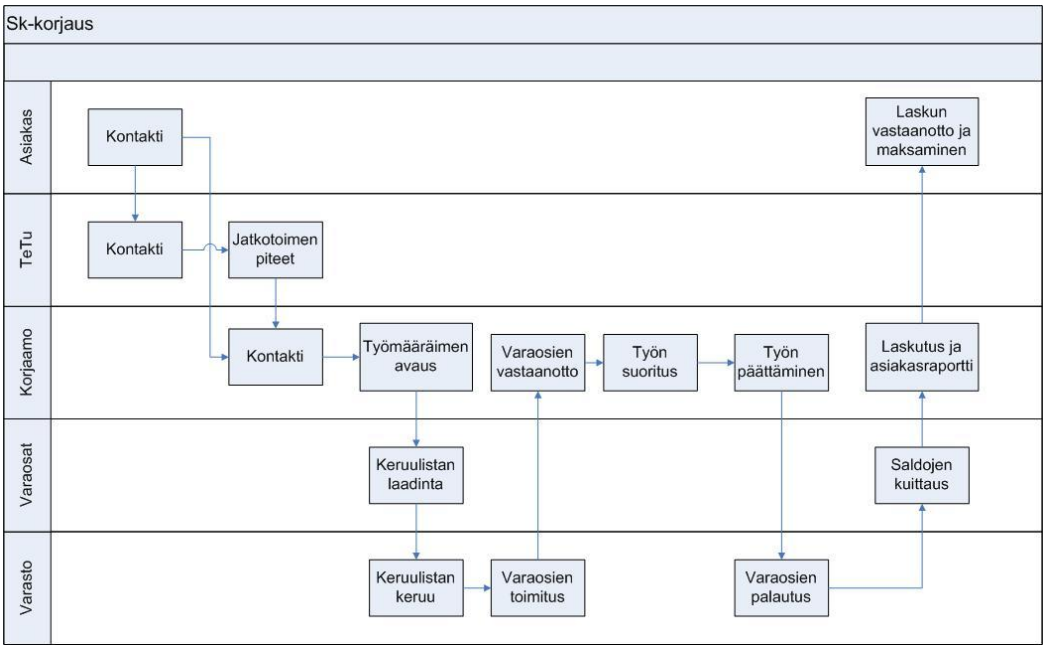
Vaihtokonekunnostus ja uuden koneen varustelu

Vaihtokonekunnostus ja uuden koneen varustelu ovat huollon toiminnassa vähiten tuottava osuus. Tällöin kyseessä on aina sisäinen asiakastyö, joten ne eivät ole niin kriittisiä toiminnan kannalta. Vaihtokonekunnostuksessa asiakkaalle myyty vaihtokone laitetaan toimintakuntoon, jotta se voidaan luovuttaa asiakkaalle. Tämä johtuu siitä, että metsäkoneet menettävät toimintakykyään, jos ne ovat niin sanotusti seisonnassa eli käyttämättöminä.

Uuden koneen varustelu tarkoittaa tehtaalta valmistuvan koneen varustelua toimitusohjeen mukaisesti. Käytännössä koneisiin lisätään asiakkaan toivomia ja kauppasopimuksessa mainittuja lisälaitteita.

5.3.2 Ydinprosessi

Ydinprosessi on yrityksen ja sen sidosryhmien toimintaa läpileikkaava toimintoketju. Ydinprosessi koostuu pienemmistä prosesseista, eli tässä tapauksessa työvaiheista. Kuvion 7 prosessikuvauksessa yrityksenä on huolto ja muut osastot toimivat sen sidosryhminä. (Hannus 1994, 366.)



Kuvio 7. Huollon ydinprosessi korkeimmalla tasolla.

John Deeren tapauksessa ydinprosessi kuvaa ketjua, joka käydään läpi kaikissa huollon toimenpiteissä. Tämä ydinprosessi kehitettiin Proserve-projektin aikana ja se tulee käyttöön koko Euroopassa.

Ketju alkaa aina asiakkaan kontaktista, useimmiten puhelinsoitosta. Tavoitteena on että ensimmäinen kontakti yrityksessä on aina huoltoneuvoja, eli Tekninen Tuki (TeTu). Tekninen tuki selvittää asiakkaan ongelman ja yrittää saada aikaan ratkaisun. Mikäli todetaan, että koneelle on tehtävä huoltotoimenpiteitä, siirtyy asia ketjussa eteenpäin korjaamolle.

Korjaamopäällikkö tekee lisätiedusteluita ja niiden perusteella kuvauksen ongelmasta. Tämän jälkeen hän avaa niin sanotun työmääräimen eli toimeksiannon huoltotyöstä. Toimeksianto sisältää ohjeet asentajalle sekä varaosamyyjälle. Varaosamyyjä tekee ongelmakuvauksen perusteella varaosatilauksen kyseiselle työlle. Tästä tulostetaan keruulista joka on varaston työkalu osien keräämisessä.

Varasto toimittaa osat korjaamolle keruulistassa osoitetun asiakkaan lokeroon. Asentaja ottaa osat sekä työmääräimen lokerosta ja aloittaa työnsä. Työn päätyttyä asentaja raportoi korjaamopäällikölle käytetyt tunnit, sekä merkitsee keruulistaan työssä käytetyt osat. Mikäli varaosia on jäänyt yli, ne palautetaan takaisin varastoon. Palautuksessa osat otetaan uudelleen varastosaldoihin ja sen jälkeen varaosamyyjä kuittaa tilauksen. Korjaamopäällikkö lisää laskuun käytetyt tunnit ja muut tarvikkeet kuten öljyt. Tämän jälkeen lasku siirtyy talousosaston haltuun, josta se toimitetaan eteenpäin asiakkaalle.

Korjaamopäällikkö tekee suoritetusta työstä myös asiakasraportin, josta ilmenee ongelman kuvaus, tehdyt korjaukset, käytetyt varaosat ja teholliset työtunnit. Raportti luovutetaan asiakkaalle joko suoraan tai koneen mukana. Liitteissä on kuvattu prosessin kulku järjestelmässä. Se on esitelty järjestelmän näyttöjen avulla.

6 Huollon moduulin testaus

Työ alkoi ohjelmistoon tutustumisella. Käytännössä tämä tapahtui viemällä läpi kuvitteellisia toimeksiantoja prosessin mukaisesti järjestelmässä. Tutustumisvaiheen jälkeen alkoi testisuunnitelman laatiminen ja testitapausten luominen. Tässä työssä ei ole mahdollista käsitellä kaikkia testitapauksia, koska niitä ei ollut suunniteltu vielä siinä vaiheessa.

6.1 Testauksen tavoitteet

Tärkeimpänä tavoitteena oli vaadittujen ominaisuuksien ja toiminnallisuuksien testaaminen. Lähdekoodin testaamiselle ei tässä vaiheessa enää ollut tarvetta. Ohjelma tuli toimittajalta valmiina runkona, johon suoritettiin räätälöinnit. Näin ollen kyse oli järjestelmän käyttöönottovaiheen toiminnallisuustestauksesta. Toiminnallisuuden testaamisessa tuli ottaa huomioon erityisesti uudet työvaatimukset.

Ominaisuuksien ja toiminnallisuuksien lisäksi tässä vaiheessa testattiin myös ohjelman käyttöliittymää. Toimittajalta saadun järjestelmän käyttöliittymää ei ollut vielä räätälöity lainkaan, joten sieltä tuli löytää turhat kentät, ja suunnitella käyttöliittymä sellaiseksi, että se on mahdollisimman hyvä jokaiselle käyttäjäryhmälle.

Edelleen testauksen aikana luotiin määrittelyt käyttäjäprofiileille. Määrittelyt koskivat niiden ulkoasua ja rajoituksia järjestelmässä. Myös paperitulosteiden ulkoasu sekä niiden sisältämä data piti määritellä.

Koska aikaisemmin oli jo määritelty eri mittareita huollon toiminnalle, tuli myös järjestelmän kirjata näitä. Erilaisia aikamittareita varten ohjelmassa tuli olla kentät, joihin vaadittuja tietoja voi syöttää.

Huollon moduulin vaatima data tuli miettiä ja validoida. Tietojen siirto tapahtui vanhasta kannasta ja testivaiheessa tuli jo samalla testata tämän siirron onnistuminen. Käytännössä tämä tarkoittaa esimerkiksi kannassa olevan metsäkoneen ja asiakkaan oikeaa

suhdetta, eli oikeat koneet ovat oikeilla asiakkailla. Tarvittaessa käytettävälle datalle tultaisiin suunnittelemaan sekä perusarvot että kooditukset.

Järjestelmä tarjosi mahdollisuuden myös monipuolisen taustadatan hyväksikäyttöön. Ennen käyttöönottoa oli mietittävä mitä näistä otetaan käyttöön, ja miten niitä hyödynnetään.

6.2 Vaatimukset huollon moduuliin

Vaatimuksia oli sekä toiminnallisia että dataan liittyviä. Toiminnallisuusvaatimukset liittyivät jo aiemmin mainittuihin prosessiuudistuksiin sekä yleensä koko prosessin läpivientiin. Testauksessa keskityttiin keskeisten toiminnallisuuksien ja ominaisuuksien toimintaan ja läpivientiin.

6.2.1 Toiminnallisuusvaatimukset

Esitietolomake

Huolto tarvitsee esitietolomakkeen johon voidaan ottaa ylös kontaktin tiedot. Koska soittaja ei ole aina asiakas (koneen omistaja), pitää soittajan tiedot saada ylös. Lomakkeessa on oltava mahdollisuus vian kuvaukseen ja resurssitilanteen tarkasteluun.

Työmääräin

Työmääräimellä pitää pystyä suunnittelemaan tuleva työ täydellisesti. Erityistä huomiota kiinnitettiin siihen, että sillä pystyy käsittelemään samanaikaisesti useita erityyppisiä töitä (takuu, normaalihuolto) sekä varaston materiaalitilannetta.

Laskutuksen tehostaminen

Laskutusta on saatava nopeammaksi ja työmääräimen sulkemisen järjestelmässä on oltava sujuvaa. Tietojen siirto huollon ja laskutukseen välillä on oltava helppoa.

Mittarit

Proserven tuloksena määriteltyjä mittareita ovat muun muassa laskutustehokkuus, huoltojen kokonaisaika ja huoltojen sisäiset aikamittarit (esimerkiksi asiakkaan kontakti -> työn aloitus).

Paperitulosteet

Paperitulosteita pitää päivittää. Asiakkaalle on saatava raportti tehdyistä töistä jo ennen laskun lähetystä.

Palvelutuotteiden hallinta

Määräaikaishuolloista on saatava luotua paketteja, jotka voidaan laskuttaa kiinteällä hinnalla ja ovat helppoja hallita.

Töiden aikataulutus ja tyypittäminen

Huollon toimeksiannot on pystyttävä tyypittämään ja niitä on pystyttävä priorisoimaan tyypin mukaan. Järjestelmän on huolehdittava siitä, että kiireellisimmät työt siirtyvät automaattisesti työjonon kärkeen.

6.2.2 Datavaatimukset

Järjestelmän vaatimaa dataa on kolmea eri tyyppiä: kriittinen data, pakollinen perusdata ja valinnainen perusdata. Näiden määrittely on järjestelmän kannalta hyvin tärkeää. Määrittely tehtiin prosessikaavioiden avulla sekä käyttämällä hyväksi Proserven aikana luotuja spesifikaatioita ja järjestelmän toimittajan tarjoamaa konsultointiapua.

Kriittinen data

Kriittinen data on se data, jota huollon moduuli tarvitsee toimiakseen vaatimusten mukaisesti. Tämä data tulee tarkastaa erittäin huolellisesti. Järjestelmän toimittaja käyttää

datan kohdalla määrittelyä monimutkainen ja se kuvaa hyvin tietojen rakennetta. Taulukko 1 esittelee järjestelmän kriittisen datan.

Taulukko 1. Kriittinen data

Nimi	Kuvaus
Konekanta	Rekisteröityjen asiakkaiden hallinnassa olevat laitteet (metsäkoneet, harvesteripäät).
Asiakasrekisteri	Asiakkaat, joilla yksilöllinen asiakasnumero.
Työntekijät	Työntekijöiden henkilökohtaiset tunnisteet.
Huoltosopimukset	Metsäkoneiden yksilölliset huoltosuunnitelmat.
PM Actions	Huollon ennakkosuunnittelun työkalut.
Varaosavarasto	Aktiivisen koodin omaavat varaosat.
Hintalistat	Huollon työn sisäinen kustannus.
Sisäiset organisaatiot	Huoltoihin osallistuvat organisaatiot (korjaamo, tekninen tuki yms.)

Prosessin kannalta pakollinen perusdata

Perusdata täydentää kriittistä dataa. Järjestelmä tarjoaa mahdollisuuden käyttää useita erityyppisiä taustatietoja, joita hyödynnetään yrityksen tarpeiden mukaisesti. Pakollinen perusdata on lueteltu taulukossa 2.

Taulukko 2. Pakollinen perusdata.

Nimi	Kuvaus
Työtyypit	Erilaiset huoltotyötyypit (hälytyskorjaus, suunniteltu korjaus yms.). Hyödynnettävissä koneen huoltohistorian tarkastelussa, tilastollisena datana.
Parametrit	Esimerkiksi paine, lämpötila, öljyn määrä. Parametrit ovat linkitettyinä koneisiin ja ohjaavat esimerkiksi PM Actioneita.
Toiminnot	Metsäkoneelle suoritettavan korjaustoimenpiteen tyyppi ja kuvaus.
Varustelutaso	Metsäkoneen varustelutaso.
Konetyyppi	Harvesteri tai kuormakone.
Sidosryhmien käyttäjät	Varaosien ostajat yms. pakolliset sidosryhmät.

Valinnainen perusdata

Valinnaista perusdataa käytetään hyväksi muun muassa tilastollisessa seurannassa. Joissain tapauksissa (esimerkiksi Rooli) tämä data liittää kriittiseen dataan lisää ominaisuuksia. Kaikkea tätä ei tarvitse käyttää, ja on päinvastoin suositeltavaa, että turha tieto siivotaan järjestelmästä pois. Valinnainen perusdata on lueteltu taulukossa 3.

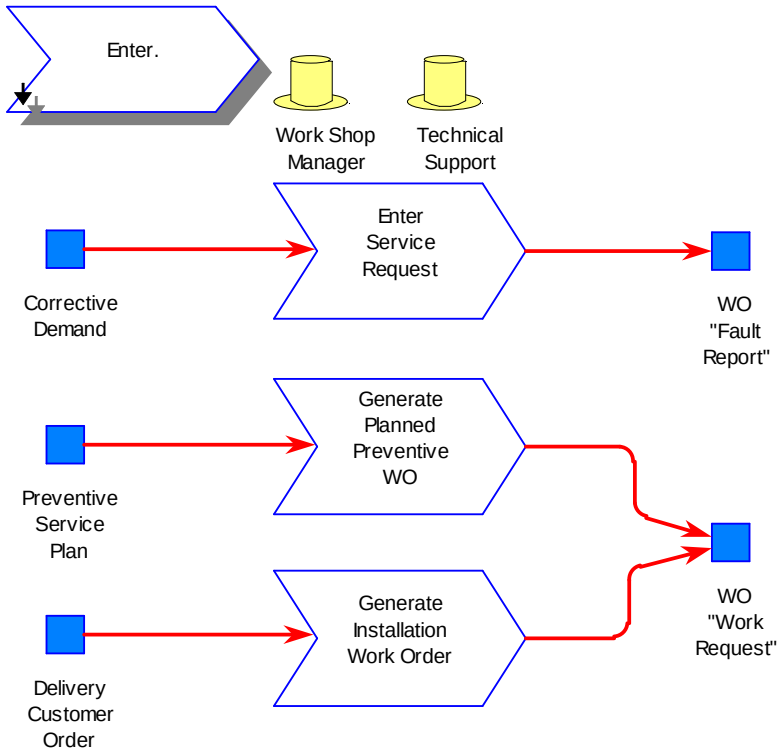
Taulukko 3. Valinnainen perusdata.

Nimi	Kuvaus
Rooli	Voidaan sitoa työntekijään, varaosiin tai toimipaikkaan. Rooliin voidaan määritellä valmiiksi esim. työn laskutushinta.
Löydös	Työmääräimen raportointiin liittyvä tieto viasta. Hyödynnettävissä koneen huolto-historiassa.
Oire	Ks. Löydös
Luokka	Löytyneen vian luokka. Mahdollista määritellä viat eri luokkiin, jota voidaan hyödyntää tilastollisessa seurannassa.
Tyyppi	Vian tyyppi, esim. vuoto, väsyminen yms.
Suoritettu toiminto	Koneelle suoritettu korjaustoimenpide.
Aiheuttaja	Vian aiheuttaja. Voidaan hyödyntää tilastollisessa seurannassa.
Tärkeys	Määrittelee huollon vasteaikoja riippuen työtyypin painotuksesta.
Huoltosopimustyyppi	Eri huoltosopimustyypit, mm. SLA (Service Level Agreement), normaali jne.

6.3 Suunnittelun toteutus

6.3.1 Testiskenaariot

Työmääräimen tuottaa kolme eri tilannetta: koneen hajoaminen, määräaikaishuolto tai asiakkaalle menevän uuden koneen lisävarustelu. Näistä kolmesta eri osa-alueesta on muodostettu testiskenaariot, jotka ryhmittävät testitapaukset. Kuvio 8 osoittaa, kuinka skenaariot eroavat toisistaan. Varsinaiset erot syntyvät siis vain prosessin käynnistävässä tapahtumassa, sekä tästä aiheutuvissa eroissa järjestelmän käytössä.



Kuvio 8. Työmääräimen generoivat tapahtumat. (IFS Solution Defining Document,2008.)

6.3.1.1 Skenaario 1: Koneen hajoaminen tai rikkoutuminen (Corrective Demand)

Prosessin laukaisee asiakkaan koneen hajoaminen. Useimmin kysymyksessä on hälytyskorjaus ja joissain tilanteissa myös suunniteltu korjaus. Koska tapauksia on lähes mahdoton ennakoida, joudutaan työmääräin tekemään manuaalisesti. Se tarkoittaa, että työmääräin luodaan käsin aloittaen prosessi alusta ilman järjestelmän tarjoamia ”oikopolkuja”.

Prosessi alkaa niin sanotun huoltopyynnön teolla, jonka pohjalta luodaan varsinainen työmääräin. Tässä skenaariossa työmääräin saa järjestelmässä statuksen vikaraportti, joka tarkoittaa korjaavaa toimenpidettä.

6.3.1.2 Skenaario 2: Ennakoiva huolto (Preventive Service Plan)

Ennakoiva huolto tarkoittaa ennalta suunniteltua huoltoa. Sen periaate on, että jokaiselle koneelle luodaan yksilöllinen huoltosuunnitelma. Se voi perustua käyttötunteihin, ennalta määriteltyihin tapahtumiin tai se voi olla kalenteripohjainen.

Järjestelmä tarjoaa varta vasten ennakoivan huollon tarpeisiin suunnitellun moduulin, joka on nimeltään Ennakoiva kunnossapito. Moduulin etuna on, että työmääräin voidaan luoda suoraan yksittäisen koneen huoltosuunnitelmasta. Tällä säästetään työaikaa verrattuna manuaaliseen tapaan. Ideaalitilanteessa metsäkoneen tietokone lähettäisi reaaliaikaista tietoa koneen käyttötunneista suoraan yrityksen järjestelmään. Täten järjestelmä ilmoittaisi aina, kun huoltoraja lähestyy, ja asiakkaalle voitaisiin lähettää kutsu huoltoon.

Moduuli sisältää valmiita huoltopaketteja, joita kutsutaan Standardityö-nimellä. Nämä paketit sisältävät valmiiksi kaiken suunnittelutiedon huollosta. Paketissa on valmiina huollossa suoritettavat työt, huoltoon tarvittavat osat, huollon kesto ja huollon kustannukset. Tällä tavoin ennakoiva huolto voidaan suunnitella hyvin pitkälle etukäteen, mikä vähentää työkuormaa huomattavasti.

6.3.1.3 Skenaario 3: Uuden koneen varustelu (Delivery Customer Order)

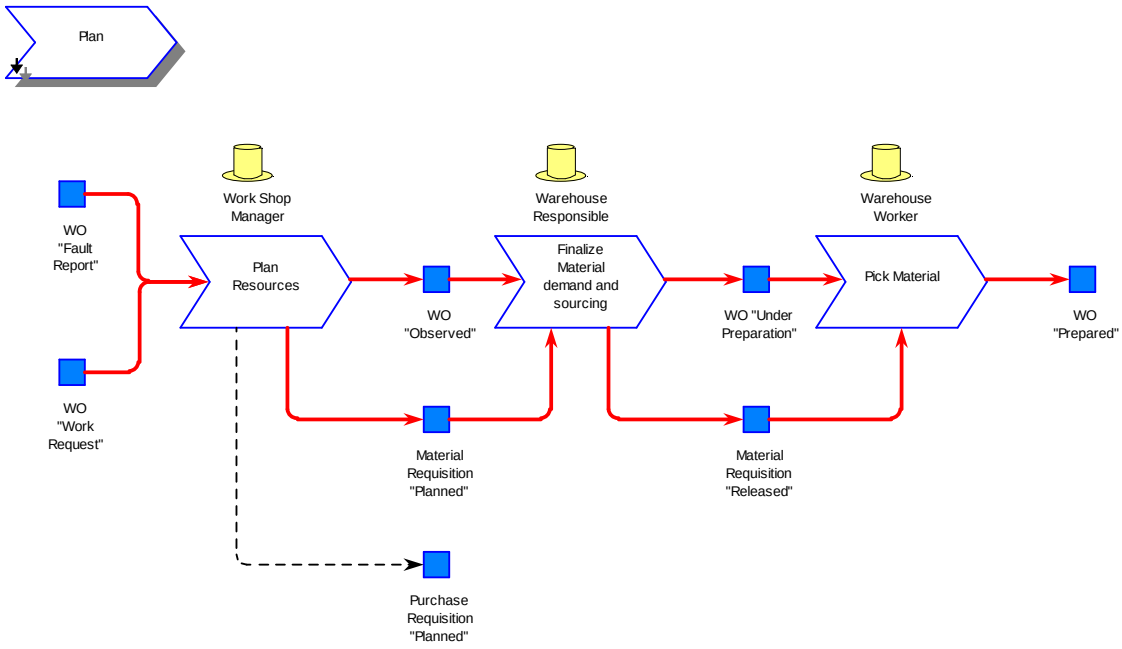
Uuden koneen varustelu eroaa normaalista huoltotyöstä siinä, että työmääräimen avaa toimitusvalvoja. Kun asiakas tilaa uuden koneen tehtaalta, luodaan samalla sisäinen asiakastilaus. Tämä asiakastilaus kerää kaikki sisäiset kustannukset tietystä projektista.

Huolto suorittaa uuden koneen varustelun kauppasopimuksen mukaisesti. Varustelusta koituvat kustannukset lisätään asiakastilaukselle ja tarkemmin niin sanotulle projektille. Projekti on aina yksi metsäkone.

Skenaario 2 ja skenaario 3 tuottavat järjestelmään työmääräimen, jolla on statuksena työpyyntö. Se tarkoittaa, ettei kyseessä ole kiireellinen työ.

6.3.2 Testitapaukset

Testauksessa on käytetty mustalaatikkotestauksen periaatteita, korkealla tasolla prosessikaaviossa. Se tarkoittaa, että testitapauksissa syöttötieto ja tuloste muodostuvat prosessien vaiheiden mukaan, eikä yksittäisiä kenttiä ole erikseen testattu. Kuvio 9 esittelee järjestelmän toimintaa huoltotyön suunnitteluvaiheessa.



Kuvio 9. Työmääräimen suunnitteluvaihe IFS:n prosessikuvauksessa (IFS Solution Definition Document,2008)

Esimerkiksi kappaleessa 6.3.1.1 esitellyssä skenaario 1:ssä ensimmäisen vaiheen tulosteena on työmääräin statuksella tarkasteltu, lisäksi tulosteina ovat myös suunniteltu -tilassa olevat materiaalivaraus ja/tai materiaalin ostotilaus. Vastaavasti syöttötietona toimii aiemmassa vaiheessa luotu vikaraportti tai työpyyntö.

Yksi testitapaus kattaa useita toimenpiteitä itse ohjelmassa, mutta näiden testaaminen tapahtuu käyttöohjeita vasten. Koska myös käyttöohjeet on testattava, saadaan tällä tavoin testattua sekä käyttöohjeet että toiminnot samalla kerralla. Mikäli testitapaus ei mene läpi käyttöohjeiden avulla, on ohjeita tällöin parannettava. Samalla toki huomataan mahdolliset virheet myös ohjelman teknisessä toteutuksessa.

Jokaisen skenaarion alle on listattu testitapauksia jotka kattavat odotettavissa olevat tapahtumat työn suorituksen aikana. Korjaavan toimenpiteen skenaario pitää sisällään lähes kaikki mahdollisuudet muuttujat, koska se kattaa koko huoltoprosessin, ja suurin osa toimeksiannoista oletettavasti edelleen päätyy kyseisen skenaarion alle.

Ennakkohuollon skenaario käsittää muuttujat määräaikaishuoltojen suunnittelussa ja keskittyy enemmän itse järjestelmän toimintoihin, ja niiden käyttökelpoisuuteen. Käytännössä siis testataan miten järjestelmän työkaluja voidaan hyödyntää, ja samalla tulee miettiä yritykselle oma ratkaisumalli.

Uuden koneen varustelun skenaario on pienin osa-alue. Sen eri muuttujat käsittävät vain muutaman eroavaisuuden prosessin aloituksessa, käytännössä puhutaan siitä kuka kyseisen prosessin käynnistää.

6.3.3 Testaus

Testaus suoritetaan manuaalisesti erillisessä testikannassa. Testikanta on avoin kaikille eri moduulien testaajille, ja tällöin tavallaan tarjoaa mahdollisuuden testata myös työprosessien toimivuutta moduulien välillä.

Testitapaukset viedään läpi prosessikaaviota seuraten ja samalla toteuttaen työtilausta testikannassa. Testausta vaikeuttaa jonkin verran se, ettei kaikkea taustadataa löydy vielä testikannasta. Kuvio 10 esittelee testitapaussuunnitelman johon on hahmoteltu myös testituloksia.

Kaikista testitapauksista tehtiin oma virheraportti, jolle kirjattiin mahdolliset puutteet ohjelman toiminnassa tai logiikassa. Virheraportit välitettiin edelleen projektin vastaavalle joka raportoi järjestelmän toimittajalle.

Scenario 1. Planned repair				
Nimi	**Lähtötilanne**	**Tehtävä**	**Status**	**Kommentit**
Suunnitelu korjaus 1.1	**Avataan uusi keikka.**	Uuden service requestin teko.	OK	
Suunnitelu korjaus 1.2	**Service request on tehty ja sen pohjalta tehdään WO**	Work orderin avaaminen service requestin pohjalta.	OK	
Suunnitelu korjaus 1.4	**Lisätään muita tietoja work orderiin.**	Work orderin suunnittelu	OK	
Suunnitelu korjaus 1.5	**Aikatauluksen käyttö**	Aikataulutus	Failed	Toiminto ei ole vielä käytössä.
Suunnitelu korjaus 1.6	**Work orderilla ei ole vielä osia.**	Tehdään material requisition	OK	Varaosakyselyä ei voi suorittaa pikanäppäimellä.
Suunnitelu korjaus 1.7	**Varaosia ei ole vielä varattu eikä toimitettu.**	Reserve ja issue	OK	HUOM! Molemmat toiminnot ovat statuksesta riippuvaisia.
Suunnitelu korjaus 1.8	**Työ on valmis.**	Työn raportointi work orderille	OK	
Suunnitelu korjaus 1.9	**Työstä on jäänyt ylimääräisiä osia**	Osien palautus (unissue)	OK	
Suunnitelu korjaus 1.10	**Work order on raportoitu.**	Work orderin tiedot siirretään customer orderille.	OK	
Suunnitelu korjaus 1.11	**Customer order on tehty.**	Tietojen siirtäminen laskutukseen.	OK	
Suunnitelu korjaus 1.12	**Osia ei ole tarpeeksi, tai ne ovat loppuneet.**	Purchase req. Tekeminen.	OK	
Suunnitelu korjaus 1.13	**Työlle otetaan ulkoista työvoimaa.**	Purchase req. Tekeminen.		
Suunnitelu korjaus 1.14	**Työ on takuukeikka**	Takuuasioiden vaikutus	Not tested	
Suunnitelu korjaus 1.15	**Työ ei valmistu suunnitellussa ajassa.**		Not tested	
Suunnitelu korjaus 1.16	**Työlle otetaan lisäosia.**	Lisätään rivejä material requisitioniin.	OK	
Suunnitelu korjaus 1.17	**Customer orderin rivit laskutetaan eri paikoista.**		Not tested	
Suunnitelu korjaus 1.18	**Työ alkaa suunniteltua aikaisemmin.**	Kirjataan oikea aloitus päivä suunniteltua aikaisemmaksi.	OK	
Suunnitelu korjaus 1.19	**Tunteja raportoidaan liikaa**	Valmiista työstö raportoidaan tunteja yli arvion.	OK	
Suunnitelu korjaus 1.20	**Lisätöiden lisääminen work orderille.**	Työrivien lisääminen työn suorituksen aikana.	OK	

Kuvio 10. Testitapaussuunnitelma, skenaario 1.

7 Yhteenveto

Ohjelmistotestauksen rooli kehitysprojektin riskien hallinnassa on merkittävä. Oikein suunniteltu ja oikein suoritettu testaus vähentää selkeästi mahdollisia myöhemmän vaiheen taloudellisia kuluja. Siksi sille pitää varata riittävästi resursseja sekä ajankäytön että rahan suhteen.

Toiminnallisessa testauksessa on tehtävä laaja ja yksityiskohtainen pohjasuunnittelu ohjelman käyttötarpeista. Tähän voidaan käyttää joko testiskenaarioita tai käyttötapauksia. Molemmista koostetaan testitapaukset, joilla ohjelman toiminta testataan, ja varmistetaan sen käyttäytyminen halutulla tavalla.

John Deere Forestry Oy aloitti ERP-järjestelmän käyttöönottovaiheen kuluvan vuoden lokakuussa. Huollon moduulin toiminnalliset vaatimukset on määritelty aiemmin samana vuonna. ERPin tavoitteena on parantaa yrityksen back office -toimintoja ja tuoda uusia mahdollisuuksia sisäiseen raportointiin. Myös toimenkuvien selkeyttäminen ja prosessien ”riisuminen” ovat ERP-järjestelmän käyttöönoton tavoitteina.

Huollon moduulin testaus alkaa joulukuussa. Tähän mennessä huollon moduuliin on tehty toimintaa kuvaavat prosessikaaviot. Testaus tapahtuu tätä prosessimallia noudattaen. Moduulille on määritelty myös taustadatat, joita se käyttää. Tätä dataa on olemassa sekä kriittistä että perusdataa.

Tällä hetkellä huollon osalta valmistellaan korjaamopäällikön käyttöliittymän ulkoasua, ja seuraavaksi siirrytään suunnittelemaan rajoituksia profiilille. Testitapaukset on kirjoitettu valmiiksi ja ne odottavat viimeisen hyväksymistestauksen alkamista. Oma työ testauksen suunnittelussa on ollut haastavaa ja mielenkiintoista. Tämän opinnäytetyön aikana olen saanut valtavasti lisätietoa testaamisesta sekä järjestelmän käyttöönottoon liittyvistä asioista. Olen myös huomannut, että muutamia harha-askeleita on tullut otettua, mutta toisaalta se on luonnollista kokemattomuuden vuoksi. Olen kuitenkin tyytyväinen tähän asti saavutettuihin tuloksiin.

Lähdeluettelo

Painetut lähteet

Black, Rex, 2007. Pragmatic Software Testing. Becoming an effective and efficient test professional. USA: Wiley Publishing, Inc.

Everett, Gerald D & McLeod Jr, Raymond. 2007. Software Testing: Testing across the entire software development life cycle. Hoboken, New Jersey, USA: John Wiley & Sons, Inc.

Hannus, Jouko. 1994. Prosessijohtaminen. Ydinprosessien uudistaminen ja yrityksen suorituskyky. Neljäs painos. Jyväskylä: Gummerus Kirjapaino Oy.

IFS 2008: Solution Defining Document. Service & repair.

John Deere 2008: Tuotespesifikaatiot D-sarja

Kremzar, Michael H & Wallace, Thomas F. 2001. ERP: Making It Happen. Implementers guide to success with Enterprise Resource Planning. USA: John Wiley & Sons, Inc.

Loveland, Scott, Miller, Geoffrey, Prewitt Jr., Richard & Shannon, Michael. 2005. Software Testing Techniques: Finding the defects that matter. Hingham, Massachusetts USA: Charles River Media, Inc.

Myers, Glenford J, 2004. The Art of Software Testing. Second edition. Hoboken, New Jersey, USA: John Wiley & Sons, Inc.

Verkkolähteet

Clifton, Marc. Advanced Unit Testing, Part I. [online] [viitattu 18.9.2003] *http://www.codeproject.com/KB/cs/autp1.aspx#White%20Box%20Testing5*

Elpida, Samara. InnoSupport. Supporting innovations in SMEs. [online]
http://www.innovation.lv/ino2/publications/leonardo_manual/en/www.innosup-port.net/webhelp/wso/index.cfm@fuseactionlearnl_id3808pl_id3554.htm

John Deere 2008. John Deere Forestry Oy:n kotisivut. John Deere. [online] [viitattu 2008]
www.deere.fi

Nair, Prabhakaran. Different Types Of Testing. [Online] [viitattu 20.11.2008]
http://rajeevprabhakaran.wordpress.com/2008/11/20/different-types-of-testing/

Smooth Lab. Next generation software testing Q & A. [online] [viitattu 2007]
http://smoothlab.com/software-testing.htm

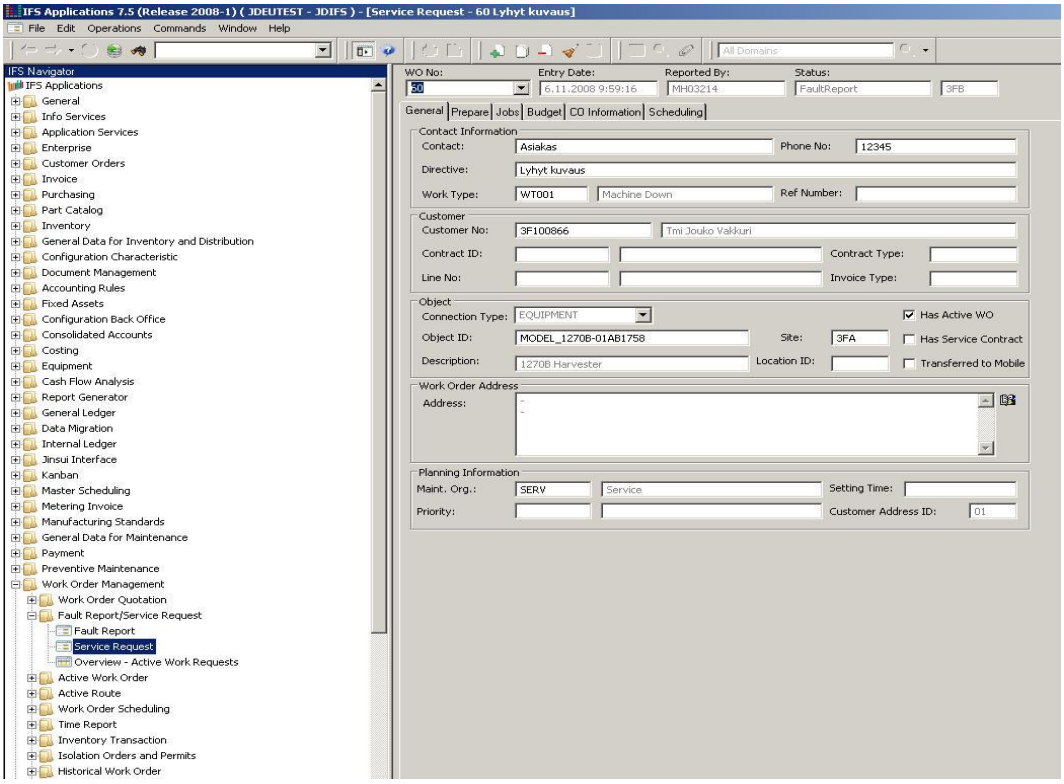
Wailgum, Thomas. ERP definition and solutions. [online] [viitattu 2008]
http://www.cio.com/article/40323/ERP_definition_and_solutions#improve

Liitteet

Huoltoprosessin pääpiirteet IFS-tauluina testikannassa

1. Avaaminen

Mikäli huolto nähdään tarpeelliseksi, avaa joko tekninen tuki tai korjaamon työnjohto Huoltopyyntö-ikkunan (kuvio 1). Tähän ikkunaan syötetään perustietoa työstä. Pakolliset datat ovat kontaktin nimi ja puhelinnumero, lyhyt kuvaus ongelmasta, työtyyppi (hälytyskorjaus, suunniteltu korjaus jne.), koneen sarjanumero, asiakasnumero ja asiakkaan nimi sekä työn suorittava organisaatio. Ylätunnisteeseen kirjautuu päivämäärä koska kyseinen huoltopyyntö on avattu, avaajan tunniste järjestelmässä sekä niin sanottu site (toimipaikka), jolta pyyntö avataan. Toimipaikka määräytyy henkilön mukaan. Kontakti ja asiakas voivat olla eri henkilöitä ja soittaja on useasti asiakkaalle työskentelevä koneen kuljettaja.

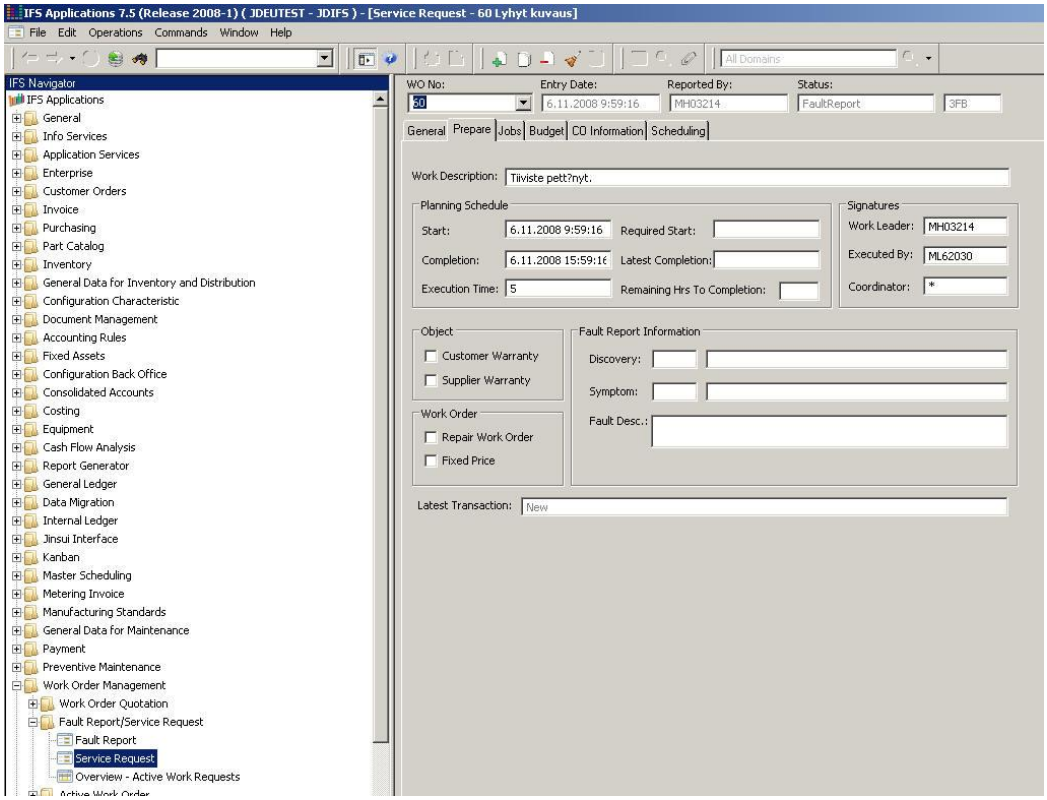


Kuvio 1. Huoltopyyntö-ikkuna IFS-järjestelmässä.

2. Huoltopyynnön suunnittelu

Huoltopyyntöä käytetään esitietolomakkeena, ja siihen voidaan jo suunnitella jonkin verran tulevaa työtä. Suunnittelun määrä riippuu siitä, avaako pyynnön tekninen tuki vai huolto. Tekninen tuki ei yleensä tee muuta kuin kirjaa pakolliset syötteet ensimmäiselle sivulle.

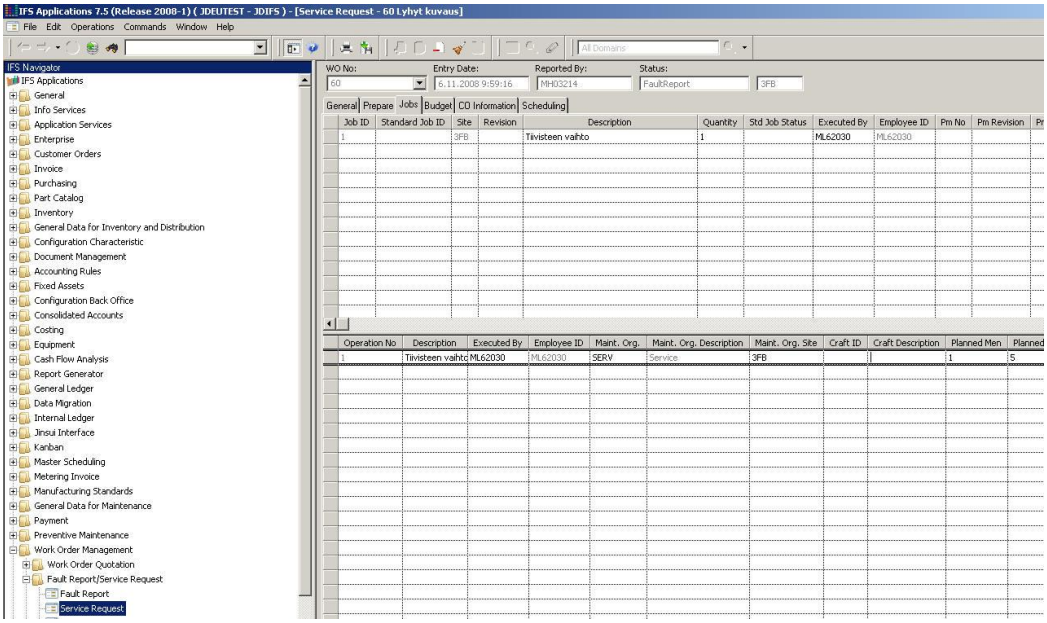
Valmistelu-välilehdelle (kuvio 2) on mahdollista suunnitella etukäteen työn aikataulua ja kertoa lisäinformaatiota koneen viasta. Aikatauluun voidaan laittaa suunniteltu aloituspäivä, suunniteltu lopetuspäivä ja arvioidut teholliset työtunnit. Nämä syötteet ohjaavat materiaalin hankinta-aikaa kyseiselle työlle.



Kuvio 2. Huoltopyynnön Valmistelu-välilehti IFS-järjestelmässä.

3. Työvaiheiden suunnittelu

Aikataulutuksen lisäksi on mahdollista lisätä jo tuleva työ tai tulevia töitä. Tämä tapahtuu Työt-välilehdellä (kuvio 3). Nämä tarkoittavat konkreettisia toimia kyseisen vian korjaamiseksi. Työnjohto voi halutessaan varata tietyn asentajan tietylle työlle jo tässä vaiheessa.

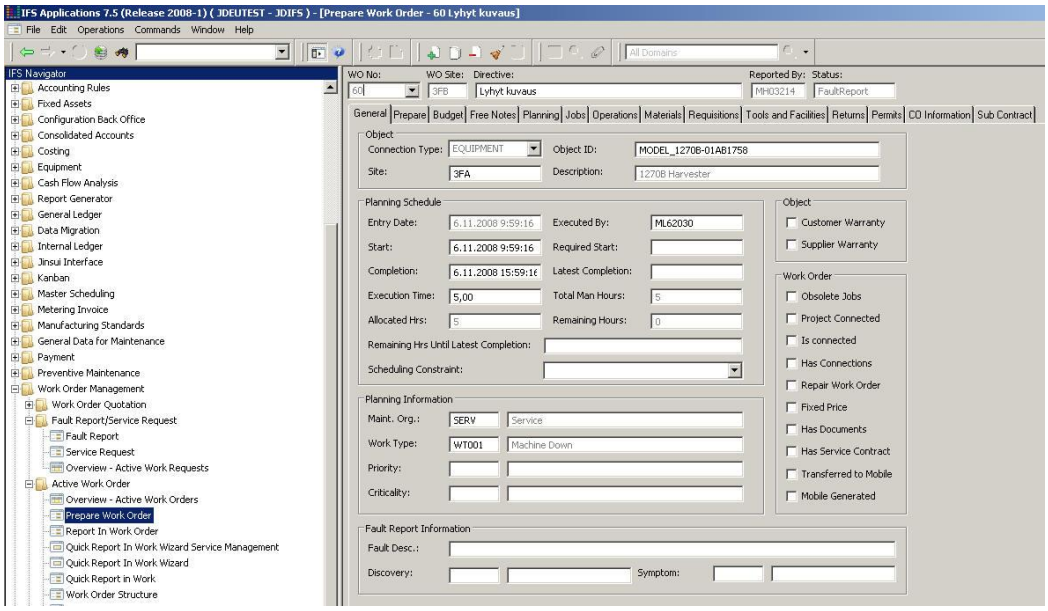


Kuvio 3. Huoltopyynnön Työt-välilehti IFS-järjestelmässä.

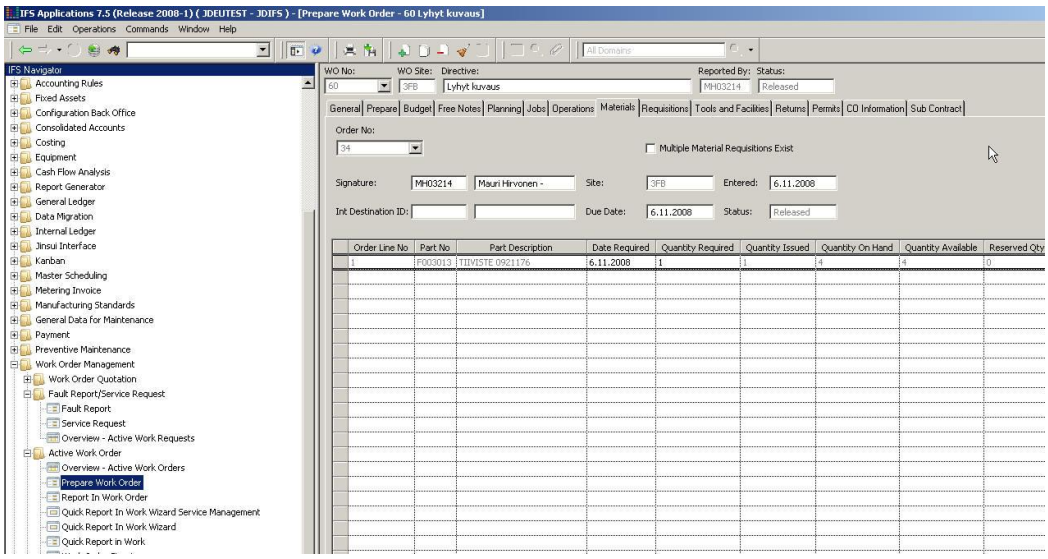
4. Työmääräimen avaus

Kun huoltopyyntö on tehty, se siirtyy jonotuslistalle odottamaan varsinaisen työmääräimen avaamista. Tässä vaiheessa työ näkyy vikaraportti-tilassa järjestelmässä. Varsinainen työmääräin tehdään huoltopyynnön pohjalta (kuvio 4) ja varsinainen suunnittelutyö tehdään sille. Työmääräin sisältää aikataulun, vian tarkemman kuvauksen, työvaiheet, työn suorittavan asentajan (tai asentajat), materiaalit (kuvio 5), mahdolliset erikoistyökalut, budjettilaskennan ja muita lisätietoja.

Työmääräimellä on eri statuksia jotka ohjaavat mm. materiaalihankintaa. Alkuvaiheessa työmääräin on vikaraportti-tilassa, siirtyen siitä eri vaiheiden jälkeen lopulta vapautettu- ja aloitettu-tilaan.



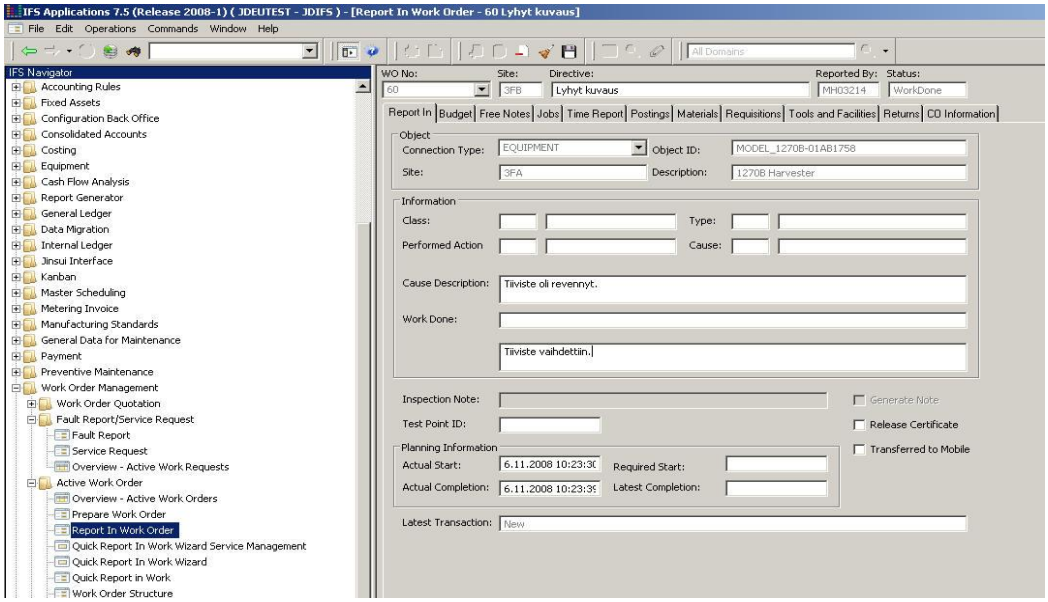
Kuvio 4. Työmääräimen etusivu IFS-järjestelmässä.



Kuvio 5. Työmääräimen materiaalivarausikkuna IFS-järjestelmässä.

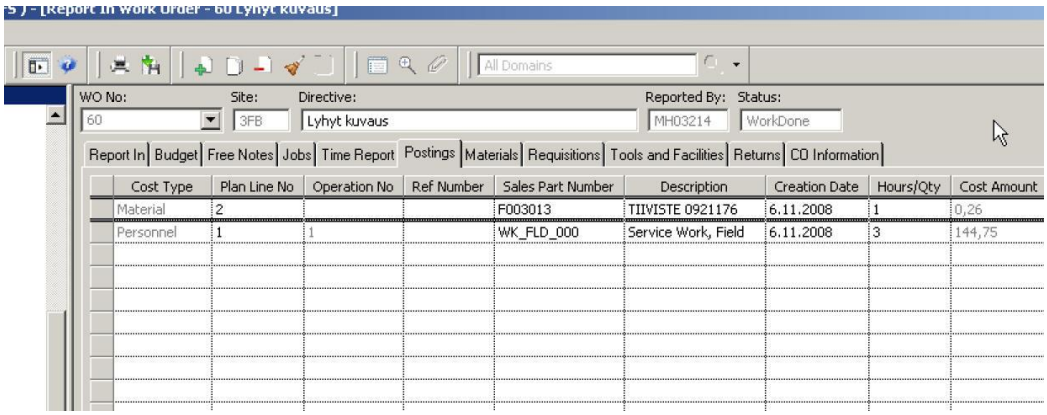
5. Työmääräimen raportointi

Kun työ on saatu valmiiksi, se tulee raportoida järjestelmään. Tässä vaiheessa käytetään raportointityökalua (kuvio 6). Asentaja syöttää tiedot työhön käytetyistä tehollisista työ-tunneista, käytetyistä varaosista sekä todelliset aloitus- ja lopetuspäivät.



Kuvio 6. Työmääräimen raportointityökalun etusivu IFS-järjestelmässä.

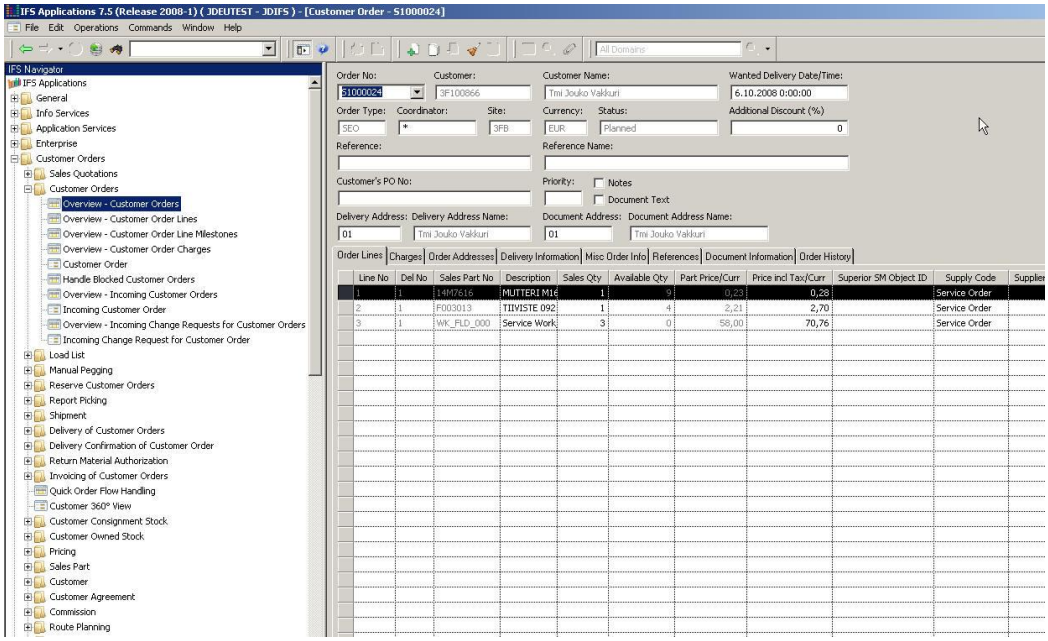
Työnjohto lisää työn kustannukset ja tiliöinnit sekä tarkastaa vielä kertaalleen syötettyjen tietojen oikeellisuuden. Lopulliset laskutettavat rivit näkyvät Tiliöinnit-välilehdellä (kuvio 7).



Kuvio 7. Laskutettavat rivit Tiliöinnit-välilehdellä IFS-järjestelmässä.

6. Myyntitilaus

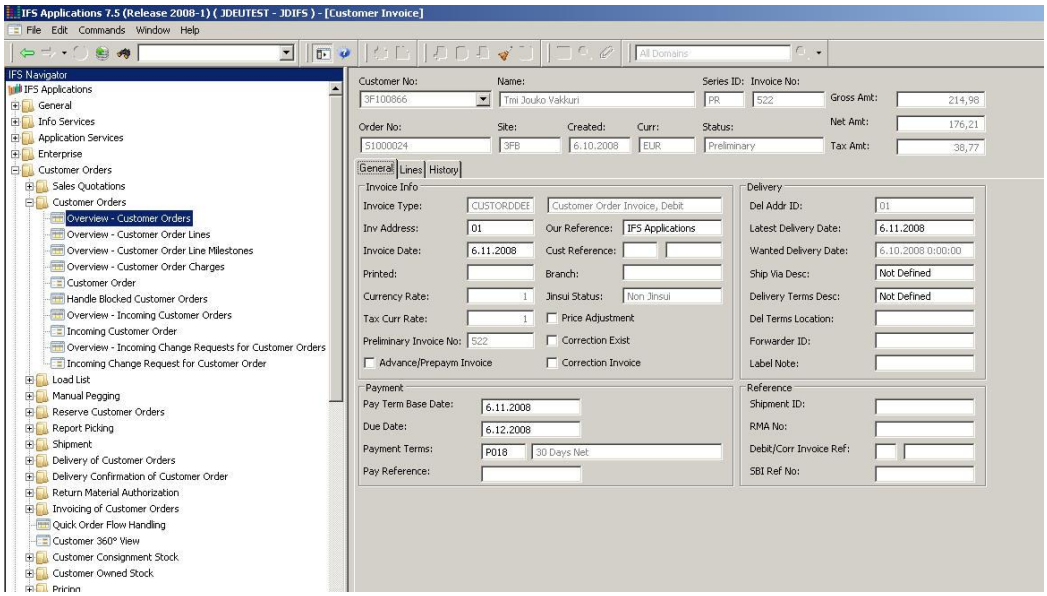
Kun työ on raportoitu, suoritetaan datan siirto myyntitilaukselle (kuvio 8). Myyntitilaustyökalu kerää kaiken tiedon kyseisestä asiakastilauksesta. Tilaus on aina asiakaskohtainen, eli mikäli yhdellä työmääräimellä on useita eri asiakkaita, luodaan kaikista erillinen tilaus. Yksi myyntitilaus voi sisältää esimerkiksi tehtaalta tilatun uuden koneen sekä sille suoritetusta varustelusta syntyneet rivit.



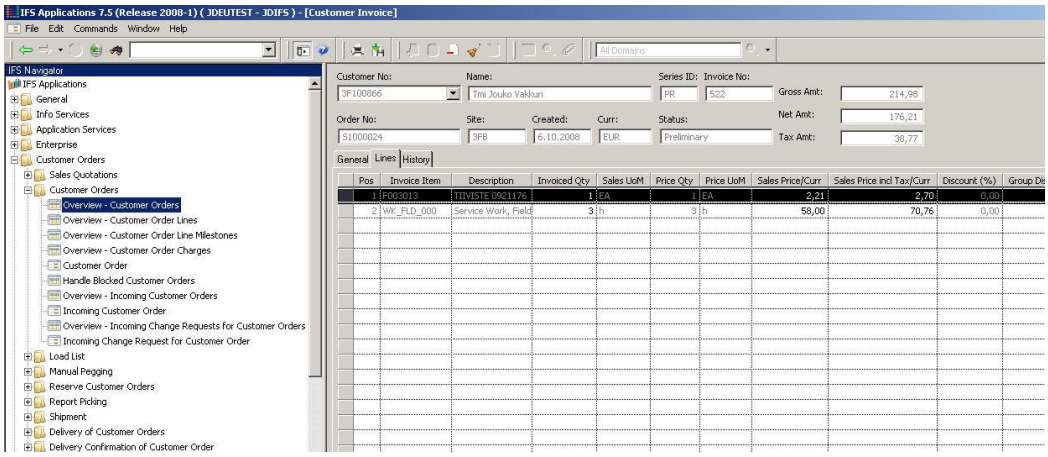
Kuvio 8. Myyntitilaus IFS-järjestelmässä.

7. Laskutus

Myyntitilauksen pohjalta tehdään varsinainen lasku (kuvio 9 ja kuvio 10) asiakkaalle. Laskuun ei tule yhtä yksityiskohtaista erittelyä kuin myyntitilaukselle, vaan siinä yksinkertaisesti näkyy yksi työrivi suoritetusta työstä ja lisäksi laskutettujen osien kustannus.



Kuvio 9. Laskutuksen hallintatyökalu IFS-järjestelmässä.



Kuvio 10. Laskun rivinäkymä IFS-järjestelmässä.