Helander Antti

OCR -PALVELINALUSTAN SUUNNITTELU

Tietojenkäsittelyn koulutusohjelma

2016

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

OCR -PALVELINALUSTAN SUUNNITTELU

Helander, Antti
Satakunnan ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma
Marraskuu 2016
Ohjaaja: Nuutinen, Petri
Sivumäärä: 51
Liitteitä: 6

Asiasanat: tekstintunnistus, sisällönhallinta, järjestelmäarkkitehtuuri, järjestelmäsuunnittelu

---

Opinnäytetyön aiheena oli suunnitella yrityksen yhteistyökumppanin energia-alalla toimivalle asiakkaalle uusi Datacap-palvelinympäristö. Asiakkaalle oli jo olemassa viisi Datacap-palvelinympäristöä kehitys-, testaus-, hyväksyntä- ja tuotantokäyttöön, mutta ne olivat ylimitoitettuja sekä ajoivat vanhoja ohjelmistoversioita. Lisäksi ympäristöjen väliset siirrot ja ylläpito olivat työläitä aikanaan tehtyjen arkkitehtuuripäätösten vuoksi. Uudella ympäristöllä haluttiin korvata nykyinen ympäristö sekä korjata siihen nykyisessä ympäristössä ilmenneitä ongelmia ja puutteita.

Opinnäytetyössä lähdettiin liikkeelle esittelemällä ensin teoriatasolla, mitä on optinen tekstintunnistus, miten sen avulla pystytään digitaalisista dokumenteista hakemaan tekstiä, mitä optisia tekstintunnistusohjelmistoja on olemassa, mikä niistä valittiin opinnäytetyössä käytettäväksi tekstintunnistusohjelmistoksi ja mitä lisäkomponentteja valittu optinen tekstintunnistusohjelmisto vaatii toimiakseen.

Teoriatason esittelyn jälkeen siirryttiin käsittelemään käytännön tasolla tutkittavaa ongelmaa. Ensin esiteltiin nykyisen ympäristön osat, verkkoarkkitehtuuri ja toimintalogiikka. Esittelyvaiheen yhteydessä perehdyttiin nykyiseen ympäristöön tarkemmin ja kerättiin haastattelujen avulla listaa ongelmista, joita nykyisessä ympäristössä oli.

Ongelmien keräämisen jälkeen otettiin selvää, onko yrityksen ja yhteistyökumppanin välillä vaatimuksia, joita tulisi ottaa huomioon uutta ympäristöä suunniteltaessa. Niitä oli ja liittyivät ympäristön saavutettavuuteen ja sen kapasiteettiin.

Selvitystyön jälkeen johdettiin sopimuksen kohdista ja jokaisesta ongelmasta vaatimuslista, joka uuden ympäristön tulisi korjata tai täyttää. Tämän jälkeen alettiin miettiä ja ottaa selvää, miten vaatimus toteutettaisiin uudessa ympäristössä. Joihinkin pystyttiin antamaan selvä ratkaisuidea, miten se toteutettaisiin uudessa ympäristössä, mutta toisiin vaatimuksiin ei pystytty tarjoamaan selkeää ratkaisuideaa. Lopputuloksena saatiin kuitenkin laadittua kattava suunnitelma, miten uutta ympäristöä lähdettäisiin rakentamaan, jotta se korjaisi mahdollisimman paljon nykyisessä ympäristössä ilmenneitä ongelmia ja täyttäisi yrityksen ja yhteistyökumppanin välisen sopimuksen vaatimukset.

PLANNING A NEW OCR SERVER ENVIRONMENT

Helander, Antti
Satakunnan ammattikorkeakoulu, Satakunta University of Applied Sciences
Degree Programme in Information Technology
November 2016
Supervisor: Nuutinen, Petri
Number of pages: 51
Appendices: 6

Keywords: text recognition, content management, systems architecture, system design

---

The purpose of this thesis was to design a new optical character recognition (OCR) server environment for the company's partner company who have a customer that works on the energy field. The customer had already five OCR server environments for developing, testing, acceptance, and production purposes but they were excessively large and deprecated. Moving applications between different environments was also troublesome because the network architecture was so complex. Because of that the current environments needed to be replace with a new one. There were also a lot of problems in the current environment which could be addressed with the new one.

The thesis started with introducing on the theoretical level what is optical character recognition (OCR), how it can be used to search text on digital documents, what types of OCR software exists on the market, which one was selected to be used in this thesis, and what additional components the selected OCR software requires to work correctly.

After theoretical part the thesis continued to introduce problems it should solve. First it introduced the parts of the current environment, its network architecture, and its working logic. After that the thesis introduced problems that exist on the current environment. The problem descriptions were collected by interviewing the developers of the current environments.

The last part of thesis was creating a requirement analysis list. It was based on the problems in the current environments and requirements that was a part of the contract between the company and the partner company. The contract included requirements that was related to the availability of the environment and its capacity.

When the requirement analysis was finished, planning how a requirement could be solved on the new environment was started. In some cases, there was a clear insight how a requirement could be solved on the new environment but some cases there was not. At the end however, a plan was successfully created on how the new environment should be built so it fixes the problems that exist on the current environment and fulfils requirements that was descripted in the contract.

SISÄLLYS

KÄYTETYT TERMIT

AD	Lyhenne sanoista Active Directory. On Microsoft-yrityksen Windows-toimialueen käyttäjätietokanta ja hakemistopalvelu, joka sisältää tietoa käyttäjistä, tietokoneista ja verkon resursseista. Se mahdollistaa keskitetyn resurssien jakamisen käyttäjille ja sovelluksille sekä tarjoaa tavan nimetä, kuvata, paikallistaa, hallita ja suojata käytössä olevia verkon resursseja.
ASCII	Merkistöstandardi, miten teksti kuvataan numeroiksi tietokoneen muistiin.
Batch	Automaattisesti käsiteltävä työjono. Jokaisesta työjonosta luodaan IBM Datacap-ohjelman Batch-kansioon alakansio uniikilla työjonotunnisteella. Sinne tallennetaan käsiteltävästä työjonosta muun muassa lokia työjonon prosessoinnin eri vaiheista ja kopiot työjonon alkuperäisistä dokumenteista.
Binaarijärjestelmä	Binaari eli 2-järjestelmä on yksinkertaisin lukujärjestelmä. Binäärijärjestelmässä on vain kaksi numeroa: 0 ja 1, joista muodostetaan kaikki luvut.
Bugi	Virhe ohjelmakoodissa.
Dashboard	Reaaliaikainen käyttöliittymä, joka näyttää organisaation tai tietokonelaitteiden avaintietojen nykyisen tilan ja historialliset trendit graafisena esityksenä.
Dekoodaus	Palauttaa koodatun tiedon siihen muotoon, jossa se oli ennen koodausta, purkaa koodin.
Domain	Tässä opinnäytetyössä domain tarkoittaa Windows-toimialuetta, joka on joukko Microsoft-yrityksen Windows-käyttöjärjestelmän sisältäviä tietokoneita. Niitä hallitaan keskite-

	tysti yhdeltä tai useammalta Windows-palvelimelta. Toimialueella on yksi tai useampi järjestelmänvalvoja, jolla on täydellinen hallintamahdollisuus toimialueen koneisiin. Jokaisella toimialueen tietokoneita käyttävällä käyttäjällä on puolestaan oma käyttäjätili. Sen avulla heillä on mahdollisuus kirjautua sisään ja käyttää toimialueen tietokoneita ja verkkoresursseja, joihin heille on määritelty käyttäjätilissä käyttöoikeus.
Enkoodaus	Vastakohta dekoodaukselle. Enkoodauksessa tiedoston tyyppi muutetaan formaatista toiseen
Failover-klusteri	Joukko servereitä, joita kutsutaan myös solmuiksi. Ne toimivat yhdessä tarjotakseen korkean saavutettavuuden ohjelmille ja palveluille. Jos yksi servereistä vikaantuu, voi toinen serveri tai solmu klusterissa ottaa vikaantuneen serverin työtaakan itselleen ilman häiriöaikaa.
Fingerprint	Uniikki tunniste, jolla sisään tulevan dokumentin tyyppi tunnistetaan perustuen sen layoutiin ja yhdistetään johonkin tunnettuun dokumenttityyppiin.
FTPS	Laajennos yleisesti käytettävään tiedostonsiirtoprotokollaan (FTP), mikä lisää tuen sekä TLS- että SSL -salausprotokolliin.
HA	Lyhenne sanoista high availability. Se on tietojärjestelmien suunnittelussa käytettävä käytäntö, joka pyrkii siihen, että järjestelmä on aina käyttäjän käytettävissä. High availability -suunnittelun avulla pyritään minimoimaan katkot ja niiden ilmaantuessa pitämään kesto mahdollisimman lyhyenä.
IBM	Lyhenne sanoista International Business Machines Corporation. Yritys, joka on tunnettu suurtietokoneiden ja raskaiden palvelimien valmistajana sekä alkuperäisen IBM PC -arkkitehtuurin kehittäjänä. 2000-luvulla IBM on vähitellen luopunut kokonaan

kuluttajatietokoneiden, lähiverkkolaitteiden ja kiintolevyjen valmistuksesta. Yhtiön liikevaihto muodostuu nykyään IT- ja liiketoimintakonsultoinnista, palvelinliiketoiminnasta sekä ohjelmistoista.

Konfiguroida — Määritellään ennalta määrätyt asetukset.

Network Load Balancing — Microsoft-yrityksen kehittämä ohjelmapohjainen kuormantasaustekniikka. Se on heartbeat-tyyppinen kuormantasaaja, joka jakaa kuorman tasaisesti eri palvelimien välille ja voidaan määritellä kuormittamaan toisia palvelimia enemmän kuin toisia. Network Load Balancing -kuormantasaaja tarjoaa myös korkean saavutettavuuden ominaisuuksia: siihen voidaan liittää monta palvelinta, mutta ulospäin käyttäjälle näytetään ainoastaan yksi yhteysosoite. Liittämisen jälkeen Network Load Balancing kysyy aktiivisesti palvelimilta niiden tilaa. Jos joku palvelin ei vastaa määritetyn ajan kuluessa, palvelin merkitään vialliseksi, eikä sille enää välitetä kyselyitä.

Merkistö — Tietokoneet käyttävät sisäisessä prosessoinnissaan binaarijärjestelmää. Jotta tekstiä voitaisiin käsitellä tietokoneella, on teksti muutettava ensin kymmenjärjestelmän numeroiksi ja ne edelleen binaarijärjestelmän luvuiksi. Jotta muuntaminen onnistuisi, jokaiselle kirjaimelle ja erikoismerkille on annettu numero, jolla sitä kuvataan tietokoneen muistissa.

.Net — Microsoftin kehittämä ohjelmien suoritusympäristö, joka sisältää kaksi pääkomponenttia: CLR ja .NET Framework Class. CLR on suoritusmoottori, jonka avulla ohjelmia pystytään suorittamaan. .NET Framework Class on luokka, joka sisältää valmiiksi luotua ja testattua koodia, jota ohjelmistokehittäjät voivat käyttää ohjelmistokehityksessä.

Ruleset	Joukko funktioita, joilla käsitellään optisesti tunnistettua tekstiä.
Skriptaus	Proseduurien automatisointitekniikka, jossa komentoja ketjutetaan merkkijonoiksi ja tallennetaan tekstitiedostona niin kutsuttuna skriptinä. Joka kerta kun skripti suoritetaan, komennot käsitellään yksi kerrallaan aivan kuin ne olisi kirjoitettu käsin pääteikkunassa.
Skripti	Komentosarja; tulkattava lyhyt tietokoneen ohjelma.
Skriptikieli	Komentosarjakieli eli skriptikieli on kieli, jolla kirjoitetaan komentosarjoja eli skriptejä. Näillä automatisoidaan tehtäviä ilman, että tarvitaan varsinaisia ohjelmointikieliä.
Säie	Perinteistä prosessia kevyempi prosessi. Säie eroaa prosessista siten, että sillä ei ole omia resursseja, vaan se käyttää sen prosessin resursseja, johon se kuuluu. Yhdessä prosessissa on silloin yksi tai useampia säikeitä
Virtualisointi	Tietotekniikassa virtualisointi on menetelmä, jolla fyysisiä resursseja voidaan käyttää ja hallinnoida erilaisina loogisina resursseina. Tämä tarkoittaa sitä, että virtualisoitujen resurssien todelliset ominaisuudet ja arvot eivät näy järjestelmille, sovelluksille tai loppukäyttäjille, jotka käyttävät näitä resursseja. Tällä tavoin yksi fyysinen resurssi, kuten palvelin tai tallennusväline voidaan jakaa toimimaan useampana loogisena resurssina. Virtualisoinnilla on useita käyttökohteita, mutta tässä opinnäytetyössä virtualisoinnilla tarkoitetaan palvelinvirtualisointia. Se on fyysisten laiteresurssien jakamista loogisiksi laiteresursseiksi, jotka voivat toimia omina palveliminaan.
XML	Metakieli, jolla kuvataan tiedon rakenne ilman ennalta määrättyjä koodeja.

1. JOHDANTO

1.1. Taustaa

Opinnäytetyö tehdään yritykselle, joka on sisällönhallintaan erikoistunut ohjelmistoyritys. Se on toiminut vuodesta 1994 ja sen yksi keskeisimmistä osa-alueista on kehittää älykkäitä arkistointiratkaisuja. Ohjelmistokehitykseen yritys käyttää IBM:n tuotteita ja on sen Premier Business Partner.

Yrityksen yhteistyökumppanin asiakas toimii energia-alalla ja toimittaa loppuasiakkailleen muun muassa öljyä, maakaasua ja erilaisia petrokemian tuotteita. Ennen nykyistä järjestelmää kaikki loppuasiakkaiden tekemät tilaukset syötettiin joko manuaalisesti tai käsin asiakkaan ERP-järjestelmään. Tämä oli kuitenkin työlästä ja hidasta. Lisäksi manuaalinen tilaustensyöttö vaati paljon henkilöstöresursseja, oli kallista ja tilausten syötön yhteydessä tuli helposti virheitä.

Uudella järjestelmällä haluttiin nopeuttaa ja automatisoida tilaustenkäsittelyä, saada kustannussäästöjä ja minimoida virheiden määrä tilaustenkäsittelyssä. Se on rakennettu yhteistyössä yrityksen ja sen yhteistyökumppanin kanssa. Yhteistyökumppani on erikoistunut dataintegraatioihin ja sen tehtävänä on muodostaa asiakkaalta tulleesta tilauksesta XML-muotoinen tiedosto ja lähettää yrityksen järjestelmään prosessoitavaksi. Lisäksi yhteistyökumppani huolehtii yritykseltä saamansa OCR-teknologialla tunnistetun tilauksen tuloksen tallentamisen takaisin asiakkaan ERP-järjestelmään.

Tilaustiedot tulevat asiakkaalta useassa erilaisessa dokumenttimuodossa käsiteltäväksi. Dokumentit voivat tulla esimerkiksi PDF-, Excel-, Word-, TIFF-, RTF-, ja HTML-tiedostoina tai sähköpostidokumenttina. Dokumenttien alkuperänä ovat esimerkiksi sähköpostit, faksit ja erilaiset sähköiset lomakkeet. Näistä muodostetaan XML-tiedosto, joka sisältää metatietoa sekä alkuperäisen tilauksen enkoodattuna. Se lähetetään yrityksen järjestelmään, jossa tilaus dekoodataan ja välitetään IBM Data-

cap-sovellukselle. Se tunnistaa OCR-, ICR- ja OMR-teknologioilla tilauksesta tiedot, jotka ovat etukäteen määritelty haettavaksi Datacap-sovelluksen asetuksissa. Haetut tiedot tallennetaan uuteen XML-tiedostoon, joka lähetetään takaisin yhteistyökumppanille. Se jatkojalostaa XML-tiedostoa ja lähettää tilauksen asiakkaalle, joka toimittaa loppuasiakkaalleen tuotteet sen perusteella.

Tällä hetkellä dokumenttien käsittelyyn on jo käytössä ympäristöt, jotka osaavat poimia dokumenteista asiakkaan pyytämät tiedot, mutta niiden kykyä prosessoida dokumentteja pitäisi tehostaa ja automatisoida. Ympäristöt ovat myös alimitoitettuja sekä ajavat vanhoja ohjelmistoversioita. Lisäksi ympäristöjen väliset siirrot ja ylläpito ovat työlästä aikanaan tehtyjen arkkitehtuuripäätösten vuoksi.

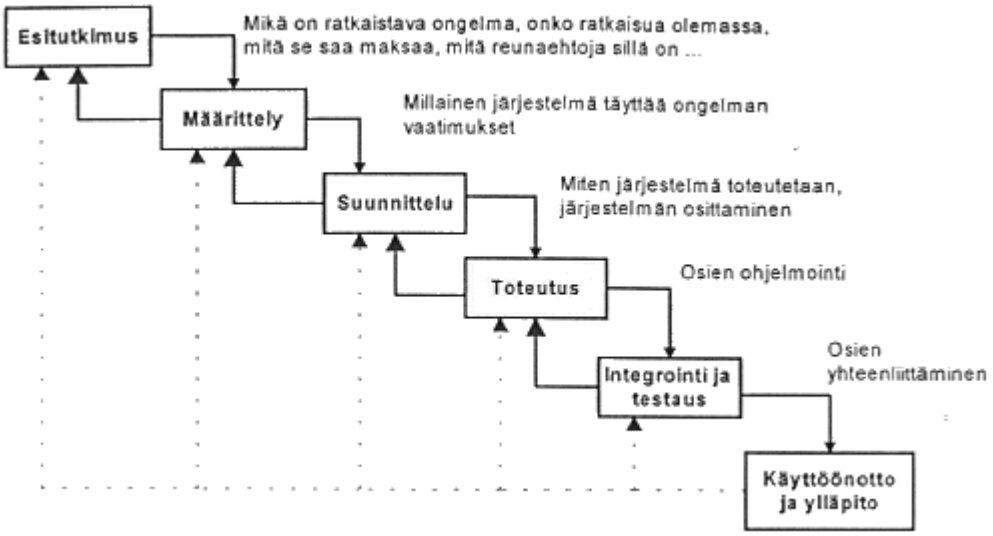
1.2. Opinnäytetyön tavoite

Opinnäytetyön tavoitteena on suunnitella uusi ympäristö, joka korjaa asiakkaan nykyisessä ympäristössä ilmenneitä ongelmia. Siinä käydään lävitse aluksi nykyisen ympäristön ongelmia ja haasteita. Niiden pohjalta laaditaan mahdollisemman kattava vaatimuslista korjauksista ja puutteista, jotka uuden ympäristön tulisi korjata. Sen jälkeen opinnäytetyössä etsitään ja esitellään ratkaisumallit siihen, miten haasteet tai ongelmat voitaisiin korjata uudessa ympäristössä. Opinnäytetyössä esitellään myös, mitä on optinen tekstintunnistus ja miten sen avulla tekstiä voidaan hakea erilaisista dokumenteista ja tallentaa myöhempää jatkokäsittelyä varten.

2. SUUNNITTELUN VAIHEET

Järjestelmän elinkaareksi kutsutaan aikaa, joka kuluu järjestelmän kehittämisen aloittamisesta sen poistamiseen käytöstä. Vaihejakomallilla tarkoitetaan tapaa, jolla ohjelmiston kehitystyö tai koko elinkaari jaetaan vaiheisiin. Tavallisin vaihejakomalli on vesiputousmalli (kuva 1). Siitä on olemassa useita erilaisia muunnelmia, mutta yleensä niistä voidaan erottaa ainakin määrittely-, suunnittelu- ja toteutusvaiheet. Koska opinnäytetyö rajoittuu optisen tekstintunnistusympäristön suunnitteluun, jäte-

tään tässä opinnäytetyössä esittelemättä toteutusvaihe. (Haikala & Märijärvi 1998, 25.)



Kuva 1. Vesiputousmalli (Haikala & Märijärvi, 1998, 25).

Määrittelyvaihetta edeltää usein esitutkimukseksi tai tarvekartoitukseksi kutsuttu vaihe. Sen tehtävänä on asettaa yleiset järjestelmätason vaatimukset, esimerkiksi varastonvalvontasovelluksen vaatimukseksi voidaan asettaa varaston kiertonopeuden kasvattaminen 10 prosentilla. Tällaisia vaatimuksia kutsutaan usein asiakasvaatimuksiksi, koska ne määrittelevät asiakkaan tarpeet, mutta eivät ota kantaa siihen, millainen järjestelmä täyttää asiakkaan vaatimukset. (Haikala & Märijärvi 1998, 25-26.)

Määrittelyvaiheessa asiakasvaatimukset analysoidaan ja niistä johdetaan järjestelmävaatimukset. Se on prosessi, jossa otetaan selvää käyttäjien odotuksista uutta tai muutettavaa järjestelmää kohtaan. Näitä odotuksia kutsutaan vaatimuksiksi ja niiden pitää olla laskettavissa olevia, relevantteja ja yksityiskohtaisia. Toisin sanoen se on käyttäjän järjestelmältä vaatima edellytys tai kyky ratkaista käyttäjän ongelma tai saavuttaa haluttu tavoite. (Haikala & Märijärvi 1998, 64; Mellon 2012; Rouse 2016a.)

Määrittelyn tuloksena syntynyttä dokumenttia sanotaan toiminnalliseksi määrittelyksi. Siinä kuvataan järjestelmän toiminnot, toteutukselle asetettavat ei-toiminnalliset vaatimukset sekä rajoitukset. Toimintojen yhteydessä määritellään järjestelmälle toteutettavat ominaisuudet ja kommunikointi muiden järjestelmien kanssa. Ei toimin-

nallisia vaatimuksia ovat esimerkiksi suoritusteho, vasteaika ja käytettävyys. Rajoituksia ovat esimerkiksi muistitila ja toteutus tietyllä ohjelmointikielellä. (Haikala & Märijärvi 1998, 27.)

Suunnitteluvaiheessa määrittelyvaiheessa kuvattujen toimintojen toteutus suunnitellaan. Suunnitteluvaiheen tarkoituksena on muuntaa asiakkaan tarpeiden mukaan tehty määrittely teknilliselle kielelle, toisin sanoen järjestelmän toteutuksen kuvaukseksi. Suunnittelu jaetaan usein kahteen tasoon: moduulisuunnitteluun ja arkkitehtuurisuunnitteluun. Moduulisuunnittelussa suunnitellaan kunkin moduulin eli järjestelmästä erotettavan loogisen kokonaisuuden rakenne. Tarkoituksena on hienontaa järjestelmä niin pieniin osiin, että osat voidaan antaa yksittäisen suunnittelijoiden tehtäväksi. (Haikala & Märijärvi 1998, 67.)

Arkkitehtuurisuunnittelu on pääasiassa osien välisen työnjaon ja rajapintojen suunnittelua. Tavoitteena on pyrkimys mahdollisimman vähän toisistaan riippuviin moduuleihin niin, etteivät yksittäisen moduulin sisällä tehdyt muutokset säteile moduulin ulkopuolelle. Näin muutosten tekeminen helpottuu ja projektin osat voidaan toteuttaa toisistaan mahdollisimman riippumattomasti. (Haikala & Märijärvi 1998, 28, 67.)

Tässä opinnäytetyössä valittiin käytettäväksi suunnittelumalliksi vesiputousmalli, koska siinä panostetaan erityisen paljon suunnitteluun. Opinnäytetyön tarkoitus puolestaan on suunnitella mahdollisimman kattavasti uusi OCR-palvelinalusta ja sen osien väliset työnjaot. Lisäksi vesiputousmalli on selkeä, ja projektin eri vaiheet seuraavat toisiaan suoraviivaisesti. (Suntuubi 2016.)

Opinnäytetyö lähtee liikkeelle luvusta kolme esittelemällä teoriatasolla, mitä optinen tekstintunnistus on, mitä optisia tekstintunnistusjärjestelmiä on olemassa ja mitä optista tekstintunnistusjärjestelmää opinnäytetyössä käytetään. Sen jälkeen kuvataan lyhyesti, mitä muita komponentteja valittu optinen tekstintunnistusjärjestelmä vaatii toimiakseen. Luvussa viisi on esitelty teoriatasolla, miten nykyinen ympäristö toimii.

Luku kuusi on vesiputousmallin esitutkimus- ja määrittelyvaihetta. Siinä on esitelty aluksi yhteistyökumppanin kanssa laaditut kapasiteetti ja sopimustason vaatimukset,

jotka uuden ympäristön on täytettävä. Tämän jälkeen luvussa siirrytään määrittelyvaiheeseen, jossa on kuvattu nykyisessä ympäristössä ilmenneitä ongelmia ja johdettu vaatimuksia niiden ratkaisemiseksi.

Luku seitsemän on vesiputousmallin suunnitteluvaihe. Siinä luvun kuusi jokaiselle vaatimukselle on ehdotettu ratkaisumalli, miten se toteutetaan uudessa ympäristössä. Luvussa käydään läpi myös arkkitehtuurisuunnittelua, koska siinä on esitelty uuteen ympäristöön asennettavien komponenttien rooleja ja työnjakoa.

3. TEKSTINHAKU DIGITAALISISTA DOKUMENTEISTA

3.1. Optinen tekstintunnistus

Tekstiä voidaan hakea kuvista ja muuntaa muokattavaksi dataksi tietokoneella. Tämä on hyödyllistä esimerkiksi tilanteissa, joissa tulostettu asiakirja, valokuva tai pdf-tiedosto halutaan muuttaa muokattavaksi tekstiksi. Sitä varten on tietokoneelle kehitetty ominaisuus, joka osaa poimia digitaalisista dokumenteista kirjaimia, yhdistää kirjaimet sanoiksi ja sanat lauseiksi. Tätä ominaisuutta kutsutaan optiseksi tekstintunnistukseksi ja siitä käytetään yleisemmin lyhennettä OCR. (Abbyy Inc 2016.)

3.2. OCR- ICR- JA OMR

Tarkkaa tapaa, jolla ihminen tunnistaa objekteja, ei ole vielä täysin ymmärretty. Tiedemiehet ovat kuitenkin havainneet kolme perusperiaatetta, joita ihmiset käyttävät objektien tunnistamiseen. Nämä kolme perusperiaatetta ovat:

1. Kyky erotella objekteja taustasta / ryhmitellä elementit: mikä kaikki kuuluu yhteen objektiin?
2. Kyky tunnistaa: minkälainen kuvio se on?
3. Kyky tunnistaa eroteltu objekti. mikä se on?

Samanlaista periaatetta käytetään optisessa tekstitunnistuksessa eli OCR-teknologiassa. (Abbyy Inc 2016.)

Kun optinen tekstintunnistusohjelma saa dokumentin käsiteltäväksi, se analysoi dokumentin rakenteen ja jakaa sivun erilaisiin elementteihin kuten tekstilohkoihin ja taulukoiksi. Rivit se jakaa sanoiksi, ne merkeiksi ja muuntaa konekielellä ymmärrettävään muotoon kuten ASCII-merkistöksi. Kun sanat ovat eroteltu merkeiksi, ohjelma vertaa niitä settiin mallikuvia. Mallikuvien avulla ohjelma luo lukuisia hypoteeseja, mikä merkki on kyseessä. Ohjelma analysoi näihin hypoteeseihin perustuen erilaisia muunnoksia rivien jakamisesta sanoihin ja sanojen jakamista merkkeihin. Kun ohjelma on prosessoinut lukuisia tämänkaltaisia mahdollisia hypoteeseja, se näyttää niistä parhaiten alkuperäistä tekstiä vastaavan käyttäjälle. (Abbyy Inc 2016; CVISION Technologies Inc 2016.)

Vaikka OCR-teknologia osaa tunnistaa laajan joukon erilaisia fontteja monesta eri kielestä, se ei osaa tunnistaa käsinkirjoitettua tekstiä. Tätä varten OCR-teknologiasta on kehitetty laajennos nimeltä ICR. Se on teknologia, jolla käsinkirjoitettua tekstiä voidaan muuntaa koneellisesti käsiteltävään muotoon. Tekstin tunnistus tapahtuu samalla tavalla kuin OCR-teknologialla, mutta ICR-teknologiaan on sisäänrakennettuna älykkyyttä tunnistamaan epäselviä tai monitulkintaisia osia tekstistä. Tällaisissa tapauksissa ICR pyrkii lähestymään epäselvää tai monitulkintaista tekstiä kuten ihminen käyttäen apunaan teknologiaan sisäänrakennettua sanakirjaa ja kielioppia. (CVISION Technologies Inc 2016; LEAD Technologies Inc 2016.)

Kolmas hyvin yleisesti käytettävä teknologia, jota käytetään yhdessä OCR- ja ICR-teknologian kanssa optiseen tekstin tunnistamiseen, on OMR-teknologia. Se on OCR- ja ICR-teknologiaa vanhempaa teknologiaa ja käytetään keräämään dataa tulostettujen lomakkeiden merkityistä kentistä kuten valintalaatikoista. Jotta OMR-lukija osaisi kerätä datan, sille määritetään valmiiksi sijainnit ja kentät, joista tieto kerätään. Tyypilliset OMR-teknologian käyttökohteet ovat esimerkiksi monivalintakaavakkeet, joista tummennetaan parhaiten sopivan kohdan valintalaatikko. (Beal 2016b; CVISION Technologies Inc 2016.)

3.3. Opinnäytetyössä käytettävä optinen tekstintunnistusohjelmisto

Markkinoilla on saatavilla erilaisia tekstintunnistusohjelmistoja. Näitä ovat muun muassa ABBYY-yrityksen FineReader, Adobe-yrityksen Adobe Acrobat, Canon-yrityksen Readiris, Nyance-yrityksen OmniPage ja IBM-yrityksen Datacap. Tässä opinnäytetyössä päädyttiin käyttämään IBM Datacap -ohjelmistokokonaisuutta optisen tekstintunnistus ohjelmistoympäristön toteutukseen, koska yritys on IBM-osakeyhtiön Premier-partneri ja saa käyttöönsä kaikki IBM:n kehittämät tuotteet. Toisena syynä ovat tekniset syyt. IBM Datacap sisältää useita OCR-moottoreita tekstintunnistamiseen ja mahdollistaa paremman tuloksen tekstintunnista-misessa. Lisäksi IBM Datacap sisältää tuen yrityksen käyttämälle ecm-alustalle ja kehittynyttä analytiikkaa tekstin tunnistamiseen ja validoimiseen. (Adobe Systems Inc 2016; Abbyy Inc 2016; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 67; I.R.I.S. S.A. 2016; Nuance Communications Inc 2016; TopTenReviews 2016.)

3.4. IBM Datacap

Opinnäytetyöhön valittu optista tekstintunnistus -teknologiaa käyttävä ohjelmistokokonaisuus on nimeltään IBM Datacap. Se käyttää datan tunnistamiseen apuina OCR-, ICR- ja OMR-tekstintunnistusteknologioita sekä lukuisia skripti- ja koodipohjaisia .NET-toimintoja. Datacap osaa tunnistaa käsinkirjoitettua tekstiä, tietokoneella tuotettua tekstiä, viivakoodeja ja poimia valintalaatikoiden arvot. (IBM Knowledge Center 2016; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 17.)

IBM Datacap ei itsessään ole sovellus, jota käytetään tunnistamaan tekstiä dokumenteista. Se on ohjelmistokokonaisuus, jota käytetään kehitettäessä sovelluksia tiettyjä yritystarpeita varten. Tällaisia tarpeita voivat olla esimerkiksi erilaisten tietojen hakeminen laskuista. Näitä tarpeita varten IBM Datacap tarjoaa joukon erilaisia funktioita muun muassa tekstintunnistamiseen dokumentista, sen validoimiseen, vahvistamiseen, luokitteluun ja viemiseen halutussa tiedostomuodossa haluttuun paikkaan, esimerkiksi tietokantaan. Näiden lukuisten funktioiden ansiosta pystytään jokaista

yritystarvetta varten luomaan oma räätälöity IBM Datacap-sovellus. Jos sovelluskehitykseen ei ole kuitenkaan saatavilla tarvittavaa funktiota, pystytään IBM Datacap-sovellukseen tekemään lisää funktioita. Tässä opinnäytetyössä termeillä IBM Datacap-sovellus tai Datacap-sovellus viitataan IBM Datacap-ohjelmistolla kehitettyyn sovellukseen ja termeillä IBM Datacap tai Datacap ohjelmistoon, jolla sovellukset on kehitetty. (IBM Knowledge Center 2016.)

3.4.1. IBM Datacap -komponentit

IBM Datacap koostuu komponenteista Datacap Server, Datacap client, Datacap Studio, Rulerunner Service, Maintenance Manager, Datacap Navigator, Datacap Web Client, Report Viewer, Application Manager ja Datacap-sovelluksen tietokannat (kuva 1). Datacap Server on keskeisin ohjelmiston osa: se huolehtii batcheista ja välittää ne työasemille ja käyttäjille, ylläpitää Datacap-sovelluksen työnkulkuun liittyviä tehtäviä, huolehtii käyttäjien tunnistamisesta ja pääsystä ohjelmiston eri osiin, yksilöi batchit, valvoo batch-jonoja ja kontrolloi pääsyä Datacap-sovelluksen tietokantoihin. Datacap Server -komponentista käytetään toisinaan myös nimeä Datacap Taskmaster ja myös opinnäytetyössä siihen viitataan joko nimellä Datacap Server tai Datacap Taskmaster. (IBM Knowledge Center 2016.)

Datacap Client -komponentin muodostavat Datacap Desktop ja FastDoc-ohjelmat, joilla käyttäjä voi ottaa yhteyttä Datacapilla tehtyihin sovelluksiin. Ohjelmat tarjoavat käyttäjälle käyttöliittymän, jonka avulla he pystyvät esimerkiksi skannaamaan dokumentteja ja validoimaan dokumenteista saadut tiedot. (IBM Knowledge Center 2016.)

Datacap Studio on ympäristö, jota käytetään Datacapilla tehtävien ohjelmien kehittämiseen ja testaamiseen. Sitä käytetään muun muassa Datacap-sovellusten konfiguroimiseen, määritellään asiakirjojen hierarkiatasot, luodaan säännöt dokumenttien tekstialueiden- ja kenttien tunnistamiseen, kehitetään rulesettejä ja testataan tehtäviä tai jo tehtyjä Datacap-sovelluksia. (IBM Knowledge Center 2016; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 9, 17.)

Rulerunner Service on Windows-palvelu ja samalla Datacapin komponentti. Se voidaan ohjelmoida suorittamaan automaattisesti batchiin liittyviä tehtäviä, jotka eivät vaadi käyttäjän huomiota. Ne voivat olla esimerkiksi dokumentin terävöittäminen, konvertointi, tunnistaminen ja siitä saadun datan tallentaminen ennalta määrättyyn sijaintiin ennalta määrätyssä muodossa. (IBM Knowledge Center 2016; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 93.)

Maintenance Manager -komponenttia käytetään Datacapilla kehitettyjen sovellusten valvomiseen. Ohjelma voidaan konfiguroida suorittamaan muun muassa seuraavia tehtäviä: monitoroimaan batchien tilaa, poistamaan tai siirtämään batchit toiseen sijaintiin ja lähettämään sähköpostia virhetilanteissa. Kaikki Maintenance Manager -tehtävät voidaan suorittaa manuaalisesti Datacap Maintenance Manager -komponentilla, asettaa suoritettavaksi osaksi IBM Datacap-sovelluksen ajonaikaista suoritusta tai asettaa suoritettavaksi automaattisesti Windows-käyttöjärjestelmän tehtävien ajoitusohjelmalla. (IBM Knowledge Center 2016; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 70.)

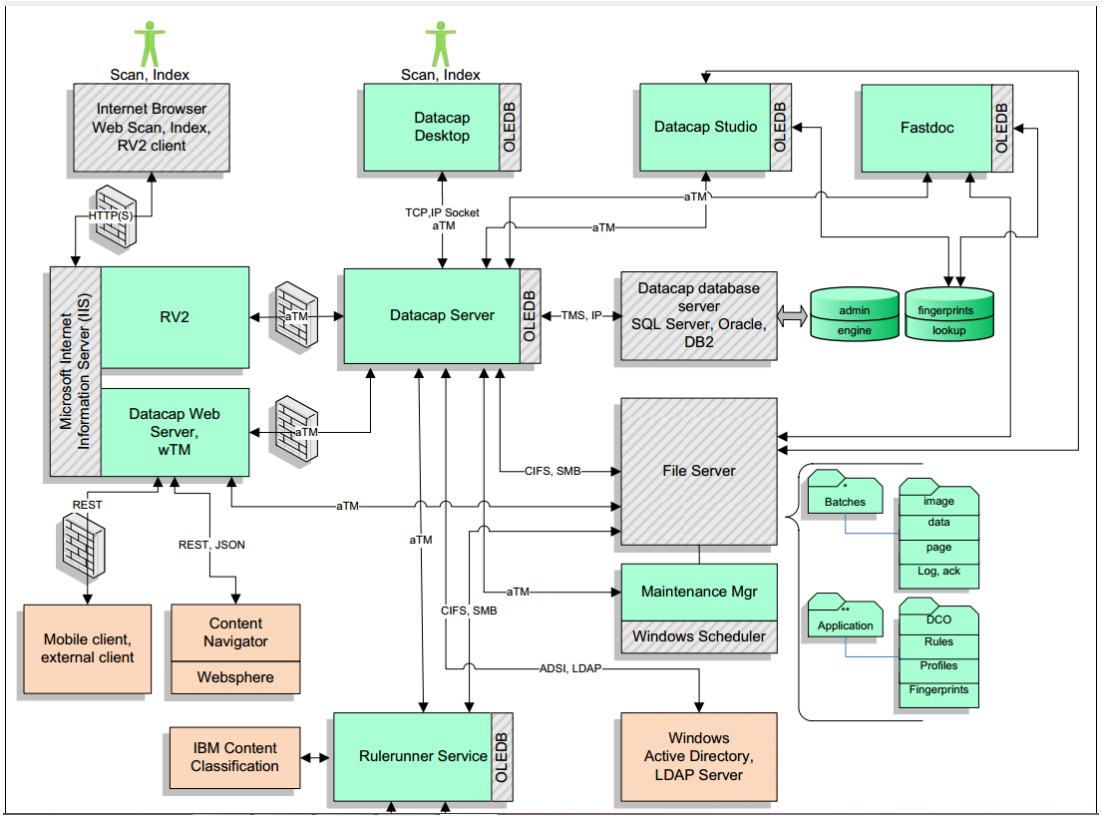
Datacap Navigator ja Datacap Web Client -komponentit ovat webpohjaisia komponentteja, joiden avulla netin kautta voidaan suorittaa samankaltaisia tehtäviä kuin Datacap Desktop- ja FastDoc-komponenteilla ilman, että käytettävälle työasemalle tarvitsee asentaa muita Datacap-komponentteja. Datacap Web Service on puolestaan komponentti, jonka avulla etäyhteyttä käyttävät ohjelmat kuten Datacap Navigator ja Datacap Web Client pystyvät kommunikoimaan verkon yli IBM Datacap-ohjelmiston kanssa. (IBM Knowledge Center 2016; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 73.)

Report Viewer -komponentti on web-pohjainen raportointiohjelma. Se tuottaa reaaliajan raportteja Datacap-sovelluksen toiminnasta, kuten batchin tilasta ja ongelmabatcheista. Tiedot raporttiin Repor Viewer saa Datacap-sovelluksen Engine-tietokannasta. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 73; IBM Knowledge Center 2016.)

Application Manager -komponentti ylläpitää rekisteriä, mihin ympäristöihin Datacapin osat ovat asennettu. Tällä tavalla eri palvelimilla sijaitsevat Datacapin osat tie-

tävät, millä palvelimella mikin Datacapin osa sijaitsee. Application Manager -komponentin kautta Datacapin osat pääsevät myös käsiksi tietoihin, joita ne tarvitsevat toimiakseen. Tällaisia tietoja ovat Datacap Server -komponentin sijainti, Datacap-sovelluksen tietokantojen sijainti, fingerprint-kirjaston sijainti ja tiedostopalvelimen sijainti. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 72.)

Edellä mainittujen komponenttien lisäksi Datacap sisältää tietokannat Administration, Engine ja Fingerprint. Administration-tietokantaan tallentuu tieto käyttäjistä, ryhmistä, työasemista, seurannasta, toiminnallisesta turvallisuudesta, ohjelman konfiguroinnista ja työnkulun konfiguroinnista. Engine-tietokantaan tallentuu tieto batcheista, niiden suoritustiloista ja erilaisia tilastotietoja. Fingerprint-tietokanta hallinnoi Datacap-sovelluksen fingerprinttien sijaintietoja. (IBM Knowledge Center 2016.)



Kuva 1. Datacap-komponentit ja niiden väliset suhteet. Turkoosilla värjätyt komponentit ovat Datacap-komponentteja. Harmaaraidalliset ovat ulkoisia komponentteja kuten tietokantoja ja tiedostojärjestelmiä. Osa niistä on sellaisia, joita ilman Datacap ei toimi. Pinkillä värjätyt komponentit ovat valinnaisia ja tarjoavat lisäominaisuuksia kuten käyttäjän tunnistamista. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 68-69.)

4. MUUT VAADITTAVAT JA VALINNAISET KOMPONENTIT

Koska Datacap-sovellukset eivät toimi ilman tiettyjä lisäkomponentteja (kuva 1), on vaadittavat lisäkomponentit esitelty tässä luvussa. Lisäksi uuden ympäristön suunnittelussa käytetään joitain IBM Datacap -ohjelmiston tukemia valinnaisia komponentteja (kuva 1). Jokaisen komponentin kohdalla on mainittu, onko se vaadittu vai valinnainen.

4.1. VMware vSphere

VMware vSphere on yksi VMware-yhtiön tarjoama virtualisointituote, jota käytetään muun muassa palvelinvirtualisointiin. Ennen vuotta 2009 VMware vSphere tunnettiin myös nimellä WMware Infrastructure. (Moonsoft 2016, Rouse 2016b.)

IBM Datacap ei vaadi toimiakseen virtualisointia. Koska yrityksessä kaikkien asiakkaiden palvelimet ovat kuitenkin virtualisoituja, on tässäkin opinnäytetyössä otettu virtualisointi huomioon. Se, miksi opinnäytetyössä päädyttiin käyttämään virtualisointialustana WMware vSphere-virtualisointituotetta, johtuu kahdesta syystä. Ensimmäinen syy on, ettei yrityksen aloittaessa ollut tarjolla muita sen tarpeita vastaavia virtualisointituotteita. Toinen syy on, ettei yritys aio tällä hetkellä vaihtaa sitä mihinkään toiseen virtuasliointialustaan. (Sysilahti henkilökohtainen tiedonanto 17.7.2016)

4.2. Microsoft Server 2012 R2

IBM Datacap-ohjelmiston komponentit vaativat alustakseen Microsoft Windows käyttöjärjestelmän (taulukko 1). Yksi IBM Datacap -ohjelmiston tukema käyttöjärjestelmä on Microsoft Server 2012 R2 Datacenter, joka on yksi versio Microsoftin kuudennesta julkaisemasta palvelinkäyttöjärjestelmästä. Muut kolme versiota ovat: Foundation, Essentials ja Standard. (IBM 2016; Minasi ym. 2014, 1-2).

Opinnäytetyössä päädyttiin käyttämään ympäristön palvelimien käyttöjärjestelmänä Microsoft Server 2012 R2 Datacenter -versiota, koska tarvittavien lisenssien luku-

määrä määräytyy virtualisointialustan fyysisten palvelinten prosessorien lukumäärän eikä ajettavien virtuaalipalvelimien lukumäärän mukaan. Yritys puolestaan hyödyntää virtualisointia paljon, ja näin ollen Microsoft Server 2012 R2 -palvelinkäyttöjärjestelmän Datacenter-versio on yritykselle kustannustehokkain vaihtoehto. (Microsoft 2016, Sysilahti henkilökohtainen tiedonanto 17.7.2016)

Taulukko 1. IBM Datacap ohjelmiston tukema käyttöjärjestelmä, versio ja bittisyys. (IBM 2016).

Microsoft käyttöjärjestelmä ja versio	**Bittisyys**
Windows 8.1 Enterprise	64-bittinen
Windows 10 Enterprise	64-bittinen
Windows 7 Professional	32-bittinen
Windows 7 Professional	64-bittinen
Windows Server 2008 R2 Datacenter Edition	64-bittinen
Windows Server 2008 R2 Enterprise Edition	64-bittinen
Windows Server 2008 R2 Standard Edition	64-bittinen
Windows Server 2012 Datacenter Edition	64-bittinen
Windows Server 2012 R2 Datacenter Edition	64-bittinen
Windows Server 2012 Essentials Edition	64-bittinen
Windows Server 2012 R2 Essentials Edition	64-bittinen
Windows Server 2012 Standard Edition	64-bittinen
Windows Server 2012 R2 Standard Edition	64-bittinen

4.3. Tiedostonjako

IBM Datacap vaatii toimiakseen tiedostonjaon, jonka pitää olla kaikkien batcheja käsittelevien Datacap-sovellusten saavutettavissa. Ne käyttävät sitä kuvatiedostojen, puretun datan ja kontrollitiedostojen tallentamiseen. Lisäksi tiedostojakoon tallennetaan tiedostot, joita tarvitaan eri Datacap-komponenttien suorittamisessa. Tällaisia tiedostoja ovat esimerkiksi fingerprint-tiedostot ja dokumentin hierarkiatason määrittelytiedosto. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 74.)

Tässä opinnäytetyössä tiedostonjako päätettiin toteuttaa Windows Server 2012 R2 -alustalle. Syy, miksi tiedostonjako päätettiin toteuttaa Windows-alustalle, on, että IBM Datacap vaatii alustakseen Windows-käyttöjärjestelmän. Lisäksi ympäristöön tulee AD. Windows-tiedostonjako puolestaan toimii paremmin Windows-verkossa etenkin, jos verkossa käytetään AD:a kuin esimerkiksi Linux-käyttöjärjestelmälle toteutettava tiedostonjako (wkiess01 2013).

Toinen syy, miksi opinnäytetyössä päädyttiin käyttämään tiedostonjakopalvelimen alustana Microsoft Server 2012 R2-palvelinkäyttöjärjestelmää, on sen tuoma parannus tiedostonjakoon. Se oli myös osasyy, miksi ympäristön palvelimien käyttöjärjestelmäksi valittiin Microsoft Server 2012 R2. Parannusta kutsutaan nimellä Continuously Available File Shares (CAFS). CAFS-teknologia hyödyntää Microsoft Server 2012 R2-palvelinkäyttöjärestelmän uutta SMB3.0-teknologiaa parantamaan tiedostonjakoa, jota käytetään dokumenttien tallentamiseen ja ohjelmien tukemiseen. (Otey 2013.)

CAFS-teknologia korjaa Windows-tiedostonjaossa esiintyviä ongelmia. Microsoft on tarjonnut jo pitkään korkeasti saavutettavaa tiedostonjakoa, mutta se on kärsinyt keskeytyksistä mahdollisissa virhetilanteissa ja soveltunut ainoastaan Microsoft Office -ohjelmiston kaltaisille ohjelmille, jotka sulkevat ja avaavat tiedostoja säännöllisesti, koska tämän kaltaiset ohjelmat pystyvät virheestä palautumisen jälkeen ottamaan uudelleen yhteyden tiedostonjakoon ja tallentamaan muutokset. (Otey 2013.)

Tämän tyyppinen ratkaisu ei kuitenkaan sovellu esimerkiksi tietokantasovelluksille, jotka vaativat tiedostojen olevan auki pitkiäkin aikoja ja kärsivät datan menetyksestä virhetilanteissa. CAFS-teknologia pyrkii ratkaisemaan tämän ongelman tarjoamalla ohjelmatukea sekä jatkuvasti saavutettavaa pääsyä tiedostonjakoon melkein nollan häiriöajalla. (Otey 2013.)

4.4. IIS-webpalvelin

IBM Datacap-ohjelmisto vaatii toimiakseen IIS-webpalvelimen, koska Datacap Web-, Datacap Web Services- ja Report Viewer -komponentit asennetaan siihen. IIS on lyhenne sanoista Internet Information Server, joka on yksi suosituimmista webpalvelimista. Sen on kehittänyt Microsoft, ja sitä käytetään ASP.NET- ja ASP-tekniikoilla toteutettujen internetpohjaisten websovellusten kehitykseen, isännöimiseen ja tarjoamiseen käyttäjille. Websovellusten kehittämiseen ja ylläpitämiseen IIS-webpalvelin tarjoaa joukon työkaluja, joilla voidaan myös kehittää ja ylläpitää hakumoottoreita. Lisäksi IIS-webpalvelimesta löytyy tuki websovelluksille, jotka ottavat yhteyttä tietokantoihin. Tässä opinnäytetyössä käytettävä IIS-webpalvelimen versio on 8.5, joka on tällä hetkellä toiseksi uusin versio IIS web-palvelimesta ja sisältyy Microsoft 2012 R2 -palvelinkäyttöjärjestelmään. (benjaminperkins 2013; Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 74; IIS 2016; Net-informations.com 2016.)

4.5. Tietokantapalvelin

Koska IBM Datacap-ohjelmistolla tehdyt sovellukset käyttävät hyödykseen tietokantoja, pitää IBM Datacap-ympäristössä olla asennettuna tietokantapalvelin. IBM Dataccap-ohjelmiston asennuksen yhteydessä asennetaan Microsoft Acces -tietokanta, mutta sitä ei tueta tuotantokäytännössä. Tämän vuoksi IBM Datacap -ympäristöön on asennettava tietokantapalvelin. Tuettuja tietokantahallintajärjestelmiä ovat DB2, Microsoft SQL Server ja Oracle Database. (IBM Knowledge Center 2016.)

Opinnäytetyössä päädyttiin käyttämään DB2-tietokannanhallintajärjestelmää, koska IBM Datacap sisältää lisenssin siihen. DB2 on relaatiotietokantojen hallintajärjestel-

mä, joka on suunniteltu tallentamaan, analysoimaan ja hakemaan dataa tehokkaasti. DB2-tietokannanhallintajärjestelmää on myös laajennettu tukemaan oliopohjaisia ominaisuuksia sekä ei relaationaalista XML-rakennetta. (Tutorialspoint 2016.)

DB2-tietokantahallintajärjerjestelmästä on olemassa useita erilaisia versioita, joista yritykset voivat valita tarpeitansa vastaavan version. Nämä versiot ovat:

- Advanced Enterprise Server Edition and Enterprise Server Edition (AESE / ESE)
- Workgroup Server Edition (WSE)
- Express -C
- Express Edition
- Enterprise Developer Edition.

Tässä opinnäytetyössä käytetään DB2 Enterprise -versiota, koska IBM Datacap vaatii toimiakseen sen. (IBM 2016; Tutorialspoint 2016.)

4.6. Käyttäjätodennus

Datacap-sovellukseen pitää määritellä käyttäjän todennustapa. IBM Datacap voidaan määritellä käyttämään sen omaa sisäistä käyttäjän todennusta tai ulkoista käyttäjän todennusta. Se tukee seuraavia käyttäjän todennustapoja:

- Datacap authentication, jota kutsutaan nimellä TMA.
- Windows Active Directory, jota kutsutaan nimellä ADSI.
- Windows Active Directory Lightweight Directory Services, jota kutsutaan nimellä ADLDS.
- Lightweight Directory Access Protocol, jota kutsutaan nimellä LDAP.
- Low-Level Lightweight Directory Access Protocol, jota kutsutaan nimellä LLLDAP.

(IBM Knowledge Center 2016.)

TMA-käyttäjätodennus on Datacap-ohjelmiston oma sisäinen käyttäjän todennustapa. Se tukee sekä käyttäjä, että ryhmätason käyttäjätodennusta. Kun käytettäväksi todennustavaksi on valittu TMA-käyttäjätodennus, käytetään todennukseen käyttäjä-

ryhmiä ja käyttäjiä, jotka ovat määritelty Datacap-sovelluksiin. (IBM Knowledge Center 2016.)

Käytettäessä käyttäjän todennukseen ADSI- tai LDAP-käyttäjätodennusta tapahtuu käyttäjän todennus domainin ryhmätasolla. Salasanaa ei vaadita käyttäjän todentamiseen, ellei sitä ole erikseen määritelty. Molemmat todennusmenetelmät tunnistavat käyttäjän Windows-käyttäjätilin tietojen perusteella. Tämä tarkoittaa sitä, että käyttäjän on kirjauduttava Datacap-palvelimelle kaksi kertaa samoilla tunnuksilla: ensin Datacap-palvelimelle ja tämän jälkeen palvelimella olevaan Datacap-komponenttiin. ADSI-käyttäjätodennus vaatii toimiakseen, että samassa domainissa, missä Datacap-koneet sijaitsevat, on asennettuna AD-palvelin. LDAP-käyttäjätodennus vaatii toimiakseen, että se kuuluu samaan domainiin kuin Datacap-koneet. Molemmat käyttäjän todennustavat tukevat käyttäjän todennusta monesta eri domainista. (IBM Knowledge Center 2016.)

Käytettäessä ADLDS- tai LLLDAP-käyttäjätodennusta käyttäjien todennus tapahtuu Datacap-komponentin kirjautumisikkunaan syötetyn käyttäjätunnuksen perusteella. Käyttäjän todennus voi tapahtua myös taustapalvelun tai -prosessin välityksellä, jolloin käyttäjätiedot kulkevat niiden kautta Datacap-sovellukselle. Toimiakseen ADLDS- tai LLLDAP-käyttäjätodennus vaatii, että jokaiselle käyttäjälle sekä Datacap prosessille- ja palvelulle on luotava oma käyttäjätunnus sekä AD:iin että jokaiseen ADLDS- tai LLLDAP-käyttäjätodennusta käyttävään Datacap sovellukseen. Molemmat käyttäjän todennustavat tukevat käyttäjän todennusta palvelimelta, joka ei ole samassa verkossa kuin Datacap-koneet. (IBM Knowledge Center 2016.)

Tässä opinnäytetyössä päädyttiin käyttämään käyttäjän todennustapana LLLDAP-käyttäjätodennusta. Syy on, että LLLDAP-käyttäjätodennus on todennustavoista joustavin, koska se tukee käyttäjien todennusta myös muista lähteistä kuin AD:sta. Lisäksi se on toiminut yrityksen muissa ympäristöissä käyttäjätodennustavoista parhaiten. (IBM Knowledge Center 2016; Olli henkilökohtainen tiedonanto 7.3.2016; Sysilahti henkilökohtainen tiedonanto 1.3.2016.)

5. NYKYINEN YMPÄRISTÖ

5.1. Ympäristön osat

5.1.1. IBM Datacap

Ympäristöt koostuvat seuraavista IBM Datacap komponenteista: Datacap Taskmaster ja Datacap Rulerunner (katso liite 1). Jotta sanomien käsittely olisi mahdollisimman tehokasta, ovat molemmat komponentit asennettu omalle palvelimelleen. Tällä tavoin palvelimet suorittavat vain yhtä tehtävää, ja Datacap-komponentit saavat kaiken tehon irti palvelimen resursseista eivätkä komponentit häiritse toistensa toimintaa. Lisäksi komponenttien hajauttaminen eri palvelimille mahdollistaa ympäristöjen helpomman laajennettavuuden.

Datacap Taskmaster -palvelimelle on asennettu Datacap Server Manager -komponentti, jota käytetään Datacap server -komponentin konfiguroimiseen ja hallinnoimiseen. Lisäksi Taskmaster-palvelimella sijaitsee Datacap.xml-tiedosto, jossa on tieto ajettavien Datacap-sovellusten sijainneista.

Datacap Rulerunner -palvelimelle on asennettu Datacap Rulerunner -komponentti. Se on asetettu ajamaan automaattisesti loppuasiakkaiden Datacap-sovelluksien työvaiheita, jotka ovat määritelty Datacap-sovelluksen asetuksissa. Se lukee sisään asiakkaan lähettämät loppuasiakkailta peräisin olevat tilausliitteet ja konvertoi ne tif-kuvatiedostoksi tai lukee suoraan Datacap-sovelluksen käyttöön, jos kyseessä on tekstidokumentti, esimerkiksi Microsoft Office Word -asiakirja. Konvertoinnin jälkeen Datacap Rulerunner -komponentti tunnistaa dokumentin rakenteen ja hakee valmiiksi määriteltyihin kenttiin tiedot dokumentista. Lopuksi Datacap Rulerunner -komponentti tarkistaa haetut tiedot ja vie ne XML-tiedostoon.

Jokaisessa ympäristössä on lisäksi oma tietokantapalvelin. Tietokantapalvelimella sijaitsevat Datacap-komponenttien tarvitsemat Engine- Fingerprint- ja Administration-tietokannat. Palvelimella käytettävä tietokantaohjelmisto on IBM DB2 Enterprise.

5.1.2. Inbound Handler

Inbound Handler on yrityksen kehittämä Java-ohjelmointikielellä toteutettu sovellus XML-sanomien purkamiseen (katso liite 2). Jotta sanomien purkaminen olisi mahdollista, on XML-sanomien rakenne ennalta sovittu asiakkaan kanssa. Ohjelman tehtäväkokonaisuudet ovat sanoman vastaanotto, sanoman tarkistus, tilausliitteen ja sen prosessointiin tarvittavan metatiedon poimiminen sanomasta, tilausliitteen lähettäminen Datacap-sovelluksen prosessoitavaksi ja siihen liittyvän metatiedon välittäminen Datacap-sovelluksen Outbound-kansioon odottamaan jatkokäsittelyä. (Korpela henkilökohtainen tiedonanto 11.2.2016.)

5.1.3. XML Processor

XML Processor on yrityksen kehittämä Java-ohjelmointikielellä toteutettu sovellus, jolla Datacap-sovelluksesta tullut XML-muotoinen tilausdata yhdistetään alkuperäisestä XML-tiedostosta saadun metadatan kanssa (katso liite 2). Ohjelman tehtävänä on muokata ja tarkistaa loppuasiakkaan Datacap-sovelluksen Outbound-kansioon tullut XML-muotoinen tiedosto, joka sisältää Datacap-sovelluksen tilausliitteestä poimimat tiedot. Tämän jälkeen se yhdistää loppuasiakkaan Datacap-sovelluksen Outbound-kansiossa olevat XML-muotoiset meta- ja tilaustiedot yhdeksi XML-muotoiseksi tiedostoksi ja lähettää sen Datacap-sovelluksen Outbound-kansiosta Outbound Handlerin Outbound-kansioon. (Korpela henkilökohtainen tiedonanto 1.2.2016.)

5.1.4. Outbound Handler

Outbound Handler on yrityksen kehittämä Java-ohjelmointikielellä toteutettu sovellus (katso liite 2). Sen tehtävänä on hakea loppuasiakaskohtaisesta Datacap-sovelluksesta tullut XML-muotoinen sanoma Outbound-kansiosta ja lähettää se takaisin asiakkaan web-palveluun ja arkistoida tämän jälkeen lähetetty sanoma. Jos Outbound Handler -sovellus ei saa yhteyttä web-palveluun, se siirtää sanoman Failed-kansioon. (Korpela henkilökohtainen tiedonanto 1.2.2016.)

5.2. Verkkoarkkitehtuuri

Nykyinen verkko koostuu seuraavista verkon komponenteista: FTPS-palvelimesta, yrityksen kehitysympäristöstä, yhteistyökumppanin testiympäristöstä, asiakkaan testiympäristöstä ja asiakkaan kahdesta tuotantoympäristöstä (kato liite 1). Tuotantoympäristöjä on verkossa kaksi, koska tuotantoympäristön pitää olla koko ajan hyvin saavutettavissa ja varmuus tähän on ratkaistu rakentamalla kaksi tuotantoympäristöä. (Sysilahti henkilökohtainen tiedonanto 12.2.2016.)

Jokainen ympäristö koostuu kolmesta palvelimesta: AD- ja tietokantapalvelimesta, Datacap taskmaster -palvelimesta ja Datacap Rule Runner -palvelimesta. Poikkeuksena ovat asiakkaan tuotantoympäristöt, joissa on kaksi Datacap Rule Runner -palvelinta. Tämä johtuu siitä, että tuotantoympäristöjen hyvän saavutettavuuden lisäksi tuotantoympäristöjen pitää olla toimintavarmoja ongelmatilanteissa, ja se on ratkaistu nykyisessä verkkoratkaisussa asentamalla kaksi Rulerunner-palvelinta varmistamaan tuotannon jatkuvuus niissä tilanteissa. (Sysilahti henkilökohtainen tiedonanto 12.2.2016.)

Asiakkaan ja yhteistyökumppanin ympäristöt sijaitsevat virtuaaliverkossa. Jokaista ympäristöä kohden on FTPS-palvelimessa oma verkkokortti ja kotihakemisto, jotta yhteistyökumppanin lähettämät sanomat menisivät oikeisiin ympäristöihin. Yhteen verkkokorttiin on määritelty yhden ympäristön IP-osoite. Sen avulla pystytään FTPS-palvelimen kautta ottamaan yhteys oikeaan ympäristöön ja sanomat pystyvät liikkumaan FTPS-palvelimen ja ympäristöjen välillä. Jokainen ympäristö puolestaan muodostaa oman suljetun verkon, jossa jokaisella on oma AD. (Sysilahti henkilökohtainen tiedonanto 12.2.2016.)

Yrityksen kehitysympäristö sijaitsee muista ympäristöistä poiketen yrityksen intranetissä. Tämä johtuu siitä, että virtuaaliverkoissa oleviin ympäristöihin ei pääse suoraan kiinni työasemilta. Kehitysympäristöön on kuitenkin saatava suora yhteys työasemilta, koska sitä käytetään sovelluskehityksessä. (Sysilahti henkilökohtainen tiedonanto 26.2.2016b.)

5.3. Toimintalogiikka

Nykyisen ympäristön toimintalogiikan kuvaus perustuu nykyisen ympäristön suunnittelijan antamiin kommentteihin. Tästä kuvauksesta löytyy myös havainnollistava ja tarkentava kuva liitteestä kaksi, jonka kronologista järjestystä tässä luvussa oleva kuvaus noudattaa.

5.3.1. Sanoman vastaanotto

Sanoman vastaanottaminen etenee seuraavasti:

- Asiakas lähettää tilauksen XML-tiedostona FTPS:n yli yritykselle. Tiedosto sisältää XML-muodossa alkuperäisen sähköpostin ja liitteet, joiden lisäksi se sisältää prosessiin liittyviä metatietoja.
- Yrityksen kehittämä Inbound Handler -sovellus siirtää tilauksen FTPS-kansiosta Processing-kansion sisälle väliaikaiseen tilauksenkäsittelykansioon ja alkaa käsitellä sitä. XML-tiedostosta poimitaan yksi tilausliite tai sähköpostin sisältö ja prosessiin liittyvät metatiedot. Samalla luodaan Outbound-niminen xml-tiedosto muodossa Sanomaid_outbound.xml, joka sisältää prosessin ja sähköpostin metatiedot.
- Sanomaid_outbound.xml-tiedosto siirretään Datacapin Outbound-kansion alle sanomaid:n mukaiseen alikansioon ja tilausliite tai sähköpostin sisältö siirretään Datacapin Inbound-kansioon.

Jos saapunut XML-tiedosto on virheellinen, Sanomaid_outbound.xml tiedostoon luodaan virheviesti ja siirretään Outbound-kansioon. Sanoma voi olla virheellinen, jos:

- OCR-partneria ei tunnisteta (Unknown OCR Partner). Tunnistus tehdään tutkimalla, onko OCR-partneria vastaavaa kansiota olemassa Datacapin Inbound-kansiossa.
- Tilauksen XML-tiedostossa tapahtui validointivirhe (Invalid XML).
- Jos tilauksessa on enemmän kuin 1 validi tilausliite (Too many documents).

Samalla alkuperäinen XML-tiedosto nimetään uudelleen lisäämällä sen loppuun _inbound.xml-pääte, siirretään archive-kansioon, ja poistetaan Processing-kansiosta väliaikainen tilauksenkäsittelykansio. (Korpela henkilökohtainen tiedonanto 1.2.2016.)

5.3.2. Sanoman käsittely

Datacapin Inbound-kansiosta Datacap ottaa liitteen käsittelyyn ja luo siitä Datacapin Outbound-kansioon alkuperäisen sanomaidn:n mukaiseen alikansioon seuraavat tiedostot:

- sanomaid:tä vastaavan xml-tiedoston, joka sisältää Datacapin OCR:n liitteestä tunnistamat tiedot
- tif-tiedostot tilauksen sivuista
- käsiteltävän batchin mukaan nimetyn .complete-tiedoston, joka kertoo, milloin Datacap on valmis.

(Korpela henkilökohtainen tiedonanto 1.2.2016.)

Xml-prosessrori tutkii aktiivisesti Datacapin Outbound-kansiota. Kun sinne luodaan kansio, joka sisältää seuraavat tiedostot: käsiteltävää batchia vastaava .complete-tiedosto, sanomaid:tä vastaava xml-tiedosto ja sanomaid:tä vastaava _outbound.xml-tiedosto XML-prosessori siirtää kansion väliaikaiseen Processing-kansioon.

Processing-kansiossa xml-prosessori tarkistaa Datacapin tuottaman sanomaid:tä vastaavan XML-tiedoston ja yhdistää sen ja Inbound Handler -ohjelman tuottaman sanomaid:tä vastaavan Sanomaid_outbound.xml-tiedoston yhdeksi tiedostoksi. Yhdistetty tiedosto sisältää prosessointiin tarvittavan metadatan ja alkuperäisen sähköpostidatan lisäksi Datacapin liitteestä tunnistamat tiedot. Tiedosto nimetään muodossa Sanomaid_response.xml, jonka jälkeen se siirretään eteenpäin Outbound handlerin Outbound-kansioon. (Korpela henkilökohtainen tiedonanto 1.2.2016.)

5.3.3. Valmiin sanoman jatkokäsittely

Outbound Handlerin Outbound-kansiosta yrityksen tekemä Outbound Handler -ohjelma siirtää tiedoston Proscessing-kansion sisälle väliaikaiseen prosessointikansioon. Sieltä Outbound Handler -ohjelma yrittää lähettää Sanomaid_response- tai Sanomaid_outbound.xml-tiedoston takaisin asiakkaan web-palveluun. Jos tiedosto saadaan lähetettyä onnistuneesti, tiedosto siirretään Archive-kansioon ja prosessointikansio poistetaan Processing-kansion sisältä. Jos web-palvelimeen ei saada yhteyttä,

Outbound Handler -ohjelma yrittää lähettää tiedoston viisi kertaa viiden sekunnin välein uudelleen. Tämän jälkeen Outbound Handler -ohjelma siirtää tiedoston Failed-kansioon. Jos tiedosto halutaan lähettää uudelleen, se pitää siirtää takaisin Outbound Handlerin Outbound-kansioon. (Korpela henkilökohtainen tiedonanto 1.2.2016.)

6. UUDEN YMPÄRISTÖN VAATIMUKSET

Tässä luvussa on listattu vaatimukset, jotka uuden ympäristön on täytettävä. Vaatimukset on jaettu sopimustason- ja kapasiteettivaatimuksiin sekä järjestelmätason vaatimuksiin. Järjestelmätason vaatimukset on johdettu nykyisen ympäristön ongelmista, ja jokaisen vaatimuksen kohdalla on esitelty ongelma, joka on tarkoitus ratkaista.

6.1. Sopimustason- ja kapasiteettivaatimukset

Sopimustason vaatimuksen mukaan ympäristön tulee olla asiakkaan saavutettavissa 24 tuntia jokaisena viikonpäivänä. Ympäristön toimittaja järjestää ympärivuorokautisen tuen jokaisena viikonpäivänä ja antaa asiakkaalle puhelinnumeron hätätilanteita varten. Saapuvista dokumenteista 80 prosenttia pitää mennä läpi kymmenessä minuutissa ja loput 20 prosenttia kahdessa tunnissa. Kapasiteettivaatimuksen mukaan järjestelmä on puolestaan suunniteltava niin, että se pystyy ottamaan käsittelyyn 2000 sanomaa päivässä ja 200 sanomaa tunnissa.

Sopimustason vaatimuksissa määritellään myös, että palveluun saavutettavuusasteen pitää olla vähintään 99 prosenttia. Lisäksi järjestelmän puhtaat päivät eli päivät, jolloin järjestelmän pitää olla saavutettavissa ja toimia ongelmitta, on 90 päivää. Saavutettavuus ja puhtaat päivät on laskettu ilman sovittuja järjestelmän huoltoja.

6.2. Järjestelmätason vaatimukset

6.2.1. Verkon hallinta on oltava helppoa ja ylläpitäjillä on oltava helppo pääsy ympäristöihin.

Nykyisen ympäristön verkkoarkkitehtuuri on hyvin monimutkainen, ja verkon ylläpitäminen on hyvin hankalaa. Etätyöpöytäyhteyden muodostaminen ympäristöjen palvelimille on monimutkaista, koska ensin pitää muodostaa etäyhteys FTPS-palvelimelle, josta otetaan virtuaaliverkon yli etäyhteys ympäristöön. Tämän jälkeen ympäristössä pitää ottaa vielä etäyhteys haluttuun palvelimeen. Ympäristöjen välillä ei ole myöskään mahdollista siirtää dataa, koska ympäristöstä toiseen ei ole suoraa pääsyä.

6.2.2. Ympäristöjen on tuettava noin 1200 Datacap-sovellusta.

Yksi keskeisin esille nouseva ongelma nykyisessä ympäristössä on, miten saadaan sovellusalustana käytettävä Datacap-alusta tukemaan suurta sovellusmäärää mahdollisimman hyvin. Tällä hetkellä loppuasiakkaita on noin 100, mutta loppuasiakkaiden määrä tulee nousemaan. Jokaiselle loppuasiakkaalle tarvitaan sanomien purkuun oma sovellus, joka käsittelee tulleet sanomat, mutta jo nyt alustassa on nähtävissä ongelmia mm. hidastumisena sovellusmäärän kasvaessa.

6.2.3. Sovellusten siirtäminen ympäristöstä toiseen on oltava mahdollisimman helppoa ja automatisoitua.

Toinen keskeinen esille noussut ongelma oli, miten sovellus saadaan siirrettyä yhdestä ympäristöstä toiseen uudella alustalla (kehitys → testaus → yhteistyökumppanin testaus → asiakkaan hyväksyntä → tuotanto), ja miten siirretään nykyisessä alustavassa olevat sovellukset uudelle alustalle.

6.2.4. Sanomien käsittely ei saa hidastua ja kaikki tulevat sanomat on poimittava ja otettava käsittelyyn.

Jostain syystä nykyisessä ympäristössä Datacap-sovelluksiin tulevien sanomien käsittely on välillä hidasta tai hidastuu. Lisäksi Datacap-sovellukset eivät ota kaikkia tulevia sanomia käsittelyyn. Selkeää syytä tähän ei tiedetä, mutta uudelleenkäynnistys yleensä korjaa ongelman.

6.2.5. Rulerunner-palvelimen konfigurointi oikeilla asetuksilla

Jostain syystä nykyisissä ympäristöissä Rulerunner-palvelimet käsittelevät Datacap-sovelluksiin tulleita sanomia hitaasti. Ongelmana on, millaiset asetukset Rulerunner-palvelimeen pitäisi konfiguroida, jotta se käsittelisi tulevat sanomat mahdollisimman nopeasti, miten uudet partnerit pitäisi ottaa ajoon, ja saako uusien partnerin ajoon ottamista automatisoitua.

6.2.6. Mahdollisista virheistä on tultava ilmoitus nopeasti ja luotettavasti.

Monitorointijärjestelmien lähettämät virheilmoitukset voivat jäädä nykyisessä ympäristössä välille, jos lähtevän sähköpostin palvelin on nurin, siihen ei saada yhteyttä tai verkkoyhteydet ulos ovat poikki.

6.2.7. Sanoma pitää saada lähetettyä uudelleen automaattisesti, jos yhteistyökumppanin web-palveluun ei saada yhteyttä.

Jos yhteistyökumppanin webserviceen ei saada yhteyttä, sanoma menee failed-kansioon, josta se pitää manuaalisesti siirtää eteenpäin.

6.2.8. Sisäisen virheen sattuessa sanoma pitää käsitellä uudelleen ja valmistuneiden Batchien poiston on oltava automaattista.

Jos loppuasiakkaiden Datacap-sovelluksissa tapahtuu sisäinen virhe käsittelyssä, ne eivät yritä automaattisesti uudelleen käsitellä sanomaa. Lisäksi niiden jättämiä vanhoja batch-kansioita ei poisteta automaattisesti Datacap-ympäristöistä minkä johdosta levytila voi joskus loppua kesken.

7. UUDEN YMPÄRISTÖN SUUNNITTELU

Uutta ympäristöä lähdettiin suunnittelemaan lähtökohdasta, että se täyttää luvussa kuusi esitetyt vaatimukset. Jotta kaikki vaatimukset tulisi ratkaistua uudessa ympäristössä ja sopimustason vaatimukset tulisi otettua huomioon sitä suunniteltaessa, johdettiin jokaisesta yksittäisestä vaatimuksesta ratkaisumalli. Jokainen niistä on esitelty tässä luvussa aliluvuittain. Lisäksi joissakin vaatimuksissa on esitelty selvä toteutusmalli, miten vaatimus tullaan toteuttamaan uudessa ympäristössä. Kaikkiin vaatimuksiin ei ole kuitenkaan olemassa selvää toteutusmallia, koska ne vaativat käytännön kokeilua ratkaisua varten tai ratkaisumalli sisältää jo toteutusmallin, miten ongelma tai haaste korjataan uudessa ympäristössä.

7.1. Verkon hallinta on oltava helppoa ja ylläpitäjillä on oltava helppo pääsy ympäristöihin.

Kuten luvussa 6.2.1. todettiin, on nykyisen ympäristön verkkoarkkitehtuuri hyvin monimutkainen, ympäristöihin pääsy hyvin monimutkaista ja verkon ylläpitäminen hyvin hankalaa. Ratkaisuna tähän on laatia koko verkkoarkkitehtuuri uusiksi. Sen sijaan, että kaikki ympäristöt ovat jokainen omassa suljetussa ympäristössään, sijoitetaan kaikki nykyiset ympäristöt samaan verkkoon. Verkkoon luodaan yksi toimialue, johon liitetään kaikkien nykyisten ympäristöjen palvelimet. Tällä tavalla verkon hallinta helpottuu, koska kaikki palvelimet sijaitsevat samassa verkossa, etäyhteyden ottamien mihin tahansa verkossa sijaitsevalle palvelimelle onnistuu ja datan siirtämi-

nen ympäristöstä toiseen tulee mahdolliseksi. (Sysilahti henkilökohtainen tiedonanto 12.2.2016.)

7.1.1. Toteutus

Uudessa ympäristössä luodaan verkkoarkkitehtuuri kokonaan uudelleen ja lähtökohtana on, että kaikki palvelimet sijaitsevat samassa aliverkossa ja samassa domainissa (katso liite 3). Uuteen ympäristöön ei tule kehitys-, testaus-, hyväksyntä- ja tuotantoympäristöille omaa suljettua verkkoa, vaan kaikkien ympäristöjen Datacap-palvelimet asennetaan samaan aliverkkoon, ja niille määritetään kiinteä IP-osoite. Se, onko Datacap-palvelin tarkoitettu kehitys-, testaus-, hyväksyntä- vai tuotantokäyttöön, määritetään uudessa ympäristössä päättämällä jokaisen Datacap-palvelimen kohdalla, minkä ympäristön tehtäviä se tulee hoitamaan. Kehitys-, testaus-, hyväksyntä- ja tuotantoympäristöille ei myöskään tule omia tietokantapalvelimia, vaan jokaiselle ympäristölle asennetaan yksi yhteinen tietokantapalvelin. Tämä vähentää tarvittavien palvelimien määrää, minkä ansiosta verkon rakenne yksinkertaistuu ja helpottaa osaltaan verkon ylläpidettävyyttä.

Tietokantapalvelimeen luodaan jokaiselle ympäristölle oma tietokanta. Loppuasiakkaiden Datacap-sovellukset sijoitetaan tietokantaan skeemoittain, jolloin jokaiselle loppuasiakkaalle ei tarvitse luoda omaa tietokantaa, eikä tietokantojen määrä pääsee kasvamaan liian isoksi. Lisäksi kaikkien ympäristöjen kaikkien loppuasiakkaiden Datacap-sovellusten batch-kansioiden luonti sekä sanomien vastaanottaminen keskitetään hallitusti yhdelle tiedostonjakopalvelimelle. Kun kaikki tiedostojenkäsittely- ja tallennus tehdään yhdellä levyjakopalvelimella, pystytään keskitetysti hallitsemaan levyresursseja, ja verkonhallintaa helpottuu, koska kaikki tiedostonkäsittely ja tallennus tapahtuvat yhdellä palvelimella eikä monella eri verkon palvelimella.

Uusi verkkoarkkitehtuuri selkeyttää verkon rakennetta. Verkon ylläpitäminen helpottuu, koska enää ei ole käytössä lukuisia erilaisia yksityisiä verkkoja vaan kaikki palvelimet sijaitsevat samassa verkossa ja samassa domainissa. Kun kaikki palvelimet ovat samassa verkossa, saadaan niiden määrää myös pudotettua, koska enää ei tarvita ympäristöille omia AD- ja tietokantapalvelimia. Se selkeyttää osittain myös verkon

rakennetta, koska mitä vähemmän verkossa on palvelimia, sitä yksinkertaisempi ja selkeämpi verkon rakenne on.

Uusi verkkoarkkitehtuuri helpottaa ylläpitäjien pääsyä palvelimille. Kun kaikki palvelimet sijaitsevat samassa aliverkossa, ylläpitäjät pystyvät muodostamaan yhteyden haluamalleen palvelimelle suoraan työasemaltaan ilman, että heidän tarvitsee ensin ottaa etäyhteys monelle eri palvelimelle, ennen kuin he pääsevät haluamalleen palvelimelle. Lisäksi uusi verkkoarkkitehtuuri mahdollistaa myös, että Datacap-sovelluksia pystytään siirtämään ympäristöjen välillä, koska palvelimet sijaitsevat samassa verkossa, ne näkevät toisensa, ja niihin pääsee käsiksi suoraan toisesta samassa aliverkossa olevasta palvelimesta.

Uusi verkkoarkkitehtuuri mahdollistaa LLDAP-käyttäjätodennuksen käyttöönottamisen (katso liite 4) ja verkon tietoturvan lisäämisen. Nykyisissä ympäristöissä käyttäjät kirjautuvat palvelimiin samoilla käyttäjätunnuksilla, joilla on täydet valtuudet muokata kaikkia palvelimen asetuksia. Se lisää riskiä, että käyttäjät tulevat vahingossa muokanneeksi asetuksia, joita eivät olisi halunneet muokata tai eivät olisi saaneet muokata. Uudessa ympäristössä pystytään estämään tämä ottamalla käyttöön LLDAP-käyttäjätodennus ja jakamalla käyttäjät käyttäjäryhmiin ylläpitäjät, sovelluskehittäjät ja käyttäjät, joilla ei ole oikeuksia tehdä mitään muutoksia palvelimiin. Tällä tavalla pystytään kontrolloimaan, mitä oikeuksia kullakin käyttäjällä on mihinkin ympäristön osiin. Se on erityisen tärkeää etenkin Datacap-komponenttien kohdalla, koska ne sisältävät paljon asetuksia, joihin kaikilla ympäristön käyttäjillä ei ole tarvetta päästä käsiksi.

LLDAP-käyttäjätodennus lisää myös seurattavuutta. Kun kaikilla käyttäjillä on omat tunnukset Datacap-palvelimille, pystytään seuramaan, kuka muutti ja mitä osaa Datacap-sovelluksesta. Se helpottaa mahdollisesti myös yhtäaikaista kehitystä, jos datacap-sovellusten moduuleja pystytään lukitsemaan. Sen testaaminen on kuitenkin mahdollista vasta joko uuden ympäristön toteutusvaiheessa tai käyttöönottovaiheessa, eikä onnistumisesta pystytä antamaan varmuutta vielä suunnitteluvaiheessa.

7.2. Ympäristön saavutettavuusasteen on oltava korkea.

Kuten luvun 6.1. sopimustason vaatimuksissa on määritelty, on koko ympäristön oltava saavutettavissa 24 tuntia viikon jokaisena päivänä. Ratkaisuna tähän on laatia kriittisistä palvelimista HA-ympäristöjä. Nämä palvelimet ovat AD-palvelin, FTPS-palvelin, tietokantapalvelin, tiedostonjakopalvelin sekä Datacap-palvelimet, jotka ovat tuotantokäytössä. Lisäksi verkon ja virtualisointialustan on oltava HA, mutta ne ovat olleet konfiguroituna HA:ksi ennen opinnäytetyön kirjoittamisen aloittamista ja jätetty siitä syystä sen ulkopuolelle.

7.2.1. HA AD:n toteuttaminen

Koska uudessa ympäristössä käyttäjien todennus ympäristön palvelimille ja Datacap-komponentteihin tapahtuu AD-palvelimella, ei käyttäjien pääsy niihin saa estyä, vaikka AD-palvelin lakkaisi toimimasta. Tilannetta pystytään korjaamaan asentamalla ympäristöön kaksi AD-palvelinta ja muodostetaan niiden avulla ympäristöön HA AD (katso liite 3). Sen toteuttaminen on teoriassa yksinkertaista: ensin asennetaan yhdelle palvelimelle AD-rooli ja luodaan siihen domain. Kun se on tehty, asennetaan toiselle palvelimelle AD-rooli ja liitetään se jo olemassa olevaan domainiin. Liittämisen yhteydessä toinen AD-palvelin kopioi ensimmäiseltä AD-palvelimelta AD:n ja domainin asetukset. Sen jälkeen asetetaan molemmat palvelimet synkronoimaan käyttäjät ja asetukset säännöllisesti, jotta molemmat palvelimet olisivat aina ajan tasalla. Kun tämä on tehty, on verkossa käytössä kaksi AD-palvelinta, joista jompikumpi toimii vuorollaan AD- ja domain-palvelimena. Jos jompikumpi palvelimista lakkaa toimimasta, osaa jäljellä oleva toimiva AD- ja domain-palvelin automaattisesti mainostaa itseään ympäristön muille palvelimille olevansa nykyinen AD- ja domain-palvelin.

7.2.2. HA FTPS-yhteyden toteuttaminen

Koska sanomien vastaanottaminen yhteistyökumppanilta ja niistä kerätyn datan lähettäminen takaisin yhteistyökumppanille tapahtuu FTPS-palvelimen kautta, ei lähettäminen tai vastaanottaminen saa keskeytyä, vaikka FTPS-yhteys lakkaisi toimimas-

ta. Yhteyden jatkuva saavutettavuus pystytään varmistamaan rakentamalla HA FTPS-yhteys käyttäen apuna Microsoft Windows 2012 R2 -palvelinkäyttöjärjestelmän tarjoamaa Network Load Balancing -ominaisuutta (katso liite 3). Sen avulla pystytään kaksi FTPS-palvelinta liittämään Network Load Balancing -ominaisuuden solmuiksi. Käyttäjille solmut näkyvät yhtenä yhteysosoitteena, mutta todellisuudessa sanomat välitetään jommallekummalle FTPS-palvelimelle. Jos toinen FTPS-palvelin lakkaa vastaamasta Network Load Balancing Managerille, se jatkaa sanomien välittämistä FTPS-palvelimelle, joka vastaa yhä sen kutsuille. Tällaisella toteutuksella pyritään siihen, ettei sanomien vastaanottaminen ja siitä saadun datan lähettäminen keskeytyisi virhetilanteissa ja taataan ympäristön korkeaa saavutettavuutta sanomien vastaanottamisen ja siitä saadun datan lähettämisen osalta.

7.2.3. HA-tiedostojaon toteuttaminen

Koska Datacap-sovellukset tarvitsevat toimiakseen tallennustilaa batch-kansioille ja käsiteltävien tiedostojen vastaanottamiseen, pitää tiedostojen säilytyspaikka olla sellainen, että se on koko ajan hyvin saavutettavissa mahdollisista virhetilanteista huolimatta. Tämä pystytään varmistamaan sijoittamalla Datacap-sovellusten batch-kansiot, tiedostojen vastaanottaminen ja niiden käsittely levyjakopalvelimelle (katso liite 3). Levyjakopalvelimella hyödynnetään levyjakoon Microsoft Windows 2012 R2 -palvelinkäyttöjärjestelmän mukanaan tuomaa CAFS-tiedostojakoa. Toteutuksessa kaksi Windows 2012 R2 -palvelinkäyttöjärjestelmän palvelinta liitetään Failover klusteri -ominaisuuteen solmuiksi. Tämän jälkeen Failover klusteriin lisätään CAFS-tiedostonjakorooli. Jos toiseen palvelimista tulee tämän jälkeen virhetilanne, siirtyy tiedostonjako toimivalle palvelimelle. Tällä tavalla pystytään minimoimaan riskiä, että Datacap-sovellukset lakkaisivat toimimasta tiedostonkäsittelyn vuoksi ja pystytään varmistamaan koko ympäristön korkeaa saavutettavuutta.

7.2.4. HA-tietokantayhteyden toteuttaminen

Koska Datacap-sovellukset vaativat toimiakseen toimivan tietokantayhteyden, pitää tietokannan olla saavutettavissa, vaikka itse tietokantapalvelimeen tulisi vika. Tietokantayhteyden varmuutta nostetaan uudessa ympäristössä asentamalla HA tietokan-

tayhteys. IBM tarjoaa Microsoft Windows -käyttöjärjestelmäalustalle asennetulle DB2-tietokantajärjstelmälle kahta tapaa tehdä HA-tietokantayhteys. Nämä ovat High Availability Disaster Recovery (HADR) ja tietokantainstanssin asentamisen Microsoft Windows -palvelinkäyttöjärjestelmän Failover klusterille. (IBM Knowledge Center 2016.)

HADR-toteutuksessa tietokannan lokidata replikoidaan aktiivisesta tietokannasta yhteen tai useampaan valmiustilassa olevaan tietokantaan. Jos aktiivinen tietokanta vikaantuu, muuttuu joku valmiustilassa olevista tietokannoista aktiiviseksi tietokannaksi, ja sovellukset ohjataan käyttämään tätä tietokantaa. (High Availability and Disaster Recovery Options for DB2 for Linux, UNIX, and Windows 2012, 6.)

Toinen tapa toteuttaa HA-tietokantayhteys Windows-palvelinkäyttöjärjestelmän DB2-tietokantapalvelimelle on luoda DB2-tietokantainstanssi Windows-palvelinkäyttöjärjestelmän failover klusteriin. Se toteutetaan niin, että ensin jokaiselle klusterin solmupalvelimelle asennetaan kopio DB2-tietokantahallintajärjestelmästä. Sen jälkeen yhdellä solmupalvelimista luodaan DB2-tietokantainstanssi failover klusteriin. Kun se on luotu, liitetään muut failover klusterin solmupalvelimet luotuun instanssiin. (High Availability and Disaster Recovery Options for DB2 for Linux, UNIX, and Windows 2012, 101-154.)

Itse HA-tietokantayhteyden logiikka failover klusterissa on yksinkertainen. Failover klusterin solmupalvelimet näkyvät tietokantayhteyttä käyttäville sovelluksille yhtenä yhteysosoitteena. Yhdellä solmupalvelimista on kuitenkin kerrallaan käynnissä db2-instanssi. Jos tähän solmupalvelimeen tulee vika, siirtyy db2-instanssi pyörimään automaattisesti jollekin toiselle solmupalvelimelle. (High Availability and Disaster Recovery Options for DB2 for Linux, UNIX, and Windows 2012, 224.)

HA-tietokantayhteys suunniteltiin ensin toteutettavaksi HADR-menetelmällä. Siinä tuli kuitenkin eteen ongelmaksi se, että aktiivisen tietokannan vikaantuessa järjestelmä ei osaa muuttaa automaattisesti valmiustilassa olevaa tietokantaa aktiiviseksi ja ottaa sitä käyttöön. Käyttöönotto olisi pitänyt tehdä manuaalisesti, mikä on huono ympäristössä, jonka saavutettavuusasteen pitää olla korkea ympäri vuorokauden.

(High Availability and Disaster Recovery Options for DB2 for Linux, UNIX, and Windows 2012, 224).

Sen sijaan tässä opinnäytetyössä HA-tietokantayhteys päädyttiin toteuttamaan Windows-palvelinkäyttöjärjestelmän failover klusterin avulla, koska se tarjoaa hyvän mahdollisuuden db2-tietokannan toimia tehokkaasti. Liiketoiminnallista dataa ei häviä, koska palvelinsolmujen työtaakan jako on automatisoitua. Klusterointi tehostaa myös työntekoa ja liiketoimintaa, koska se vähentää häiriöaikaa ja ihmisten huomiontarvetta. Lisäksi klusterointia suositellaan käytettäväksi ympäristöissä, joissa palvelimille on asennettuna saman tuoteperheen käyttöjärjestelmä, ja käytetään käyttöjärjestelmä riippuvaisia klusterointiratkaisuja. (High Availability and Disaster Recovery Options for DB2 for Linux, UNIX, and Windows 2012, 8.)

7.2.5. HA tuotantoympäristön toteuttaminen

Asiakkaan uuden ympäristön tuotantoympäristö suunnitellaan kokonaan uudelleen. Tuotantoympäristön korkeaa saavutettavuutta ei toteuteta enää luomalla asiakkaalle kahta erillistä tuotantoympäristöä, vaan siitä suunnitellaan HA-ympäristö IBM:n virallisten ohjeiden mukaan (katso liite 1). Datacap-ympäristön toteuttamiseen HA-ympäristöksi on olemassa useita erilaisia tapoja. Niistä on esitelty tässä opinnäytetyössä lyhyesti vain tavat, jotka keskittyvät luomaan Datacap-ympäristöstä HA-ympäristön hyödyntäen Datacap Taskmaster- ja Rulerunner-palvelimia.

Ensimmäinen tapa toteuttaa Datacap-ympäristöstä HA-ympäristö on skaalata Rulerunner-palvelimia horisontaalisesti. Tässä toteutustavassa asennettuna on useita keskitettyjä Datacap Taskmaster -palvelimia. Niihin on liitettynä yksi tai useampia Rulerunner-palvelimia, jotka jakavat prosessoitavan kuorman tasaisesti keskenään. Rulerunner-palvelimet voivat olla joko yksi- tai monisäikeisiä. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 175-177.)

Toinen tapa toteuttaa Datacap-ympäristöstä HA-ympäristö on Rulerunner-palvelimien skaalaaminen vertikaalisesti ja horisontaalisesti. Vertikaalinen toteutus-

tapa tarkoittaa Rulerunner-palvelimen fyysisten resurssien kasvattamista ja jakamista. Se voidaan toteuttaa kahdella eri tavalla. Ensimmäisessä tavassa Rulerunner-palvelimen prosessointitehoa kasvatetaan parantamalla sen fyysisiä ominaisuuksia esimerkiksi lisäämällä palvelimen keskusmuistia, päivittämällä palvelimen prosessoreita tai parantamalla kiintolevyjen kirjoitusnopeutta. Toinen tapa on lisätä Rulerunner-palvelimen käyttämien säikeiden määrää. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 175-177.)

Vertikaalisessa ja horisontaalisessa toteutustavassa hyödynnetään molempia Rulerunner-palvelimien skaalaustekniikoita. Siinä prosessointitehoa kasvatetaan sekä tehostamalla Rulerunner-palvelimen fyysisiä ominaisuuksia, että lisäämällä ympäristön Rulerunner-palvelimien määrää. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 176.)

Kolmas tapa toteuttaa Datacap-ympäristöstä HA-ympäristö on skaalata Taskmaster-palvelimia. Taskmaster-palvelimia voidaan lisätä ympäristöön ja konfiguroida ne aktiivi-aktiiveiksi tai aktiivi-passiiveiksi. Aktiivi-aktiivi konfiguroinnissa Datacap-ympäristöön on asennettu vähintään kaksi Taskmaster-palvelinta. Molemmilla on eri ip-osoite ja verkkonimi. Jos joku Taskmaster-palvelimista lakkaa toimimasta, ohjataan sovellukset käyttämään jäljellä olevia Taskmaster-palvelimia. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 179.)

Aktiivi-passiivi konfiguroinnissa ympäristöön on asennettuna vähintään kaksi Taskmaster-palvelinta, joista yksi on vuorollaan aktiivinen palvelin ja loput passiivisia. Ne on konfiguroitu identtisiksi, ja kaikilla on sama ip-osoite ja verkkonimi. Jos aktiivinen palvelin lakkaa toimimasta, käynnistetään joku passiivista palvelimista manuaalisesti. Se jatkaa tehtävien jakamista siitä, mihin edellinen aktiivinen Taskmaster-palvelin jäi. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 179.)

Neljäs tapa toteuttaa Datacap-ympäristöstä HA-ympäristö, on skaalata sekä Taskmaster, että Rulerunner-palvelimia. Tällaisen toteutuksen etuna on, että kaikkia Datacap-palvelimia pystytään lisäämään ympäristöön koska tahansa, eikä ympäristöön tarvitse asentaa erillisiä network load balancingeja. Toteutusmallissa ympäris-

töön on asennettuna vähintään kaksi Taskmaster- ja Rulerunner-palvelinta. Kaikki Taskmaster-palvelimet jakavat saman tietokannan, jonne ne kirjoittavat tietoa batchien käsittelyn vaiheista ja pystyvät sen ansiosta jakamaan saman Datacap-sovelluksen tehtäviä ja batcheja Rulerunner-palvelimille. Jos joku Taskmaster-palvelimista vikaantuu eikä pysty jatkamaan Datacap-sovelluksen ja batchien jakamista Rulerunner-palvelimille, pystyvät muut Taskmaster-palvelimet jatkamaan saman sovelluksen tehtävien ja batchien jakamista Rulerunner-palvelimille jaetun tietokannan ansiosta. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 181.)

Rulerunner-palvelimet ovat puolestaan konfiguroitu ottamaan yhteyttä yhteen Taskmaster-palvelimeen aktiivisemmin kuin toiseen. Tällä tavalla ne ovat kuitenkin tietoisia kaikkien Taskmaster-palvelimien olemassaolosta. Jos aktiivisemmin yhteyttä otettava Taskmaster-palvelin lakkaa kuitenkin vastaamasta Rulerunner-palvelimille, ne aloittavat automaattisesti käyttää aktiivisesti muuta yhä pystyssä olevaa Taskmaster-palvelinta. Jos puolestaan itse Rulerunner-palvelin lakkaa vastaamasta Taskmaster-palvelimille, osaavat ne olla välittämättä kaatuneesta Rulerunner-palvelimesta ja jatkaa tehtävien ja batchien jakamista pystyssä olevien Rulerunner-palvelimien kesken. (Implementing Document Imaging and Capture Solutions with IBM Datacap 2015, 181.)

Tässä opinnäytetyössä päädyttiin toteuttamaan tuotantoympäristön korkea saavutettavuus skaalaamalla sekä Datacap Taskmaster- että Rulerunner -palvelimia (katso liite 5). Tämä lähestymistapa mahdollistaa sen, ettei sanomien käsittely keskeydy, koska tuotantoympäristössä on asennettua kaksi Taskmaster-palvelinta. Jos toinen niistä lakkaa vastaamasta Rulerunner-palvelimille, pystyvät ne ottamaan yhteyttä toiseen Taskmaster-palvelimeen ja jatkamaan sanomien käsittelyä keskeytyksettä. Jos puolestaan yksi Rulerunner-palvelimista lakkaa vastaamasta Taskmaster-palvelimille, ei sanomien käsittely keskeydy sen takia, vaan Taskmaster-palvelimet jatkavat sanomien välittämistä jäljellä olevien Rulerunner-palvelimien kesken.

Tämä lähestymistapa korjaa tuotantoympäristön osalta myös luvussa 6.2.2. esiintyneen ongelman, miten saadaan Datacap-alusta tukemaan suurta Datacap-sovellusmäärää. Kun Datacap-sovelluksia alkaa tulla tuotantoympäristöön liikaa, ja

tuotantoympäristö alkaa hidastua, sen toimintavarmuutta ja suorituskykyä pystytään parantamaan lisäämällä ympäristöön lisää Datacap Taskmaster- ja Rulerunner -palvelimia.

7.3. Ympäristöjen on tuettava noin 1200 Datacap-sovellusta.

Kuten luvussa 6.2.2. todettiin, on nykyisen ympäristön ongelmana, miten saadaan sovellusalustana käytettävä Datacap-alusta tukemaan suurta määrää Datacap-sovelluksia. Ongelmaa lähdetään selvittämänään ensin testaamalla, pilotoimalla ja mittaamalla Datacap-alustan suorituskykyä. Sen jälkeen saatujen tulosten perusteella lähdetään etsimään ratkaisua IBM Datacap Redbooks -kirjasta ja internetistä.

7.3.1. Toteutus

Luvussa 7.2.6. esiteltiin tapa, miten saadaan tehtyä tuotantoympäristöstä korkeasti saavutettava ympäristö lisäämällä Datacap Taskmaster- ja Rulerunner-palvelinten määrää. Samalla tavalla pystytään ratkaisemaan ongelma, miten saadaan sovellusalustana käytettävä Datacap tukemaan isoja määriä sovelluksia. Kaikkiin muihin ympäristöihin paitsi tuotantoympäristöön asennetaan aluksi yksi Datacap Taskmaster- ja Rulerunner-palvelin (katso liite 6). Kun Datacap-sovelluksia alkaa tulla ympäristöihin enemmän, eivätkä Taskmaster-ja Rulerunner palvelimet pysty prosessoimaan niitä kyllin nopeasti, lisätään ympäristöihin lisää Datacap Taskmaster- ja Rulerunner-palvelimia. Lisäämisen ansiosta vältetään palvelimien hidastuminen ja kaatuminen liian suuren sovellusmäärän vuoksi, pystytään tasapainottamaan palvelimien kuormitusta hajauttamalla Datacap-sovelluksia eri Datacap Taskmaster- ja Rulerunner-palvelimille ja mahdollistetaan, että ympäristöt pystyvät tukemaan mahdollisimman isoja määriä Datacap-sovelluksia.

Toinen tapa saada Datacap-alusta tukemaan noin 1200 Datacap-sovellusta, on skaalata luvussa 7.2.6. mainitun mallin mukaan Rulerunner-palvelimia sekä vertikaalisesti että horisontaalisesti. Vaikka sovellusmäärän lisääntymistä pystytään tukemaan ympäritöissä helposti lisäämällä Datacap Taskmaster- ja Rulerunner-palvelimien määrää, se voi johtaa tilanteeseen, jossa etenkin Rulerunner-palvelinten määrä voi

nousta kohtuuttoman isoksi. Se puolestaan monimutkaistaa verkon rakennetta ja vaikeuttaa verkon ylläpitämistä. Tätä voidaan ehkäistä lisäämällä Rulerunner-palvelimien prosessoreiden määrää sekä yhden prosessorin ytimien määrää, koska mitä enemmän Rulerunner-palvelimessa on säikeitä, sitä enemmän sovelluksia yhdellä Rulerunner-palvelimella voi ajaa. Yhdellä Rulerunner-palvelimella voi olla säikeitä 150 prosenttia enemmän kuin palvelimen prosessorissa tai prosessoreissa on ytimiä. (IBM Knowledge Center 2016).

7.4. Sovellusten siirtäminen ympäristöstä toiseen on oltava mahdollisimman helppoa ja automatisoitua.

Luvussa 6.2.3 nousi uutta ympäristöä suunniteltaessa ongelmaksi, miten Datacap-sovellukset siirretään uudessa ympäristössä toiseen, ja miten Datacap-sovellusten siirtäminen onnistuu nykyisestä ympäristöstä uuteen. Datacap-sovellusten siirtäminen ympäristöstä toiseen onnistuu suunnittelemalla siirtoskriptit, joilla Datacap-ohjelmistolla tehtyjä Datacap-sovelluksia pystytään siirtämään ympäristöstä toiseen. Samoin sovellusten siirtäminen vanhasta ympäristöstä uuteen onnistuu laatimalla sitä varten erikseen siirtsoskriptit. Itse siirtoskriptien suunnittelu on jätetty tämän opinnäytetyön ulkopuolelle.

7.5. Sanomien käsittely ei saa hidastua.

Luvussa 6.2.4. esiteltiin ongelma, että jostain syystä Datacap-sovellukseen tulevien sanomien käsittely on välillä hidasta tai hidastuu. Ongelmaa lähdetään selvittämään ajamalla uudessa ympäristössä uusimmalla Datacap-ohjelmistoversiolla sanomia läpi sovelluksista. Saatujen tulosten perusteella lähdetään selvittämään internetistä ja IBM Datacap Redbooks-kirjasta, oliko vikana vanha Datacap-ohjelmistoversio, onko kyseessä bugi Datacap-ohjelmistossa vai onko Datacap-ohjelmalla tehdyt sovellukset konfiguroitu väärin.

7.6. Kaikki tulevat sanomat on poimittava ja otettava käsittelyyn.

Luvussa 6.2.4. esiteltiin ongelma, että jostain syystä nykyisissä ympäristöissä Datacap-sovellus ei ota kaikkia tulevia sanomia käsittelyyn. Ongelmaa lähdetään selvittämään näkemällä se konkreettisesti. Saatujen havaintojen perusteella yritetään selvittää ratkaisua esimerkiksi internetistä ja IBM Datacap Redbook -kirjasta, onko kyseessä jokin väärin määritelty Datacap-ohjelman osa vai mahdollisesti bugi Datacap-ohjelmassa.

7.7. Rulerunner-palvelimen konfigurointi oikeilla asetuksilla

Luvussa 6.2.5. esiteltiin ongelma, miten uudet partnerit pitäisi ottaa ajoon ja saako uusien partnerin ajoon ottamista automatisoitua. Ratkaisu aloitetaan selvittämällä, miten uusien partnerien ottaminen ajoon tehdään nykyisin ja miten se tehtäisiin uudessa ympäristössä. Sen jälkeen lähdetään selvittämään internetistä ja IBM Datacap Redbook-kirjasta, miten partnerien ottaminen ajoon Rulerunner-palvelimella pitäisi tehdä käytännössä. Jollei niistä ole apua, lähetetään IBM:n tukeen palvelupyyntö ongelmasta. Lisäksi mietitään, olisiko uusien partnerien ottaminen ajoon mahdollista automatisoida. Lopullinen toteutus tehdään skriptauksella joko dynaamisesti tai konfiguroidusti, mutta se on jätetty tämän opinnäytetyön ulkopuolelle.

7.8. Mahdollisista virheistä on tultava ilmoitus nopeasti ja luotettavasti.

Luvussa 6.2.6. esiteltiin ongelma, että monitorointijärjestelmien lähettämät virheilmoitukset voivat jäädä välille, jos lähtevän sähköpostin palvelin on nurin, siihen ei saada yhteyttä tai verkkoyhteydet ulos ovat poikki. Ratkaisu ongelmaan on konfiguroida monitorointijärjestelmä niin, että se tarkistaa sähköpostiviestin perille menon. Jos se ei mennyt perille, konfiguroidaan monitorointijärjestelmä lähettämään viesti uudelleen tietyn ajan kuluttua. Tämän lisäksi mietitään, pystytäänkö virheilmoitus lähettämään sähköpostiviestin lisäksi tekstiviestillä tai muulla vastaavalla tavalla, kuten asentamalla ympäristöön Dashboard keräämään dataa sen toiminnasta.

7.9. Sanoma pitää saada lähetettyä uudelleen automaattiesti, jos yhteistyökumppanin web-palveluun ei saada yhteyttä.

Luvussa 6.2.7 esiteltiin ongelma, jossa yhteistyökumppanin webserviceen ei saada yhteyttä, niin sanoma menee failed-kansioon. Sieltä se pitää manuaalisesti siirtää eteenpäin. Ratkaisuna kokeillaan sanoman jättämistä odottamaan niin kauaksi aikaa, että yhteistyökumppanin webserviceen saadaan uudelleen yhteys. Tämän jälkeen yritetään siirtää sanoma uudelleen sinne.

7.10. Sisäisen virheen sattuessa sanoma pitää käsitellä uudelleen.

Luvussa 6.2.8. esiteltiin ongelma, että jos Datacapissa tapahtuu sisäinen virhe käsittelyssä, se ei yritä automaattisesti uudelleen käsitellä sanomaa. Ratkaisuna konfiguroidaan Datacap Maintenance Manager -komponentti nollaamaan batchin käsittelytila. Kun käsittelytila on nollattu, osaa Rulerunner Servce -komponentti ottaa batchin uudelleen käsittelyyn.

7.11. Valmistuneiden Batchien poiston on oltava automaattista.

Luvussa 6.2.8. esiteltiin ongelma, ettei Datacap -ympäristöistä poisteta vanhoja batch-kansioita automaattisesti. Tämä pystytään ratkaisemaan uudessa ympäristössä konfiguroimalla Datacap Maintenance Manager -komponentti poistamaan tai arkistoimaan valmistuneet batchit.

7.12. Suunnittelun lopputulos

Suunnittelun lopputuloksena ympäristöön rakennetaan 20 uutta palvelinta:

- Kehitys- ja testaus-ympäristöihin tulee yhteensä kuusi palvelinta ja ympäristöjä tulee yhteensä kolme kappaletta: yrityksen oma kehitysympäristö, yhteistyökumppanin testiympäristö ja asiakkaan testiympäristö. Jokaista ympäristöä kohden tarvitaan yksi Taskmaster- ja Rulerunner-palvelin.

- Asiakkaan tuotantoympäristöön tulee yhteensä viisi palvelinta. Koska ympäristön tulee olla saavutettavissa 24 tuntia jokaisena viikon päivänä, ympäristöstä suunnitellaan korkeasti saavutettava. Se toteutetaan asentamalla ympäristöön kaksi Datacap Taskmaster-palvelinta ja kolme Datacap Rulerunner-palvelinta.
- Ympäristöön tulee kaksi AD-palvelinta. Koska käyttäjien todennus palvelimille ja Dataca-sovelluksiin tapahtuu AD:n kautta, ja palvelimiin sekä Datacap-sovelluksiin pitää päästä kirjautumaan, vaikka AD-palvelin olisi poissa käytöstä esimerkiksi laiterikon vuoksi, suunnitellaan ympäristöön yksi kahdennettu AD-palvelin, jonka domainiin kaikki palvelimet liitetään.
- Ympäristöön tulee kaksi FTPS-palvelinta. Koska FTPS-palvelinta käytetään sekä sanomien vastaanottamiseen yhteistyökumppanilta, että lähettämiseen yhteistyökumppanille, eikä sanomien vastaanottaminen tai lähettäminen palvelimelta saa keskeytyä jommankumman palvelimen vikaantuessa, suunnitellaan yksi kahdennettu FTPS-palvelin, johon on pääsy jokaisesta ympäristöstä.
- Ympäristöön tulee kaksi levyjako-palvelinta. Koska Datacap sovellukset tarvitsevat paljon levytilaa sanomien käsittelyyn, eikä sanomien käsittely saa keskeytyä palvelimien vikaantuessa, suunnitellaan yksi kahdennettu levyjakopalvelin, johon on pääsy kaikista ympäristöistä.
- Ympäristöön tulee kaksi tietokantapalvelinta. Koska IBM Datacap-sovellukset vaativat toimiakseen tietokantayhteyden, eivätkä Datacap-sovellukset saa kaatua, vaikka tietokantapalvelimeen tulisi vika, suunnitellaan yksi kahdennettu tietokantapalvelin, johon kaikkien Datacap sovelluksien tietokannat tallennetaan.

8. LOPUKSI

Opinnäytetyön tarkoitus oli suunnitella uusi OCR-palvelinalusta niin, että se korjaisi mahdollisimman paljon nykyisessä ilmenneitä ongelmia, parantaisi korkeaa saavutettavuutta ja nopeuttaisi sanomien käsittelyä. Suunnittelu oli haastavaa, koska nykyi-

sessä ympäristössä oli paljon ratkaisemattomia ongelmia. Niiden selvittämiseen kului paljon aikaa ja suunnittelua haittasi etenkin Datacap-ohjelmiston kohdalla dokumentaation niukkuus ja vaikeaselkoisuus. Lisää vaikeutta suunnittelun toi Datacap-ohjelmisto kokonaisuutena, koska se on käsitteellisesti hyvin laaja ja sen toimintalogiikan sekä komponenttien välisten suhteiden ymmärtäminen oli haastavaa ja aikaa vievää.

Suunnittelun haasteellisuudesta huolimatta onnistuttiin opinnäytetyössä nostamaan uuden ympäristön automaatioastetta sekä löytämään ratkaisumalli moneen ongelmaan, miten ne korjattaisiin uudessa ympäristössä. Joissakin tapauksissa ongelma johtui yksinkertaisesti nykyisen ympäristön monimutkaisuudesta. Tällaisia ongelmia olivat esimerkiksi Datacap-sovelluksien siirto ympäristöjen välillä ja etätyöpöytäyhteyden muodostaminen ympäristöihin. Jotkut ongelmat olivat puolestaan sellaisia, ettei niitä oltu osattu ottaa huomioon nykyistä ympäristöä suunniteltaessa. Näitä olivat esimerkiksi levytilan täyttyminen batch-kansioista sekä sanomien uudelleen käsittely.

Jotkut ongelmat nykyisessä ympäristössä johtivat puolestaan siihen, ettei se täyttänyt kunnolla sopimustaso- ja kapasiteettivaatimuksia. Nämä ongelmat liittyivät lähinnä korkean saavutettavuuden vaatimuksiin. Nykyisessä ympäristössä ei oltu tehty kriittisistä palvelimista kunnollisia HA-toteutuksia tai ne olivat tehty hyvin monimutkaisesti. Esimerkiksi nykyisen ympäristön tuotantoympäristön korkeaa saavutettavuus on toteutettu hyvin monimutkaisesti, vaikka saatavilla olisi ollut yksinkertaisempiakin tapoja toteuttaa se.

Nykyisessä ympäristössä oli lisäksi haasteita, joihin ei opinnäytetyössä pystytty antamaan selvää vastausta, miten ne voitaisiin korjata uudessa ympäristössä. Näitä olivat esimerkiksi Rulerunner-palvelimen konfigurointi oikeilla asetuksilla ja sanomien käsittelyn hidastuminen. Tämä johtuu yksinkertaisesti siitä, että haasteet vaativat käytännön kokeiluja ennen kuin pystytään toteamaan, mistä ne johtuvat ja onko niihin saatavilla ratkaisua. Joskus haasteeseen ei ole edes saatavilla ratkaisua. Tällaisissa tilanteissa on yritettävä löytää toisenlainen tapa ratkaista se.

Vaikka opinnäytetyössä onnistuttiin laatimaan suunnitelma, miten uusi ympäristö tulisi rakentaa, ei se ole aukoton. Vasta toteutusvaiheessa nähdään, toimivatko suunnitellut ratkaisuideat ongelmien korjaamiseksi vai ei. Suunnitelma ei myöskään kerro, miten ongelma tai haaste pitäisi ratkaista käytännössä. Toteutusvaiheessa saattaa vastaan tulla myös ongelmia, joita ei ole otettu tai pystytty ottamaan huomioon suunnitteluvaiheessa. Hyvä suunnitelma antaa kuitenkin paremmat edellytykset lähteä toteuttamaan uutta ympäristöä, koska siinä on muun muassa jo mietitty ympäristön HA-kuvioita sekä palvelimien välistä työnjakoa.

Tämä opinnäytetyö ei myöskään kata kaikkea, mitä uutta ympäristöä suunniteltaessa tulisi ottaa huomioon, koska siitä tulisi muuten liian laaja kokonaisuus. Tällaisia ovat muun muassa ympäristön toipumissuunnitelma, Datacap-komponenttien varmuuskopiointisuunnitelma sekä ympäristöjen tietokantojen luonnin ja varmuuskopioinnin suunnittelu. Hyvin tehty järjestelmäsuunnittelu helpottaa kuitenkin niiden laatimista, koska siinä on jo otettu huomioon esimerkiksi toipumista nopeuttavia ominaisuuksia, joita ovat esimerkiksi palvelimien rakentaminen HA:ksi.

LÄHTEET

Abbyy Inc. Viitattu 26.9.2016. http://www.abbyy.com/

Adobe Systems Inc. Viitattu 26.9.2016. http://www.adobe.com/#

Beal, V. 2016. OMR - optical mark recognition. Viitattu 9.5.2016. http://www.webopedia.com/TERM/O/OMR.html

Benjaminperkins 2013. What's new in IIS 8.5 – Microsoft Internet Information Services 8.5 New Features. Viitattu 26.7.2016. https://blogs.msdn.microsoft.com/benjaminperkins/2013/06/25/whats-new-in-iis-8-5-microsoft-internet-information-services-8-5-new-features/

CVISION Technologies Inc. Viitattu 9.5.2016. http://www.cvisiontech.com/

Dorfman, M. & Thayer, R. H. Software Requirements Engineering. Los Alamitos, CA: IEEE Computer Society Press, 1997.

Haikala, I., Märijärvi J. 1998. Ohjelmisto tuotanto. 6. p. Helsinki: Suomen Atk-kustannus.

High Availability and Disaster Recovery Options for DB2 for Linux, UNIX, and Windows 2012. IBM: IBM Redbooks. Viitattu 10.9.2016. http://www.redbooks.ibm.com/abstracts/sg247363.html?Open

IBM. Viitattu 17.7.2016. http://www.ibm.com/fi-fi/

IBM Knowledge Center. Viitattu 8.10.2016. https://www.ibm.com/support/knowledgecenter/

Implementing Document Imaging and Capture Solutions with IBM Datacap. 2015. IBM: IBM Redbooks. Viitattu 11.9.2016. http://www.redbooks.ibm.com/abstracts/sg247969.html?Open

IIS. Viitattu 26.7.2016. http://www.iis.net/

I.R.I.S. S.A. Viitattu 26.9.2016. http://www.irislink.com/EN-FI/c983/IRIS---The-World-leader-in-OCR--PDF-and-Portable-scanner.aspx

Korpela, J. 2016. Ohjelmistosuunnittelija, Elinar Oy. Pori. Suullinen haastattelu 1.2.2016. Haastattelijana Antti Helander. Muistiinpanot haastattelijan hallussa.

Korpela, J. 2016. Ohjelmistosuunnittelija, Elinar Oy. Pori. Suullinen haastattelu 11.2.2016. Haastattelijana Antti Helander. Muistiinpanot haastattelijan hallussa.

LEAD Technologies Inc. Viitattu 9.5.2016. https://www.imaging-components.com/

Mellon, S. 2012. A Framework for Software Product Line Practice, Version 5.0. Viitattu 25.4.2016. http://www.sei.cmu.edu/productlines/frame_report/req_eng.htm

Minasi, M., Greene, K., Booth, C., Butler, R., McGabe, J., Panek, R., Rice, M. & Rotch, S. 2014. Mastering Windows Server 2012 R2. Viitattu 24.7.2016. https://raghavenv12.files.wordpress.com/2014/10/mastering-windows-server-2012-r2-by-sybex.pdf

Moonsoft Oy. Viitattu 25.7.2016. http://www.moonsoft.fi/index.aspx

Net-informations.com. Viitattu 26.7.2016. http://net-informations.com

Nuance Communications Inc. Viitattu 26.9.2016. http://www.nuance.com/index.htm

Olli, I. 2016. Järjestelmäsuunnittelija, Elinar Oy. Pori. Suullinen haastattelu 1.3.2016. Haastattelijana Antti Helander. Muistiinpanot haastattelijan hallussa.

Otey, M. 2013. Windows Server 2012: Implement Continuously Available File Shares. Viitattu 25.7.2016. http://windowsitpro.com/windows-server-2012/windows-server-2012-implement-continuously-available-file-shares

QuinStreet Inc. Viitattu 13.5.2016. http://www.webopedia.com/

Rouse, S. 2016a. Requirements analysis (requirements engineering). Viitattu 25.4.2016. http://searchsoftwarequality.techtarget.com/definition/requirements-analysis

Rouse, S. 2016b. VMware vSphere. Viitattu 25.7.2016. http://searchvmware.techtarget.com/definition/VMware-vSphere

Suntuubi. Viitattu 29.9.2016. http://hybridimenetelma.suntuubi.com/?cat=1

Sysilahti, P. 2016. Pääsuunnittelija, Elinar Oy. Pori. Suullinen haastattelu 1.3.2016. Haastattelijana Antti Helander. Muistiinpanot haastattelijan hallussa.

Sysilahti, P. 2016. Pääsuunnittelija, Elinar Oy. Pori. Suullinen haastattelu 21.2.2016. Haastattelijana Antti Helander. Muistiinpanot haastattelijan hallussa.

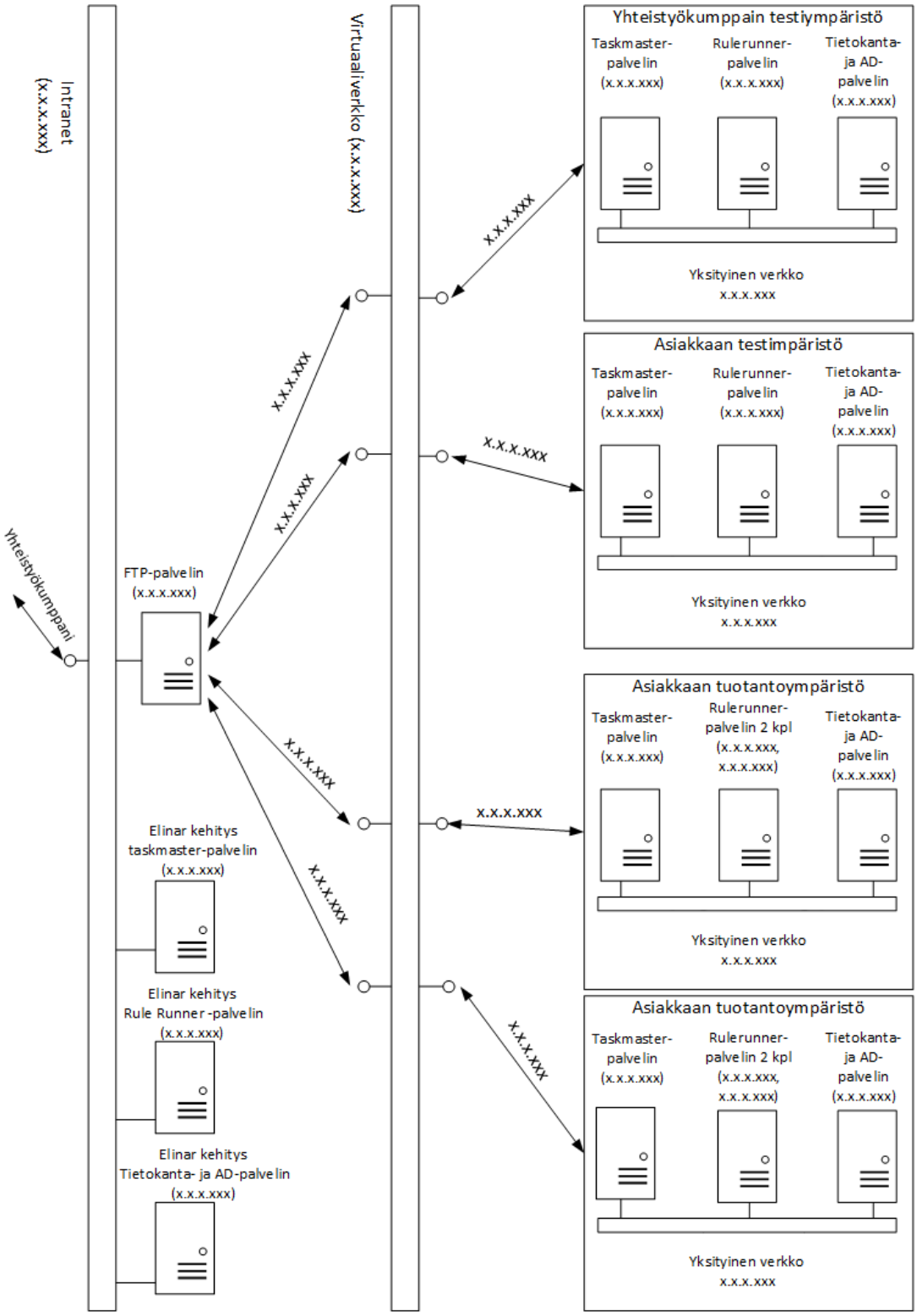
Sysilahti, P. 2016. Pääsuunnittelija, Elinar Oy. Pori. Suullinen haastattelu 17.7.2016. Haastattelijana Antti Helander. Muistiinpanot haastattelijan hallussa.

TopTenReviews. Viitattu 26.9.2016. http://www.toptenreviews.com/

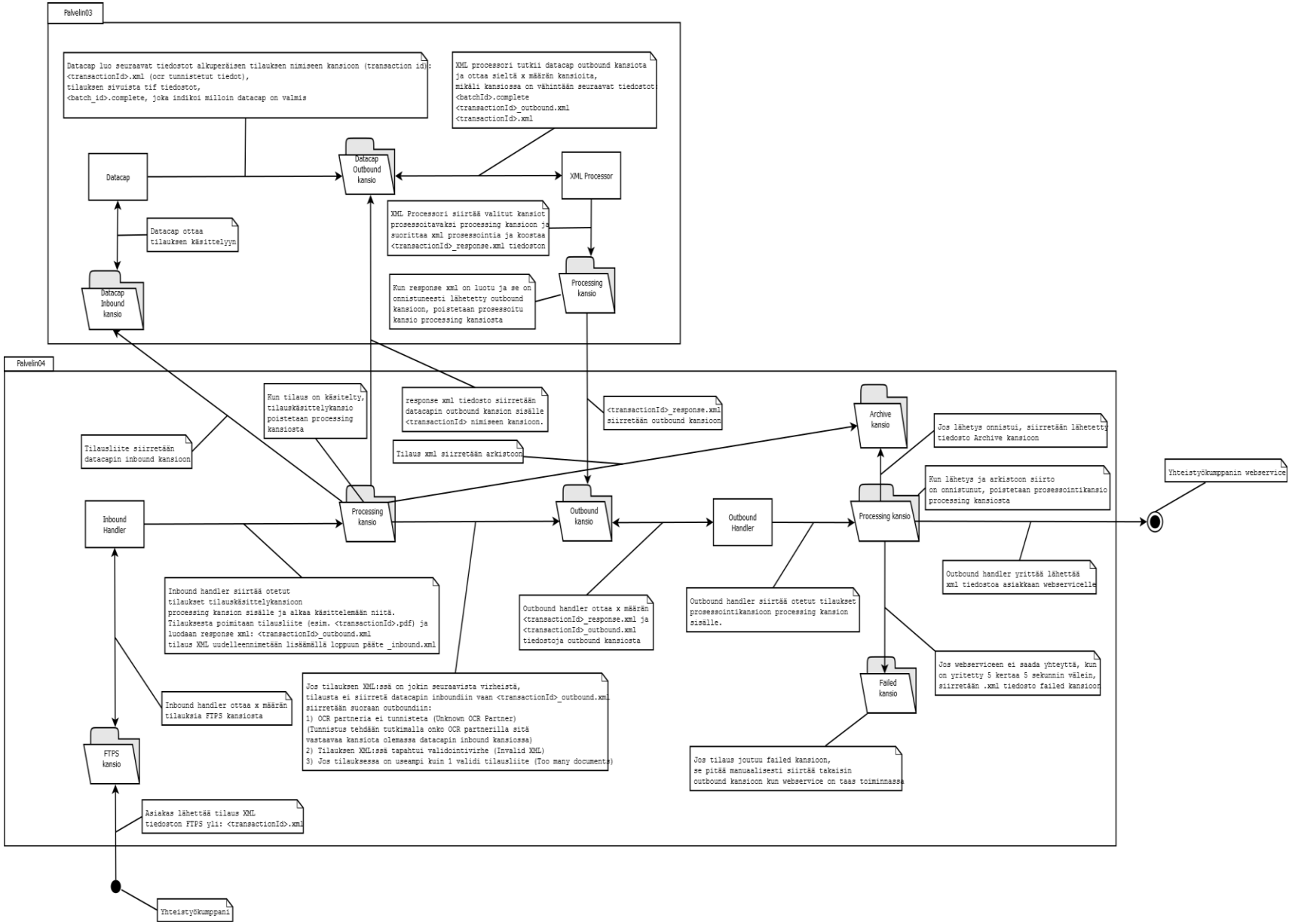
Tutorialspoint. Viitattu 26.7.2016. http://www.tutorialspoint.com/index.htm

Wkiess01. 2013. Linux Versus Windows File Server. Viitattu 25.7.2016. https://community.spiceworks.com/topic/368213-linux-versus-windows-file-server

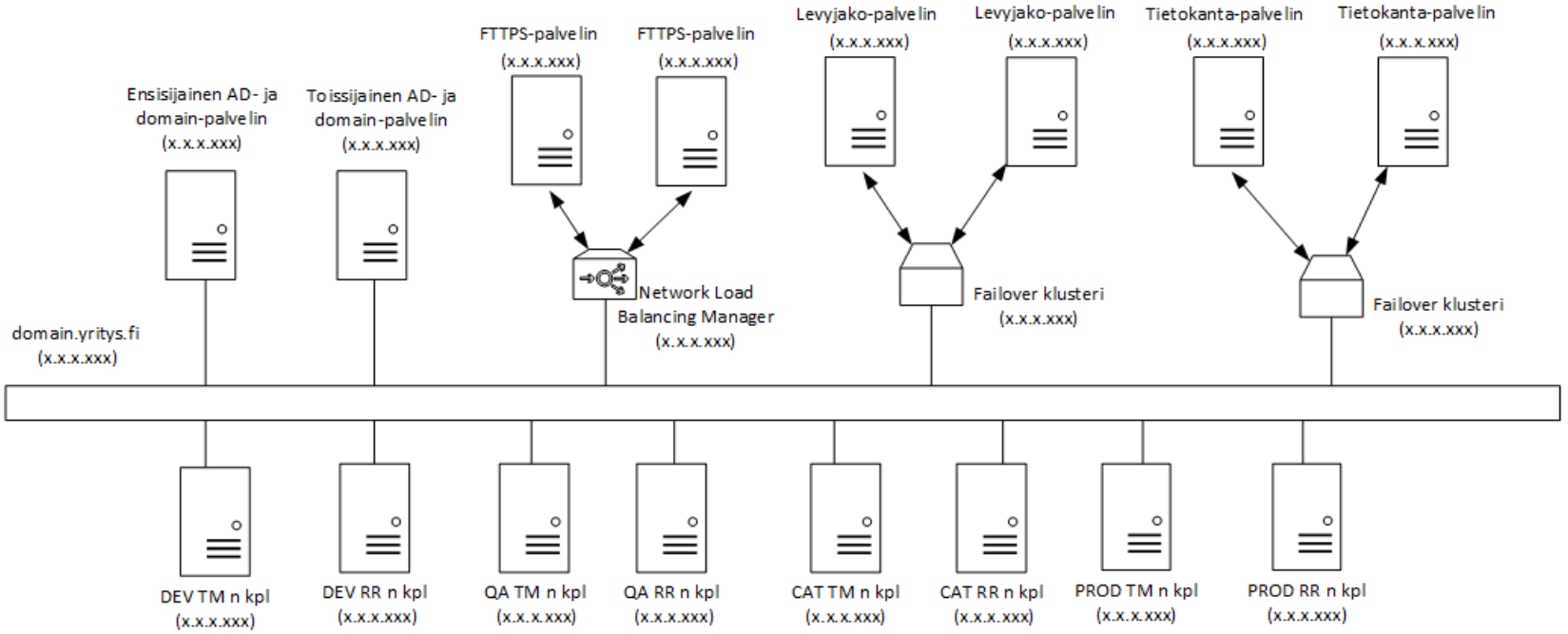
LIITE 1


Intranet
(x.x.x.xxx)
Virtuaaliverkko (x.x.x.xxx)
Yhteistyökumppani
FTP-palvelin
(x.x.x.xxx)
Elinar kehitys
taskmaster-palvelin
(x.x.x.xxx)
Elinar kehitys
Rule Runner -palvelin
(x.x.x.xxx)
Elinar kehitys
Tietokanta- ja AD-palvelin
(x.x.x.xxx)
x.x.x.xxx
x.x.x.xxx
x.x.x.xxx
x.x.x.xxx
x.x.x.xxx
x.x.x.xxx
x.x.x.xxx
Yhteistyökumppain testiympäristö
Taskmaster-palvelin
(x.x.x.xxx)
Rulerunner-palvelin
(x.x.x.xxx)
Tietokanta- ja AD-palvelin
(x.x.x.xxx)
Yksityinen verkko
x.x.x.xxx
Asiakkaan testimpäristö
Taskmaster-palvelin
(x.x.x.xxx)
Rulerunner-palvelin
(x.x.x.xxx)
Tietokanta- ja AD-palvelin
(x.x.x.xxx)
Yksityinen verkko
x.x.x.xxx
Asiakkaan tuotantoympäristö
Taskmaster-palvelin
(x.x.x.xxx)
Rulerunner-palvelin 2 kpl
(x.x.x.xxx,
x.x.x.xxx)
Tietokanta- ja AD-palvelin
(x.x.x.xxx)
Yksityinen verkko
x.x.x.xxx
Asiakkaan tuotantoympäristö
Taskmaster-palvelin
(x.x.x.xxx)
Rulerunner-palvelin 2 kpl
(x.x.x.xxx,
x.x.x.xxx)
Tietokanta- ja AD-palvelin
(x.x.x.xxx)
Yksityinen verkko
x.x.x.xxx

LIITE 2

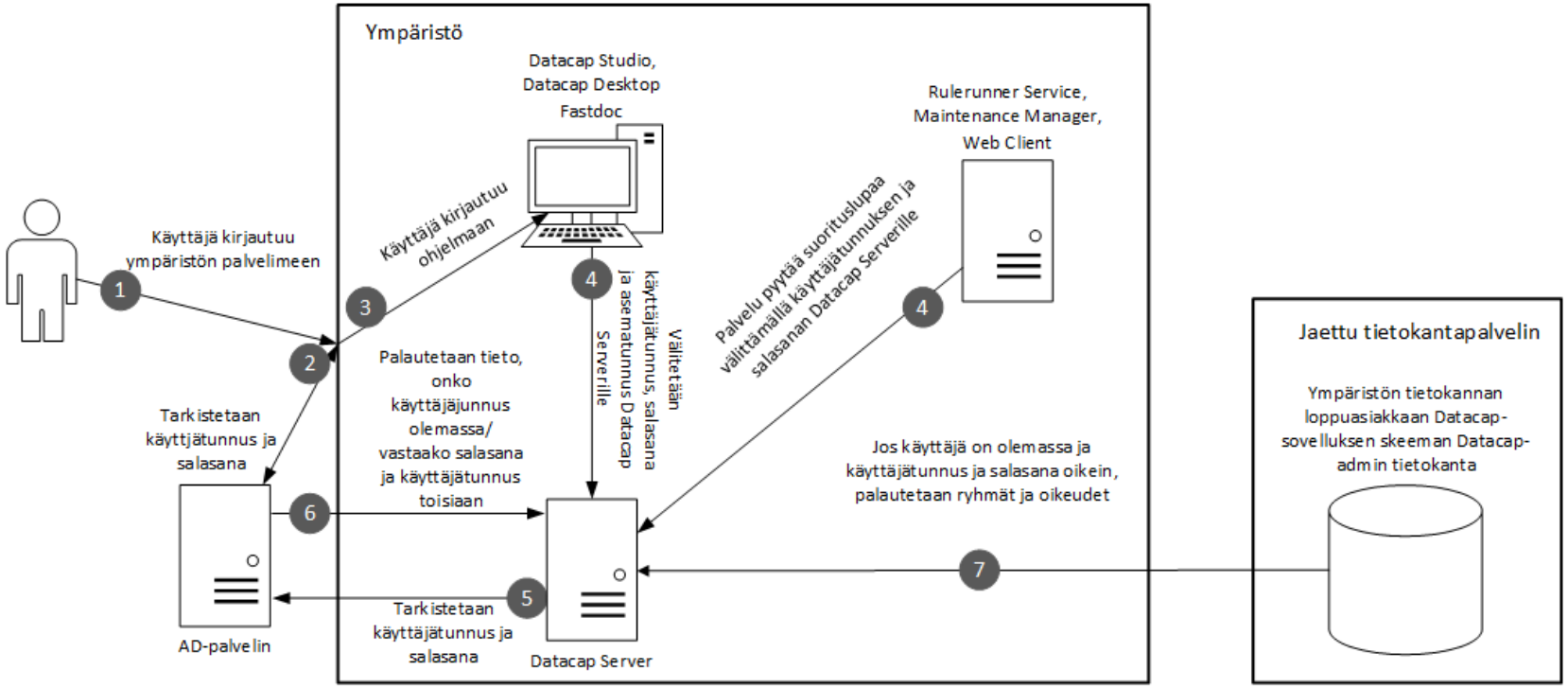

Palvelin03
Datacap luo seuraavat tiedostot alkuperäisen tilauksen nimiseen kansioon (transaction id):
<transactionId>.xml (ocr tunnistetut tiedot),
tilauksen sivuista tif tiedostot,
<batch_id>.complete, joka indikoi milloin datacap on valmis
XML processori tutkii datacap outbound kansiota
ja ottaa sieltä x määrän kansioita,
mikäli kansiossa on vähintään seuraavat tiedostot:
<batchId>.complete
<transactionId>_outbound.xml
<transactionId>.xml
Datacap
Datacap Outbound kansio
XML Processor
Datacap ottaa tilauksen käsittelyyn
XML Processori siirtää valitut kansiot prosessoitavaksi processing kansioon ja suorittaa xml prosessointia ja koostaa <transactionId>_response.xml tiedoston
Datacap Inbound kansio
Kun response xml on luotu ja se on onnistuneesti lähetetty outbound kansioon, poistetaan prosessoitu kansio processing kansiosta
Processing kansio
Palvelin04
Kun tilaus on käsitelty, tilauskäsittelykansio poistetaan processing kansiosta
response xml tiedosto siirretään datacapin outbound kansion sisälle <transactionId> nimiseen kansioon.
<transactionId>_response.xml siirretään outbound kansioon
Archive kansio
Jos lähetys onnistui, siirretään lähetetty tiedosto Archive kansioon
Tilausliite siirretään datacapin inbound kansioon
Tilaus xml siirretään arkistoon
Yhteistyökumppanin webservice
Kun lähetys ja arkistoon siirto on onnistunut, poistetaan prosessointikansio processing kansiosta
Inbound Handler
Processing kansio
Outbound kansio
Outbound Handler
Processing kansio
Outbound handler yrittää lähettää xml tiedostoa asiakkaan webservicelle
Inbound handler siirtää otetut tilaukset tilauskäsittelykansioon processing kansion sisälle ja alkaa käsittelemään niitä.
Tilauksesta poimitaan tilausliite (esim. <transactionId>.pdf) ja luodaan response xml: <transactionId>_outbound.xml
tilaus XML uudelleennimetään lisäämällä loppuun pääte _inbound.xml
Outbound handler ottaa x määrän <transactionId>_response.xml ja <transactionId>_outbound.xml tiedostoja outbound kansiosta
Outbound handler siirtää otetut tilaukset prosessointikansioon processing kansion sisälle.
Jos webserviceen ei saada yhteyttä, kun on yritetty 5 kertaa 5 sekunnin välein, siirretään .xml tiedosto failed kansioon
Failed kansio
Inbound handler ottaa x määrän tilauksia FTPS kansiosta
Jos tilauksen XML:ssä on jokin seuraavista virheistä, tilausta ei siirretä datacapin inboundiin vaan <transactionId>_outbound.xml siirretään suoraan outboundiin:
1) OCR partneria ei tunnisteta (Unknown OCR Partner)
(Tunnistus tehdään tutkimalla onko OCR partnerilla sitä vastaavaa kansiota olemassa datacapin inbound kansiossa)
2) Tilauksen XML:ssä tapahtui validointivirhe (Invalid XML)
3) Jos tilauksessa on useampi kuin 1 validi tilausliite (Too many documents)
FTPS kansio
Jos tilaus joutuu failed kansioon, se pitää manuaalisesti siirtää takaisin outbound kansioon kun webservice on taas toiminnassa
Asiakas lähettää tilaus XML tiedoston FTPS yli: <transactionId>.xml
Yhteistyökumppani

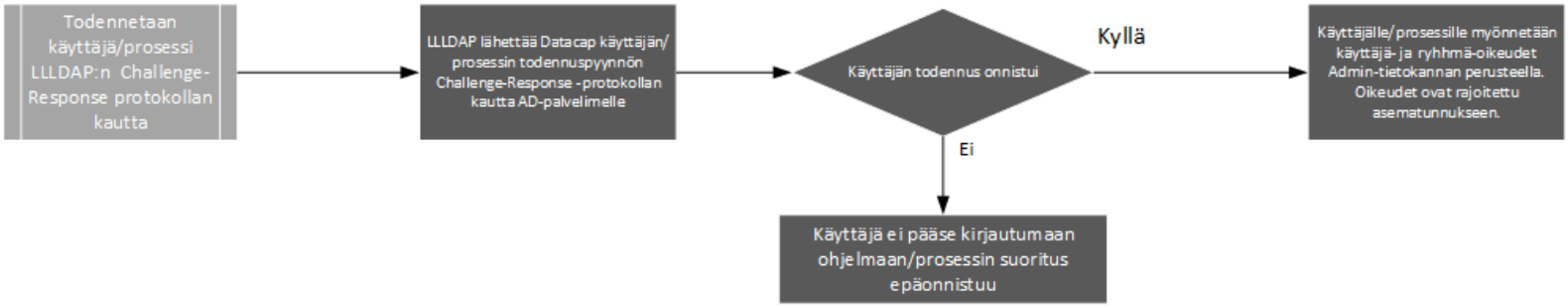
LIITE 3

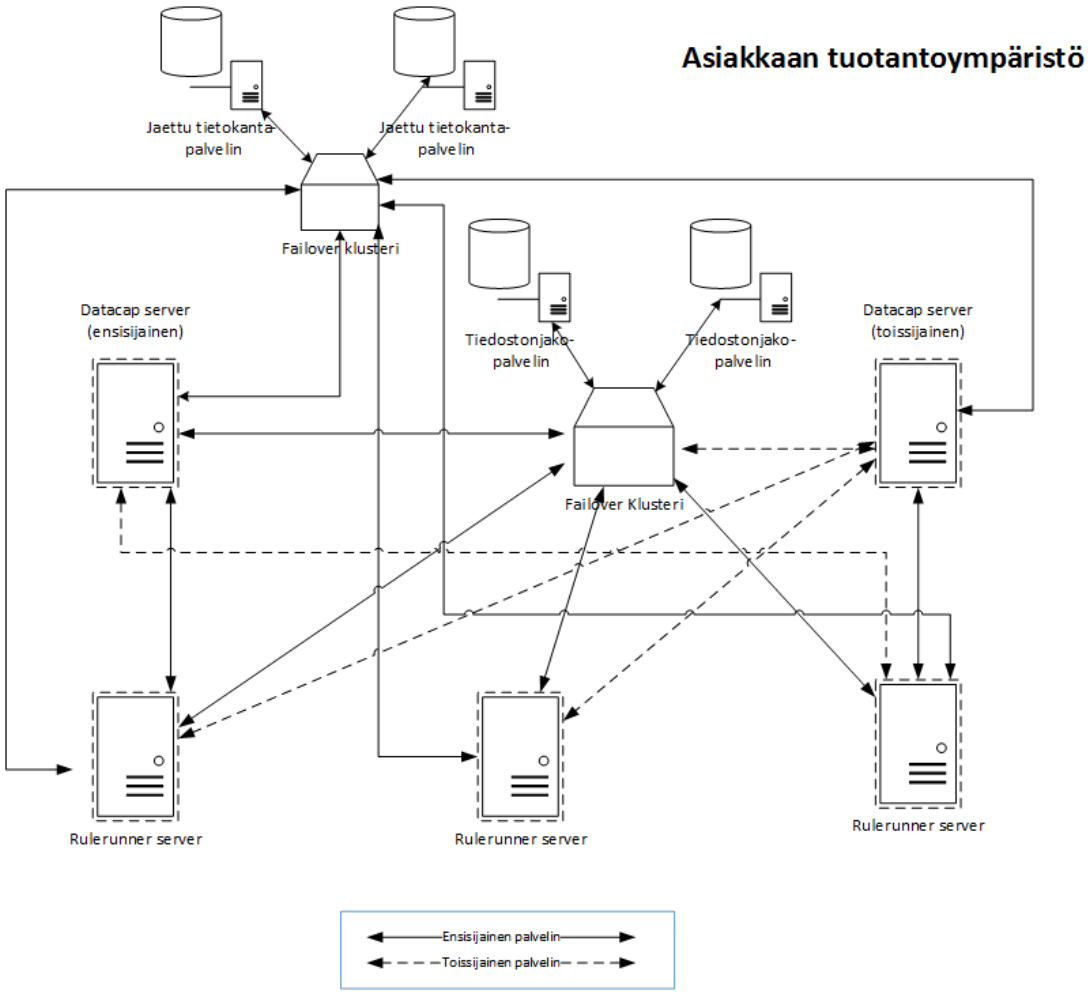


- DEV= yrityken kehitysympäristö
- QA = yhteistyökumppanin testiympäristö
- CAT = asiakkaan testiympäristö
- PROD = asiakkaan tuotantoympäristö
- TM = Taskmaster-palvelin
- RR = Rulerunner-palvelin

Liite 4


Ympäristö
Datacap Studio,
Datacap Desktop
Fastdoc
Rulerunner Service,
Maintenance Manager,
Web Client
Käyttäjä kirjautuu
ympäristön palvelimeen
Käyttäjä kirjautuu
ohjelmaan
Palvelu pyytää suorituslupaa
välittämällä käyttäjätunnuksen ja
salasanan Datacap Serverille
Välitetään
käyttäjätunnus, salasana
ja asematunnus Datacap
Serverille
Palautetaan tieto,
onko
käyttäjäjunnus
olemassa/
vastaako salasana
ja käyttäjätunnus
toisiaan
Tarkistetaan
käyttjätunnus ja
salasana
Jos käyttäjä on olemassa ja
käyttäjätunnus ja salasana oikein,
palautetaan ryhmät ja oikeudet
Tarkistetaan
käyttäjätunnus ja
salasana
AD-palvelin
Datacap Server
Jaettu tietokantapalvelin
Ympäristön tietokannan
loppuasiakkaan Datacap-
sovelluksen skeeman Datacap-
admin tietokanta
1
2
3
4
4
5
6
7


Todennetaan
käyttäjä/prosessi
LLLDAP:n Challenge-
Response protokollan
kautta
LLLDAP lähettää Datacap käyttäjän/
prosessin todennuspyynnön
Challenge-Response -protokollan
kautta AD-palvelimelle
Käyttäjän todennus onnistui
Kyllä
Käyttäjälle/prosessille myönnetään
käyttäjä- ja ryhhmä-oikeudet
Admin-tietokannan perusteella.
Oikeudet ovat rajoitettu
asematunnukseen.
Ei
Käyttäjä ei pääse kirjautumaan
ohjelmaan/prosessin suoritus
epäonnistuu


Asiakkaan tuotantoympäristö
Jaettu tietokanta-palvelin
Jaettu tietokanta-palvelin
Failover klusteri
Datacap server (ensisijainen)
Tiedostonjako-palvelin
Tiedostonjako-palvelin
Datacap server (toissijainen)
Failover Klusteri
Rulerunner server
Rulerunner server
Rulerunner server
Ensisijainen palvelin
Toissijainen palvelin

Yrityksen kehitys ja testaus, yhteistyökumppanin testaus ja asiakkaan hyväksyntäympäristö