TIETOKANNAN MIGRAATIO PILVIPALVELUUN

Ammattikorkeakoulututkinto

Tieto- ja viestintätekniikka, insinööri (AMK)

Kevät, 2017

Noora Pohjalainen

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Tieto- ja viestintätekniikka | Tiivistelmä
Tekijä Noora Pohjalainen | Vuosi 2023
Työn nimi Tietokannan migraatio pilvipalveluun
Ohjaaja Jari Mustajärvi

Opinnäytetyön tavoitteena on toteuttaa ohjelma, jolla mahdollistetaan vanhojen IBM Notes -tietokantojen arkistointi Sharepoint -sivulle. Sovelluksella kerätään tiedot tietokannasta asetustiedostojen avulla ja siirretään ne valitulle tiedonhallintajärjestelmälle käyttäen OneDrive -pilvitallennuspalvelua.

Työn lopputuloksena kehittyi C# -ohjelmointikielellä tuotettu sovellus, jossa käytetään Visual Studio -ohjelmaan sisäänrakennettua .NET Framework -alustaa, ja siihen kuuluvia luokkakirjastoja apuna. Sovelluksessa hyödynnetään SharePoint -kirjastoa, johon kerätyt tiedot ja tiedostot tietokannasta sijoitetaan

Sovelluksella voidaan mahdollistaa IBM Notes -tietokantojen siirto Sharepoint -sivustolle. Pienin haastein ja mahdollisen kehitystyön edetessä, saa sovelluksesta kehitettyä tehokkaamman ja yksinkertaisemman ratkaisun jatkon kannalta.

Avainsanat C#, SharePoint, .NET
Sivut 21 sivua

HAMK
HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Information and Communication Technology
Author Noora Pohjalainen
Subject Database migration to cloud service
Supervisors Jari Mustajärvi

Abstract
Year 2023

The goal of the thesis is to develop a program that provides a solution for archiving old IBM Notes databases to a SharePoint page. The application collects data from the database using configuration files and transfers them to the selected data management system by using the OneDrive cloud storage service.

The program was produced in C# programming language and Visual Studio program which includes the built-in .NET Framework platform that uses class libraries as an aid. From the database collected data and files will be placed in SharePoint library.

The application enables the transfer of IBM Notes databases to a SharePoint page. With small challenges and development work it is possible to develop more efficient and simpler solution.

Keywords C#, SharePoint, .NET
Pages 21 pages

Sisällys

1 Johdanto 1

2 Ohjelmat ja tekniikat 2

2.1 SharePoint 2

2.1.1 Tiedostokirjasto 2

2.1.2 Luettelot 3

2.2 OneDrive 3

2.3 IBM Notes 4

2.4 C# 4

2.5 .NET Framework 4

2.5.1 CLR 6

2.5.2 Luokkakirjastot 6

2.6 INI-tiedostotunniste 8

3 Sovellus 9

3.1 Sovelluksen toteutus 9

3.2 Sovelluksen toimintaperiaate 9

3.3 Alustus 10

3.4 Tiedonsiirto 11

3.5 Sovelluksen arviointi 17

4 Pohdinta 19

Lähteet 21

Kuvat, taulukot ja kaavat

Kuva 1. .NET Frameworkin toimintaperiaatteesta (Thomson, 2023).....6

Kuva 2. Kaksi tapaa, kuinka luokkia voidaan luoda C# -ohjelmointikielellä (Microsoft, 2022). 7

Kuva 3. Kokoonpanotiedosto 10

Kuva 4. CreateIndex -metodi, jolla tietokenttä tekstitiedosto luodaan.....12

Kuva 5. Metodi, jolla Sharepoint tietuekirjaston sarakkeet päivitetään tekstitiedoston mukaan. 14

Kuva 6. ExtractAttachments -metodi, jolla liitetiedostot viedään Sharepoint tietuekirjastoon vastaaviin kansioihin.....16

1 Johdanto

Työn tarkoituksena oli kehittää migraatiosovellus vanhojen tietokantojen siirtämiseen toiseen järjestelmään. Aikoinaan käytössä ollut IBM Notes on korvaantunut Microsoft Office 365 -tuotteella, jolloin IBM Notes -tietokannoissa säilytetyt vanhat lomakkeet ja sähköpostit ovat olleet vaikeasti saatavilla, eikä tuotteen tukemista jatketa enää. Vuoden 2022 loppuun mennessä järjestelmä poistetaan kokonaan HCL:n portaalista sekä lisenssijärjestelmästä. Suurin osa näistä tietokannoista halutaan pitää tallessa, jolloin migraatiosovellukselle on käytännön tarve. Tiedonsiirron kohteeksi oli valittu SharePoint -tiedonhallintajärjestelmä, yksi Microsoft Office 365 -tuotteista.

Opinnäytetyön tavoitteena on selonteko prosessista, miten migraatiosovelluksella luodaan onnistunut tietokannan siirto Sharepoint -sivustolle. Opinnäytetyössä selostetaan sovelluksen toimivuus ja joitain sen keskeisiä toimintatapoja.

Sovellus toteutettiin käyttämällä .NET ohjelmistokomponenttikirjastoa Microsoft Visual Studio -ympäristössä ja C# -ohjelmointikieltä. Työssä luodun sovelluksen avulla migraatio IBM Notes -kannasta kerätyt tiedot siirrettiin onnistuneesti SharePoint -alustalle.

2 Ohjelmat ja tekniikat

2.1 SharePoint

SharePoint on kaiken kokoisille yrityksille tarkoitettu selainpohjainen alusta, jolla pystytään yhdistämään yrityksen sisällön- ja dokumenttihallinta. Yleisin SharePoint -sivuston käyttötapa on tiedostojen tallentaminen, käsittely sekä jakaminen, mutta sitä käytetään myös intranet- ja ryhmätyösivustona jakamaan yritys kohtaisia uutisia ja päivityksiä. Se tarjoaa ajantasaisen tiedon, millä tahansa laitteella tai selaimella. Osana Microsoft 365 -tuoteperhettä SharePoint -sivuja voidaan yhdistää myös muita Microsoft Office 365 -tuotteita kuten Outlook ja Teams. (Microsoft, 2022)

SharePoint -käsitteellä voidaan tarkoittaa useampaa eri tuotetta. Microsoft SharePoint Online on pilvipalvelu, joiden sivuille voidaan yhdistää muita Microsoft 365 -tuotteita kuten Teams -tiimien sisällöt sekä tiedostokirjastoista voi käynnistää Power Automate -työnkulkuja. SharePoint Serverillä voidaan hyödyntää uusimmat nykyaikaiset ominaisuudet: esimerkiksi PowerApps ja Power BI integrointi, verkko-osat, luettelot, tiedostokirjastot sekä uudistetut sivustosivut. Vuonna 2013 viimeksi julkaistu SharePoint Designer 2013 on maksuton ohjelma, jota käytetään ulkoisten sisältötyyppien muokkaamiseen ja työnkulku ratkaisujen rakentamiseen. OneDrive synkronointi on työpöytäohjelma, joka mahdollistaa tiedostojen offline-käytön synkronoimalla SharePoint -sivusto paikalliselle tietokoneelle. Pääsääntöisesti OneDrive sisältyy kaikkiin Microsoft 365 -tuotteisiin, mutta se on myös mahdollista ladata ilmaiseksi. (Mircosoft, 2022)

2.1.1 Tiedostokirjasto

SharePoint -sivuston oletusarvoinen sivusto on tiedostokirjasto, joka luodaan automaattisesti sivunluonnin yhteydessä. Se tarjoaa turvallisen tallennuspaikan tiedostoille sekä yksinkertaisen tiedostojen hallinnoinnin. Tiedostojen siirtäminen onnistuu helposti myös kansioiden välillä. Tiedostokirjastot sisältävät luettelon tiedostoista, kansioista ja tiedoston avaintiedot, jota voi käyttää tiedostojen järjestämiseen ja löytämiseen.

Tarvittaessa sivustoon voidaan lisätä myös muita tiedostokirjastoja, jos halutaan rajata tiedostojoukon käyttöoikeuksia. (Microsoft, 2022)

Tiedostokirjastoihin on mahdollista lisätä tiedostojoukko -sisältötyyppi, joka yhdistää toisiinsa liittyvät tiedostot yhdeksi näkymäksi. Tiedostojoukon -sisältötyppiin voidaan määrittää metatietosarakkeet, sallitut sisältötyypit ja muokata tiedostojoukon aloitussivua. tai joilla yhdistetään joukko tiedostoja. Tiedostojoukko -sisältötyyppejä voi olla useampi samassa kirjastossa.

2.1.2 Luettelot

Toinen tapa tallentaa, jakaa ja käsitellä tietoa on Microsoft Luettelo. Se on kokoelma tietoja, joihin voi lisätä sarakkeita, jonka mukaan luodaan näkymiä, ryhmitellään tietoja, muotoillaan sekä korostetaan tärkeimmät tiedot. Se voi sisältää esimerkiksi kuvia, linkkejä, henkilöitä sekä päivämääriä. Luettelokohteeseen on myös mahdollista liittää liitetiedostoja. Luettelon avulla voidaan seurata esimerkiksi resursseja, yhteystietoja ja varastoja. (Microsoft, 2022)

2.2 OneDrive

OneDrive on Microsoftin tarjoama pilvitallennuspalvelu, jolla voi tallentaa ja varmuuskopioida omia henkilökohtaisia tiedostoja tai jakaa tiedostoja ja kansioita muille käyttäjille. Sitä voi käyttää myös varmuuskopiona henkilökohtaiselle tietokoneelle turvatakseen tärkeät tiedostot. Pilvipalvelu myös mahdollistaa tallennettujen tiedostojen offline-käytön, jos verkkoyhteys ei ole saatavilla. (Microsoft, 2022)

2.3 IBM Notes

IBM Notes, nykyään tunnetaan nimellä HCL Notes, oli ensimmäisiä kehitettyjä työryhmäohjelmistoja. Se oli pääsääntöisesti yrityksille tarkoitettu työpöytäsovelluksia sähköposti- ja kalenteriratkaisuihin, joka sisälsi sähköpostin, osoitekirjan, tehtävälistan, internet-selaimen sekä tietokannan. IBM Notes -sovellukset eli tietokannat toimivat IBM Domino palvelimelta, nykyisin HCL Domino, joka mahdollistaa reaaliaikaisen viestittelyn, tiedostojaon, ääni- ja videopuhelut sekä ryhmäviestit. Sovellus voi olla käyttäjän käytössä oleva paikallinen tai palvelimelle kaikille käytössä oleva, joita voidaan mahdollisesti rajoittaa erilaisilla käyttöoikeuksilla. Notes -tietokanta ei ole relaatiotietokanta, vaan koostuu tiedosto näkymistä. (Ravi, 2021)

2.4 C#

C# on moderni, olio-orientoitunut ja tyyppiturvallinen Microsoftin kehittämä ohjelmointikieli, joka alun perin suunniteltiin .NET ohjelmistokehitykseen. Pääasiassa ohjelmointikieltä käytetään työpöytä-, mobiili- ja web-sovellusten kehittämiseen. Sen ominaisuudet, kuten automaattinen muistinhallinta sekä kehittynyt virheiden hallinta, auttaa luomaan kestäviä ja luotettavia sovelluksia. (Coldberry, 2022)

C#-ohjelmointikielellä luotu ohjelma ajetaan .NET -ohjelmakehitys alustassa, jossa luotu koodi käännetään välikielelle Intermediate Language -muotoon. Suorituksen aikana välikieli latautuu CLR virtuaalikoneelle, joka kääntää ohjelmakoodista tehdyn välikielen natiiviin muotoon. Näin tehdään, jotta IL-koodi voi olla vuorovaikutuksessa useiden eri ohjelmointikielien kanssa, jolloin sitä ei tarvitse kääntää useiksi eri sovelluksiksi eri käyttöjärjestelmiä varten. (Thompson, 2023)

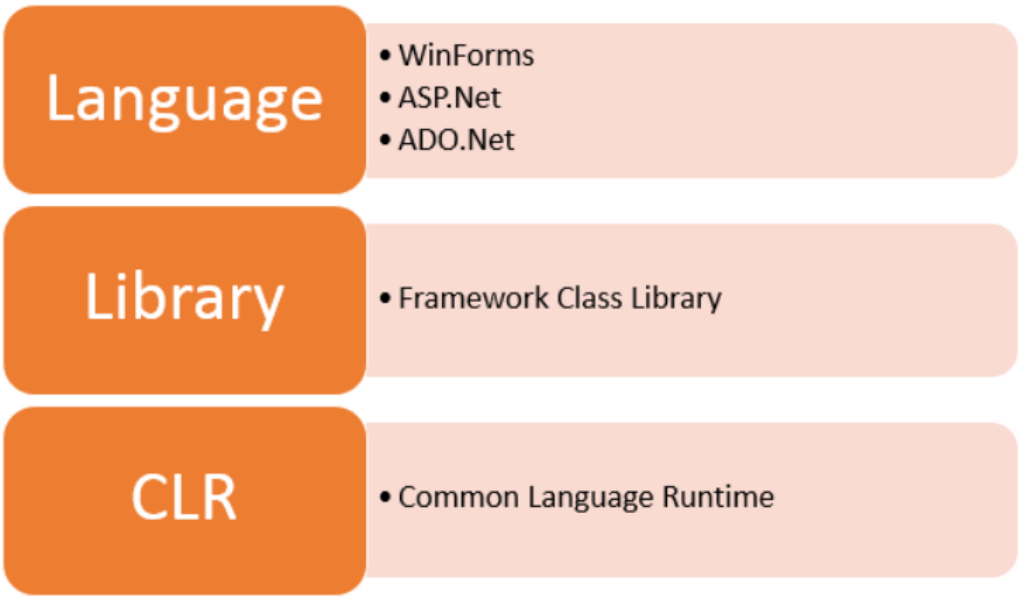
2.5 .NET Framework

.NET Framework on tekniikka, jolla tuetaan pääsääntöisesti Windows -sovellusten rakentamista ja käyttöä, joka on suunniteltu täyttämään seuraavat tavoitteet:

- Edistää koodin turvallista suorittamista
- Johdonmukaisen ja oliokeskeisen ohjelmointiympäristön
- Yhdenmukaistaa kehitystyötä
- Mahdollisuus integroida alusta minkä tahansa muun koodin kanssa. (Microsoft, 2023)

Kuva 1 kuvaa .NET Framework -alustan toimintaperiaatetta. Se on avoimen lähdekoodin kehitysalusta, joka koostuu CLR eli Common Language Runtime sekä luokkakirjastoista ja tukee useimpia käytettyjä ohjelmointikieliä. Alusta tarjoaa useita mahdollisuuksia ohjelmistokehityksessä kuten muistin hallintaa, tyyppiturvallisuutta ja kestävyyttä. Se on kehitetty yhteensopivaksi versioiden välillä, jolloin kehitystyössä ei ole merkitystä millä versiolla alkuperäinen ohjelmisto on kehitetty. (Microsoft, 2023)

Pääasiassa .NET Frameworkin ohjelmistokehitys tehdään Microsoft Visual Studio -ympäristössä. Se on mahdollista ladata omana pakettina Microsoftin omilta sivuilta tai käyttää Visual Studio -sovelluksen sisäänrakennettua .NET Framework -alustaa.



Kuva 1. .NET Frameworkin toimintaperiaatteesta (Thomson, 2023).

2.5.1 CLR

CLR eli Common Language Runtime tarkoittaa ajonaikaista ympäristöä, joka on .NET -alustalla ajettava virtuaalinen ajo-ohjelma, joka lataa ja suorittaa ohjelman. Se muuttaa ohjelmakoodin natiiviksi koodiksi, jolloin se toimii rajapintana alustan ja käyttöjärjestelmän välillä.

Se hallitsee muistia, turvallisuutta sekä kokoamista. Ajettavaa ohjelmakoodia kutsutaan hallittavaksi koodiksi eli toisin sanoen CLR tarjoaa hallitun ohjelmanajo-ohjelman .NET -kehitysalustalle kehittäen sen tietoturvallisuutta sekä integraation useiden ohjelmointikielien kanssa. (Microsoft, 2023)

2.5.2 Luokkakirjastot

.NET Framework sisältää useita eri luokkakirjastoja, jotka ovat kokoelma metodeista ja funktioista. Ne ovat oliokeskeisiä luokkakirjastoja, joka mahdollistaa yleiset ohjelmistokehitykseen tarvittavat tavoitteet, kuten tiedonkeruu, tiedostojen hallinta sekä tietokantayhteydet. Luokkakirjastot sisältävät myös erilaisia tyyppejä, joita voidaan käyttää

sovellusten ja palveluiden kehittämiseen kuten esimerkiksi konsoli -sovellusten, Windows lomakkeiden sekä verkkosovellusten. Esimerkiksi on olemassa luokkakirjasto, jolla voidaan käsitellä tiedostoon kirjoittamista ja lukemista. (Microsoft, 2023)

C# Copy

```
System.Console.WriteLine("Hello World!");
```

C# Copy

```
using System;
```

C# Copy

```
Console.WriteLine("Hello World!");
```

Kuva 2. Kaksi tapaa, kuinka luokkia voidaan luoda C# -ohjelmointikielellä (Microsoft, 2022).

Kuten kuvassa 2 voi nähdä, kirjastoja voidaan käyttää kahdella tapaa: .NET -alusta käyttää nimiavaruuksia hallinnoimaan luokkia käyttämällä System. etuliitettä metodin edessä tai käyttää sitä päätason direktiivinä, jolloin kirjaston etuliitettä ei tarvitse enää mainita metodissa.

Useimmat metodit esimerkiksi WinFormsien kehitykseen käytettävässä Visual Studio -sovelluksessa voidaan lisätä kirjastojen määrää valitsemalla ’References’ ja lisäämällä uusi luokkakirjasto. Yleisin käsitys luokkakirjastoita on .dll -tiedostopääte

Tässä työssä on käytetty Visual Studio -sovelluksen sisäänrakennettua .NET alustaa ja luokkakirjastoja. Se toteutettiin C# -ohjelmointikielellä, Microsoft 365 -tuotteilla ja .NET -ohjelmistokehityksen tekniikoin.

Työssä käsiteltävä sovellus toteutettiin C#-ohjelmointikielellä ja suurelta osin Microsoft 365 -tuotteilla ja .NET -ohjelmistokehityksen tekniikoin.

2.6 INI-tiedostotunniste

INI-tiedostotunniste on kokoonpanotiedosto eli alustustiedosto, jota käytetään asetusten asettamiseen Windows -käyttöjärjestelmiin. Se sisältää asetukset ja niiden osiot, joissa asetuksen osiot määritellään merkkijonoilla. Tiedostot ovat ainoastaan avattavissa tekstieditoreilla kuten Microsoft NotePad tai Microsoft WordPad. (ReviverSoft, 2023)

3 Sovellus

3.1 Sovelluksen toteutus

Sovellus toteutettiin Microsoft Visual Studio -sovelluksella .NET -ohjelmistokehitysalustaa käyttäen. Tiedonsiirron jokainen vaihe on eritelty omiin osioihinsa: kansion luominen, tietokenttien päivitys, liitteiden lisääminen.

Koko sovelluksen toiminnallisuus perustuu C#-ohjelmointikielen ja .NET -ohjelmistokehitysalustan yhteistyöhön. Visuaalinen ulkoasu tehtiin Windows Forms UI -teknologialla, jossa käytettiin teknologian tarjoamia komponentteja. Ohjelmakoodilla käytetään .NET -ohjelmistokehityksen tarjoamia koottuja kirjastoja, joilla sovelluksen eri toiminnot suoritetaan.

3.2 Sovelluksen toimintaperiaate

Migraatiosovelluksen kehittäminen kokonaisen tietokannan migraatioon on suuri prosessi, johon vaaditaan monta vaihetta. Kehitystyön alussa oli tärkeää selvittää, mitkä tietokentät halutaan IBM Notes -tietokannasta säilyttää ja mitä jätetään pois. Tiedonsiirtoon ei haluta sisällyttää tietoja, millä ei ole tarvetta. Nämä käydään läpi tarkastelemalla tietokannan lomakkeita ja niiden ominaisuuksia, kuten mitkä tietokentät ovat olemassa sekä mitkä valitaan säilytettäväksi. Tarvittavat tietokentät räätälöidään asiakkaan pyynnön mukaan.

Työssä käytettävään tietokantaan sisältyy kahdenlaisia eri lomakkeita, jotka siirretään Sharepoint -dokumenttikirjastoon kehitettyä sovellusta käyttäen. Tietokannassa on kahdenlaisia lomakkeita, tapaukset ja muistiinpanot. Työn tarkoituksena on yhdistää liitetyt tapaukset ja muistiinpanot keskenään yhtenäiseksi kansioksi dokumenttikirjastossa, jolloin jokainen kansio kuvaa yhtä tapausta sisältäen muistiinpanot.

3.3 Alustus

Ennen ohjelman käynnistämistä luodaan INI-tiedosto, joka on kokoonpanotiedosto käytettävän ohjelman alustamiseen. Tiedostolla määritellään parametrit mistä ja mitkä tiedot IBM Notes -tietokannasta halutaan tuoda. Tiedostoon määritellään Sharepoint -sivun nimi, kansio ja osoite sekä IBM Notes -tietokannan polku sekä nimi. Myös Sharepoint -dokumenttikirjaston nimi, tietokantaan asetetun näkymän nimi, tietokannan tietokentät ja niihin yhdistetyt sarakkeet Sharepoint -sivustosta. Jokainen lomakekenttä lisätään omalle riville tiedostoon ja jokainen kenttätyyppi määritellään joko teksti- tai päivämääräkentäksi.

```
DocSpace title¤
DocSpace folder¤
SharePoint site¤
SharePoint document¤
Notes server¤
Notes database¤
DocID view¤MigrationView
Files view¤MigrationFiles
Field¤Code:S:DocumentCode
Field¤Rev:S:Version
Field¤Date:D:VersionDate
Field¤Tree:S:TableOfContents
Field¤Title:S:Subject
```

Kuva 3. Kokoonpanotiedosto

Kokoonpanotiedoston mukaan lisätään määritellyt Sharepoint -sarakkeet lisäämällä ne manuaalisesti Sharepoint -tiedostokirjastoon. Kokoonpanotiedoston rakenne esitellään kuvassa 3.

Tiedostokirjastoon on ennen työnalkua luotu kaksi sisältötyyppiä, joilla tapaukset ja muistiinpanot erotellaan omiin tyyppeihinsä. Kahdella eri sisältötyypillä voidaan hallita lisättyjä sarakkeita sekä tallentaa tietoa dokumenttikirjastoon valittuun sisältötyyppiin.

3.4 Tiedonsiirto

Kokoonpanotiedoston ollessa valmis, on sovellus valmis yhdistettäväksi IBM Notes -tietokannan kanssa. Sovellus kirjautuu Sharepoint sivulle tunnuksilla, jotka on määritelty sovellukseen, jonka jälkeen se yhdistää IBM Notes -kantaan.

Ensimmäinen vaihe on kerätä tunnistetiedot (ID) kaikista tietokannassa sisältävistä lomakkeista omaan tekstitiedostoon siihen luodusta näkymästä. Kuvassa 4 voi nähdä minkä metodin sovellus käynnistää, jolla se luo tekstitiedoston kokoonpanotiedostossa valittuun polkuun ja kirjoittaa lomakkeiden tunnistetiedot omille riveille. Tunnistetiedon päätteeksi lisätään erotinmerkki sekä nelinumeroinen numerosarja, joka kertoo tulevan Sharepoint -dokumenttikirjaston kansion numeron. Jokaiselle tietokantaan tallennettuun lomakkeeseen kehittyy oma tunnistetieto ja se auttaa yhdistämään tietokannasta haetut tiedot myöhemmässä vaiheessa.

```
void CreateIndex()
        {
            Core.LogStart("Creating index");
            doc.First();
            if (StartPoint.Text != "")
                doc.GetByKey(StartPoint.Text);
            int i = 0;
            StreamWriter f = new StreamWriter(ProjectDir(indexfilename));
            string orig = "";
            string s = "";
            do
            {
                orig = " ";// subject;
                int n = Fields.GetCount();
                int x = 0;
                //x = 3;
                s = "";
                object[] o;
                while (x < n)// - 2)
                {
                    if (Fields.Value(x).StartsWith("="))
                    {
                        s += (doc.Evaluate(Fields.Value(x).Substring(1))[0]).ToString() + Core.csvsep;
                    }
                    else
                    {
                        o = Fields.GetValue(doc, x);
                        if (o == null)
                        {
                            s += "" + Core.csvsep;
                        }
                        else if (o.Length == 0)
                        {
                            s += "" + Core.csvsep;
                        }
                        else if (o[0] == null)
                        {
                            s += "" + Core.csvsep;
                        }
                        else if (Fields.GetType(x) == "N")
                        {
                            string oxsep = "";
                            foreach (object ox in o)
                            {
                                s += oxsep + Notes.GetName(ox.ToString());
                                oxsep = "; ";
                            }
                            s += Core.csvsep;
                        }
                        else
                        {
                            s += Notes.GetName(o[0].ToString()) + Core.csvsep;
                        }
                    }
                    x++;
                }
                f.WriteLine
                (
                    orig + Core.csvsep +
                    Core.GetId(doc.Id) + Core.csvsep +
                    s
                );
                doc.Next();
                UpdateProgressBar(i);
                i++;
            } while (doc.Valid);
            f.Flush();
            f.Close();
            Core.LogStop();
        }
```

Kuva 4. CreateIndex -metodi, jolla tietokenttä tekstitiedosto luodaan.

Kun tunnistetiedot lomakkeista on haettu, luodaan tekstitiedosto, johon on tuotu kaikki valitut tietokentät tietokannasta. Tietokenttä tekstitiedosto sisältää kaikkien valittujen

tietokenttien tiedot selkeytettynä. Lomakkeet usein täytetään käyttäjien toimesta manuaalisesti, jolloin se antaa mahdollisuuden virheille kuten esimerkiksi erikoismerkit, ylimääräiset välilyönnit ja monia muita. Kun sovellus kirjoittaa tekstitiedostoa, käy se läpi valitut kentät, poistaen samalla erikoismerkit. Ennen tietokenttien tietoja, tiedostoon on lisätty aikaisemmasta tekstitiedostosta luodut nelinumeroiset numerosarjat, jolla tunnistetaan, mihin tapaukseen tietokenttien tiedot kuuluvat.

Kun molemmat tekstitiedostot on luotu, tunnistetiedot sekä tietuekentät, luodaan kansiot tunnistetietojen perusteella nelinumeroisen numerosarjan mukaan. Kyseinen numerosarja on päätetty sovellusta tehdessä indeksoimaan suuren määrän kansoita. Kansiot luodaan hakemalla kokoonpanotiedoston mukaan määritellyn parametriksi määritellyn Sharepoint -tietuekirjaston mukaan. Sovellus hakee tietuekirjaston ja kirjoittaa määriteltyyn listaan tunnistetietojen perusteella tiedostokansiot tietuekirjastoon. Sovellus hakee kansioiden nimet rivi riviltä tekstitiedostosta ja luo kansiot niiden mukaan.

```
private void UpdateFolderFieldsButtonClick(object sender, EventArgs e)
        {
            Core.LogStart("Updating folder fields");
            StreamReader f = new StreamReader(ProjectDir(indexfilename));
            SharePoint.Clear();
            if (sharepointlibrary == "")
            {
                MessageBox.Show("SharePoint Library setting missing.");
                return;
            }
            SharePoint.GetList(sharepointlibrary);
            int i = 0;
            string subdir = SubDir.Text;
            if (StartPoint.Text != "")
                i = int.Parse(StartPoint.Text);
            for (int j = 0; j < i; j++) f.ReadLine();
            SetProgressBar(Core.idmap.Count, i);
            while (!f.EndOfStream)
            {
                string[] s = f.ReadLine().Split(Core.csvsep);
                if (CheckFolders.Checked)
                {
                    if (!Directory.Exists(subdir + SharePointDir(s[1])))
                    {
                        Core.LogLine(s[1]);
                    }
                }
                else
                {
                    Core.LogLine(s[1] + "," + s[2] + "," + s[3]);
                    SharePoint.GetFolder(subdir + s[1]);     // Id
                    for (int j = 0; j < Fields.fields.Count; j++)
                    {
                        if (s[j + 2].Trim() == "") { }
                        else
                            SharePoint.SetFolderFieldValue(Fields.Name(j), s[j + 2].Replace("¦", "\n"));  // Code
                    }
                    SharePoint.UpdateFolder();
                    if (i % 50 == 0)
                    {
                        Stopwatch x = new Stopwatch();
                        x.Start();
                        SharePoint.Execute();
                        x.Stop();
                        changeslabel.Text = x.Elapsed.ToString();
                        IdlePanel.BackColor = Color.FromArgb(0, 255, 64);
                        //Core.Draw(0);
                        IdlePanel.BackColor = Color.Red;
                    }
                }
                UpdateProgressBar(i++);
            }
            SharePoint.Execute();
            IdlePanel.BackColor = Color.FromArgb(0, 255, 64);
            f.Close();
            Core.LogStop();
        }
```

Kuva 5. Metodi, jolla Sharepoint tietuekirjaston sarakkeet päivitetään tekstitiedoston mukaan.

Kansioiden luonnin jälkeen päivitetään sarakkeet eli kansioiden tietuekentät. Kuvassa 5 näkyy, miten sovellus lukee tietuekenttä tiedostoa erottelemalla erotinmerkin mukaan mihin tietuekenttään tieto kuuluu Sharepoint kirjastossa. Ohjelma hakee tietuekirjaston kansio numero, joka yhdistetään tietuekenttä -tiedoston vastaavaan riviin. Aikaisemmin kokoonpanotiedostossa määriteltiin IBM Notes -tietuekentät sekä Sharepoint sarakkeet vastaamaan toisiaan.

Sarakkeiden päivityksen jälkeen tuodaan IBM Notes -tietokannasta liitetiedostot, jotka liitetään oikeisiin kansioihin tekstitiedostoissa olevien nelinumeroisten numerosarjojen avulla. Näitä varten on oma näkymä tehty tietokantaan, joille tuodaan oma asennustiedosto. Ohjelma ajaa metodin, jolla tuodaan listaus kaikista liitetiedostoista samalla tekniikalla kuin aikaisemmissa osuuksissa.

Ennen liitetiedostoiden vientiä Sharepointiin, synkronoidaan tietuekirjasto paikalliselle OneDrive levylle. Paikallisella levyllä voidaan tietuekirjastoa hallita Resurssienhallinnan kautta. Asennustiedostoon, jolla kaikki tekstitiedostot on luotu, muutetaan kohdejärjestelmänpolku kohdentamaan synkronoitua SharePoint tietuekirjastoa. Oletuksena polku näyttäisi tältä:

C:\Users\username\library

Liitetiedostot viedään tekstitiedoston sekä IBM Notes -tietokannassa olevan näkymän perusteella. Kuvassa 6 on ExctractAttachments -metodi, jolla sovellus avaa IBM Notes -tietokannan liitetiedostonäkymästä ja aloittaa tiedostojen siirtämisen käyttäen tekstitiedostoa tunnistaakseen oikean kansion yhdistämällä tiedot toisessa tiedostossa olevan tunnistetyypin mukaan.

```
void ExtractAttachments(bool namesonly = false)
        {
            view.Open(filesview);
            if (namesonly)
                Core.LogStart("Extracting attachment names");
            else
                Core.LogStart("Extracting attachments");
            doc.First();
            if (StartPoint.Text != "")
                doc.GetByKey(StartPoint.Text);
            int i = 0, n = 0, m = 0, o = 0;
            string path = "";
            string filedir = "";
            Core.SortIdMapByDocId();
            while (doc.Valid)
            {
                path = Core.onedrive + Core.pathsep + SubDir.Text + Core.GetId(doc.Id);
                Core.LogLine(i + ": " + path);
                if (namesonly)
                {
                    n += doc.ExtractFileNames(path + Core.pathsep + filesfilename);
                }
                else
                {
                    filedir = path; // + Core.pathsep + "Files";
                    //Directory.CreateDirectory(filedir);
                    try
                    {
                        n += doc.ExtractFilesDirect(filedir);
                    }
                    catch (Exception e)
                    {
                        MessageBox.Show("DSM DocID: " + doc.Id + " " + e.Message);
                        try
                        {
                            //doc.Sign();
                        }
                        catch
                        {
                            Core.Log(Core.crlf + "Cannot process: " + doc.Id + Core.crlf);
                            o++;
                        }
                        //doc.Save();
                        m++;
                        //n += doc.ExtractFiles(filedir);
                    }
                }
                doc.Next();
                UpdateProgressBar(i);
                i++;
            }
            if (namesonly)
                Core.Log(n + " attachments names extracted. ");
            else
                Core.Log(n + " attachments extracted, " + m + " documents fixed, " + o + " skipped. ");
            Core.LogStop();
        }
```

Kuva 6. ExtractAttachments -metodi, jolla liitetiedostot viedään Sharepoint tietuekirjastoon vastaaviin kansioihin.

Kun liitetiedostot on lisätty kansioihin Sharepoint tietuekirjastossa ja OneDrive on päivittynyt ajan tasalle, voidaan Resurssienhallinnan kautta luoda tietuekirjastoon uusi kansio seuraavaa vaihetta varten. Uuden kansion luomiseen käytetään Komentokehotetta, koska IBM Notes -tietokannan suuren koon vuoksi toimintoa ei ollut mahdollista toteuttaa Resurssienhallinnan kautta.

Komentokehotteen avulla, siirtymällä Sharepoint tietuekirjaston OneDrive hakemistoon ja luomalla uusi hakemisto. Kun uusi hakemisto on luotu, siirretään luodut kansiot tietuekirjastosta hakemiston alle käyttäen alla olevaa komentoa:

```
for /d %n in (*) do move %n x
```

Komento siirtää kaikki kyseisessä hakemistossa olevat tiedostot ja kansiot uuteen kansioon, jonka jälkeen seuraavalle lomaketyypille tehdään samanlainen prosessi.

Kun molemmat lomaketyypit on lisätty Sharepoint tietuekirjastoon, yhdistetään nämä hakemalla luodusta uudesta hakemistosta ensimmäisen vaiheet tietuekirjaston kansiot toisen vaiheen kansioihin käyttämällä tietuekenttä tiedostoa.

3.5 Sovelluksen arviointi

Kehitetty sovellus tietokannan siirtoa varten toimi käyttötarkoitukseen ja tavoite saavutettiin, mutta vaatii vielä mahdollista kehitystä tietokirjastojen yhdistämiseen SharePointissa. C# ohjelmointikielellä kehittyi hyvin toimiva tiedon tuonti IBM Notes -tietokannasta, vaikka onkin osittain tehty hieman vanhentuneella tekniikalla, kuten INI-tiedostojen käyttämisellä asetusten alustamiseen.

Tällä hetkellä sovellus toimii parhaiten tietokantoihin, joka ei sisällä sisäisiä tiedostoja kuten esimerkiksi muistiinpanoja ja tapahtumia, eli sisältäisi vain yhden tason sen sijaan, että yhdistettäisiin kahta eri tasoa. Pääsääntöisesti IBM Notes -kannat ovat olleet sähköposti- ja työnkulkuhallinnan sovelluksena, jolloin eritasoisia tietokantoja on paljon. Suurin osa tietokannoista olivat myös mukautettuja, mikä vaikeutti sovelluksen käyttämistä muissa tietokannoissa.

Sovelluksen yksi kehitystapa olisi luoda kansiot jokaiselle eri lomakkeelle, jolloin vältytään OneDriven käytöltä kokonaan. Kirjaston synkronointi paikalliselle levylle on tehokkaampaa kuin paikalliselta levyltä Sharepoint -sivulle, mikä hidastaa tiedonsiirtoa huomattavasti.

On olemassa monia valmiita ratkaisuja samantyyppiseen tiedonsiirtoon, kuten itselle tuttu Sofor. Heidän työtapansa oli huomattavasti sujuvampaa ja nopeampaa kuin kehitetyllä sovelluksella, mutta kyseinen tapa erotteli tietokannassa olevat lomakkeet omille Sharepoint listoille.

Sovelluksen avulla kehitettiin sivusto, joka toimii arkistona lopetettujen IBM Notes tietokantojen sijaan. Hakutoiminto ja käytettävyys on huomattavasti selkeämpi käyttäjille sekä mahdollisuus mukautettuihin käyttöoikeuksiin ja ulkonäkömuutoksiin on mahdollista.

4 Pohdinta

Opinnäytetyön tavoitteena oli toteuttaa sovellus, jolla pystytään hakemaan tietoa IBM Notes -tietokannasta ja siirtää haettu tieto toiseen järjestelmään. Sen tarkoituksena on säilyttää mahdollisimman paljon tietoa tietokannasta ja siirtää tieto käytettäväksi Sharepoint -alustalle. Sharepoint -sivuston tarkoitus oli korvata aikaisempi IBM Notes -tietokanta sisältöineen.

Itse tietokannan korvaaminen onnistui kokonaisuudessaan ja tuotokseen oltiin tyytyväisiä. Lopputulokseen ei tosin päästy vain kehitetyn sovelluksen kanssa vaan, jouduttiin rinnalle tekemään erinäisiä muita sovelluksia tukemaan pääsovellusta tai manuaalisesti muuttamaan asetustiedostoja sopivaksi.

Tiedonsiirto IBM Notes -tietokannasta Sharepoint -tietuekirjastoon onnistui monen tietokannan kohdalla, mutta itsessään sovellus ei ole täydellinen. Työn kehitys oli jatkuvaa sen edetessä ja osa ratkaisuista mitä matkanvarrella kehitettiin, päätyi lopulliseen sovellukseen. Silti uusien tietokantojen tiedonsiirrossa jouduttiin kehittämään paljon uusia ratkaisuja työvaiheiden parantamiseksi.

Esimerkiksi kun tietokantojen koko kasvoi yli 50 gigatavua, ei kansioita voitu enää siirtää OneDrive -pilvipalvelua käyttämällä. Synkronointi paikallisen levyn ja OneDriven välillä ei ollut enää tehokasta, kun tiedonsiirtoon saattoi mennä useita päiviä. Jotta nämä ongelmat vältettiin, kehitettiin toisia sovelluksia tukemaan työssä kuvailtua sovellusta. Isoimpien tietokantojen kohdalla löydettiin ratkaisu kansioiden siirtoon siirtämällä ne suoraan Sharepointin puolella OneDriven ja Komentokehotteen sijaan.

IBM Notes -järjestelmä käsitteenä ei ole ollut tuttu ennen tätä aihetta, koska ei sitä juuri enää käytetä, eikä järjestelmää tueta miltään osalta enää. IBM Notes on pääosin lopetettu, joka teki aiheeseen perehtymisen hankalaksi muulta osin kuin, mitä työtä tehdessä opin. Suurin osa löydetyistä lähteistä kertoi saman asian, minkä olen teoriaosuuteen kirjoittanut. Vaikka sovelluksen teossa pääsi näkemään, miltä IBM Notes -kannat näyttävät on vaikeaa syventyä aiheeseen muuten.

Opinnäytetyön aikana olen oppinut paljon uusia tekniikoita ja mahdollisuuksia esimerkiksi SharePoint -alustan kanssa, mitä en ollut aikaisemmin tiennyt. Tiedon jakaminen ja sen rajaaminen Sharepoint sivustoilla on aika lailla rajatonta. Uusien tekniikoiden kuten Power Automate ja erilaiset integraatio mahdollisuudet Sharepoint sivuilla antaa mahdollisuudet vielä parempaan kehitykseen.

Vaikka Visual Studio ja C# -ohjelmointikieli on ollut tuttu entuudestaan, sain paljon uutta informaatiota niiden käytöstä sovelluskehitykseen. Esimerkiksi sovelluksessa hyödynnetyt kirjastot olivat aivan uusia asioita, ja miten paljon on mahdollisuuksia yhdistää erinäisiä järjestelmiä keskenään näiden avulla. Valittu kieli toteutettuun sovellukseen oli C# ohjelmointikieli, jonka ominaisuuksista oli hyötyä eri toimintojen tekemiseen. Lisättävät luokkakirjat SharePoint ja IBM Notes -kehitykseen olivat suuri apu sovelluksen toteutuksessa.

Lähteet

Barbara Thompson. (06.05.2023) What is .NET Framework? Explain Architecture & Components [kuva 1] https://www.guru99.com/net-framework.html

Codeberry.(2022) C#-ohjelmoinnin perusteet: Aloittelijan opas C#-ohjelmointikieleen. https://codeberryschool.com/blog/fi/c-sharp-ohjelmoinnin-perusteet/

Microsoft (29.9.2022) [kuva 2] A tour of C# https://learn.microsoft.com/fi-fi/dotnet/csharp/tour-of-csharp/#net-architecture

Microsoft (30.03.2023) Get started with .NET Framework https://learn.microsoft.com/en-us/dotnet/framework/get-started/overview

Microsoft. (2022) Mikä on Sharepoint?https://support.microsoft.com/fi-fi/office/mik%C3%A4-on-sharepoint-97b915e6-651b-43b2-827d-fb25777f446f

Microsoft. (2022) Mikä tiedostokirjasto on? https://support.microsoft.com/fi-fi/office/mik%C3%A4-tiedostokirjasto-on-3b5976dd-65cf-4c9e-bf5a-713c10ca2872

Microsoft. (2022) Luettelon luominen https://support.microsoft.com/fi-fi/office/luettelon-luominen-0d397414-d95f-41eb-addd-5e6eff41b083

Ravi. (2021) Complete History of IBM Lotus Notes to HCL Notes) https://www.stellarinfo.com/blog/complete-history-ibm-lotus-notes-hcl-notes/

ReviverSoft (2023) .INI Tiedostopääte https://reviversoft.com/fi/file-extensions/ini

sphpr-admin. (2021) Mikä se Sharepoint oikein on? https://m365hpr.fi/mika-se-sharepoint-oikein-on/