LAHDEN AMMATTIKORKEAKOULU
Lahti University of Applied Sciences

MICROSOFT TEST MANAGERIN KÄYTTÖ TIETOVARASTOYMPÄRISTÖN VALVONNASSA

LAHDEN
AMMATTIKORKEAKOULU
Tekniikan ala
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka
Opinnäytetyö
Kevät 2014
Samu Niemelä

Lahden ammattikorkeakoulu
Tietotekniikan koulutusohjelma

NIEMELÄ, SAMU:

Microsoft Test Managerin käyttö tietovarastoympäristön valvonnassa

Ohjelmistotekniikan opinnäytetyö, 40 sivua

Kevät 2014

TIIVISTELMÄ

Tässä opinnäytetyössä tutkittiin Microsot Test Manager -ohjelmiston soveltuvuutta suorittamaan ja raportoimaan päivittäin tehtäviä valvontoja. Päivittäiset valvonnat ovat ryhmä tiettyjen testitapausten suorittamista asiakkaan tietovarastoympäristössä. Microsoft Test Manager -ohjelmiston taipuminen ohjatun valvonnan suorittamiseen ja lopputulosten raportointiin oli perusteltua. Valvonnan tarkoituksena on havaita normaalista poikkeavat tapahtumat mahdollisimman helposti ja nopeasti sekä kyetä raportoimaan syyt miksi poikkeamia tapahtuu.

Työ tehtiin CGI Suomi Oy:n Lahden toimipisteelle. CGI Suomi Oy on Suomen toiseksi suurin it-palveluyritys ja CGI maailman viidenneksi suurin it-palveluyritys. CGI Suomi Oy työllistää Suomessa noin 3000 henkilöä. Yritysten tarjoamia palveluja ovat konsultointi, tietojärjestelmien integraatiot, IT- ja liiketoimintaprosessien ulkoistamispalvelut sekä IT-palveluiden tuottaminen.

Opinnäytetyössä toteutettiin eräälle yrityksen asiakasympäristölle valvontasuunnitelma Microsoft Test Manager -ohjelmistolla. Valvontasuunnitelman tarkoituksena oli ohjeistaa valvonnassa, kerätä historiatietoja ympäristön valvonnasta ja lisätä asiakkaalle tarjottavaa raportointia ympäristöstään.

Asiasanat: OLAP, tietovarasto, ETL, Microsoft Test Manager, business intelligence, valvonta

Lahti University of Applied Sciences
Degree Programme in Information Technology

NIEMELÄ, SAMU: Using Microsoft Test Manager for monitoring a data warehouse system

Bachelor's Thesis in n software engineering, 40 pages

Spring 2014

ABSTRACT

The objective of this thesis was to investigate how the Microsoft Test Manager software works in monitoring data warehouse environments. In monitoring you complete a series of tests in these environments. The purpose of monitoring is to detect irregular event in the environments easily and fast. You also have to report the reasons why these irregularities occur.

The work was accomplished for CGI Finland Oy, its Lahti´s office. CGI Finland Oy is the second biggest IT solution company in Finland. The company offers consulting, systems integration, IT and business services outsourcing and providing IT solutions.

In this thesis a monitoring plan was executed to one of the company's clients with the Microsoft Test Manager software. The purpose of the monitoring plan is to guide the person who does the monitoring. It also collects historical data from the environment and provdes the customer with information about the'r environment.

Key words: OLAP, data warehouse, ETL, Microsoft Test Manager, business intelligence, monitoring

SISÄLLYS

1 JOHDANTO

Yritykset tuottavat jatkuvasti suuren määrän liiketoimintaansa liittyvää tietoa. Tällaista perinteistä tietoa ovat esimerkiksi pankkitileillä tehtävät tilitapahtumat ja yrityksen myyntitiedot, mutta nykytekniikka mahdollistaa tiedonkeruun myös uusilla toimialoilla. Tämän kerätyn tiedon avulla yritys pyrkii ennustamaan ja analysoimaan, miten sen omaa toimintaa kyetään tehostamaan ja parantamaan. Ongelmana on se, että yritykset eivät osaa tai eivät halua panostaa tämän valtavan datamassan jalostamista siihen pisteeseen, että siitä olisi konkreettista hyötyä niille. Sen lisäksi suurimmissa yrityksissä tämä datamassa on pirstaloituna eri osa-alueilla ja keskittynyt näiden toiminta-alueidensa intresseihin.

Tietovarastointi ja business intelligence ovat nopeimmin kehittyviä tietotekniikan aloja, sillä yritykset haluavat hyödyntää valtavia datamassoja, joita se kerää jatkuvasti omasta liiketoiminnastaan. Yritykset ovat kohdanneet riittämättömyyttä toteuttaa vaativia analysointi- ja raportointijärjestelmiä organisaatioidensa sisällä. Tietovarastointi (DW, data warehouse) keskittää tämän valtavan datamassan, joka koostuu useista eri lähteistä, yksittäiseen ja hyvin hallittuun tietokantaan sekä yhdenmukaistaa datan. Liiketoiminnan hallinta tai toisin sanoen BI-ratkaisut analysoivat tätä dataa tietovarastosta ja esittävät sen sellaisessa formaatissa, joka auttaa ymmärtämään yrityksen liiketoiminnan tapahtumia.

Kun yritysten kiinnostus tietovarastoja ja BI-ratkaisuja kohtaan kasvaa, vaikuttaa se myös suoraan tietovarasto- ja BI-projektien läpivientiin. Projektit ovat jatkuvasti monimutkaisempia, datan määrä kasvaa ja aikataulut kiristyvät. Tuotteen on silti edelleen oltava mahdollisimman hyvin määritykset täyttävä ja toimiva kokonaisuus. Projektin siirtyessä kehityksestä tuotantoon ja ylläpidon piiriin on tärkeää, että tietovarasto on vakaa ja viimeistelty. Tietovarastolle tuotteena saadaan näin lisää arvoa. Erilaisia poikkeamia esiintyy silti korkealaatuisissakin ratkaisuissa. Näihin poikkeamiin täytyy reagoida nopeasti ja palauttaa ympäristö poikkeamasta normaalitilaan. Mitä nopeammin tilanteisiin reagoidaan ja havaitut poikkeamat korjataan, sitä parempaa laatua asiakas kokee saavansa.

Tietovarastoympäristöjen toteutukset poikkeavat toisistaan paljonkin riippuen työkaluista ja käyttöjärjestelmäympäristöistä. Ylläpidossa ympäristöjen valvontojen käytännöt vaihtelevat asiakkuuksien mukaan ja henkilöstön vaihtuvuus voi olla tiheää. Tärkeää olisikin pitää reagointiajat poikkeamiin ja niiden palautukset tasaisen lyhyinä riippumatta henkilöstöstä. Ongelmana on uusien henkilöiden perehdyttäminen toimenpiteisiin, mikä vie aikaa ja resursseja tiimin muilta jäseniltä.

Tässä opinnäytetyössä tutkitaan Microsoft Test Manager -ohjelmiston soveltuvuutta tukea ja ohjata asiakasympäristöjen valvontaa. Ohjelmistolla luodaan valvontasuunnitelma CGI Suomi Oy -yrityksen erään asiakkaan tietovarastoympäristölle. Valvontasuunnitelman on tarkoitus toimia perehdytysmateriaalina valvontojen suorittamiseen. Sen lisäksi se kuvaa sovittuja valvonnan käytäntöjä ja toimenpiteitä poikkeamien ratkaisuun. Lisäksi valvontasuunnitelman on tarkoitus kerätä historiatietoa poikkeamista ja tuottaa raportointia asiakkaalle heidän ympäristöstään.

2 TIETOVARASTOINTI

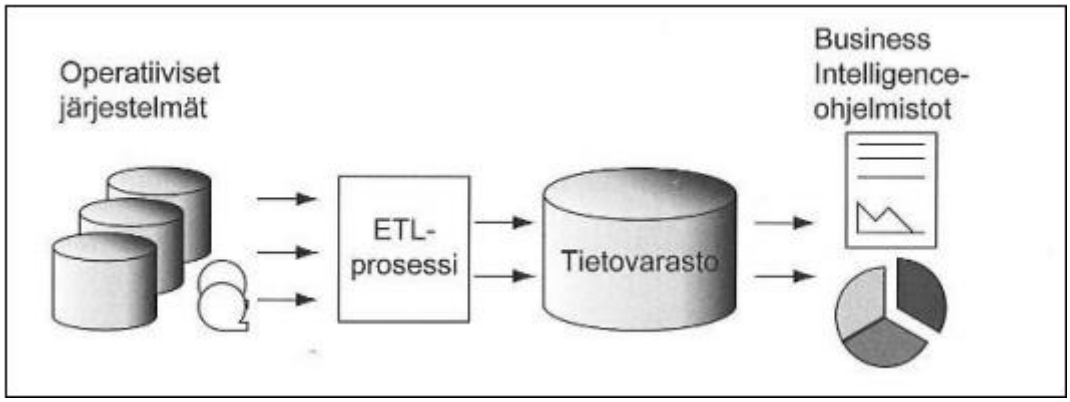
2.1 Tietovaraston määrittely

Yrityksissä tieto on arvokas resurssi eli pääoma. Sitä kertyy valtavia määriä ja sitä tallennetaan suuria määriä yrityksen sisällä, mutta tieto on hajallaan ympäri organisaatiota, eikä sitä ei ole mallinnettu kunnolla, mikä aiheuttaa ongelmia sen ylläpidossa. Operatiiviset järjestelmät eivät sovellu tiedon yhtenäistämiseen ja analysointiin kovinkaan hyvin, sillä niillä tuotetut raportit ovat työläitä rakentaa ja riskinä on se, että tietojen yhdistäminen eri järjestelmistä ei onnistu. Raporttien sisällön kysely on raskasta tietokantojen taulujen läpikäyntiä, mikä hidastaa järjestelmää. Operatiiviset järjestelmät eivät tue tehokasta ja nopeaa raporttien ja kyselyjen tekemistä. Tietovarasto on suuniteltu yhdistämään ja yhtenäistämään tietoja eri lähteistä yhteen hallittuun paikkaan, josta tiedosta voidaan jalostaa informaatiota päätöksenteon avuksi. Näillä business intelligenceksi kutsutuilla ratkaisuilla voidaan esittää tieto helposti käyttäjälle.(Hovi, Koistinen & Hervonen 2009, XI.)

Tietovarastosta on useita määrittelyjä, mutta eniten tunnustusta on saanut W.H.Inmonin, jota pidetään tietovaraston keksijänä, määritelmä tietovarastosta. Inmon määrittelee tietovaraston seuraavanlaisesti:

Tietovarasto on kokoelma aihekohtaisia integroituja tietokantoja, jotka ovat suuniteltu tukemaan päätöksentekoa, ja joiden jokainen tietoyksikkö on suhteessa johonkin ajan hetkeen. (Data Warehouse Definition 2013).

> *Tietojen raportointiin ja analysointiin vaaditaan oma tietokanta, joka on erityisesti suuniteltu tätä tarkoitusta varten. Tietovarastokanta, joka on toteutettu edellä mainitulla tavalla, tukee tehokkaasti business intelligence -työkalujen käyttöä.* (Hovi ym. 2009, 14.)



KUVIO 1. Tietovarastoinnin kuvaus (Hovi ym. 2009, 14).

Kuviossa 1 kuvataan prosessia jota kutsutaan jalostusketjuksi. Eri perusjärjestelmien tietokannoista tuodaan tietoa keskitettyyn tietovarastoon yleensä kerran päivässä. Tietoa jalostetaan matkalla tietovarastoon raakatiedosta raportointi- ja kyselykäyttöön sopivaan muotoon. Tietovarastossa oleva tieto on vain lukukäyttöön tarkoitettu, jolloin operatiivisen järjestelmän kanta sekä tietovarasto pysyvät ajan tasalla keskenään. (Hovi ym. 2009, 14.)

Ensimmäinen vaihe käsittää operatiivisten järjestelmien tietokannat. Tänne kerääntyy kaikki järjestelmiin syötetyt tiedot, ja siellä ne ylläpidetään ajan tasalla. Seuraavassa vaiheessa tieto näistä operatiivisista järjestelmistä siirretään ja käytetään ETL-prosessin läpi. ETL-vaiheessa (Extract – Transform - Load) luetut tiedot muokataan tietovarastoon sopivaksi muodoksi, mikä tarkoittaa yleensä tiedon yhtenäistämistä ja yhdistämistä. Käsittelyn jälkeen tiedot ladataan tietovarastoon. Tietovarasto on, kuten edellä on mainittu, suunniteltu nimenomaan tiedon nopeaa ja helppoa hakua varten. Tietovarastoon tallennetaan monen vuoden tiedot, ja tämä historioituminen mahdollistaa trendianalyysien tarkastelun. Viimeisenä kohtana kuviossa 1 esiintyy BI-työkalujen käyttö. Näillä työkaluilla tehdään valmisraportteja, parametroitavia raportteja ja uusia kyselyjä tietovarastosta. (Hovi ym. 2009, 14-15.)

2.2 Historia

Raporttien tuottaminen yritysten johdolle ATK:n avulla on ollut esillä jo ATK:n syntymisen aikoihin. Ensimmäiset viitteet olivat jo 60-luvulla julkaistussa Sven Hedin kirjassa, jossa kerrottiin ATK:n avulla tuottamaan raportteja yritysten johdolle. Raporttien toteuttamiseen vaadittiin silti vielä noin 20 vuotta, ennen kuin komponentit sallivat sen toteuttamisen.

Tietotekniikan kehittyessä syntyivät erilaiset MIS-järjestelmät (Management Information Systems), DSS-järjestelmät (Decision Support Systems) eli päätöksenteon tukijärjestelmät sekä EIS-järjestelmät (Executive Information Systems), jotka menestyivät vaihtelevasti. Ongelmana näissä järjestelmissä oli päättää, millä tasolla, mitä tietoa ja millä välineillä sitä tarjotaan johdolle. Tieto oli summattu korkealle tasolle, ja hetken niitä käytettyään oli tarve porautua tarkemmalle tasolle raporteissa. Näissä järjestelmissä se ei ollut mahdollista, joten käyttö väheni ja lopulta loppui kokonaan johdon toimesta. (Hovi ym. 2009, 10.)

1980-luvulla puhuttiin jo informaatiokannoista tai lyhyesti infokannoista erotukseksi operatiivisille kannoille. Isoissa yrityksissä lanseerattiin käyttöön raportointiin erikoistuneita palveluyksiköitä, joita kutsuttiin nimeltä Infocenter. Näihin palveluyksiköihin luotiin omia kantoja, joihin tarpeen mukaan kopioitiin erilaisia tietoja operatiivisista järjestelmistä. Suunta oli jo selvästi nykyisten tietovarastojen kaltaiseen suuntaan, mutta näiden järjestelmien suunnittelu oli vielä vajavaista ja tietoja ei kopioitu johdonmukaisesti kokonaisuutena vaan kysynnän mukaan (Hovi ym. 2009, 10.)

Tekniikan kehittyessä vaiheeseen, jossa jokaisella oli henkilökohtainen tietokone ja käytössään taulukkolaskentaohjelma, kuka tahansa pystyi analysoimaan tietoa. Tämän kautta huomattiin, että tiedot olivat usein virheellisiä, mikä johti pahimmillaan vääriin päätöksiin. Tämän lisäksi tiedot olivat puutteellisia ja vanhentuneita ja eri henkilöillä saattoi olla samoista tiedoista eri versio. Tietojen määrityksiä ja kuvauksia ei ollut. (Hovi ym. 2009, 10.)

Tietovarastointi terminä esiintyi ensimmäisen kerran 80-luvun lopulla Devlinin ja Murphyn artikkelissa ”An architecture for business and information system”. Siltikin tietovaraston eli Data Warehousen keksijänä ja suurimpana kehittäjänä pidetään monien toimesta W.H.Inmonia. Inmon on julkaissut useita kirjoja liittyen tietovarastointiin, joista ensimmäinen, ”Building the Data Warehouse” julkaistiin 1990.

Data Vault –tietovarastointimalli on suhteellisen tuore, 2000-luvun alussa kehitetty, malli, joka on suunniteltu nimenomaan tietovarastointiin. Se on hybridimalli, jossa yhdistetään kolmannen normaalin mallin ja tähtimallin parhaita osia. Ongelmana vanhoissa malleissa on jäykkyys ja hitaus muutoksille. Yritysten tarve tiedolle muuttuu jatkuvasti, ja tietovaraston täytyy olla valmis muuttumaan näihin muutoksiin nopeasti. Data Vault on joustava, skaalautuva ja johdonmukainen malli toteuttamaan yritysten tietovarasto. (Linstedt 2013.)

2.3 Tähti- ja lumihiutalemalli

2.3.1 Faktataulu

Moniuloitteisessa mallinnuksessa faktataulu sisältää suorituskykyä kuvaavia arvoja, jotka ovat peräisin liiketoiminnan prosesseista. Faktataulut sisältävät yleensä kahden tai useamman viiteavaimen (Foreign Key), jotka ovat yhteydessä dimensiotaulujen perusavaimiin (Primary Key). (Kimball, Margy 2013, 10-12). Faktataulun perusavain koostuu dimensiotaulujen perusavaimista yhdisteltynä. Faktataulusta löytyviä tietoja lasketaan yleensä yhteen erilaisilla dimensioista saatavilla tasoilla. (Hovi, Huotari, Lahdenmäki 2005, 135). Kuviossa 2 on esimerkki eräästä mallista, jolta jälleenmyynnin faktataulu voisi näyttää.

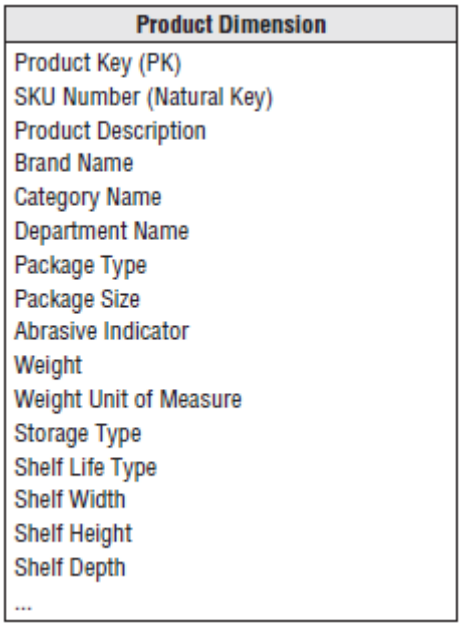
KUVIO 2. Esimerkki faktataulusta.

Koska faktataulu sisältää tapahtumarivejä, lisätään sinne jatkuvasti uusia rivejä. Vanhoja rivejä ei päivitetä, jotta historiatieto säilyy. Tämän takia faktataulu kasvaa jatkuvasti. Faktataulut vievätkin moniulotteisissa malleissa noin 90% kokonaistilasta. (Kimball, Margy 2013, 10-12.)

Faktataulun rivi edustaa kaikkien dimensioiden risteyskohtaa. Näin ollen mikäli kaikkia dimensioarvoja ei faktariville löydy, ei tietoa tallenneta faktatauluun. Tästä syystä faktataulua voidaan kutsua harvaksi. (Hovi, 1997, 76-77).

2.3.2 Dimensiotaulut

Toisin kuin faktataulussa, dimensiotaulujen tiedot ovat yleensä melko staattisia ja niistä saadaan erilaisia ryhmittelyjä ja hakukenttiä faktataulun rivejä summaaville kyselyille. Dimensiotaulut ovat yleensä huomattavasti pienempiä kuin faktataulut, mutta kuten kuviosta voidaan todeta, sisältävät ne huomattavan paljon enemmän eri sarakkeita ja attribuutteja. (Hovi ym. 2005, 135.) Dimensiotaulut pitävät sisällään faktatauluissa esiintyvien lukujen kirjalliset kuvaukset. Ne esittävät suurta roolia moniuloitteisessa mallissa, sillä niiden avulla siitä saadaan ymmärrettävä. (Kimball, Margy 2013, 14.)



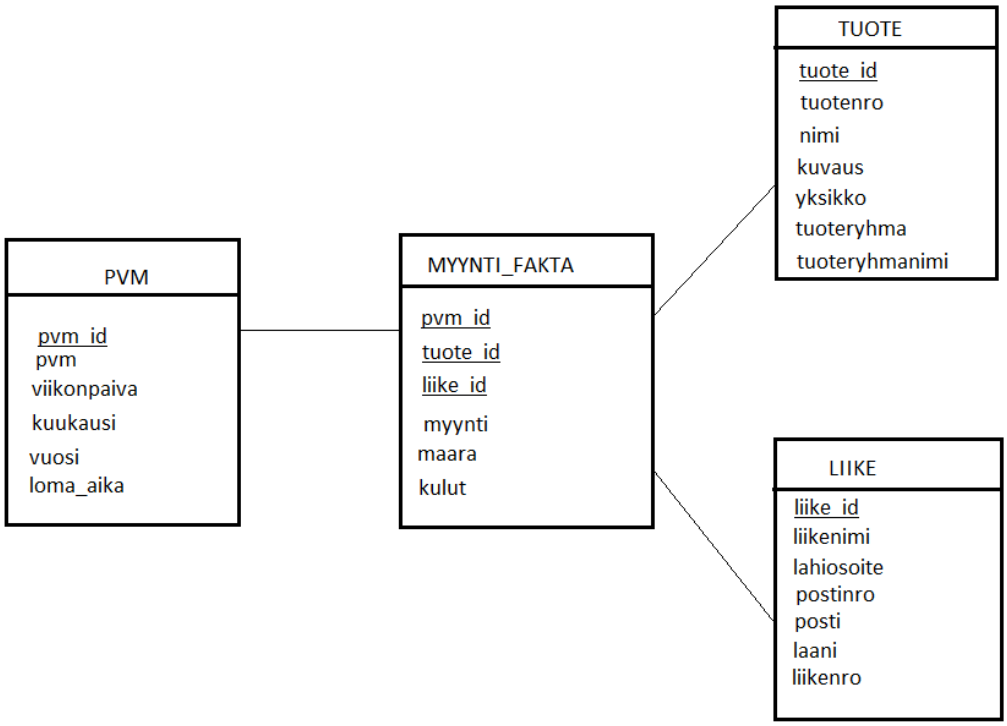
KUVIO 3. Esimerkki dimensiotaulusta.

Dimensiotauluissa, toisin kuin faktataulussa, tiedot voivat muuttua ajan myötä. Esimerkiksi kuvion 3 dimensiotaulussa voisi tuoteryhmän (Category Name) arvo muuttua. Se miten näihin muutoksiin suhtaudutaan, on tiedettävä etukäteen halutaanko historiatietoa säilyttää, jolloin rivien tietoja ei päivitetä, vai kirjoitetaanko olemassa olevan tietojen päälle. Mikäli muutokset tehdään olemassa oleville riveille, kyetään tietoja vertailemaan edellisiin, kuin muutokset olisivat olleet voimassa jo pidempään. Mikäli muutoksia ei tehdä, ei vertailu historiaan ole yhtä helppoa, mutta historia näkyy oikeana. (Hovi ym. 2009, 41.)

2.3.3 Tähtimalli

Ralph Kimball kiinnostui tietovarastoinnista ja aloitti jalostaa moniuloitteisen suunnittelun menetelmää ns. tähtimallia. Kimballia pidetäänkin dimensionaalisen suunnittelun ja tähtimallin ykkösnimenä. (Hovi ym. 2009, 10-11). Tähtimallin (Star Schema) nimi juontuu sen rakenteesta, joka muistuttaa tähteä sakaroineen ja se tukee hyvin moniuloitteista ja numeerista tietoa. Tähtimallin avulla voidaan simuloida moniuloitteisuutta käyttäen kuitenkin perinteistä relaatiotietokantaa. Tähtimalli soveltuu hyvin tietovarastointiratkaisuihin, sillä sen tavoitteena on tuottaa kyselyt ja raportit mahdollisimman helpoksi toteuttaa ja suorituskyvyltään

tehokkaiksi. (Hovi ym. 2009, 36-39). Tähtimallissa keskellä on faktataulu, jota ympäröi pienemmät dimensio- eli ulottuvuustaulut. (Hovi ym. 2005, 135).

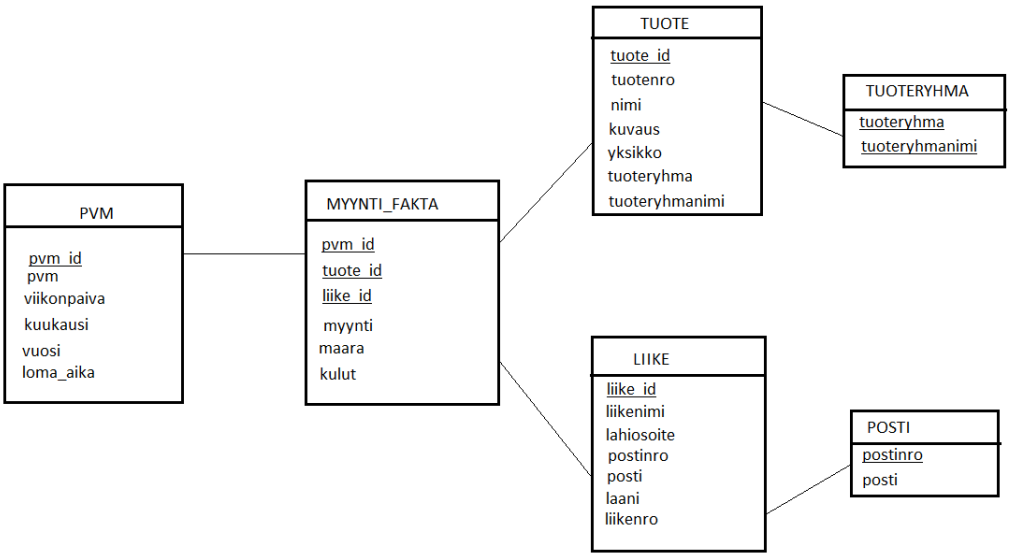


KUVIO 4. Tähtimallin esimerkki.

Kuviossa 4 on esimerkki tähtimallista, jossa on keskellä faktataulu ja sen ympärillä kolme dimensiotaulua. Perusavaimet on alleviivattu. Faktataulussa tietoja ei toisteta eli se on normalisoitu. Dimensiotauluissa sen sijaan tietoa toistetaan eli ne ovat denormalisoidussa muodossa. Tämän avulla suorituskykyä parannetaan. Haut ovat tämän myötä tehokkaampia ja mahdollistaa helpot ja yksinkertaiset kyselyt. (Hovi ym. 2005, 135.)

2.3.4 Lumihiutalemalli

Kuten yllä käytiin läpi, dimensiotauluissa tapahtuu toisteisuutta, esimerkiksi kuvion 4 TUOTE-taulussa on tuoteryhma ja tuoteryhmanimi attribuutit. Mikäli nämä dimensiotaulut myös normalisoidaan, syntyy tähtimallin variaatio, lumihiutalemalli. Kuviossa 5 nähdään lumihiutalemalli jatkettuna kuviosta 4.



KUVIO 5. Lumihiutalemallin esimerkki.

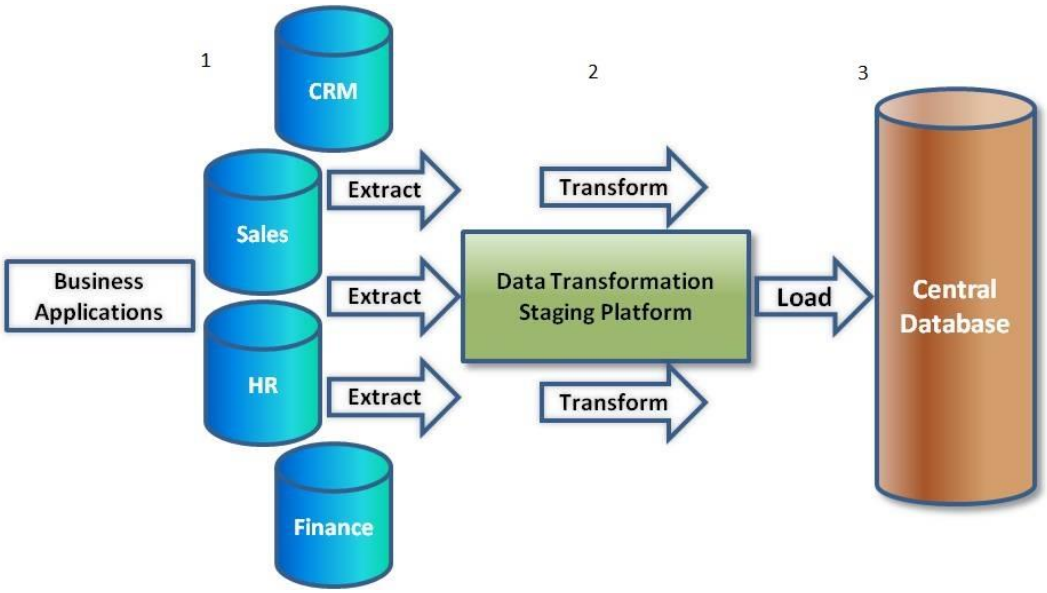
Tähtimallia pidetään yleisesti parempana kuin lumihiutalemallia, sillä se on yksinkertaisempi ja antaa paremman suorituskyvyn. BI-ratkaisujen kannalta asia ei ole näin selkeä. Tietyt BI-välineet toimivat paremmin lumihiutale-, toiset tähtimallin kanssa yhteen. (Hovi ym. 2009, 40.)

2.4 ETL-prosessi

Tietovarastoon on tuotava säännöllisesti uutta dataa, jotta se palvelisi parhaalla mahdollisella tavalla omaa tarkoitustaan, eli tukisi päätöksentekoa ja liiketoimintoa kehittämisen apuna. Prosessia, jossa tieto haetaan yhdestä tai useammasta operatiivisesta järjestelmästä, muokataan tietovarastokantaan sopivaksi muodoksi ja kirjoitetaan lopuksi tietokantaan, kutsutaan ETL (Extract-Transform-Load) prosessiksi. Tämän prosessin tavoitteena on tehdä työtä yöllisissä eräajolatauksissa ja jalostaa tietoa valmiiksi, jotta päiväsaikaan tapahtuvat kyselyt kannasta tapahtuvat nopeasti ja helposti. (Hovi ym. 2009, 48-49).

ETL-prosessien toteutus on pohjimmiltaan tietokantapohjaista eräajosovelluksen ohjelmointia. Prosessin luontia varten on kehitetty erityisiä ETL-työkaluja, jotka sisältävät paljon valmiita toiminnallisuuksia. Näiden työkalujen avulla ETL-prosessi on helpompi toteuttaa ja hallita kokonaisuutena. Tästä huolimatta, ETL-

prosessin suunnittelu, toteutus ja testaus vie suurimman osan tietovarastoprojekteissa. Tämä prosessi on syytä suunnitella ja toteuttaa huolella, sillä tämän prosessin avulla tehostetaan tiedon raportointia ja analysointia BI-välineiden avulla. (Hovi ym. 2009, 48-49).



KUVIO 6. ETL-prosessin kuvaus.

Kuviossa 6 kuvataan ETL-prosessia. Ensimmäinen (1) vaihe on raakadatan hakeminen operatiivisistä järjestelmistä sekä ulkoisista tietolähteistä työalueelle (Staging Area). Tämä voidaan tehdä joko työntö- tai vetomenetelmällä. (Hovi ym. 2009, 48-49). Koska usein ei tarkasti tiedetä mitä tietoa tarvitaan, ladataan tietoa mahdollisimman paljon. Näin ollen halutun tiedon rajaaminen tapahtuu myöhemmässä vaiheessa. (Oracle. 2005).

Seuraava vaihe (2) on tietojen yhtenäistäminen. Tarkoituksena on muokata lähdejärjestelmän tai siirtotiedostojen rakenteissa olevat tiedot tietovarastokannan helppokäyttöisiin, raportointia tukeviin rakenteisiin. Esimerkkinä voisi pitää päivämäärätietojen muuntamista yhteiseen sovittuun muotoon tai henkilötunnuksen oikean muodon tarkistamista. Myös mahdolliset duplikaattirivit etsitään ja poistetaan työalueella. Tällaista tietojen korjaamista ja selvittelyä kutsutaan ”tietojen siivoamiseksi” tai puhdistamiseksi (data cleansing). (Hovi ym. 2009, 56-57.)

Viimeisenä (3) tapahtuu tietojen lataaminen tietovarastoon. Tapahtuma- tai faktatyyppiset rivit lisätään yleensä vahojen perään. Staattiset, dimensiotyyppiset taulut päivitetään joko päälle tai historioidaan ja uudet rivit lisätään. Tiedot siirretään tietovarastoon kaikki tieto kerrallaan käyttämällä sovelluksia, joilla toteutetaan tietovarastoon vienti. (Hovi ym. 2009, 58-59.)

2.5 Business intelligence

BI-ratkaisuilla tarkoitetaan yritysten ja julkisten organisaatioiden liiketoiminnallisen informaation esittämistä ymmärrettävässä muodossa. Tämä informaatio auttaa henkilöstöä ohjaamaan toimintaansa oikeaan suuntaan. BI-ratkaisut pohjautuvat pitkälti tietovarastotototeutuksiin ja niihin tehtäviin kyselyihin. BI-ratkaisut käsittävät yleensä valmiiden raporttien katsomista, tekevät uusia hakuja tietovarastoista, hyödyntävät moniuloitteista analysointia tai katsovat valmiita tunnuslukuja tuloskorttiratkaisujen läpi. (Hovi ym. 2009, 75.)

2.5.1 Business intelligence historia

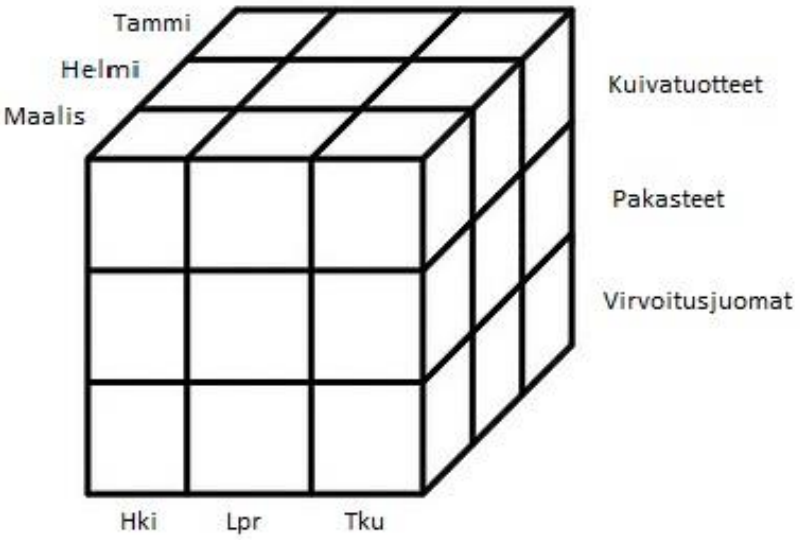
80-luvun lopulla markkinoille ilmestyi ensimmäisiä alkeellisia verkkopohjaisia BI-järjestelmiä, joita kutsuttiin ryhmäpäätöksenteon järjestelmiksi (Group DSS). Nykymuotoisista BI-ratkaisuista alettiin puhua 1989, kun Howard Dresner alkoi käyttää yhteistä termiä business intelligence nykyäänkin käytössä olevista menetelmistä ja teknologioista. Relaatiokantojen ja varsinkin SQL-kielen yleistyminen 90-luvulla edesauttoivat BI-ratkaisujen kehittymistä. Ensimmäistä kertaa kyettiin suorittamaan liiketoimintalähtöisiä kyselyjä SQL-kielen avulla ja BI-ratkaisuilla pyrittiin näiden kyselyjen tekemistä helpottamaan ja tuomaan jokapäiväiseen työelämään. Moniulotteinen analysointi (OLAP) syntyi 90-luvun alussa ja erilaiset web-sovellukset 90-luvun lopussa. Nämä tekniikat ovat edelleen suosituimpia puhuttaessa BI-ratkaisuista. (Hovi ym. 2009, 77.)

2.6 OLAP

OLAP tulee sanoista On-Line Analytical Processing, ja sillä tarkoitetaan moniuloitteista analysointia. OLAP on yksi yleisimmistä BI-ratkaisuista, jolla

tietovarastossa olevaa tietoa kuvataan. OLAP on teknologiaratkaisu, joka tarjoaa moniuloitteisen näkymän tietoon liiketaloudellista analysointia varten. Tällä pyritään mahdollisimman nopeisiin kyselyihin sekä helppokäyttöisyyteen. OLAP-ratkaisut perustuvat tyypillisesti ulottuvuuksissa sijaitsevien hierarkioiden hyödyntämiseen ja tietojen summaamiseen eli aggregointiin. Hierarkioiden hyödyntämistä käytetään porautumisessa. OLAP-ratkaisuun mallinnetaan dimensioiden hiearkioita, jolloin käyttäjä voi tarvittaessa porautua karkeamman tason luvuille. Periaatteessa tämä tarkoittaa sitä, että käyttäjä voi tutkia miten summataso muodostuu yksityiskohtaisemmalla tasolla. (Hovi ym. 2009, 91.)

OLAP:n avulla yrityksen tietoja kyetään tarkastelemaan useasta eri näkökulmasta. Esimerkkinä voidaan ottaa yrityksen myynti. Myyntitietoja voidaan tarkastella esimerkiksi aikajaksoittain, liikkeittäin tai tuoteryhmittäin. OLAP:sta puhuttaessa esiintyy myös usein sana kuutio. Vaikka OLAP:n rakennetta voidaankin kuvata hyvin kuutiona, sisältää se yleensä enemmän ulottuvuuksia kuin kolme kappaletta. Yksi keskeinen ja hyvin yleinen näistä ulottuvuuksista on aika. (Hovi, 1997, 57-59.)



KUVIO 7. Esimerkki kolmiuloitteisesta kuutiosta.

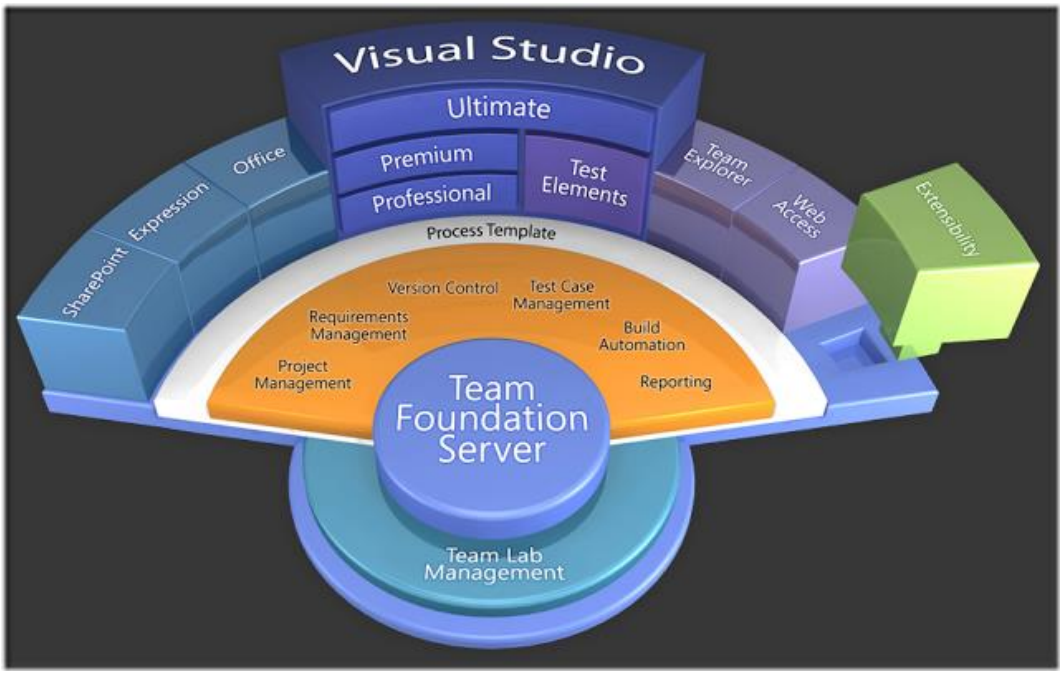
Kuviossa 7 on esimerkkinä kolmiuloitteinen kuutiomalli, joka kuvaa yllä olevaa esimerkkiä jonkin yrityksen myynnistä. Sen dimensioina esiintyvät aika, tuote ja paikkakunta. Faktatietona toimii myyntimäärä. Näin saadaan dimensioiden risteyskohdissa tietoa tietyn kuun, tietyn tuotteen ja tietyn paikkakunnan myyntimäärästä.

3 VISUAL STUDIO APPLICATION LIFECYCLE MANAGEMENT

Microsoft Visual Studio Application Lifecycle Management (ALM) on Microsoftin sateenvarjotermi tuotekokonaisuudelle, jolla kyetään hallinnoimaan sovelluksen elinkaarta alusta loppuun. Versionhallinan lisäksi Visual Studio ALM tarjoaa tuotteita koko sovelluksen elinkaaren hallintaan. Ajatuksena on tarjota välineet aina sovelluksen suunnittelusta ja mallintamisesta kehitystyöhön, testaamiseen ja levittämiseen saakka. Tämän lisäksi tästä löytyy myös välineitä perusprosessienm, kuten työnjaon, versionhallinnan, seurannan ja tiimin jäsenten välisen kommunikoinnin tehostamiseen. Visual Studio ALM perustuu valmiisiin prosessimalleihin, joita voi tarvittaessa jalostaa. (Järvinen 2008, 4-5.)

Microsoft Visual Studio ALM esiintyi ensimmäisen kerran Visual Studio 2005:ssä, jossa se tunnettiin Microsoft Visual Studio Team System nimellä. Tätä ennen Microsoft tarjosi versionhallintaan sopivaa Visual SourceSafea, mutta tämä ei saanut laajaa kiitosta käyttäjien joukossa. Nimi uudistettiin ALM:ksi Visual Studio 2010 version julkaisun yhteydessä. Uusin julkaisu tätä kirjoittaessa on Visual Studio 2013 -versio.

Perinteiset RAD-kehittimet (Rapid Application Development) toimivat hyvin yksittäisen kehittäjän työkaluna, mutta ne eivät sovellu laaduntarkkailuun tai projektijohtamiseen. Mikäli tiimin jäsenten määrä ja roolit kasvavat, ei organisoitu projektijohtaminen enää ole mahdollista rajallisilla työkaluilla. Tällainen RAD-kehitin on esimerkiksi Visual Studio -ohjelmisto ilman Visual Studio ALM -kokonaisuutta. (Järvinen 2008, 228.)



KUVIO 8. Visual Studio ALM rakenne. (Microsoft's Approach to Application Lifecycle Management 2012)

Visual Studio Application Lifecycle Management koostuu seuraavista osista:

1. Visual Studio
2. Visual Studio Test Professional
3. Team Foundation Server (TFS)
4. Visual Studio Lab Management.

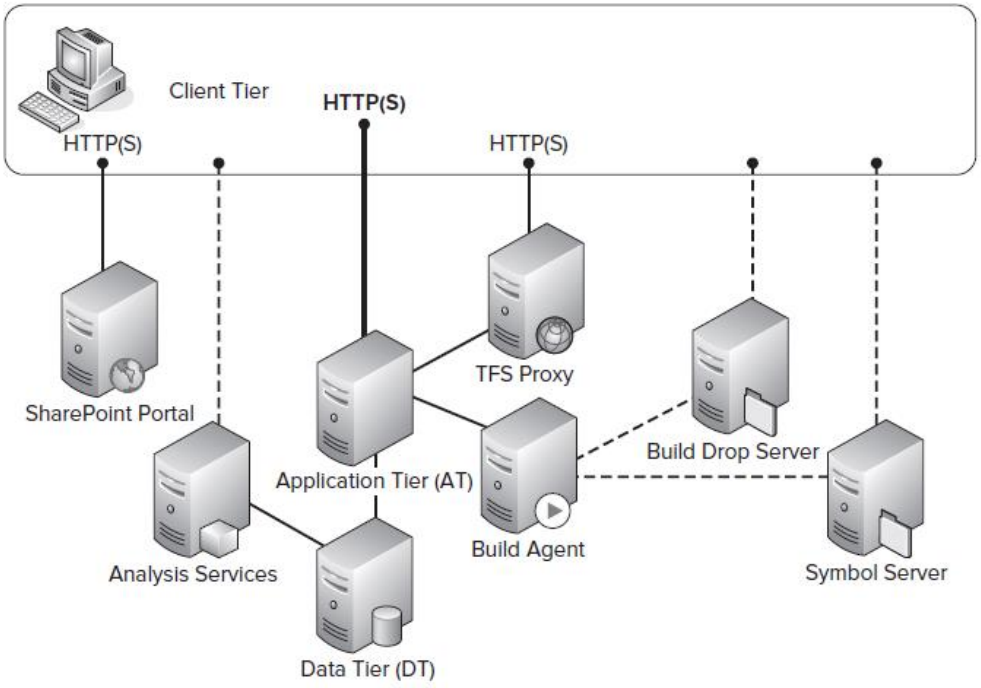
Kuviossa 8 avataan Visual Studio ALM rakennetta. Sen keskeisenä osana on palvelinohjelmisto Team Foundation Server. TFS sisältää muun muassa projektinhallintatyökalut (Project Management, Reporting, Requirements Management, Work Item Tracking), versionhallinnan (Version Control) sekä testitapausten hallinan (Test Case Management). Kuviosta nähdään myös, että korkeammalla tasolla löydetään niin sanotut Client-ohjelmistot, kuten Visual Studio, Office ja SharePoint (Blankenship,Woodward, Holliday, & Keller 2013, 4.)

3.1 Team Foundation Server

Microsoft ALM:n oleellisin osa on Team Foundation Server. Ohjelmisto toimii palvelimena asiakasohjelmille, joita ovat Visual Studion versiot 2005-versiosta alkaen. (Järvinen, J. 2008, 4-5). Team Foundation Server sisältää seuraavat ominaisuudet:

1. projektinhallintatyökalut
2. versionhallinta
3. build-prosessin hallinta
4. testitapausten hallinta
5. raportointityökalut
6. palautetyökalut
7. kehitysympäristön hallintatyökalut
8. työlistan hallinta (Work Item Tracking, WIT).

(Blankenship ym. 2012, 4).



KUVIO 9. Team Foundation Serverin arkkitehtuuri. (Blankenship ym. 2012, 60.)

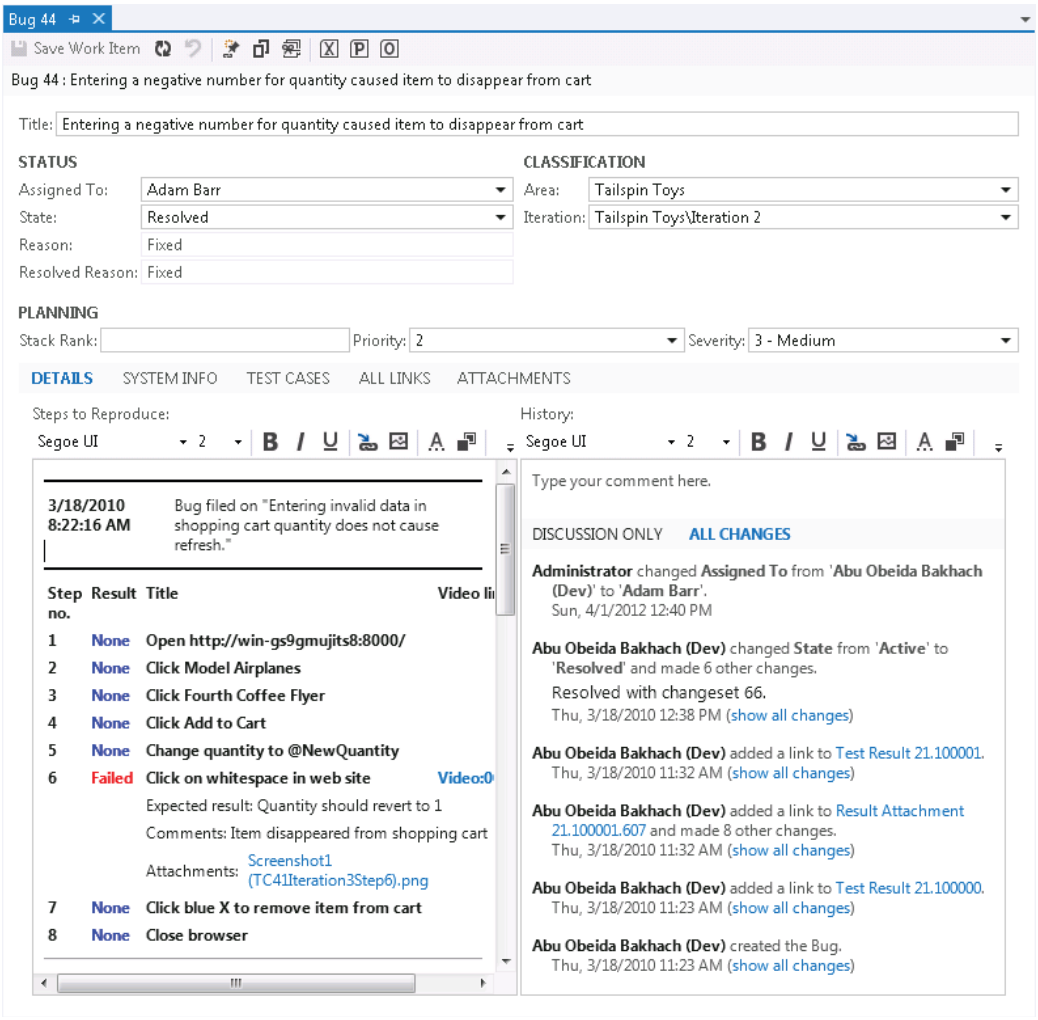
Team Foundation Server jakaantuu kahteen eri tasoon: sovellustasoon (application tier) ja datatasoon (data tier). Sovellustaso sisältää web services -palveluita, joilla kommunikoidaan client-ohjelmistojen kanssa. Se myös sisältää mahdollisuuden kommunikoida serverin kanssa web clientin kautta. Näin ollen client-ohjelmistoja, kuten Visual Studiota, ei tarvitse asentaa. (Blankenship ym. 2012, 4). Datataso on rakennettu SQL Serverin pohjalle. TFS tallentaa siis kaikki tiedot SQL Server -tietokantaan, joka toimii TFS:n tietovarastona. Esimerkiksi versionhallinnan tiedot, tehtävälistat ja käyttäjätiedot tallennetaan tietovarastoon.Tallennetusta tiedosta saadaan raportteja TFS:n toiminnoista, ja koko TFS:n instanssin varmuuskopiointi ja palautus onnistuvat tietovarastosta käsin. (Järvinen 2008, 234).

3.2 Work Item

Työlista koostuu työtehtävistä (work item). Työtehtävä on Team Foundation Serverin perustyökalu projektin hallinnalle. Microsoft itse määrittää työtehtävän näin: ”...kantaan tallennettu merkintä, jolla seurataan projektin työnjakoa ja edistystä.”. (Blankenship ym. 2012, 247.)

Työtehtävätyyppejä on useita erilaisia. Esimerkiksi Task-tyyppinen työtehtävä kuvaa työvaihetta, joka on saatettava valmiiksi. User Story-tyyppisellä työtehtävään voidaan kirjata lisätietoja rakennettavasta ohjelmistosta ja ongelma- ja virhekohtia voidaan kuvata Bug tai Issue -tyyppisillä tehtävillä. (Blankenship ym. 2012, 247.)

Kuviossa 10 on esimerkkinä Bug-tyyppinen työtehtävä. Suurin osa kentistä ovat yksiselitteisiä, kuten kenelle työtehtävä on osoitettu ja mikä työtehtävän tila on.



KUVIO 10. Esimerkki työtehtävästä. (Blankenship ym. 2012, 249.)

3.3 Visual Studio Test Professional

Visual Studio Test Professional on tuotekokonaisuus, jonka tärkein tarjoama ohjelmisto on Microsoft Test Manager (MTM). MTM on suunnattu testausroolin omaaville henkilöille ja se on suoraan yhteydessä Team Foundation Serverin kautta projektiin. Microsoft Test Managerin avulla luodaan testaussuunnitelmia, jotka auttavat määrittämään ja hallinnoimaan projektiin käytettävää resursseja testauksen osalta.Se esiteltiin ensimmäisen kerran Visual Studion 2010 versiossa. (Johnson, B. 2013, 992.)

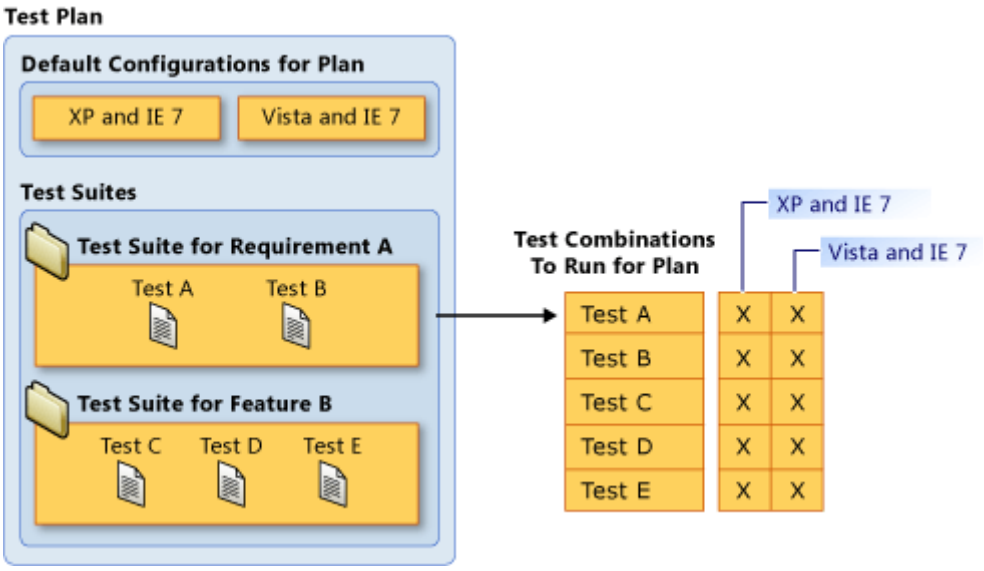
3.4 Microsoft Test Manager

MTM on työkalu, jolla suunnitellaan, toteutetaan ja valvotaan testausaktiviteetteja. Se on integroitu suoraan Team Foundation Serveriin ja mahdollistaa luoda, päivittää ja priorisoida työkohtia (work items). Ylätasolla MTM hallitsee kyseisiä kohtia:

- testaussuunnitelmia
- testauksen suunnittelua
- testauksen suorittamista ja analysointia
- virheiden keruuta ja raportointia
- työlistan hallintaa
- testiympäristön hallintaa

(Rossberg, Olausson 2012, 323-324.)

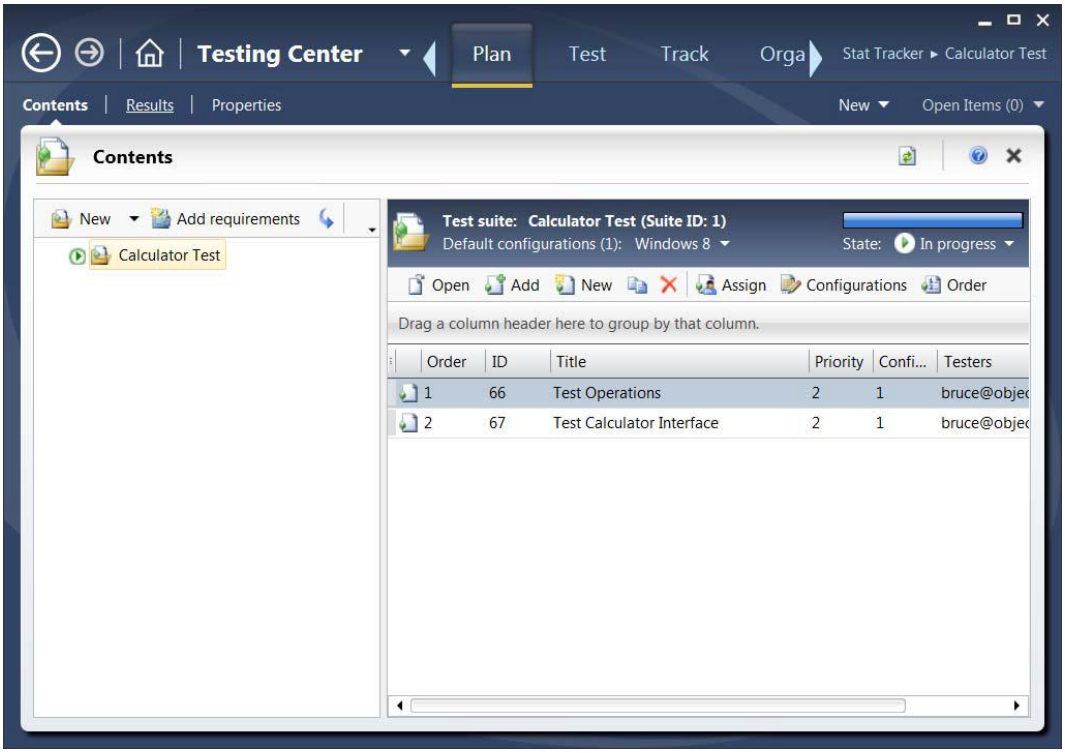
Testausta varten MTM sisältää kolmiportaisen hierarkian. Ylimpänä on testaussuunnitelma (Test Plan), seuraavana on testiryhmä (Test Suite) ja viimeisenä testiapaus (Test Case). Testaussuunnitelma ja testiryhmät auttavat organisoimaan testitapauksia, jotka sisältävät itse testattavat kohdat. Kuviossa 11 käydään läpi MTM:n rakenne.



KUVIO 11. Microsoft Test Managerin rakenne.

Testaussuunnitelma (Test Plan)

Testaussuunnitelma auttaa määrittämään ja valvomaan kaikkia kohtia, joita testataan jokaisessa julkaisussa. Testaussuunnitelma luodaan jokaiselle julkaisulle, jotta kehitystiimin tekemien muutosten toimivuus voidaan varmistaa.



KUVIO 12. Microsoft Test Manager (Johnson 2013, 993.)

Testiryhmä (Test Suite)

Testaussuunnitelman sisällä voidaan testitapaukset ryhmitellä testausryhmien mukaan. MTM sisältää kolme erilaista testiryhmää:

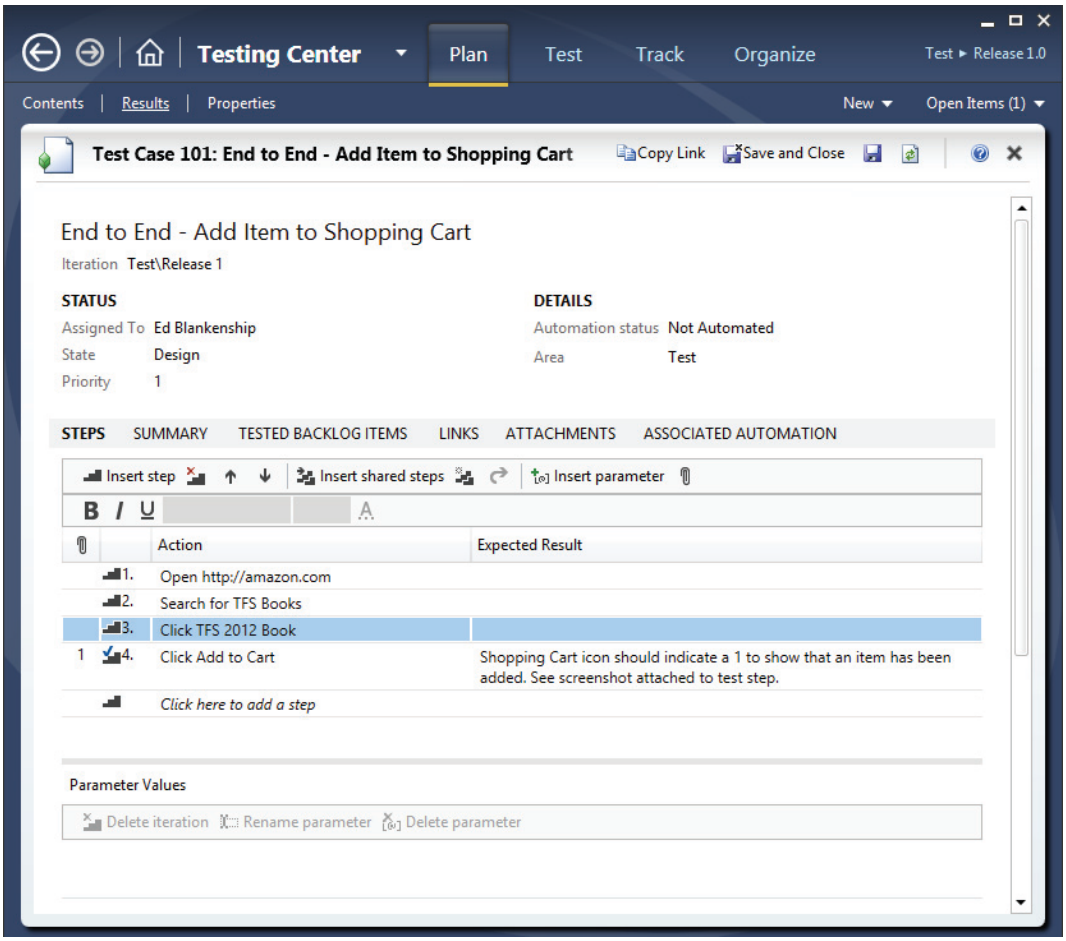
1. vaatimuspohjaiset testiryhmät
2. kyselypohjaiset testiryhmät
3. staattiset testiryhmät

(Blankenship ym. 2012, 634.)

Testitapaus (Test Case)

Testitapaukset sisältävät toiminnat, jotka testaajan täytyy suorittaa ja todeta oikeaksi. Näin varmistetaan että sovellus suoriutuu kuten on haluttu. Esimerkki testitapauksesta voisi olla seuraava: käyttäjä pystyy siirtymään verkkosivustolle ja luomaan tunnuksen. Kuviossa 13 nähdään esimerkki testitapauksesta. (Blankenship ym. 2012, 634.)

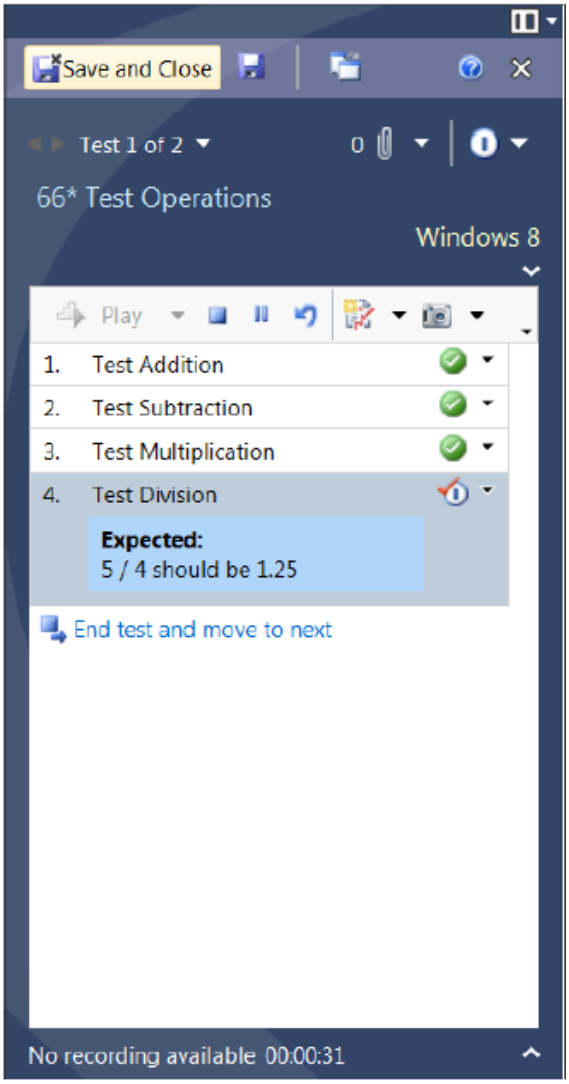
Testitapauksen rakenne noudattaa TFS:ssä olevia Work Itemiä. MTM testitapauksissa on lisätty erityinen Steps-osio, jonka kohtia testaajan tulisi noudattaa. Nämä kohdat ovat tapahtumia, joilla applikaation toimivuutta testataan. Koska testitapauksen rakenne noudattaa Work Itemin rakennetta, voidaan testitapaus liittää mihin tahansa Work Itemiin. (Blankenship ym. 2012, 634.)



KUVIO 13. Esimerkki testitapauksesta (Blankenship ym. 2012, 635)

3.4.1 Test Runner

Test Runner on Microsoft Test Managerin työkalu, jolla testitapauksia ajetaan. Kuviosta 14 nähdään Test Runner käynnistettynä. Työkalu listaa testitapauksen askeleet ja ohjaa sen kulkua. Askeleen käytyä läpi, voidaan se merkitä joko onnistuneeksi (Passed) tai virheelliseksi (Failed). (Johnson 2013, 994.)



KUVIO 14. Esimerkki Test Runner -työkalusta (Johnson 2013, 994.)

Kuviossa 14 nähdään askeleiden yläpuolella työkalurivi, jolla testiä hallitaan. Testitapaukseen voidaan lisätä tiedostoja, kommentteja ja kuvakaappauksia, jotka voivat auttaa testitapausten tuloksia analysoitaessa. (Johnson 2013, 995.)

4 VALVONTASUUNNITELMAN TOTEUTUS

4.1 Johdanto

Valvontasuunnitelma suunniteltiin ja toteutettiin CGI Suomi Oy:n yritykselle Lahden toimipisteelle. Työn tarkoituksena oli tarkastella Microsoft Test Manager 2012 -ohjelmiston soveltuvuutta tukemaan päivittäisten valvontojen suorittamista ja raportointia. Yksi tavoitteista oli myös suunnitella ja toteuttaa valvontasuunnitelma niin, että se toimisi ohjeistuksena valvonnan suorittamiselle.

Valvonnat tehdään asiakasympäristöittäin tapauskohtaisesti, eli jokainen ympäristö valvotaan tiettyjen sovittujen käytäntöjen mukaisesti. Tämä on johtanut tilanteeseen, jossa käytännöt on dokumentoitu eri paikkoihin sekalaisesti ja valvontojen raportointia ei ole mahdollista saada selville yhdestä paikasta. Opinnäytetyön tarkoituksena on saattaa valvontakäytännöt ja valvonnan lopputulosten raportointi yhden työkalun alle. Valvontasuunnitelma toimisi perehdytyksenä valvontaan ja tuottaisi informaatiota asiakkaalle ympäristön toiminnasta.

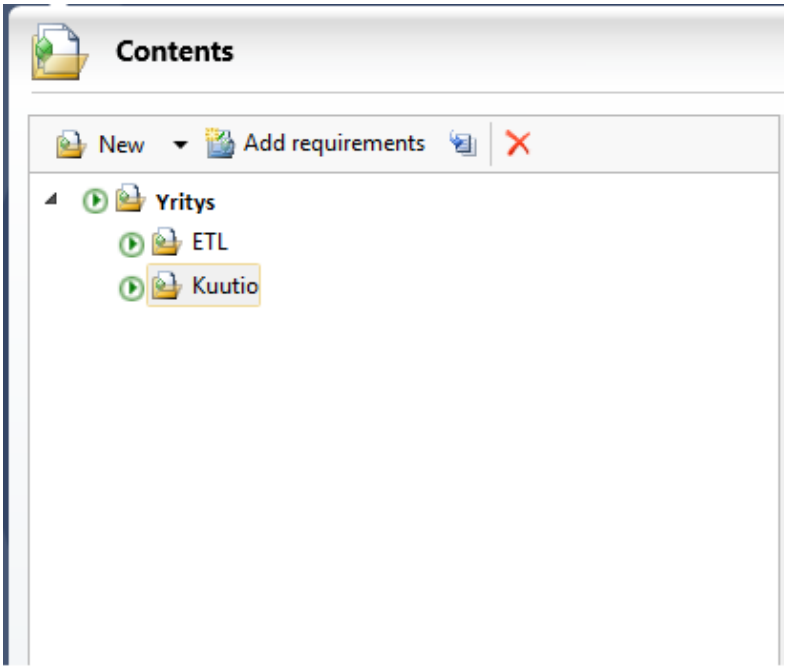
Valvontasuunnitelman suunnittelun ja toteutuksen apuna toimi vanhempi tiimin jäsen, jonka vastuualue on asiakasympäristöjen valvonta. Valvontasuunnitelma toteutettiin yhdelle yrityksen asiakasympäristöistä, jossa valvontaa tehtiin päivittäin. Valvontasuunnitelman ideana oli tuoda valvonnan tulokset yhden työkalun alle, jolloin historiatietoa kerrytetään ja tietoa keskittyy yhteen pisteeseen. Valvontojen käytännöt pyrittiin myös kuvaamaan testitapauksissa yksityiskohtaisesti, jolloin uuden henkilön olisi helppo tehdä valvonta.

4.2 Valvontaryhmien toteutus

Valvontasuunnitelman toteuttaminen aloitettiin syventymällä valvonnan vaiheisiin ja vastuualueisiin. Valvonnat asiakasympäristöissä käsittävät yleensä kaksi pääkohtaa: ETL-ajojen läpiviennin tarkastuksen ja kuutioiden prosessoitumisen. Näiden lisäksi asiakasympäristöissä on tiettyjä asiakaskohtaisia ratkaisuja, joiden läpivientejä valvotaan. Ympäristö, jonka valvonnan pohjalta valvontasuunnitelma toteutetaan MTM-työkalulla, valvotaan ETL-ajojen läpivientien onnistumista sekä

kuutioiden prosessoitumista. Palvelimelta, jolla ETL-ajot suoritetaan, tulee sähköposti ajojen tilanteesta aamulla. Viestien tarkistuksen jälkeen otetaan yhteys asiakkaan ympäristöön ja tarkistetaan ETL-ajojen mahdolliset virhetilanteet. Kuutioiden prosessoitumisesta, joka suoritetaan toisella palvelimella, tulee myös tilanneraportti sähköpostiin. Kuutiot, jotka ovat kaatuneet tai kesken, käydään tarkistamassa asiakasympäristössä sähköpostin tarkistuksen jälkeen, mikäli sille on tarvetta.

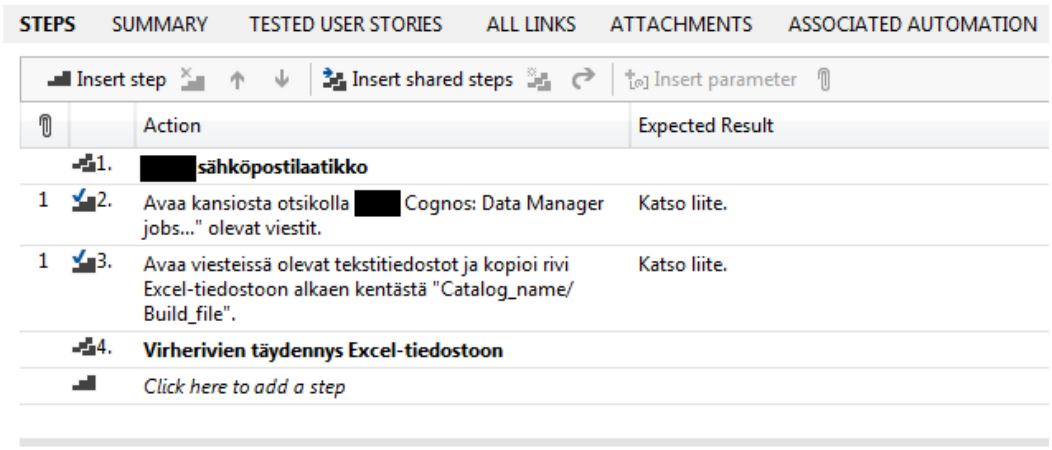
Valvontasuunnitelma rakennettiin näiden kahden, ETL-ajojen ja kuutioiden, ympärille niin että näille luotiin omat osiot eli testiryhmät. Näissä ryhmissä tärkeimpinä testitapauksina on ETL-ajojen läpivientien onnistuminen sekä kuutioiden prosessoituminen. Nämä testaukset käydään läpi joka kerta, mikä synnyttää historiatietoa. Tämän historiatiedon avulla voidaan raportoida asiakkaalle ympäristön toiminnan vakautta sekä tutkia sisäisesti minkä tyyppisiä virheitä ympäristössä esiintyy. Muut testitapaukset näiden testiryhmien ympärille luotiin valvonnan läpivientien näkökulmasta. Testitapausten askeleet (steps) suunniteltiin mahdollisimman yksityiskohtaisesti, ja niissä käytettiin hyväksi askeleihin liitettäviä kuvia. Askeleiden oletettavat tulokset onkin pääosin kuvattu kuvaliitteellä.



KUVIO 15. Valvontasunnitelman rakenne.

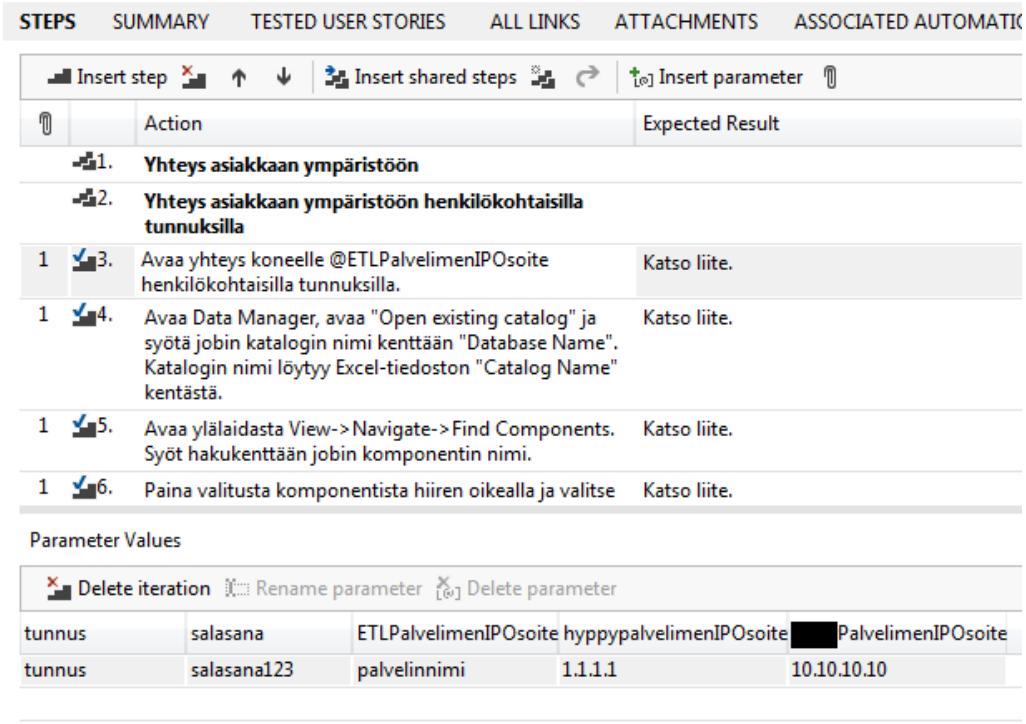
4.2.1 Testitapausten rakenne

Valvontaryhmien testitapauksissa esiintyy monia samantyyppisiä askeleita (steps). ETL-ajojen ja kuution prosessoinnin valvonnan kulku muistuttavatkin toisiaan. Askeleiden turhan toistamisen välttämiseksi on niiden suunnitteluvaiheessa pyritty havaitsemaan yhtenäiset välivaiheet ja luomaan niille oma askel. Tämän jälkeen siitä on muodostettu jaettu askel (shared step), jonka voi jakaa eri testitapausten kesken. Näin vältytään saman askeleen luomisesta eri testitapauksissa, ja myös muutosten hallinta helpottuu kun pystytään muokkaamaan keskitetysti yhtä jaettua askelta. Tällaisia tapauksia esimerkiksi olivat sähköpostien paikantaminen ja asiakkaan ympäristöön kirjautumiset. Kuviossa 16 on avattu eräs testitapahtuma, jossa on jaettuja askeleita. Ne näkyvät käyttöliittymässä tummempina.



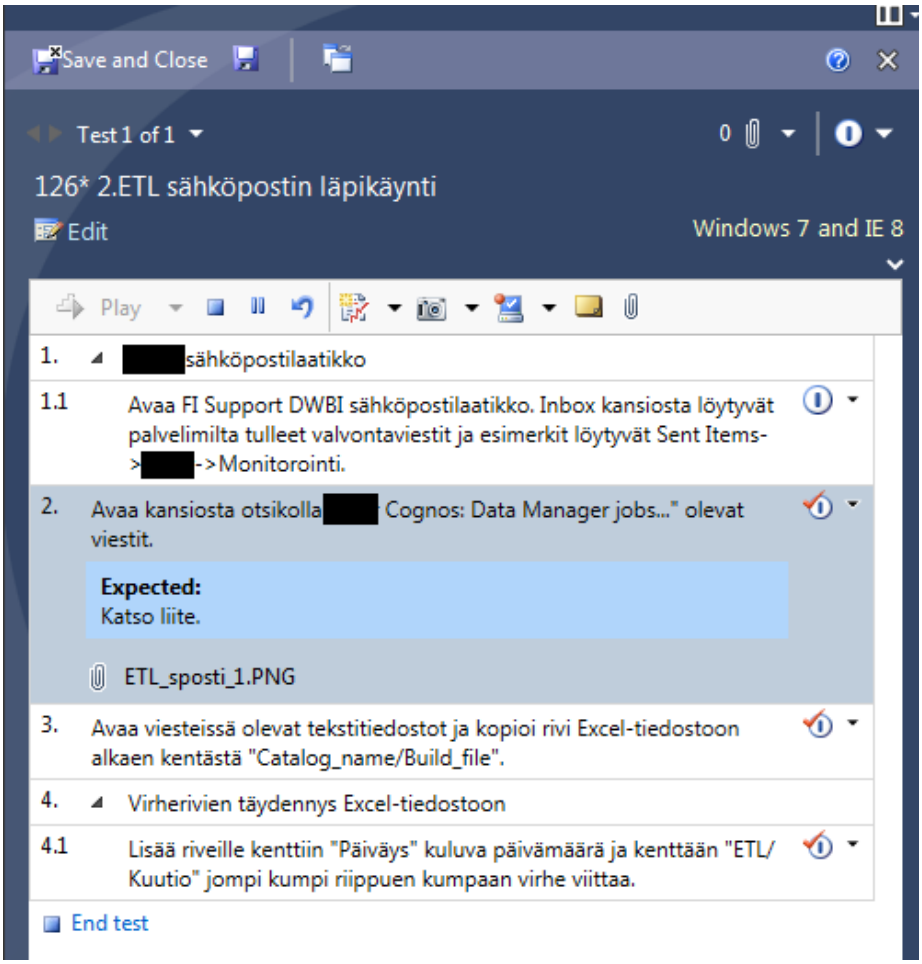
KUVIO 16. Esimerkit jaetuista askeleista.

Toinen muutoksia tukeva ominaisuus askelten luomisessa oli parametrien käyttö. Askeleiden kuvauksiin voidaan lisätä parametreja, joiden arvot testiä ajettaessa voidaan määrittää jokaisessa iteraatiossa erikseen. Testitapauksia ei tarvitse käydä läpi useampaan kertaan, mutta parametrien käyttö lisää muokattavuutta. Valvontasuunnitelmassa käytettiin parametreja määrittämään asiakasympäristön IP-osoitteita ja yhteisiä tunnuksia. Näin mikäli muutoksia tapahtuu, voidaan parametrien arvoja muuttaa keskitetysti yhdestä paikasta. Kuviossa 16 nähdään testitapauksessa määritellyt parametrit sekä niille asetetut esimerkkiarvot.



KUVIO 17. Parametrisointi Microsoft Test Managerissa.

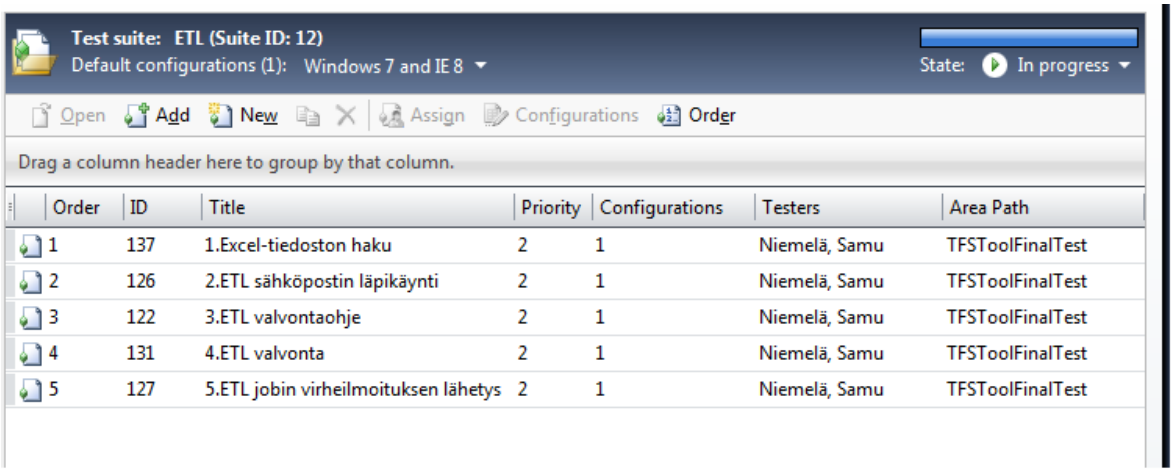
Askeleita määritettäessä kuvauskentän lisäksi jokaiselle askeleelle voidaan määrittää odotettu tulos. Sanallisen kuvauksen lisäksi tähän kenttään voidaan liittää myös tiedostoja. Useissa testitapausten askeleissa onkin käytetty sanallisen oletetun tuloksen kuvauksen sijaan kuvakaappausta. Askeleen lopputuloksesta on otettu kuvakaappaus, ja se on liitetty askeleeseen liitteeksi. Kun valvontaa tekevä henkilö suorittaa testitapausta, tulee askeleissa liitteenä näkyviin kuvakaappaus, jonka voi avata valvonnan aikana ja tarkistaa lopputulos. Kuviossa 18 on esimerkki, miltä testitapaus näyttää Test Runnerissa.



KUVIO 18. Esimerkki liittestä askeleessa Test Runner -ohjelmassa.

4.2.2 ETL-ajojen valvontaryhmä

ETL-ajojen valvontaryhmä sisältää neljä eri valvontatapausta ja kuviossa näkyy ETL-ryhmän testitapaukset. ETL-ajojen valvonnan kattamiseksi ryhmään muodostui neljä testitapausta: ensimmäisessä testitapauksessa käydään läpi ETL-valvontaviestien käsittely, toisessa käydään läpi, mistä ETL-ajoja voi hallinnoida, kolmas testitapaus käsittelee itse ajojen läpivientiä ja neljäs sisältää virhetilanteissa toimimisen. Testitapaukset on kuvattuna kuviossa 19.



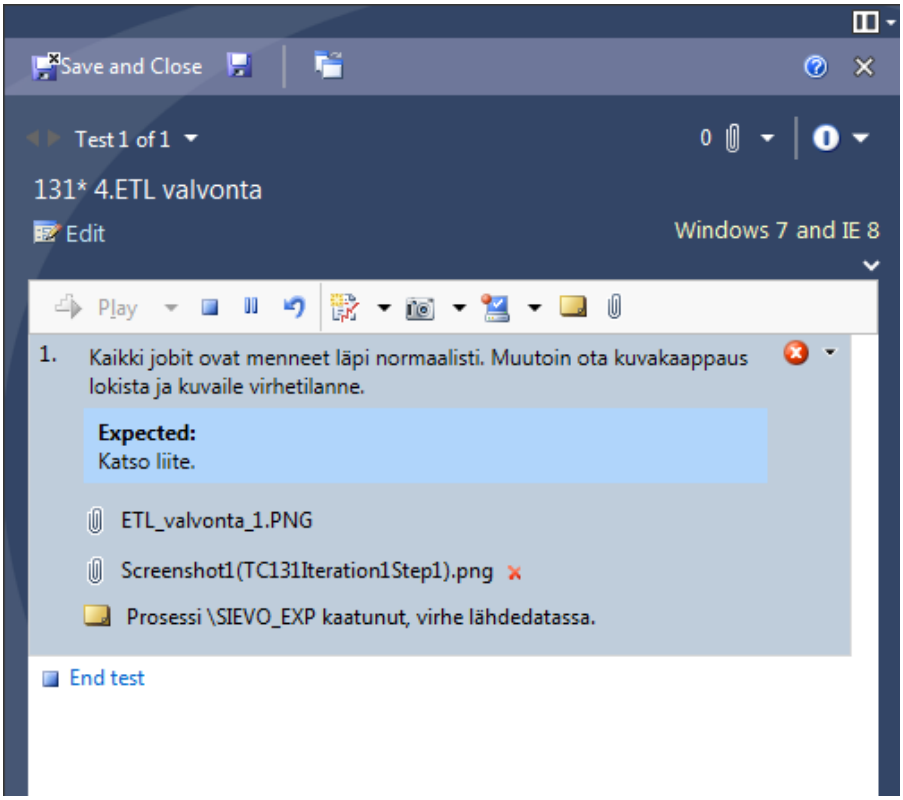
KUVIO 19. ETL-testiryhmän rakenne.

Ensimmäisessä testitapauksessa paikannetaan ja kuvataan yhteinen Excel-tiedosto, johon kerätään kaikki virheelliset ja keskeneräiset ajot ja kuutiot päivittäin. Excel-tiedosto on toiminut yhtenä raportointivälineenä asiakkaalle tällä hetkellä, mutta raportointi on tämän kautta melko karkealla tasolla. Tarkoituksena on MTM:n avulla saada raportoinnista tarkemman tason tietoa, kuten virhetilanteita aiheuttaneet syyt.

Toisessa testitapauksessa käydään läpi asiakkaan palvelimelta saapuvat sähköpostit, joissa on ETL-ajojen tilat. Sähköpostilaatikkoon saapuvat ilmoitukset virheellisistä ja keskeneräisistä ajoista. Sähköpostilaatikkoon voi saapua myös vain ilmoitus, että kaikki ETL-ajot ovat menneet normaalisti läpi. Sähköposteissa olevissa liitteissä on ilmoitettu, mitkä ajot ovat kaatuneet tai kesken. Nämä rivit siirretään Excel-tiedostoon, ja rivien avulla etsitään kyseiset ajot palvelimelta.

Kolmannessa testitapauksessa käydään läpi itse ETL-ajojen läpikäynti. Testitapauksessa on käytetty hyödyksi jaettuja askeleita sekä parametrisointia. Tärkeimpinä kohtina on opastaa miten kaatuneen tai keskeneräisen ETL-ajon tila ja lokitiedot löydetään käyttämällä asiakkaan ympäristössä olevaa työkalua. Sen lisäksi testitapauksessa käydään läpi, miten kaatuneen tai keskeneräisen ETL-ajon suhteen pitää toimia. Tapauksessa käydään läpi yleisimmät toimenpiteet, jotka valvoja voi itse tehdä ajon suhteen ja mitä jatkotoimenpiteet suoritetaan tämän jälkeen. ETL-ajojen toimenpiteet ja valvonta ovat hieman suoraviivaisempia kuin kuutioiden prosessoinnin virhetilanteiden selvittäminen.

Neljäs testitapaus on se tapaus, johon ETL-ajojen virheet raportoidaan. Tapauksessa on yksi askel, jossa ajot ovat menneet läpi normaalisti. Mikäli tämä ehto ei täyty, testiä suorittaessa merkitään askel epäonnistuneeksi ja selvitetään virhetilanteet sanallisesti. Sen lisäksi, mikäli lokitiedosto on saatavilla, otetaan siitä kuvakaappaus. Näin jää myös lokitiedostosta jälki virhetilanteeseen. Kuviossa 20 on testitapaus käynnissä ja esimerkki siitä, miten askel merkitään virhetilanteessa. Askel merkitään virheelliseksi, ja mahdollisesta lokitiedostosta otetaan kuvakaappaus liitteeksi askeleeseen. Askeleeseen myös liitetään kommenttikenttä, jossa virhetilanne selostetaan lyhyesti. Kun tarkastellaan ETL-ajojen läpivientejä, virhetilateita on helpompi analysoida ja tutkia näiden toimenpiteiden jälkeen.

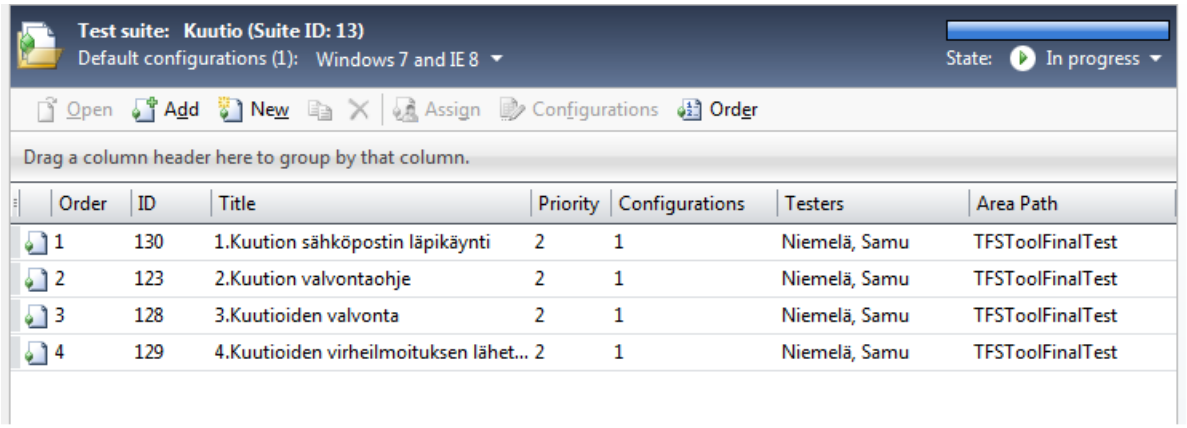


KUVIO 20. ETL-valvontatestitapaus.

Viimeisessä testitapauksessa käydään läpi virhetilanteiden ilmoitus asiakkaille ja täydennetään Excel-tiedosto puuttuvien osioden kohdalta. Tässä tärkeimpinä kohtina on virhetilanteissa oikeiden jakeluryhmien määrittely sähköposteja lähettäessä. Toinen tärkeä kohta on sähköpostissa olevan lomakkeen oikean tyyppinen täyttö tietyissä kohdissa.

4.2.3 Kuutioiden valvontaryhmä

Kuutioiden valvontaryhmät ovat rakenteeltaan samanlaiset ETL-ajojen valvontaryhmien kanssa. Ryhmät ja niiden askeleet ovat tarkoituksellisestikin viety samantyyppisiksi, jotta jaettuja askeleita voidaan käyttää tehokkaasti hyödyksi. Kuutioiden prosessoinnin virhetilanteiden valvonta on monimutkaisempaa. Virheen laatu tulee tunnistaa ja tehdä oikeat toimenpiteet tämän perusteella.



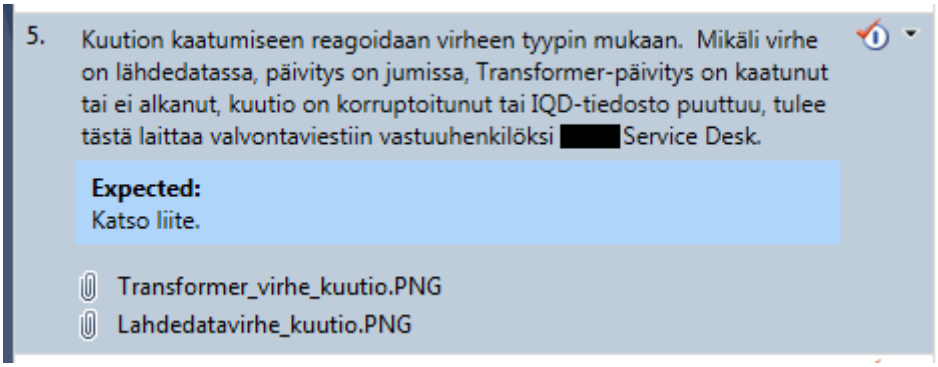
KUVIO 21. Kuution valvontaryhmän rakenne.

Ensimmäinen testitapaus ohjeistaa sähköpostien läpikäynnin, kuten edellä olleen ETL-ajojen vastaava testiryhmä. Sähköposti sisältää listan kuutioista, jotka ovat joko kaatuneet prosessoinnin aikana tai ovat kesken viestin lähetyksen tapahtuessa. Listan riveille tehdään samat toimenpiteet kuin ETL-ajojen vastaavassa ryhmässä.

Kuution valvontaohjeen suunnittelu oli kenties haastavin testitapauksista. Kuutioiden tilat tarkistetaan asiakkaan ympäristössä olevan portaalin kautta, joka listaa kaikki kuutiot ja niiden tilat. Kuutioilla on myös lokitiedostot, joissa kuvataan kuution prosessoinnin vaiheet. Mikäli kuution prosessointi on kaatunut, ensimmäiseksi tarkistetaan portaalin kautta lokitiedoston kautta, mihin prosessointi on kaatunut.

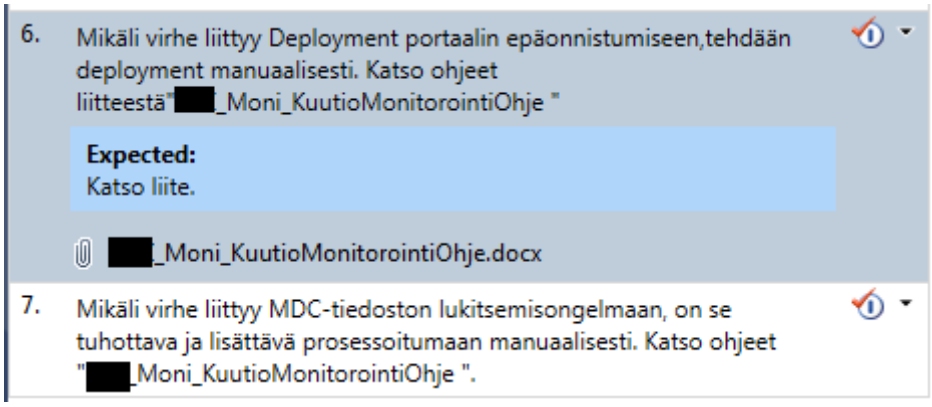
Virhetilanteesta riippuu, tekeekö jatkotoimenpiteet valvoja vai asiakkaan toimihenkilö. Testitapauksen askeleissa oli otettava huomioon kaikki mahdolliset virhetilanteet ja niiden aiheuttamat toimenpiteet.

Yleisten askelten jälkeen tulee vastaan ensimmäinen askel, jossa on kuvattu virhetilanteet, jotka eivät vaadi valvojalta muita toimenpiteitä kuin virheilmoituksen tekeminen asiakkaalle. Tämän askeleen liitteenä ovat kuvakaappaukset virheiden esimerkeistä, jotka on otettu aikaisemmista lokitiedoista. Virhetekstit lokitiedostoissa ovat hyvin pitkälti samanlaiset virhetilanteissa, joten malliesimerkit antavat suuntaa virheen syylle. Kuviossa 22 on avattuna yllä kuvattu askel.



KUVIO 22. Kuution valvontaohjeen askel.

Seuraavana ovat tilanteet, jotka vaativat toimenpiteitä valvojalta ja jokainen virhetilanne on sijoitettu omaan askeleeseen. Näin varmistetaan jokaisen virheen tarkka kuvailu ja se mitä toimenpiteitä se vaatii. Kuten edellisessä askeleessa, on myös tässä liitteenä malliesimerkki vastaavanlaisen virheen lokitiedostosta. Tämän lisäksi on liitteenä Word-tiedostoja, joissa virheen esiintymisen aiheuttamat toimet kuvataan seikkaperäisesti. Näin askeleen kuvaus pysyy selkeällä tasolla, ja liitteenä olevasta tiedostosta saadaan lisätietoa. Kuviossa 23 voidaan nähdä kaksi esimerkkiä askeleesta, joissa on kuvattu virhetilanne ja sen liitteet.



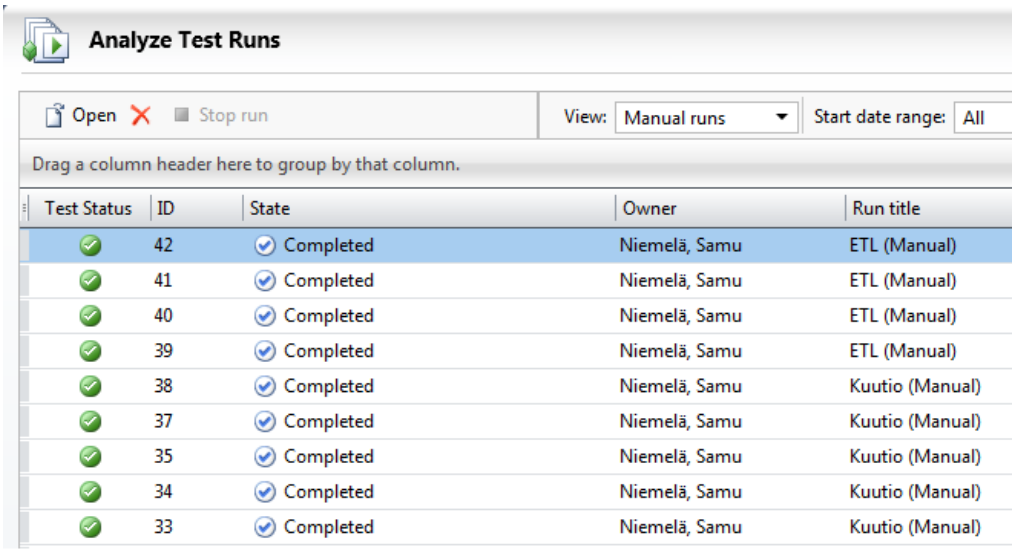
KUVIO 23. Valvontaohjeen askel.

Kaksi seuraavaa testitapausta ovat rakenteeltaan samantyyppiset kuin ETL-valvonnan testiryhmässä. Kolmannessa testitapauksessa merkitään kuutioiden prosessoinnin läpiviennin onnistuminen ja neljännessä testitapauksessa ohjeistetaan virheilmoituksen lähetys asiakkaalle ja Excel-tiedoston päivitys.

4.3 Raportointi

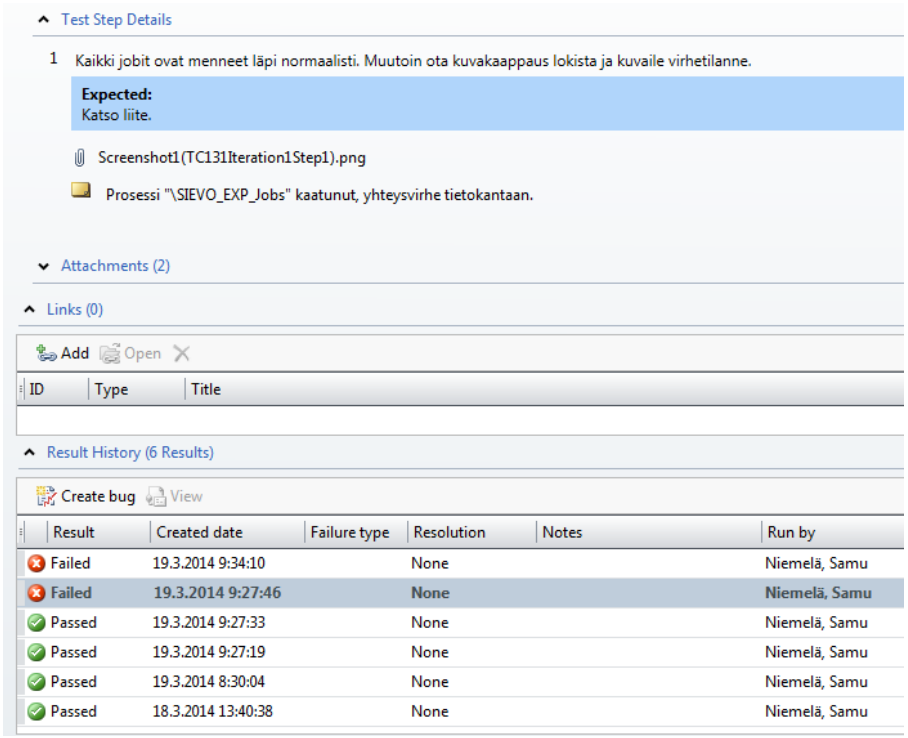
Sekä ETL- että kuution valvontaryhmässä on yksi testitapaus, jonka lopputuloksella raportoinnin näkökulmasta katsottuna on merkitystä. Muiden testitapausten tarkoitus on pitkälti toimia ohjeistuksena valvojalle. Tästä syystä niiden lopputuloksilla ei ole niin suurta merkitystä.

Testitapausten lopputuloksia pystytään analysoimaan testitapauskohtaisesti. Microsoft Test Manager tallentaa jokaisen ajokerran sen omaan tietokantaan, joten testitapausten historiointi tapahtuu automaattisesti. Listasta voidaan valita tietyn päivän testitapaus tai virheeseen päättynyt testitapaus ja avata se tarkempaa analysointia varten.



KUVIO 24. Testitapausten tulosten listaus.

Tietyn testitapauksen avattuaan, saadaan lisätietoja lopputuloksesta. Kuviossa 25 on esimerkki virheellisestä testitapauksesta ja sen lisätiedoista. Alaosassa on kyseisen testitapauksen historiatietoja menneistä tuloksista. Tulokset voidaan vaihtaa listasta valitsemalla.



KUVIO 25. Virheellisen testitapauksen lokitiedosto.

Testitapauksen askeleet kuvataan Details-osiossa, ja sieltä nähdään, mikä askel on merkitty virheelliseksi. Valitsemalla askeleen avautuu sen kuvaus, oletettu lopputulos, siihen liitetyt liitteet sekä kommentit, jotka on lisätty testiajon aikana. Virhetilannetta tarkasteltaessa voidaan nopeasti tutkia, onko virheitä esiintynyt aikaisempina päivinä tai etsiä lisätietoja muista virhetilanteisiin kulkeutuneista valvontakerroista. Testitapauslokiin voidaan myös jälkikäteen lisätä kommentteja, kuten esimerkiksi toimenpiteet, jotka tehtiin ratkaistaessa ongelmaa.

5 YHTEENVETO

Tavoitteena testaussuunnitelman suunnittelussa ja toteutuksessa oli muodostaa selkeä ja havainnollistava valvonnan käytäntöjen läpikäynti Microsoft Test Manager -ohjelmiston avulla. Sen lisäksi valvontasuunnitelman tarkoituksena oli kerätä valvontojen lopputuloksista historiatietoja ja tuottaa asiakkaalle lisäarvoa ympäristöstään. Tarkoituksena oli koota ympäristöön liittyvät valvontakäytännöt saataville yhdestä paikasta ja niiden seikkaperäinen läpikäynti eri testitapauksissa. Näin se toimisi valvontaan perehdytyksenä uudelle henkilölle. Kaiken kaikkiaan projektin läpivienti oli onnistunut. Asetetut tavoitteet täyttyivät suurimmalta osin, ja valvontasuunnitelman käyttö tuo lisää arvoa työtehtävissä muun muassa virheiden historioinnilla.

Valvontasuunnitelman suunnittelu ja toteutus vietiin läpi nopeassa aikataulussa, minkä takia nämä kohdat suoritettiin pitkälti samaan aikaan.Valvontasuunnitelman toteutusta tehdessä tuli selväksi, että selkeällä suunnitelmalla ja syvemmällä valvontakäytäntöjen perehtymisellä testitapaukset olisivat tulleet selkeämmiksi. Luonnoksia testatessa esille nousi selkeitä puutteita askeleissa ja niiden korjaus vei aikaa. Selkeä suunnitteluvaihe olisi vähentänyt tuotoksen iterointia ja lopputulos olisi ollut hieman laadukkaampi. Suunnitelmaa olisi pitänyt testata useammalla koehenkilöllä, mutta aikataulun takia siihen ei ollut mahdollisuutta. Sen lisäksi sopivia henkilöitä ei ollut tarjolla kyseisellä ajan hetkellä.

Käytin valvontasuunnitelmassa monipuolisesti Microsoft Test Manager -ohjelmiston ominaisuuksia hyödyksi. Testitapaukset toteutettiin käyttäen jaettuja askeleita ja parametrisointia, jotta suunnitelman muokattavuus onnistuu helposti. Kuvakaappausten käyttö askelten oletetuissa lopputuloksissa on havainnollinen tapa kuvata lopputulemaa. Poikkeamissa tehtävät toimenpiteet käydään askeleissa läpi sellaisella tasolla, jotta valvoja kykenee toimimaan itsenäisesti. Tämä säästää resursseja tiimin jäseniltä, koska peruskohdat käydään läpi valvontasuunnitelmassa.

Valvontasuunnitelman seuraava vaihe on ottaa valvontakierrossa käyttöön, jotta historiatietoa saadaan kerrytettyä. Tämän kautta raportointia ongelmatilanteista

saadaan tuotettua asiakkaalle. Valvontasuunnitelmaa testataan perehdytyksessä tulevien harjoittelijoiden kohdalla, jolloin saadaan todenmukaista palautetta siitä, millä tavoin suunnitelmaa voidaan kehittää. Valvontasuunnitelmien käyttönottoa muiden asiakasympäristöjen kohdalla on myös tarkoitus tutkia.

Raportoinnin kehittäminen on myös kehityksen kohteena jatkossa. Team Foundation Serverin kautta saatavat oletusraportit testitapausten kulusta vaativat sen päivittämistä uudempaan versioon. Nyt Microsoft Test Manager -ohjelmiston kaikki mahdolliset ominaisuudet eivät ole käytössä tämän asian myötä. Oletusraporttien käyttöönoton jälkeen omien kustomoitujen raporttien luominen olisi seuraava askel. Näin saataisiin juuri haluttua ja oikeanlaista informaatiota testitapausten lopputuloksista.

LÄHTEET

Blankenship, E.,Woodward, M., Holliday, G. & Keller, B. 2013. Professional Team Foundation Server 2012. Indianapolis, Indiana: John Wiley & Sons, Inc

Data Warehouse Definition. 2013. [viitattu 20.12.2013]. Saatavissa: http://www.1keydata.com/datawarehousing/data-warehouse-definition.html

Hovi, A., Koistinen, H. & Hervonen, H. 2009. Tietovarastot ja Business Intelligence. Porvoo: WSOY

Hovi, A., Huotari, J., Lahdenmäki, T. 2005. Tietokantojen suunnittelu & indeksointi. Porvoo: WS Bookwell

Hovi, A. 1997. Data Warehousing, Tietovarastotekniikka. Gummerus ja Kirjapaino Oy

Johnson B. 2013. Professional Visual Studio 2012. Indianapolis, Indiana: John Wiley & Sons, Inc

Järvinen, J. 2008. Visual Studio 2008 –käsikirja. Porvoo: WSOY

Kimball, R., Margy, R. 2013. The Data Warehouse Toolkit Third Edition. Indianapolis, Indiana: John Wiley & Sons, Inc

Linstedt D. Data Vault Basics. [Viitattu 12.1.2014] Saatavissa:http://danlinstedt.com/about/data-vault-basic

Microsoft's Approach to Application Lifecycle Management. 2012. [viitattu 5.1.2014] Saatavissa: http://2.bp.blogspot.com/-4a5KKUNeHkw/T_Vd-pCVA8I/AAAAAAAAA5g/pBJMRGhgQPI/s1600/Visual+Studio+2010+Stadium+Diagram_thumb.png

Oracle. 2005. Oracle® Database Data Warehousing Guide 10g Release 2 (10.2) [viitattu 20.12.2013]. Saatavissa: http://docs.oracle.com/cd/B19306_01/server.102/b14223/ettover.htm

Rossberg, J., Olausson, M. 2012. Pro Application Lifecycle Management with Visual Studio 2012.

Running Manual Tests Using Test Runner. [viitattu 3.4.2014]. Saatavissa: http://msdn.microsoft.com/en-us/library/vstudio/dd286725(v=vs.110).aspx