Julius Tostlebe

AINEISTOJEN JA METATIETOJEN PAKETOINTI PITKÄAIKAISSÄILYTYKSEEN APACHE NIFI -SOVELLUKSELLA

Opinnäytetyö

Liiketalouden ammattikorkeakoulututkinto

Tietojenkäsittelyn koulutus

2024

Kaakkois-Suomen ammattikorkeakoulu

Kaakkois-Suomen ammattikorkeakoulu

Tutkintonimike	Tradenomi (AMK)
Tekijä	Julius Tostlebe
Työn nimi	Aineistojen ja metatietojen paketointi pitkäaikaissäilytykseen Apache Nifi -sovelluksella
Toimeksiantaja	Disec Oy
Vuosi	2024
Sivut	30 sivua
Työn ohjaaja	Miia Liukkonen, Toni Säämänen

TIIVISTELMÄ

Tämän opinnäytetyön tavoitteena oli ottaa käyttöön ja integroida Apache Nifi -palvelinsovellus Tieteen tietotekniikan keskuksen tarjoamiin työkaluihin. Nämä työkalut mahdollistavat pitkäaikaissäilytykseen tarkoitettujen pakettien luonnin, ja yhdessä Apache Nifin kanssa ne automatisoivat tämän luontiprosessin. Työn toimeksiantajana toimi Disec Oy, jolla oli tarve kehittää automatisoitu prosessi näiden pakettien luomiseen.

Opinnäytetyön teoriaosuus tarjoaa kattavan yleiskatsauksen pitkäaikaissäilytyksen periaatteista keskittyen erityisesti digitaalisen datan säilyttämiseen ja tallentamiseen tavalla, joka takaa sen pysyvän ymmärrettävänä ja ehjänä pitkien aikavälien yli. Erityistä huomiota kiinnitetään PAS- palveluun ja METS-dokumenttiin, jotka ovat keskeisiä käsitteitä pitkäaikaissäilytyksessä.

Käytännön osassa tehdään paketoinnin automatisointia käyttäen Apache Nifiä, joka on yritystason hallintatyökalu datan keräämiseen, muuntamiseen ja reitittämiseen. Apache Nifi mahdollistaa monimutkaisten tietovirtojen hallinnan. Työssä kehitettiin Docker- pohjainen ratkaisu, joka yhdistää Nifin monipuoliset tietovirrat Tieteen tietotekniikan keskuksen työkalujen kanssa. Tämä mahdollisti datan käsittelyprosessien hallinnan ja automatisoinnin. Työssä käsitellään myös metadatan luontia, paketointia ja METS- dokumentin validointia, jotka ovat olennaisia prosesseja pitkäaikaissäilytyksen onnistumiseksi.

Projektin tuloksena kehitettiin kolme eriteltyä prosessiryhmää Apache Nifissä, jotka mahdollistavat pitkäaikaissäilytykseen soveltuvien siirtopakettien luomisen. Nämä prosessiryhmät tehostavat dataintegraatiota ja tarjoavat mallin, joka voidaan helposti siirtää ja soveltaa uusiin ympäristöihin Docker- konttien avulla. Tämä parantaa työn siirrettävyyttä ja skaalautuvuutta. Ratkaisun myötä Disec Oy pystyy nyt automatisoimaan ja tehostamaan pitkäaikaissäilytyksen prosessejaan, mikä vähentää manuaalisen työn tarvetta ja lisää toiminnan tehokkuutta.

Asiasanat: pitkäaikaissäilytys, Apache Nifi, dataintegraatio, automatisointi

South-Eastern Finland University of Applied Sciences

Degree title	Bachelor of Business Administration
Author	Julius Tostlebe
Thesis title	Packaging of materials and metadata for long-term preservation using Apache Nifi
Commissioned by	Disec Oy
Time	2024
Pages	30 pages
Supervisor	Miia Liukkonen, Toni Säämänen

ABSTRACT

The objective of this thesis was to implement and integrate the Apache Nifi server application with tools provided by the CSC – IT Center for Science. These tools facilitate the creation of packages intended for long-term preservation, and together with Apache Nifi, automate this creation process.

The theoretical part of the thesis provided a comprehensive overview of the principles of long-term preservation, focusing specifically on the preservation and storage of digital data by ensuring that data remains coherent and intact over long periods of time. Special attention were given to the PAS service and the METS document, which are key concepts in long-term preservation.

In the practical part, automation of packaging using Apache Nifi, an enterprise-level management tool for data collection, transformation and routing, was conducted. Apache Nifi enables the management of complex data flows. A Docker-based solution was developed, combining Nifi's versatile data flows with the tools from the CSC – IT Center for Science. This facilitated the management and automation of data processing tasks. The work also addressed the creation of metadata, packaging and validation of the METS document, which are essential processes for successful long-term preservation.

As a result of the project, three distinct process groups were developed in Apache Nifi, enabling the creation of transfer packages suitable for long-term preservation. These process groups enhance data integration and provide a model that can be transferred easily and applied in new environments using Docker containers, improving the portability and scalability of the work. This solution now enables Disec Oy to automate and enhance their long-term preservation processes, reducing the need for manual labor and increasing operational efficiency.

Keywords: long-term preservation, Apache Nifi, data integration, automation

SISÄLLYS

1 JOHDANTO

Tämän opinnäytetyön aiheena ja tavoitteena on toteuttaa tiedon virtausjärjestelmän käyttöönotto palvelinsovelluksena. Toimeksiantajalla on suuri tarve automatisoida aineiston tiedon virtauksia heidän kehittämäänsä arkistointipalvelua varten. Yhdellä monipuolisella palvelinsovelluksella voidaan yhdistää useita erilaisia tiedon virtausprosesseja, jotka voidaan suorittaa yhdeltä alustalta. Erilaista aineistoa on palvelinsovelluksen käyttöönoton myötä tarkoitus muuttaa palveluun sopivaksi muodoksi ja tuoda se rajapinnan kautta arkistointipalveluun.

Toimeksiantaja työlle on palvelutuotantoa harjoittava Disec Oy. Heidän Yksaniminen arkistointipalvelunsa on tuottanut jo yli 15 vuoden ajan muistinhallintaratkaisuja niin isoille kuin pienille yrityksille. Disecillä on asiakkaina monia eri organisaatioita, joilla kullakin on erilaiset tarpeet aineiston muokkaamiselle. Nykyisin aineistoon tehtävät muokkaukset ovat täysin asiakaskohtaisia ja ratkaisuiksi halutaan yleisesti hyödynnettäviä komponentteja, joita voisi käyttää. Mahdolliset aineistomuokkaukset eivät palvelinsovelluksen avulla vaatisi arkistointipalvelun erillistä päivittämistä ja ohjelmointia, vaan tiedot haettaisiin automaattisesti arkistointipalvelusta rajapinnan kautta ja käsiteltäisiin erillään palvelinsovelluksessa.

Koska Disecillä on paljon eri muotoista aineistoa ja toteutettava palvelinsovellus mahdollistaisi erittäin laajat tiedon virtausprosessit, keskitytään tässä opinnäytetyössä erityisesti digitaalisten aineistojen ja niiden metatietojen paketointiin pitkäaikaissäilytystä varten. Kehittämisongelmaksi näin ollen muodostuu, kuinka toteuttaa tiedon virtausta hallinnoiva palvelinsovellus, joka pystyisi muuttamaan ja paketoimaan dataa oikeaan muotoon pitkäaikaissäilytystä varten. Toteutuksesta rajataan pois automatisoitu tiedon tuonti ja vienti pois palvelinsovelluksesta ja keskitytään erityisesti tietojen paketointiin pitkäaikaissäilytystä varten.

2 PITKÄAIKAISSÄILYTYKSEN PAKETOINNIN PERIAATTEET

2.1 Pitkäaikaissäilytys

Pitkäaikaissäilytystä voidaan pitää tämän opinnäytetyön tärkeimpänä käsitteenä. Sillä tarkoitetaan digitaalisen datan säilyttämistä ja tallentamista tavalla, joka takaa sen pysyvän ymmärrettävänä ja ehjänä useiden kymmenien, jopa satojen vuosien ajan. Ajan myötä laitteiden, ohjelmistojen ja tiedostomuotojen vanhentuessa on tärkeää varmistaa, että data säilyy käyttökelpoisena ja yhteensopivana uudempien teknologioiden kanssa. Keskeisessä asemassa pitkäaikaissäilytyksessä ovat metatiedot, jotka kuvailevat aineiston historiaa, sisältöä, alkuperän ja tietoa siitä, kuinka dataa voidaan käyttää. (Opetus- ja kulttuuriministeriö 2024.)

Pitkäaikaissäilytys voidaan jakaa kolmeen tasoon: bittitason, loogiseen ja semanttiseen säilyttämiseen. Bittitason säilyttämisessä keskeistä on datan muuttumattomuuden varmistaminen. Se edellyttää jatkuvaa eheyden seurantaa vertailemalla datan kopioita toisiinsa, datan virheiden korjausta, tallennusteknologioiden päivittämistä tarpeen mukaan ja datan maantieteellistä hajauttamista. Tämä taso muodostaa pitkäaikaissäilytyksen perustan. (Kansalliskirjasto s.a.)

Loogisen tason säilyttämisessä keskitytään datan tulevaisuuden käyttökelpoisuuden varmistamiseen. Sitä voidaan toteuttaa esimerkiksi migraation avulla, eli muuttamalla data toiseen tiedostomuotoon, tai emulaation avulla, jossa alkuperäinen käyttöympäristö on toteutettu modernimmassa laitteistoympäristössä. Näillä keinoilla tiedostot pysyvät avattavina ja tulkittavina uusilla ohjelmistoilla myös tulevaisuudessa. (Kansalliskirjasto s.a.)

Semanttisella tasolla puolestaan pyritään säilyttämään aineiston ymmärrettävyys myös tulevaisuudessa. Tämä vaatii tuntemusta kohdeyhteisöstä ja aineiston sisällöstä. Esimerkiksi kielimuutokset voivat vaikuttaa aineiston saavutettavuuteen, kuten latinankielisten väitöskirjojen käännöstarve nykypäivänä osoittaa. (Kansalliskirjasto s.a.)

Nämä säilyttämisen tasot muodostavat kokonaisuuden, joka varmistaa digitaalisen datan pitkäaikaisen säilymisen ja hyödyntämisen eri teknologioilla pitkälle tulevaisuuteen. Pitkäaikaissäilytyksen tehokkuus ja toimivuus edellyttää, että aineisto on alusta alkaen tallennettu ja kuvailtu asianmukaisesti kansallisia standardeja noudattaen. (Kansalliskirjasto s.a.)

2.2 PAS- palvelu ja METS- dokumentti

Pitkäaikaissäilytystä toteutetaan PAS- palvelussa. Se tarjoaa kirjastoille, arkistoille ja museoille keskeisen kansallisen palvelun digitaalisten tietovarantojen pitkäaikaissäilyttämiseen. PAS- palvelun toteutuksesta vastaava CSC- Tieteen tietotekniikan keskus Oy tarjoaa avoimen lähdekoodin työkaluja pitkäaikaissäilytykseen tarvittavien pakettien luomiseen. (Opetus- ja kulttuuriministeriö 2024.)

PAS- palveluun tuodaan digitaalisia objekteja siirtopaketteina (SIP, engl. Submission Information Package), jotka on varustettu riittävillä metatiedoilla. Vain oikein muodostetut paketit voidaan prosessoida PAS- palveluun säilytyspaketeiksi (AIP, Archival Information Package), joista rakennetaan tarvittaessa jakelupaketteja (DIP, Dissemination Information Package) aineiston siirtämiseksi takaisin eri järjestelmiin. (Opetus- ja kulttuuriministeriö 2024.)

Siirtopaketti koostuu eri metatietoelementeistä, jotka kuvaavat ja määrittelevät aineistoa. Siirtopaketin on noudatettava METS- profiilia, joka on määritetty käytettäväksi PAS- palveluissa. METS- profiili tarjoaa yksityiskohtaisen teknisen kuvauksen METS- dokumentin (kuva 1) sisällöstä ja rakenteesta. Se tarkentaa kansainvälistä METS XML- skeemaa, eli kertoo millä tavalla METS- metatietostandardia sovelletaan kansallisissa PAS- palveluissa. Siirtopaketin sisältämien objektien on myös oltava yhteensopivia tässä profiilissa määriteltyjen tiedostomuotojen kanssa. (Opetus- ja kulttuuriministeriö 2024.)

Kuva 1. METS-dokumentin rakenne (Opetus- ja kulttuuriministeriö 2024)

METS- dokumentti on jaettu viiteen pääosaan: otsake, kuvaileva metatieto, hallinnollinen metatieto, rakenteellinen metatieto ja säilytettävät digitaaliset objektit. Otsake sisältää tietoja, kuten paketin luontipäivämäärän ja sen luoneen organisaation. Kuvailevassa metatiedossa kuvataan aineistoa yhdellä tai useammalla standardiformaatilla. Hallinnollinen metatieto tarjoaa teknisiä tietoja säilytettävästä aineistosta, käyttörajoituksia sekä tiedot aineiston alkuperästä. Se voi myös sisältää tietoja fyysisestä aineistosta, josta digitaalinen aineisto on peräisin. Rakenteellinen metatieto selittää, kuinka eri digitaaliset objektit liittyvät toisiinsa. Se voi esimerkiksi kuvata digitoidun kirjan sivujärjestystä. Tässä järjestyksessä jokainen sivu on oma tiedostonsa. Tiedosto- osio viimeistelee kokonaisuuden tarjoamalla viitteet itse säilytettäviin objekteihin ja niiden mahdollisiin eri ilmentymiin. (Opetus- ja kulttuuriministeriö 2024.)

3 PAKETOINNIN AUTOMATISOINNIN TYÖKALUT

3.1 Apache Nifi

Tieteen tietotekniikan keskuksen tarjoamat työkalut siirtopakettien luontiin ovat komentoriviltä ajettavia työkaluja. Jokainen työkalun eri osa, eli skripti, pitää erikseen itse ajaa komentoriviltä ja määritellä kyseisen skriptin parametrit. Tämän automatisointiin voidaan käyttää Apache Nifi- palvelinsovellusta (jäljempänä Nifi). Se on yritystason hallintatyökalu, jolla voidaan toteuttaa erilaisia tietovirtoja. Nifi auttaa keräämään, muuntamaan ja reitittämään dataa skaalautuvalla ja luotettavalla tavalla reaaliajassa. Nifin avulla voidaan tehdä dataintegraatio prosesseja sujuvampien tietovirtojen aikaansaamiseksi. Avoimen

lähdekoodin palvelinsovelluksena se on erittäin skaalautuva, turvallinen ja käyttäjäystävällinen alusta, jolla voi luoda erittäin monimutkaisia tietovirtoja. (Jain 2022.)

Nifin konsepti pohjautuu läheisesti Flow Based Programming- ohjelmoinnin pääideoihin. Flow Based Programming määrittelee prosessit verkostoiksi, jotka kommunikoivat keskenään tietopakettien avulla. Paketit liikkuvat määriteltyjä reittejä pitkin ja nämä reitit on määritelty prosessien ulkopuolella. Näin jokaista prosessia voi yhdistellä loputtomasti uusien verkostojen aikaansaamiseksi, jotka luovat niille uusia käyttötarkoituksia. (Morrison s.a.)

Nifissä FlowFile edustaa jokaista järjestelmän läpi kulkevaa objektia ja Nifi pitää kirjaa sen attribuuttijonoista sekä siihen liittyvästä sisällöstä. Prosessorit suorittavat varsinaisen työn eli prosessori suorittaa jonkinlaista datan reititystä, muunnosta tai välitystä järjestelmien välillä ja ne pääsevät käsiksi FlowFilen sisältöön. Yhteydet tarjoavat linkin prosessorien välillä. Nämä toimivat jonoina ja mahdollistavat eri prosessorien välisen vuorovaikutuksen. Näitä jonoja voidaan priorisoida dynaamisesti ja niille voidaan myös asettaa rajoja. Flow Controller ylläpitää tietoa siitä, miten prosessorit yhdistetään ja hallinnoi resursseja, joita kaikki prosessorit käyttävät. Flow Controller toimii välittäjänä, joka helpottaa FlowFile-tiedostojen vaihtoa prosessorien välillä. Prosessiryhmä on tietty joukko prosessoreja ja niiden yhteyksiä, jotka voivat vastaanottaa tietoa sisääntuloliittymien kautta ja lähettää tietoa ulostuloliittymien kautta. Tällä tavoin prosessiryhmät mahdollistavat täysin uusien komponenttien luomisen yksinkertaisesti muiden komponenttien yhdistelmällä. Omien prosessorien lisäksi Nifi pystyy myös ajamaan ulkoisia skriptejä prosessoreina osana tietovirtaa. (Apache Nifi 2024.)

Tietovirrat voivat olla monimutkaisia, mutta Apache Nifi mahdollistaa näiden virtojen visualisoinnin. Sillä voidaan rakentaa erilaisia tietovirtoja käyttäen graafiseen käyttöliittymään perustuvaa lähestymistapaa. Se auttaa vähentämään monimutkaisuutta ja mahdollistaa käyttäjien nähdä muutokset reaaliajassa. (Jain 2022.)

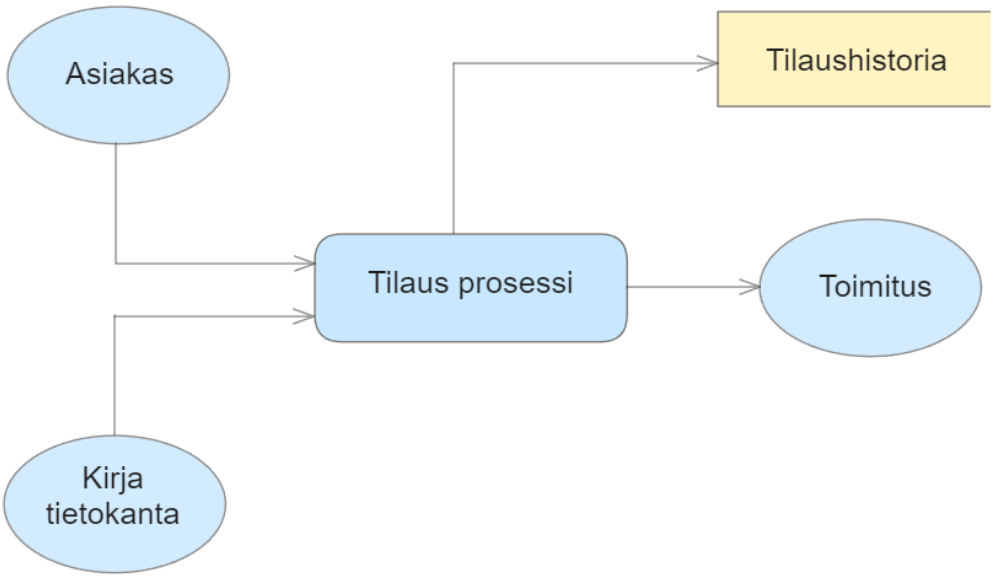
3.2 Tietovirrat

Tietovirtojen avulla tarkastellaan, kuinka data liikkuu komponentista toiseen tai järjestelmästä toiseen - toisin sanoen tietovirta on datan siirtoa lähteestä määränpäähän. Ne vaihtelevat tyypiltään ja monimutkaisuudeltaan riippuen niiden tarkoituksesta ja toiminnasta. Termiä tietovirta käytetään monissa yhteyksissä, mutta tässä sillä tarkoitetaan automatisoitua ja hallittua datan liikkumista järjestelmien välillä. (Apache Nifi 2024.)

Yksinkertainen tietovirta voi olla taulun peilaus lähteestä toiseen, joka sisältää datan tarkan kopioinnin lähteestä määränpäähän sekä arvojen että rakenteen osalta. Tällainen tietovirta ei vaadi tietokartoitusta tai- muunnoksia. Monimutkaisempi tietovirta voi sisältää datan käsittelyn useista eri lähteistä, joissa data yhdistetään, muunnetaan, suodatetaan ja jaetaan useisiin eri määränpäihin. (Modern analyst s.a.)

Tietovirtaa voidaan esittää tietovirtakaaviolla, joka näyttää miten tieto liikkuu prosessista toiseen ja tiedon suhteita järjestelmässä. Yksinkertaista tietovirtaa kirjan tilaamisesta voidaan kuvastaa kaaviolla (kuva 2). Siinä soikiot edustavat entiteettejä, jotka ovat tietojen lähteitä tai kohteita. Pyöristetyt suorakulmiot edustavat prosesseja, jotka ottavat tietoa sisääntulona, suorittavat toiminnon tiedoilla ja tuottavat tuloksen. Avonaiset suorakulmiot edustavat tietovarastoja ja nuolet tietovirtoja objektien välillä.

a



Kuva 2. Yksinkertainen tietovirta

Tietovirtojen tunnettuja haasteita ovat muun muassa järjestelmävikaantumiset, kapasiteetin ylittyminen, epäselvät rajat, muuttuvat tavoitteet, kehityserot ja turvallisuus. Vuosien saatossa tietovirrat ovat olleet yksi välttämättömistä pahoista järjestelmien arkkitehtuurissa. Nykyään kuitenkin on useita aktiivisia ja nopeasti kehittyviä tekniikoita, jotka tekevät tietovirtojen suunnittelusta mielenkiintoisempaa. (Apache Nifi 2024.)

3.3 Datamigraatio, dataintegraatio ja ETL- prosessi

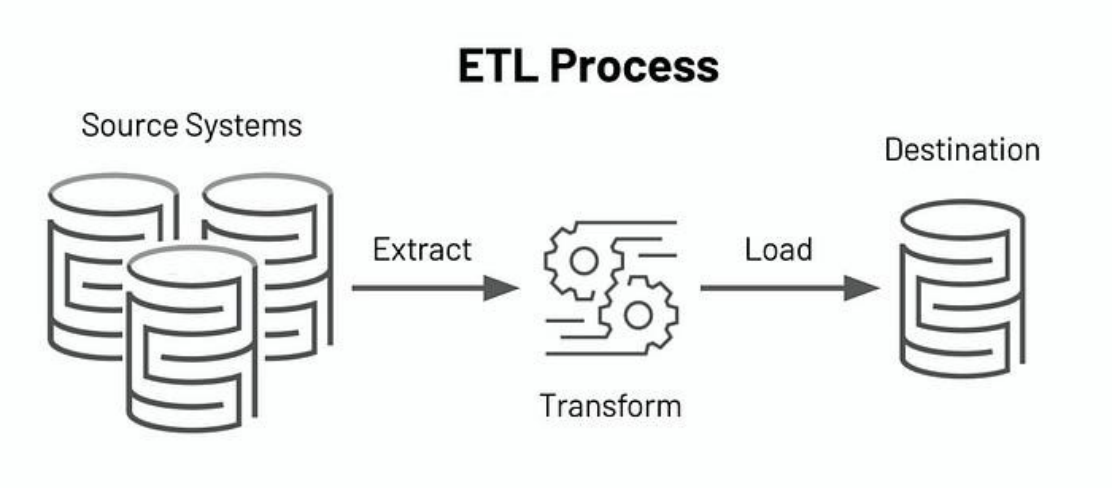
Tietovirrat kertovat, kuinka data liikkuu ja integroituu organisaation eri osa-alueiden välillä. Se myös sisältää yksityiskohtaisempia prosesseja, kuten datamigraation, dataintegraation ja ETL-prosessin. Datamigraatiolla tarkoitetaan kertaluontoista prosessia, jossa dataa siirretään vanhasta järjestelmästä uuteen pysyvästi. Dataa voidaan siirtää sovellusten, tallennusjärjestelmien, tietokantojen, datakeskusten ja liiketoimintaprosessien välillä. Siirto tapahtuu vain yhteen suuntaan ja sillä ei ole mahdollisuutta paluuseen. Koko prosessiin sisältyy datan valinta, valmistelu, muuntaminen ja siirtäminen siten, että siirretty data on oikeanlaatuista, oikeassa paikassa oikeaan aikaan. Jokaisen organisaation datan migraatiotavoitteet ja -prosessit ovat yksilöllisiä. Niiden on otet-

tava huomioon erilaisia tekijöitä, kuten kustannukset, aikataulu, tekniset vaatimukset, vaikutus liiketoimintaan, datan häviämisen mahdollisuus, vaatimustenmukaisuusvaatimukset ja paljon muuta. (Morris 2020; Cloudflare s.a.)

Termejä datamigraatio ja dataintegraatio saatetaan joskus käyttää vaihdellen, mutta ne tarkoittavat eri asioita. Datamigraation voidaan olettaa tapahtuvan kerran, kun taas dataintegraatio on jatkuva prosessi sisältäen asteittaisen datan muuttumisen. Se on prosessi, joka yhdistää tietoja monista lähteistä yhtenäiseen muotoon, joka puolestaan voi mahdollistaa tiedon hyödyntämisen päätöksenteossa ja analysoinnissa. Prosessi kattaa useita vaiheita ja niiden tarkoitus on varmistaa tiedon yhdenmukaisuus, tarkkuus ja saatavuus integraation aikana. Tämä saavutetaan käyttämällä työkaluja ja strategioita, jotka tukevat eri lähteistä peräisin olevan tiedon yhtenäistämistä. (Cloudflare s.a.; IBM s.a.)

Dataintegraatio tarjoaa useita etuja, jotka mahdollistavat organisaatioille paremmin perusteltujen päätösten tekemisen, toimintojen tehostamisen ja kilpailuedun saavuttamisen. Muita dataintegraation keskeisiä etuja ovat mm. parantunut datan laatu, tehokkuuden kasvu, tiedon hyödyntäminen. Dataintegraatiota käytetään laajasti eri toimialoilla vastaamaan erilaisiin tarpeisiin ja haasteisiin. Käyttökohteisiin lukeutuu muun muassa datan varastointi, datajärvien luonti ja BI-raporttien tekeminen. Datan varastoinnissa integraatiota käytetään luomaan keskitetty tietovarasto raportointia ja analytiikkaa varten. Datajärvet ovat suuria tietosäiliöjä, joissa voidaan säilöä valtavia määriä dataa sen alkuperäisessä muodossa. BI- raportit tarjoavat uusia näkemyksiä liiketoiminnan eri osa-alueista, mikä puolestaan auttaa ymmärtämään paremmin organisaation suorituskykyä ja suunnittelemaan strategisia toimia. (IBM s.a.)

On olemassa useita erilaisia datan integrointimenetelmiä, joilla kaikilla on omat vahvuutensa ja heikkoutensa. Integrointimenetelmän valinta riippuu niin organisaation tarpeista, suorituskykyvaatimuksista kuin budjetistakin. Yksi näistä menetelmistä on niin sanottu ETL- prosessi, jota kuvassa 3 on visualisoitu (extract, transform, load). (IBM s.a.)



Kuva 3. ETL-prosessi (Halder 2023)

ETL- prosessissa data poimitaan (extract) ensin sen alkuperäisestä lähteestä, muunnetaan (transform) sitten haluttuun muotoon ennen sen lataamista (load) datan säilytysjärjestelmään kuten tietokantaan. Muunnosvaiheessa dataa voidaan puhdistaa ja validoida, jotta se vastaa tarpeita. ETL- prosessi sopii erityisen hyvin tapauksiin, joissa datan laadun ja johdonmukaisuuden ylläpitäminen on tärkeää, sillä se mahdollistaa perusteellisen datan käsittelyn ennen sen tallentamista lopulliseen kohteeseen. (IBM s.a.)

4 PAKETOINTIPROSESSI

Toimeksiantajana tässä opinnäytetyössä toimii Disec Oy, joka tarjoaa sähköisiä tiedonhallintajärjestelmiä eri yrityksille. Eräs Disec Oy:n asiakkaista on ilmaissut tarpeen siirtää osan arkistoiduista tiedoistaan pitkäaikaissäilytykseen. Tämän projektin tarve oli paketoida pitkäaikaissäilytykseen soveltuvia paketteja sekä automatisoida tämä prosessi. Projektin tavoitteena on kehittää automatisoitu ratkaisu, jossa hyödynnetään Tieteen tietotekniikan keskuksen tarjoamia työkaluja yhdistettynä Apache Nifi- palvelinsovellukseen.

4.1 Apache Nifi Docker-kontissa

Palvelinsovellusta ryhdyttiin työstämään luomalla erillinen Docker- kontti kehitysympäristöksi. Tämä mahdollisti palvelinsovelluksen siirtämisen ja käyttämisen myöhemmin eri palvelinympäristöissä. Palvelinsovelluksen käyttöönotto muualla on myös konttien avulla jatkossa yksinkertaista ja helppoa. Kontti luodaan tekemällä *dockerfile*-tiedosto (kuva 4), johon määritellään kaikki mitä

konttiin halutaan asentaa. Tässä tapauksessa kontin käyttöjärjestelmäksi valittiin AlmaLinux linux- jakelu. Tieteen tietotekniikan keskuksen tarjoamat työkalut ovat AlmaLinux 9- yhteensopivia ja ne asennetaan RPM- paketinhallintajärjestelmällä, jota AlmaLinux tukee ja tämä tekee koko käyttöönottoprosessista vaivattomampaa.

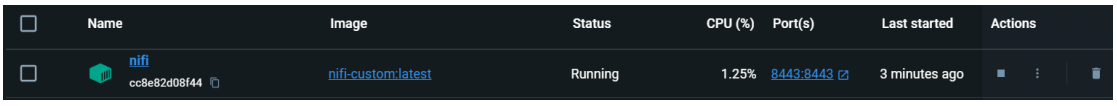
```
FROM almalinux:latest

RUN dnf -y update
RUN dnf -y install java-11-openjdk python3 python3-pip wget unzip dnf-plugins-core

# Determine JAVA_HOME and add it to /etc/profile.d/java.sh
RUN echo "export JAVA_HOME=$(dirname $(dirname $(readlink -f $(which java))))" > /etc/profile.d/java.sh

ENV NIFI_VERSION 1.25.0
ENV NIFI_BASE_DIR /opt/nifi

RUN wget https://dlcdn.apache.org/nifi/${NIFI_VERSION}/nifi-${NIFI_VERSION}-bin.zip -P /tmp/ && \
    unzip /tmp/nifi-${NIFI_VERSION}-bin.zip -d /opt/ && \
    mv /opt/nifi-${NIFI_VERSION} ${NIFI_BASE_DIR} && \
    rm /tmp/nifi-${NIFI_VERSION}-bin.zip

RUN rpm --import https://pas-jakelu.csc.fi/RPM-GPG-KEY-pas-support-el9
RUN dnf config-manager --add-repo=https://pas-jakelu.csc.fi/pas-jakelu-csc-fi.repo
RUN dnf config-manager --set-enabled crb
RUN dnf -y install epel-release
RUN dnf -y install --nogpgcheck https://mirrors.rpmfusion.org/free/el/rpmfusion-free-release-9.noarch.rpm

RUN dnf -y install \
    python3-file-scraper-full \
    python3-dpres-siptools \
    python3-dpres-ipt \
    python3-dpres-specification-migrator \
    python3-dpres-signature \
    python3-dpres-sip-compiler \
    python3-metax-access

RUN pip3 install numpy pandas

RUN groupadd -r nifi && \
    useradd -r -g nifi -d ${NIFI_BASE_DIR} -s /sbin/nologin nifi && \
    chown -R nifi:nifi ${NIFI_BASE_DIR}

USER nifi
EXPOSE 8080 8443
WORKDIR ${NIFI_BASE_DIR}
CMD ["/bin/bash", "-c", "source /etc/profile.d/java.sh && ./bin/nifi.sh run"]
```

Kuva 4. Dockerfile- tiedosto.

Muita määrityksiä Dockerfile- tiedostossa on Apache Nifi:n asennus ja sen tarvitsema Java 11- paketti. Tieteen tietotekniikan keskuksen työkalut ja kaikki niiden käyttöön tarvittavat paketit asennetaan myös valmiiksi kontille. Kun kaikki tarvittava on listattu tiedostoon, voidaan siitä luoda Docker- kuva (kuva 5).

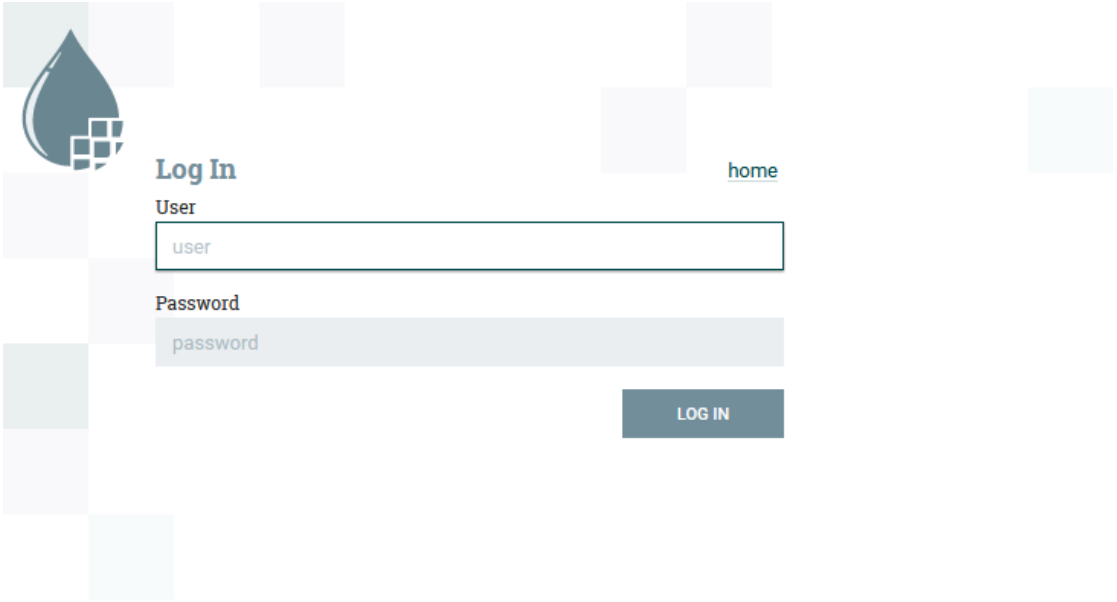
Kuva 5. Docker- kuvan luonti.

Kuvan luonnin jälkeen voidaan uusi kontti ottaa käyttöön ajamalla kometori-villä komento *docker run -d -p 8443:8443 nifi-python*. Tässä -d tarkoittaa, että kontti ajetaan taustalla (detached mode) ja -p 8443:8443 määrittelee porttien välityksen, jossa isäntäkoneen portti 8443 yhdistetään kontin porttiin 8443. *nifi-python* on luodulle kuvalle annettu nimi. Docker Desktop- ohjelmaa (kuva 6) käyttämällä näkee kontin nykyisen tilan ja sen että kontti on käynnistynyt onnistuneesti.



Kuva 6. Docker Desktop

Docker Desktop tarjoaa palvelulinkin, jonka avulla pääsee suoraan kontissa pyörivän ohjelmiston graafiseen käyttöliittymään. Tässä se ohjaa suoraan Apache Nifi:n kirjautumissivulle (kuva 7).

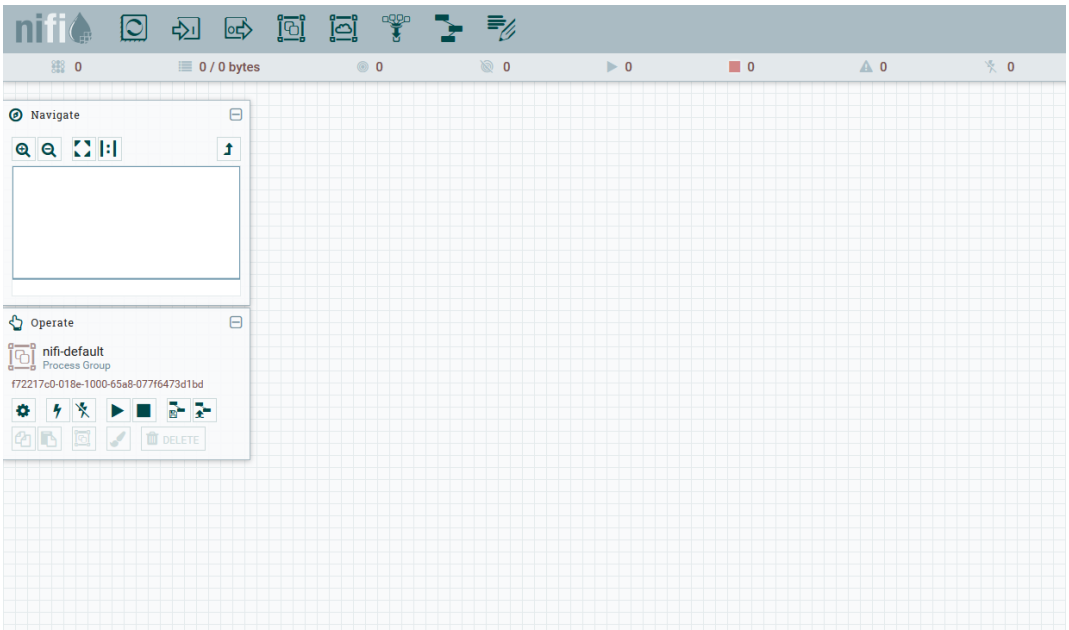


Kuva 7. Apache Nifi- kirjautuminen

Apache Nifi luo satunnaisen käyttäjätunnuksen ja salasanan ensimmäisellä käynnistyksellä. Nämä kirjautumistiedot löytyvät Nifi:n lokitiedostosta, josta käyttäjän on ne haettava kirjautuakseen.

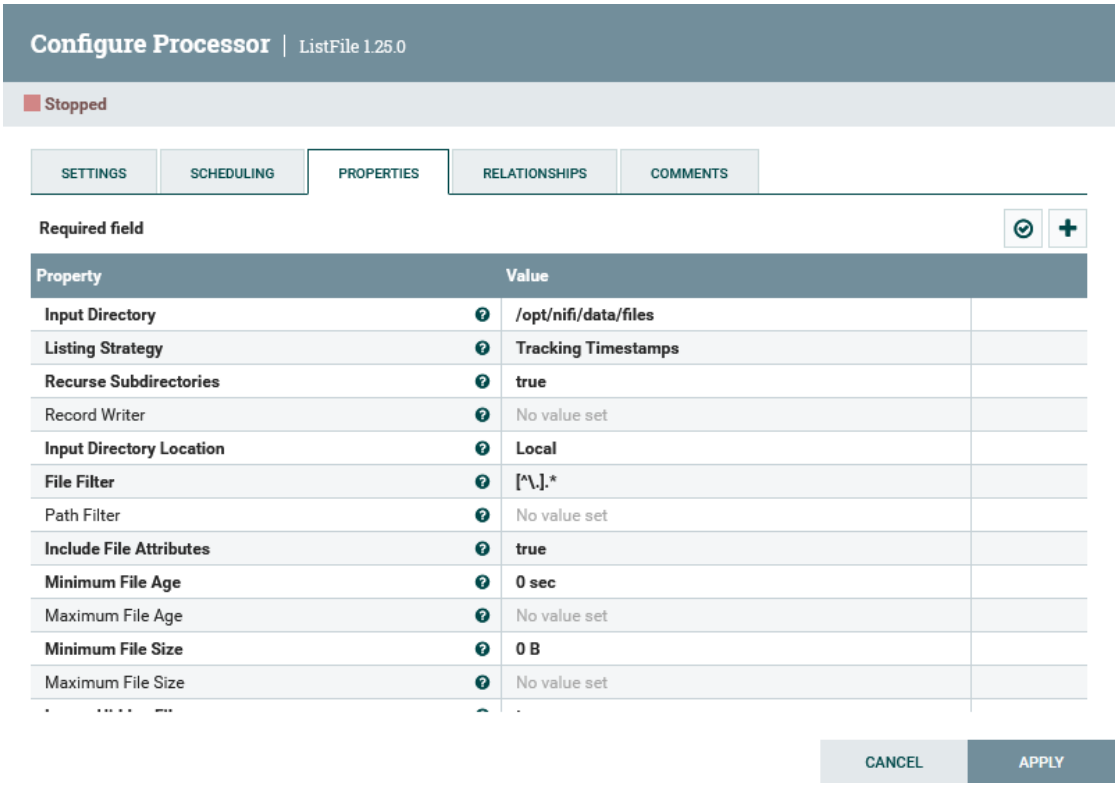
4.2 Metadatan luonti

Onnistuneen kirjautumisen jälkeen käyttäjälle avautuu tyhjä Apache Nifi -käyttöliittymä (kuva 8), johon voidaan luoda tietovirtoja ja tietoa käsitteleviä prosessoreita.



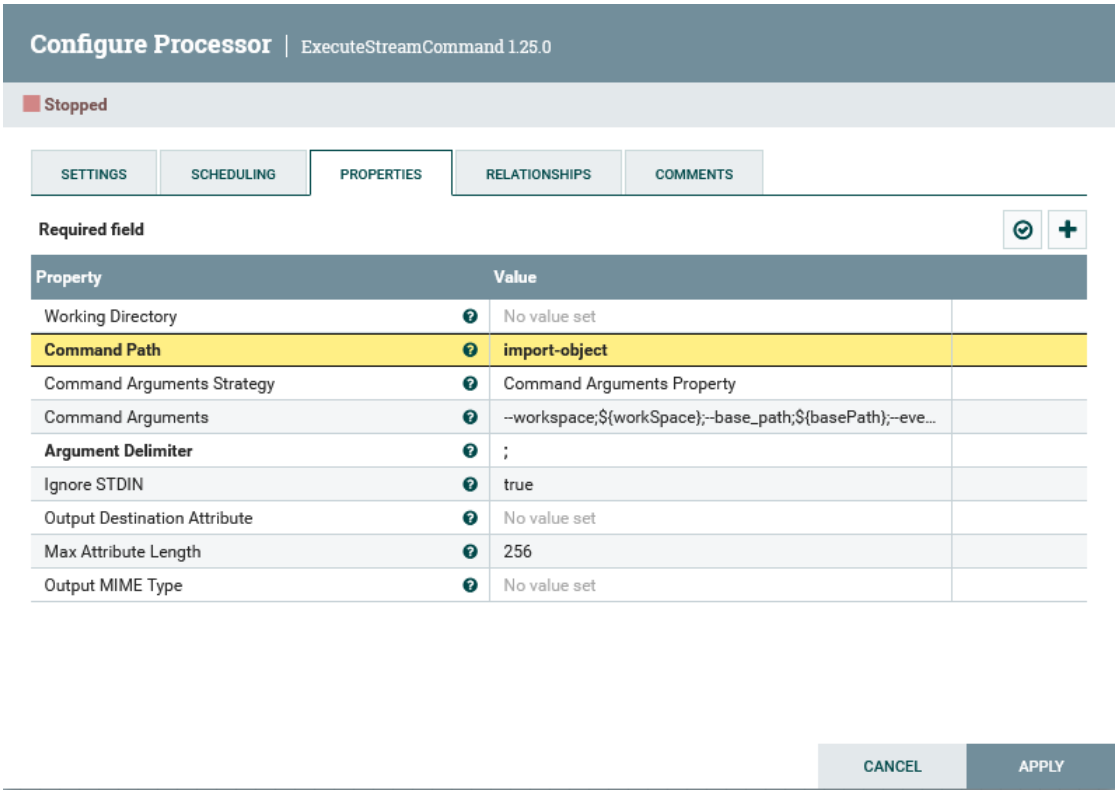
Kuva 8. Tyhjä Apache Nifi -käyttöliittymä.

Toimeksiannon mukaisesti ja Tieteen tietotekniikan keskuksen työkalujen ohjeita noudattaen aloitetaan teknisen metadatan luonti tiedostoista. Tämän tietovirran tarkoituksena, ETL- prosessia mukaillen, on poimia (extract) kansiosta käsiteltävät tiedostot, muuntaa (transform) ja luoda tiedostoista teknistä metadata-aineistoa ja lopuksi tallentaa (load) aineisto kansioon. Tiedostojen poimiminen tapahtuu Nifi: n omalla ListFile- prosessorilla (kuva 9). ListFile pitää listaa tiedostoista sille asetetusta kohdekansiosta ja lähettää kansion tiedostoista muodostetun FlowFile:n seuraavalle prosessorille käsiteltäväksi. FlowFile sisältää tarvittavia tietoja tiedostosta ja näitä tietoja käytetään myöhemmin hyödyksi. ListFile tarkkailee myös kansion tiedostoja aikaleiman perusteella ja hylkää tiedostot, jotka ovat identtisiä jo käsiteltyjen tiedostojen kanssa.



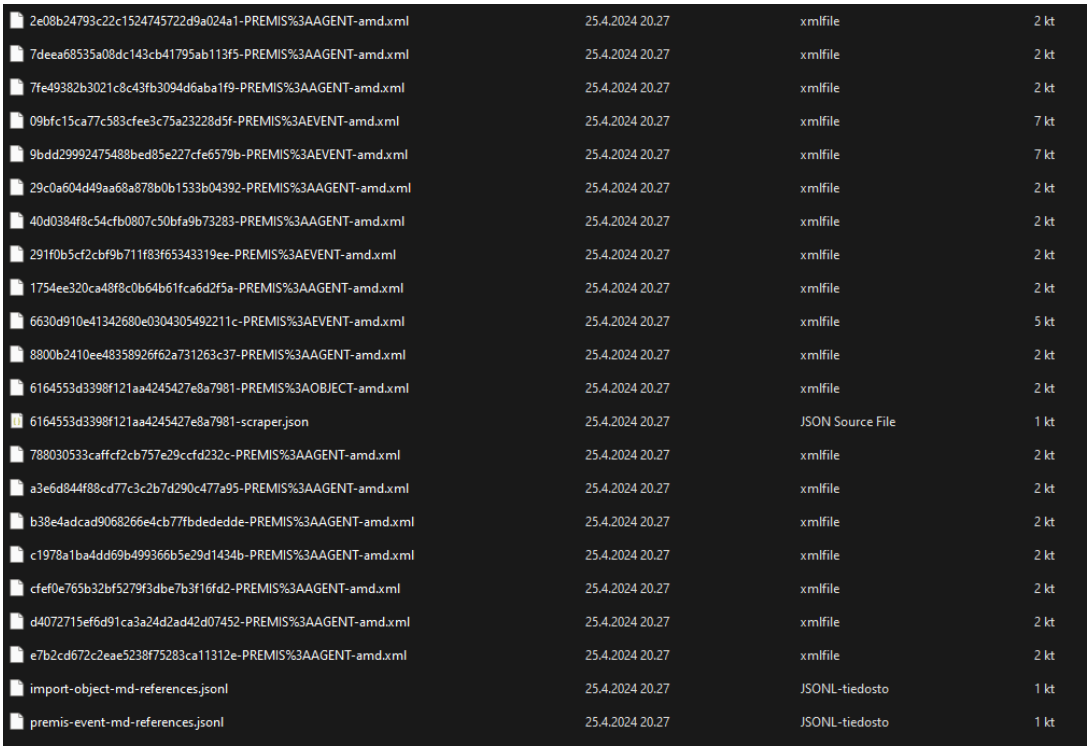
Kuva 9. ListFile-prosessori.

Tässä tietovirran prosessissa seuraavana prosessorina on Tieteen tietotekniikan keskuksen tekemä *import-object*-skripti (kuva 10). Se luo annetusta tiedostosta METS- dokumenttiin teknisen metatiedon *workspace*-kansioon. Ulkoisia skriptejä voidaan Nifissä ajaa ExecuteStreamCommand-prosessorilla. Prosessoriin määritellään ajettava skripti, kansio mihin metatietoaineisto tehdään ja tiedosto, josta metatieto tehdään.



Kuva 10. Import- object- skripti prosessorissa.

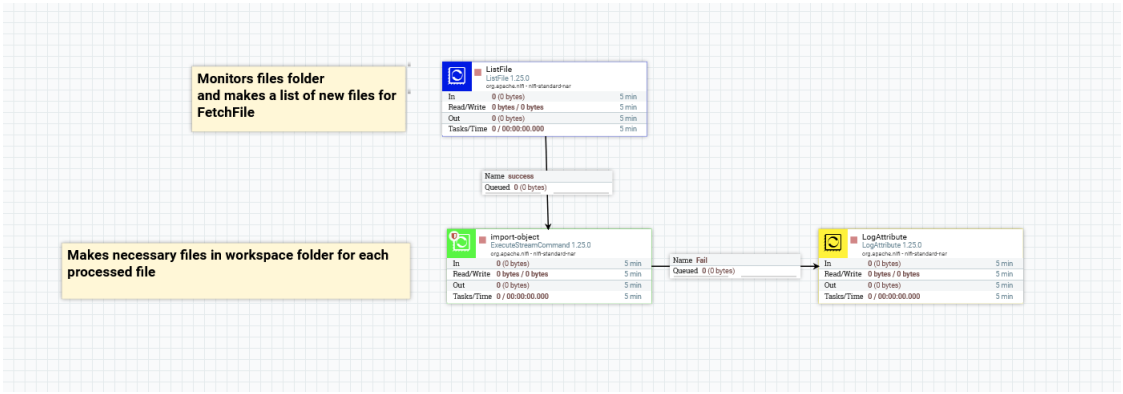
Ajettavan skriptin polkua ei tarvitse erikseen määritellä, koska työkalut on lisätty asennuksen yhteydessä Docker- kontin ympäristömuuttujaan (PATH). Näin skriptejä voidaan ajaa suoraan mistä tahansa kontin sijainnista. *Import-object*- skripti luo annetusta tiedostosta teknisen metadatan tiedostot *workspace*- kansioon (kuva 11).



2e08b24793c22c1524745722d9a024a1-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
7deea68535a08dc143cb41795ab113f5-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
7fe49382b3021c8c43fb3094d6aba1f9-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
09bfc15ca77c583cfee3c75a23228d5f-PREMIS%3AEVENT-amd.xml	25.4.2024 20.27	xmlfile	7 kt
9bdd29992475488bed85e227cfe6579b-PREMIS%3AEVENT-amd.xml	25.4.2024 20.27	xmlfile	7 kt
29c0a604d49aa68a878b0b1533b04392-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
40d0384f8c54cfb0807c50bfa9b73283-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
291f0b5cf2cbf9b711f83f65343319ee-PREMIS%3AEVENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
1754ee320ca48f8c0b64b61fca6d2f5a-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
6630d910e41342680e0304305492211c-PREMIS%3AEVENT-amd.xml	25.4.2024 20.27	xmlfile	5 kt
8800b2410ee48358926f62a731263c37-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
6164553d3398f121aa4245427e8a7981-PREMIS%3AOBJECT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
6164553d3398f121aa4245427e8a7981-scraper.json	25.4.2024 20.27	JSON Source File	1 kt
788030533caffcf2cb757e29ccfd232c-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
a3e6d844f88cd77c3c2b7d290c477a95-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
b38e4adcad9068266e4cb77fbdededde-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
c1978a1ba4dd69b499366b5e29d1434b-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
cfef0e765b32bf5279f3dbe7b3f16fd2-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
d4072715ef6d91ca3a24d2ad42d07452-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
e7b2cd672c2eae5238f75283ca11312e-PREMIS%3AAGENT-amd.xml	25.4.2024 20.27	xmlfile	2 kt
import-object-md-references.jsonl	25.4.2024 20.27	JSONL-tiedosto	1 kt
premis-event-md-references.jsonl	25.4.2024 20.27	JSONL-tiedosto	1 kt

Kuva 11. Workspace-kansion rakenne import- object- skriptin jälkeen.

Jokainen tiedosto, joka halutaan sisällyttää siirtopakettiin, ajetaan *import- object-* skriptin läpi ja jokaisesta tiedostosta tehdään meta-aineistot. Kokonaisuutena on tietovirtaprosessi Nifin graafisessa käyttöliittymässä (kuva 12), joka tekee määritellyistä tiedostoista teknistä meta-aineistoa *workspace-* kansioon.



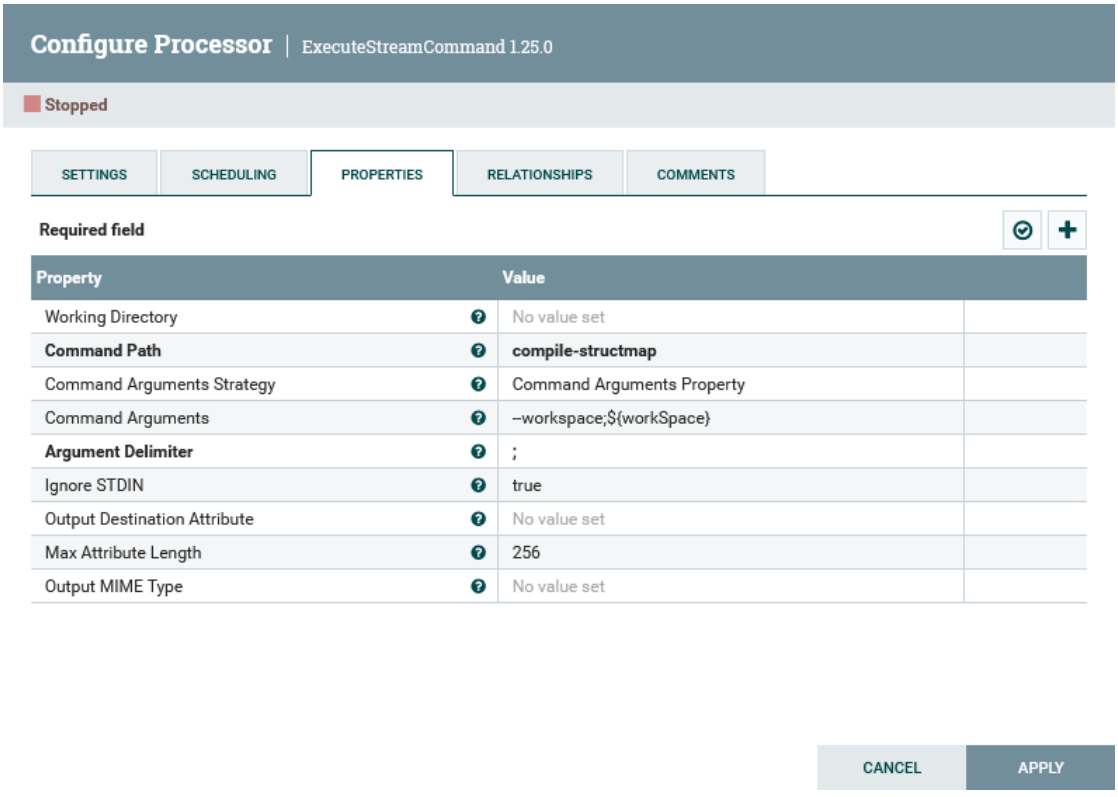
Kuva 12. Apache Nifi import- prosessoriryhmä.

Sinisellä merkatut prosessorit ovat Nifin omia prosessoreita ja vihreät ulkoisia skriptejä. Keltaiset merkityt prosessorit ovat virheiden kirjausprosessoreita, jotka kirjaavat mahdolliset virheet prosessoinnin yhteydessä.

4.3 Metadatan paketointi

Kun kaikki halutut tiedostot on käsitelty ja niistä on luotu meta-aineisto kansioon, automatisoidaan lopullisen siirtopaketin (SIP) luonti Nifillä ja Tieteen tietotekniikan keskuksen työkaluilla. Tämä seuraava ETL- prosessi eritellään edellisestä omaksi prosessiryhmäkseen Nifin *Process Group*- toiminnolla. Tällä saadaan eriteltyä eri prosesseista vastuussa olevat prosessorit ja kokonaisuus pysyy eheänä ja helpommin ymmärrettävänä.

Tämän prosessiryhmän aloittaa Nifin oma *GenerateFlowFile*- prosessori, jota käytetään tässä lähettämään käynnistyskäsky koko ryhmälle. Se lähettää ensimmäiselle *compile-structmap*- skriptille (kuva 13) käskyn suorittamiseen.

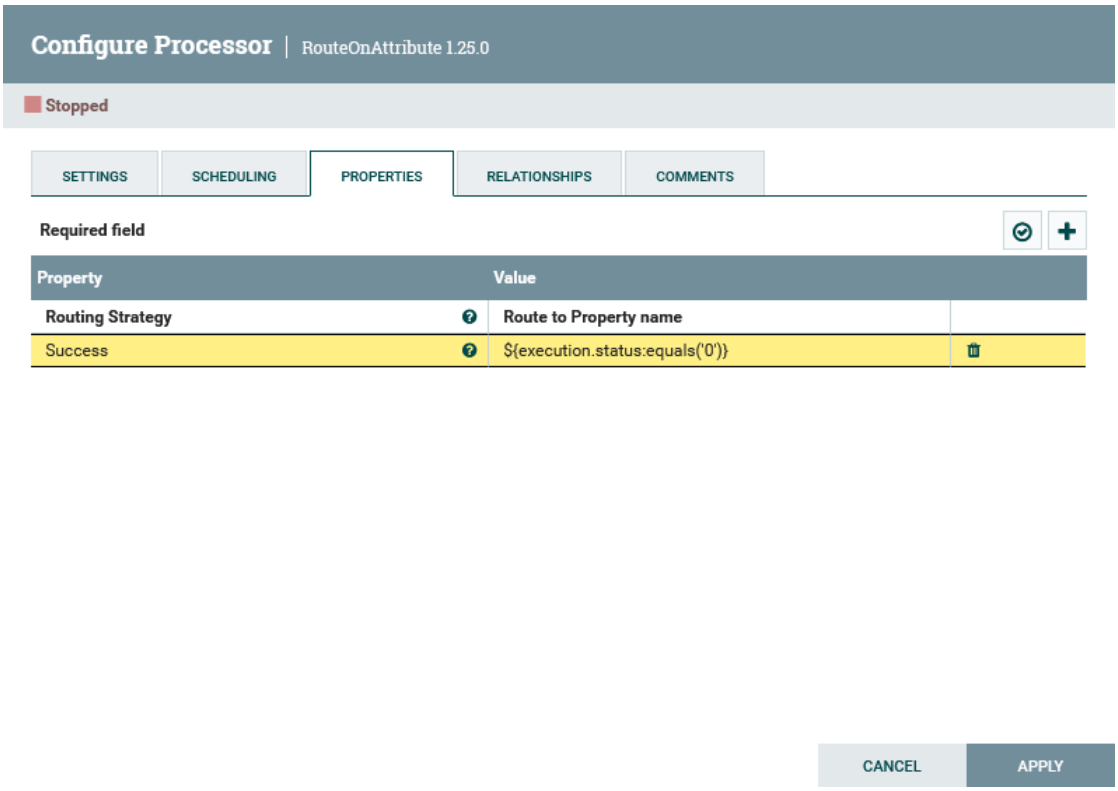


Kuva 13. compile-structmap-skripti prosessorissa.

Skripti muodostaa tiedostoja *workspace*- kansion sisällöstä (kuva 14), joka sisältää tiedosto-osion ja rakenteellisen kartan METS- dokumentista. Näiden avulla luodaan lopullinen siirtopaketti (SIP).

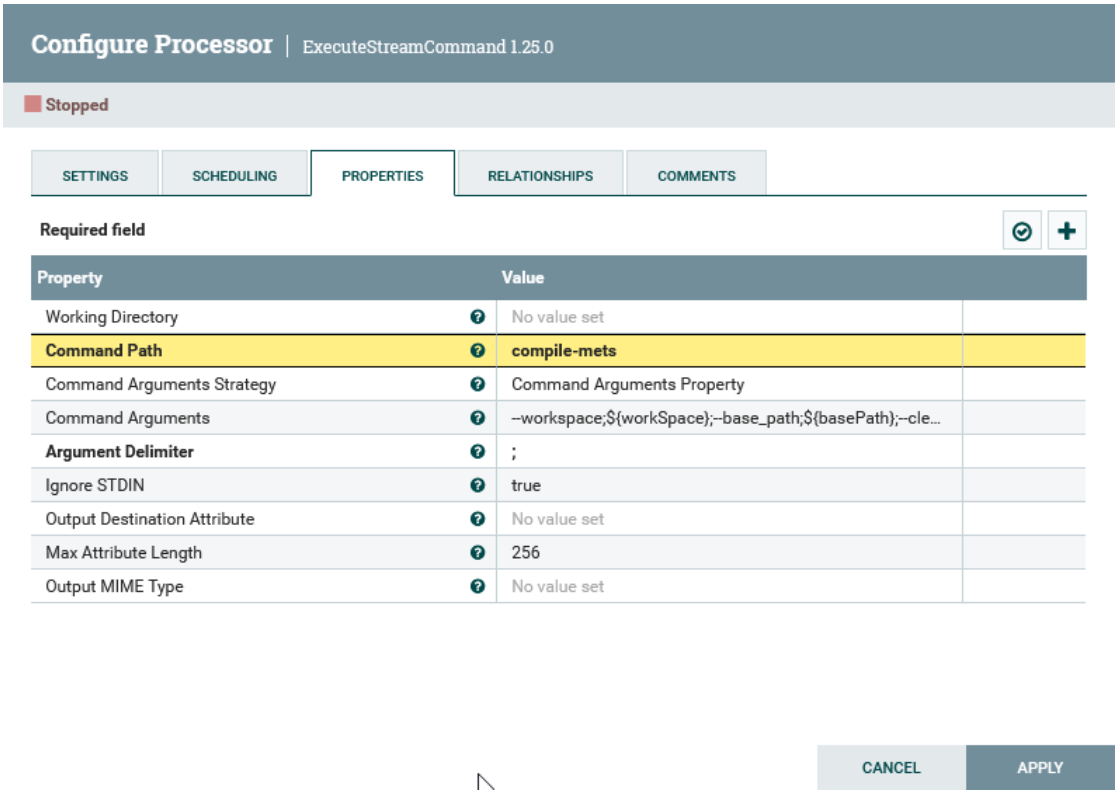
Kuva 14. tiedosto-osio ja rakenteellinen kartta workspace- kansiossa.

Tässä prosessiryhmässä ajetaan monia skriptejä peräkkäin, joten tieto edellisen skriptin onnistuneesta ajosta on tärkeä ennen kuin voidaan ajaa seuraava skripti. Tähän voidaan käyttää Nifistä löytyvää *RouteOnAttribute- p*rosessoria (kuva 15). Tässä yhteydessä sitä käytetään vastaanottamaan tietoa edellisen skriptin ajon tilasta. Siihen määritellään muuttuja *${execution.status:equals('0')}*. Kun edellisen skriptin tila on 0, eli ajo on onnistunut, antaa se käskyn suorittaa seuraavaksi määritelty skripti prosessiryhmässä.



Kuva 15. RouteOnAttribute- prosessorin määrittely.

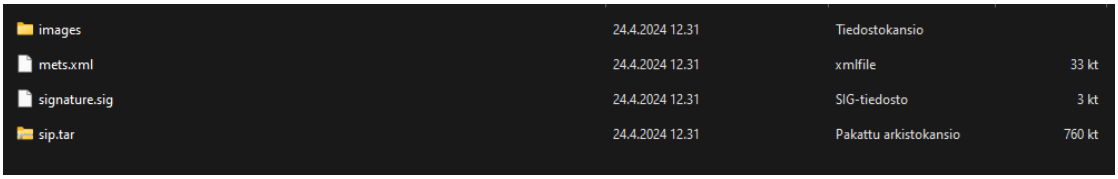
Onnistuneen skriptin ajon jälkeen voidaan luoda lopullinen METS- dokumentti SIP-pakettia varten *compile-mets*- skriptillä (kuva 16). Se kokoaa kaikista aiemmin luoduista meta-aineistosta ehjän METS- dokumentin.



Kuva 16. compile-mets- skripti prosessorissa.

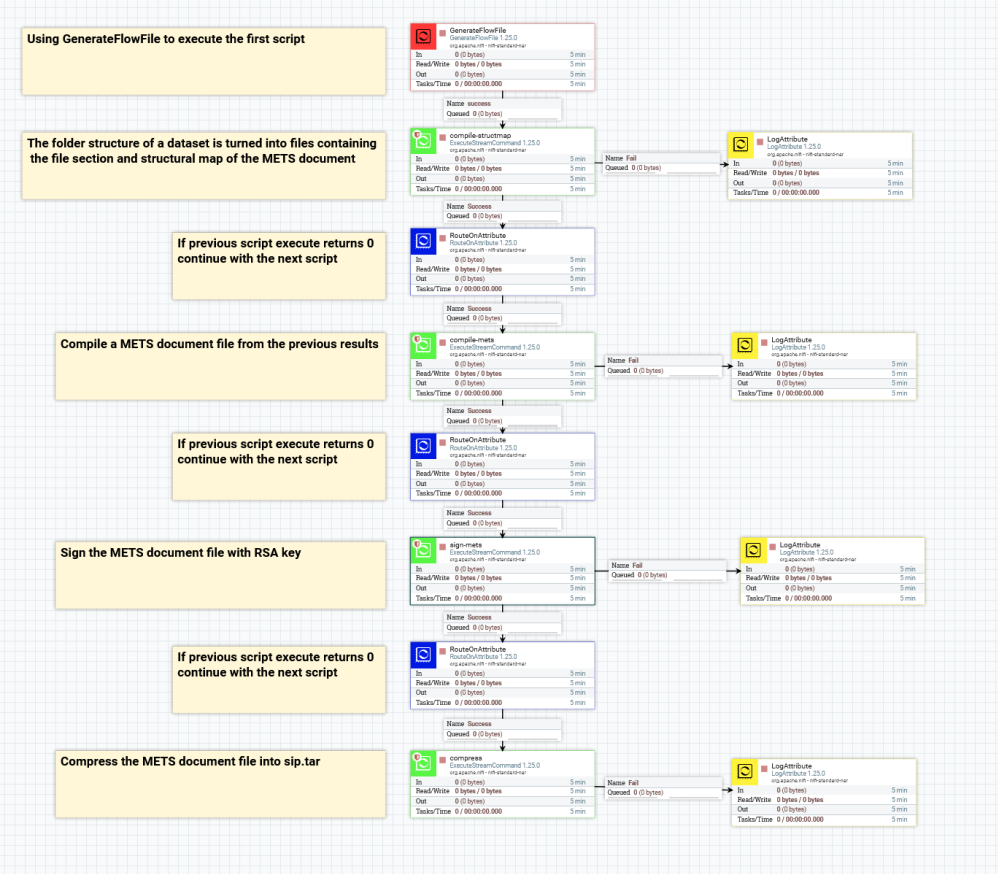
Aineiston eheyden ja kiistämättömyyden varmistamiseksi allekirjoitetaan luotu dokumentti digitaalisesti. Tähän tarvitaan *sign-mets*- skriptin lisäksi yksityistä ja julkista avainta, joilla allekirjoitus onnistuu. Skriptiin määritellään avainten hakemisto, josta se hakee avaimet ja allekirjoittaa dokumentin.

Viimeisenä prosessiryhmässä on *compress*- skripti. Se paketoi kaikki aiemmin luodut tarvittavat tiedot yhdeksi *.tar*- tiedostoksi. Tämä on lopullinen siirtopaketti, jonka voi viedä PAS- palveluun. Skriptin ajon jälkeen *workspace*- kansio sisältää kansion tuoduista digitaalisista objekteista, METS- dokumentin, digitaalisen allekirjoitustiedoston ja pakatun *sip.tar*- tiedoston (kuva 17).



Kuva 17. Workspace-kansion rakenne.

Kokonaisuutena tämä prosessiryhmä on kuvattuna alusta loppuun Nifin graafisessa käyttöliittymässä (kuva 18).



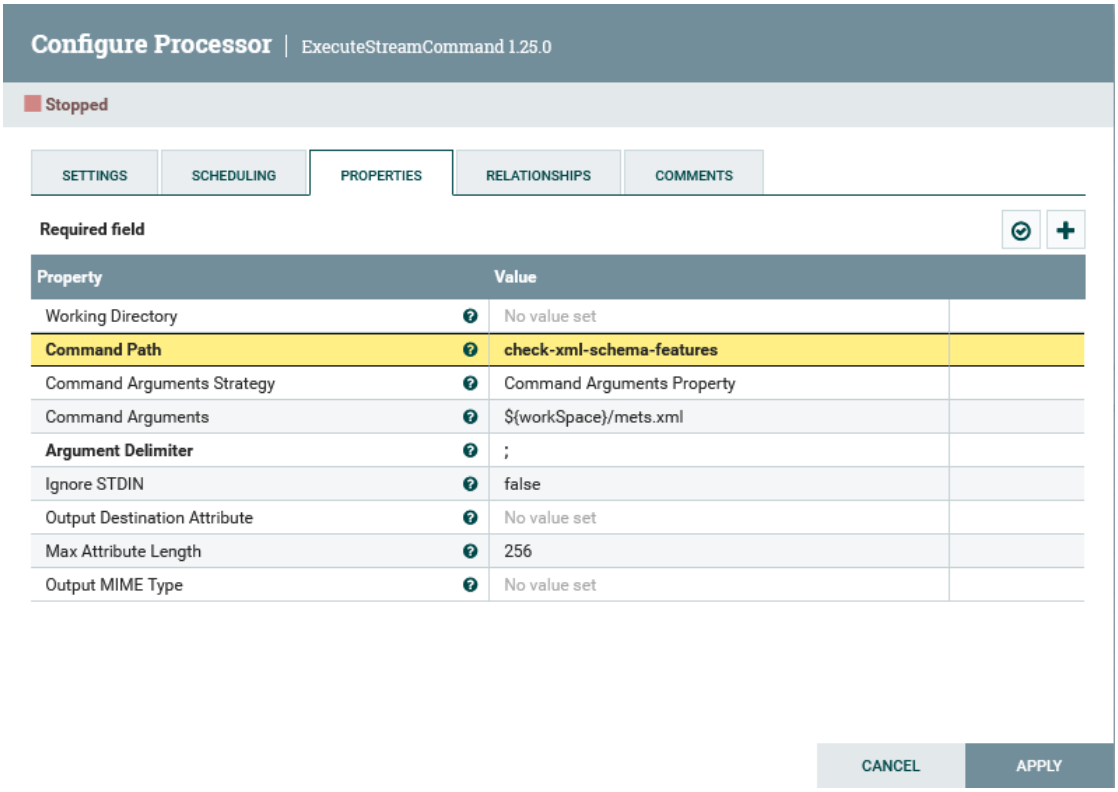
Kuva 18. Verify & sign- prosessiryhmä.

Punainen *GenerateFlowFile* aloittaa prosessin ja jokaisen vihreän skriptiajon jälkeen ovat siniset *RouteOnAttribute*- prosessorit, jotka ajoittavat skriptien suorittamista. Keltaiset *LogAttribute*- prosessorit vastaavat virheiden kirjauksesta.

4.4 METS-dokumentin validointi

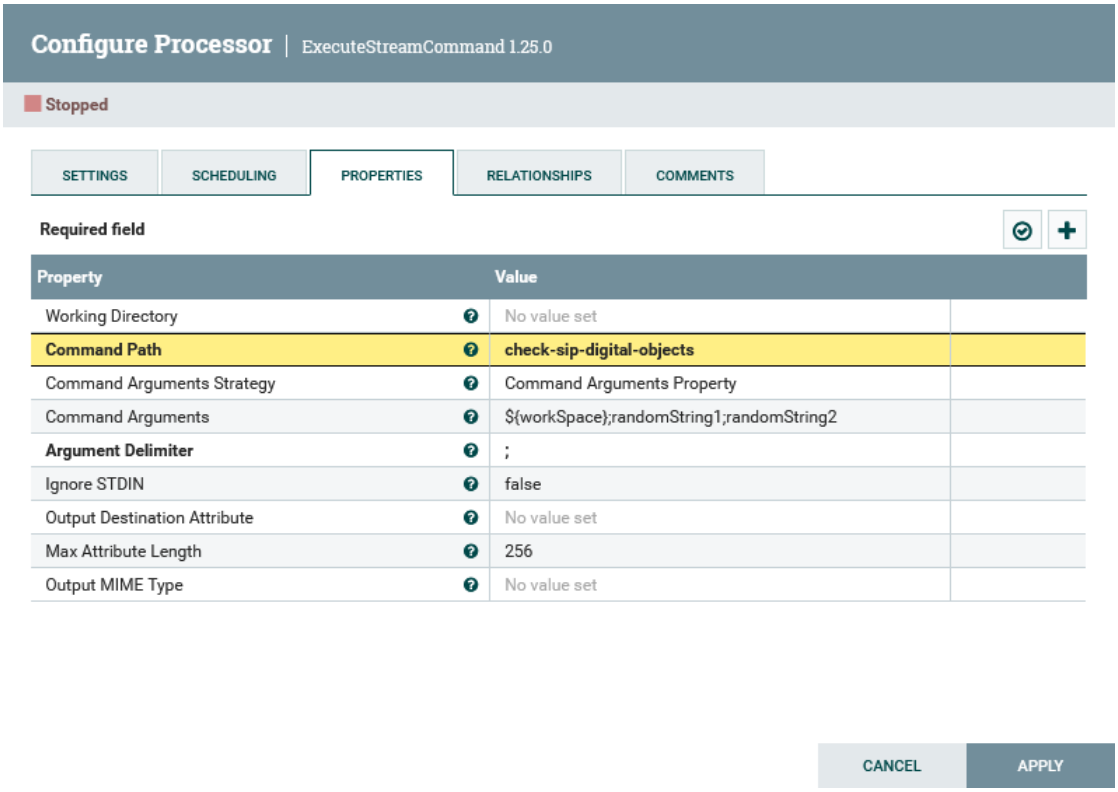
Pitkäaikaissäilytyksen bittitason mukaisesti on tärkeää varmistaa datan muuttumattomuus validoimalla. Tieteen tietotekniikan keskus on tehnyt datan validoimiseen erilliset työkalut. Näiden työkalujen käyttö voidaan automatisoida Nifillä. Validointi koostuu kolmesta eri skriptin ajosta, *check-xml-schema-features*, *check-sip-digital-objects* ja *check-sip-file-checksums*.

Prosessin aloitus tehdään *GenerateFlowFile*- prosessorilla, joka aloittaa ensimmäisen *check-xml-schema-features*-skriptin- (kuva 19) ajon. Se validoi, että luotu METS- dokumentti vastaa METS- profiilia, jota PAS- palvelu käyttää.



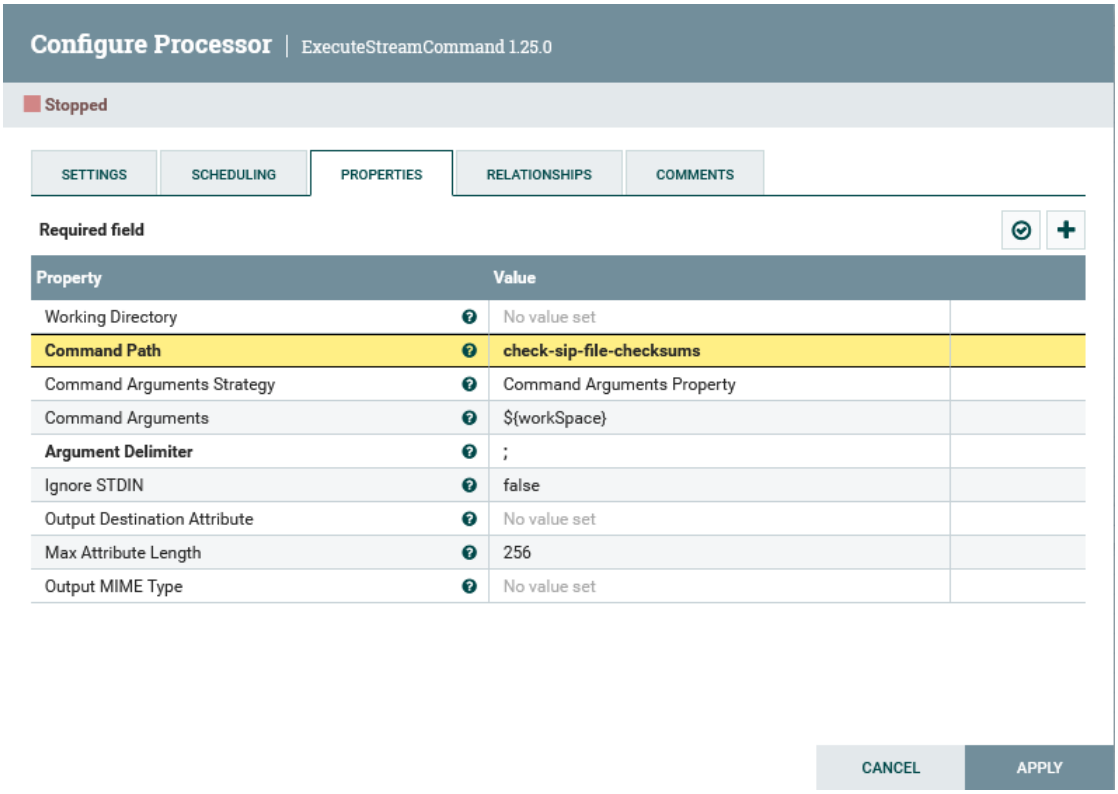
Kuva 19. check-xml-schema-features-skripti prosessorissa.

Seuraavana tekijänä validointiprosessissa on digitaalisen aineiston validointi tehdyssä siirtopaketissa. Tähän käytetään *check-sip-digital-objects*- skriptiä (kuva 20), joka tarkistaa siirtopaketin digitaalisten objektien vastaavuuden METS- dokumentin määritelmiin. Tämä skripti varmistaa myös, että kaikki dokumentissa mainitut tiedostot löytyvät paketista ja niiden metadata vastaa PAS- palvelun käyttämää METS- profiilia.



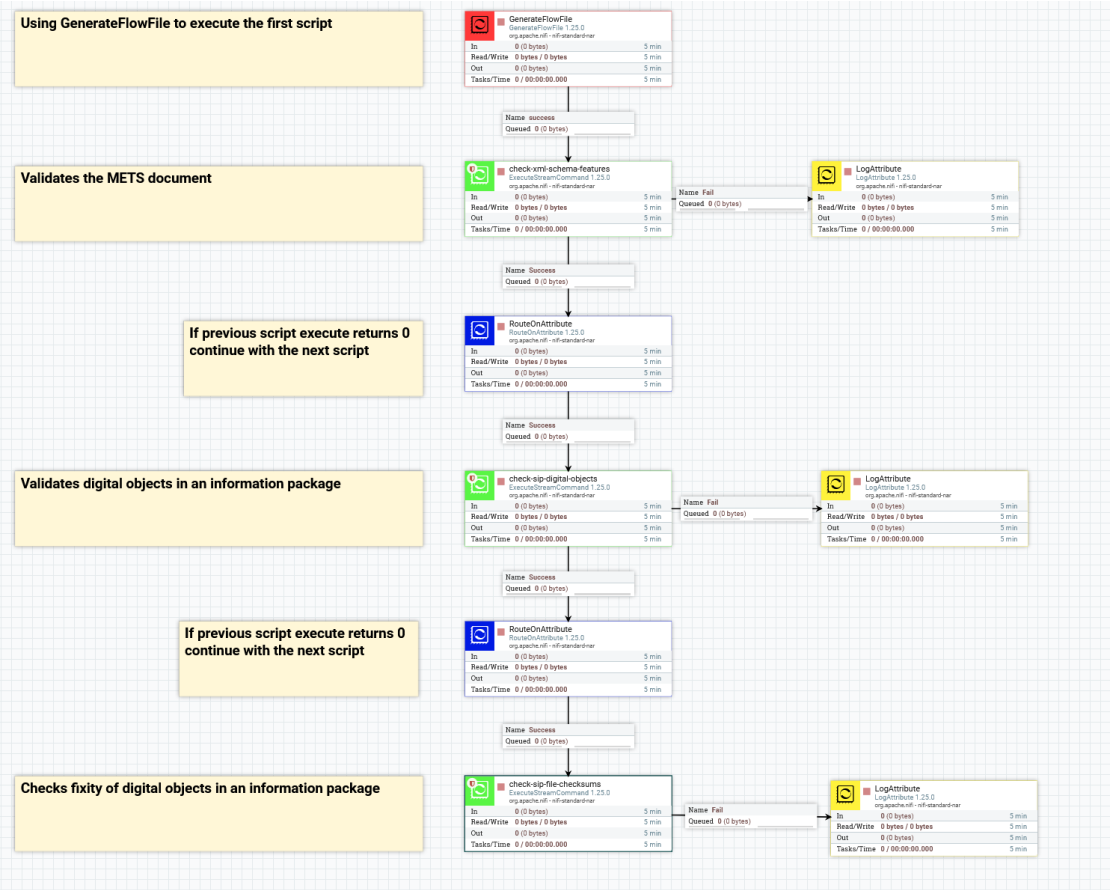
Kuva 20. Check-sip-digital-objects-skripti prosessorissa.

Viimeinen validointi tapahtuu *check-sip-file-checksums*-skriptillä (kuva 21). Se varmistaa, että tiedot eivät ole vahingoittuneet tai muuttuneet prosessin aikana vertailemalla tiedostojen tarkistussummia *(checksumeja).*



Kuva 21. Check-sip-file-checksums-skripti prosessorissa.

Kokonaisuutena valmis validointiprosessiryhmä (kuva 22) Nifin graafisessa käyttöliittymässä mukailee aiempia prosessiryhmiä.



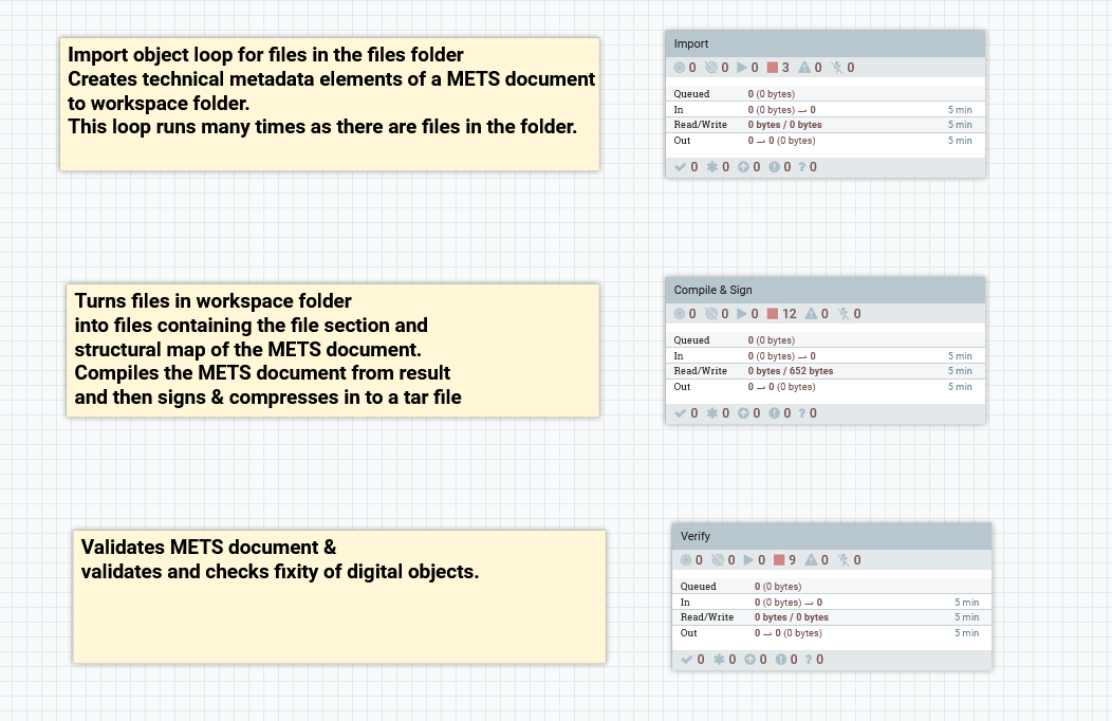
Kuva 22. Validate- prosessiryhmä.

Edellisten tapaan punainen *GenerateFlowFile* käynnistää prosessin ja jokaisen vihreän skriptiajon jälkeen seuraa sininen *RouteOnAttribute*- prosessori, joka aikatauluttaa skriptien suorittamisen. Mahdollisen validointivirheen sattuessa, kirjataan ja annetaan käyttäjälle siitä tieto keltaisilla *LogAttribute*- prosessoreilla.

5 LOPPUTULOS

Tämä opinnäytetyön tarkoituksena oli selvittää, kuinka Apache Nifi soveltuu pitkäaikaissäilytyspakettien luonnin automatisointiin ja kuinka Tieteen tietotekniikan keskuksen luomat työkalut voidaan saada käyttöön Apache Nifissä. Työn lopputuloksena on kolme eriteltyä prosessiryhmää Apache Nifissä (kuva

23.), joista jokaisella on oma tärkeä tehtävänsä tässä dataintegraatioprosessissa. Työkalut ja prosessit toimivat odotetusti ja kokonaisuudella voidaan saada luotua ja validoitua pitkäaikaissäilytykseen soveltuvia siirtopaketteja.



Kuva 23. Prosessiryhmät Apache Nifissä.

Nifissä on mahdollista tallentaa kaikki määritellyt prosessiryhmät ja prosessorit mallitiedostoiksi, jotka edelleen voidaan ottaa käyttöön muissa Nifi- ympäristöissä. Tämä ominaisuus, yhdessä Nifin ajamisen kanssa Docker- kontissa, mahdollistaa kokonaisuuden siirtämisen kehitystietokoneelta palvelinkoneelle.

Edellä luotu työ antaa hyvän pohjan jatkokehitykselle, jossa voidaan jatkaa automatisointiprosessia hakemalla Yksasta REST- rajapinnan kautta aineistoa pitkäaikaissäilytyksen paketointiprosessiin. Valmis paketti voitaisiin siirtää automaattisesti PAS- palveluun ja hyväksytyn siirron jälkeen voitaisiin antaa tieto siirron onnistumisesta.

6 PÄÄTÄNTÖ

Opinnäytetyöni päättyessä on sopiva hetki arvioida koko projektin kulkua ja sen onnistumista. Sain Diseciltä useita aihe- ehdotuksia. Niistä valitsin itselleni mielenkiintoisimman, joka tuntui uudelta ja innostavalta. Aiheen syvällinen kä-

sittely Apache Nifin avulla osoittautui haastavaksi mutta opettavaiseksi kokemukseksi. Projektin alussa Nifin käyttötarkoitus ja rooli opinnäytetyössäni olivat epäselviä, mutta ne tarkentuivat projektin edetessä. Tässä opinnäytetyössä Nifin rooli tarkentui aineiston pitkäaikaissäilytyksen paketointiin.

Raportissa tuli esille Nifin ominaisuuksia, käyttötarkoitus ja sen mahdollisuuksia. Edellä mainittujen seikkojen esittäminen onnistui hyvin. Tarkoin rajattu aihe hieman rajoitti syvempää palvelinsovelluksen tarkastelua, mutta raportin aikatauluun nähden koin sen kuitenkin sopivaksi. Mahdollisuuksien mukaan jatkan mielelläni projektin kehitystä eteenpäin.

Projektin aikataulutus oli suurin haaste. Uskon, että opinnäytetyöni tarjoaa arvokasta tietoa Disecille heidän jatkokehitystään varten ja lisäksi Apache Nifin käyttöönottoon. Nifin monipuoliset datan käsittelykyvyt tarjoavat merkittäviä etuja arkistointipalveluja tarjoavalle yritykselle.

Kokonaisuutena projekti on ollut hieno oppimiskokemus, joka on haastanut tekijää uuden oppimisille ja opitun soveltamiseen. Mainittakoon vielä Disecille kiitos, että sain mahdollisuuden projektiin.

LÄHTEET

Apache Nifi. 2024. Apache Nifi Overview. WWW-dokumentti. Saatavissa: https://nifi.apache.org/documentation/v2/ [viitattu 13.3.2024]

Cloudflare s.a. What is data migration? | Definition and common processes. WWW-dokumentti. Saatavissa: https://www.cloudflare.com/learning/cloud/what-is-data-migration/ [viitattu 20.3.2024]

CSC - IT Center for Science Ltd. 2018. GitHub. Paketointityökalu. Saatavissa: https://github.com/Digital-Preservation-Finland/dpres-siptools [viitattu 22.4.2024]

IBM s.a. What is data integration?. WWW-dokumentti. Saatavissa: https://www.ibm.com/topics/data-integration [viitattu 13.3.2024]

Jain, O. 2022. Apache NiFi Data Ingestion: A Comprehensive Guide 101. Blogi. Päivitetty 27.2.2024. Saatavissa: https://hevodata.com/learn/apache-nifi-data-ingestion/ [viitattu 14.3.2024]

Kansalliskirjasto s.a. Pitkäaikaissäilytys. Verkkojulkaisu. Saatavissa: https://www.kansalliskirjasto.fi/fi/vapaakappaletoimisto/verkkoaineistot/pitkaaikaissailytys [viitattu 11.4.2024]

Modern analyst s.a. What is a Data Flow?. WWW-dokumentti. Saatavissa: https://www.modernanalyst.com/Careers/InterviewQuestions/tabid/128/ID/6119/What-is-a-Data-Flow.aspx [viitattu 16.4.2024]

Morris, J. 2020. Practical Data Migration. 3. uudistettu painos. Swindon: BCS, The Chartered Institute for IT. E-kirja. Saatavissa: https://kaakkuri.finna.fi/Record/nelli29_mamk.4100000011509350?sid=4203114360 [viitattu 19.3.2024]

Morrison, P. s.a. Flow-based Programming. WWW-dokumentti. Saatavissa: https://jpaulm.github.io/fbp/index.html [viitattu 18.4.2024]

Opetus- ja kulttuuriministeriö. 2024. Aineistojen ja niiden metatietojen paketointi pitkäaikaissäilytykseen. PDF-dokumentti. Saatavissa: https://urn.fi/URN:NBN:fi-fe2020100578093 [viitattu 06.04.2024]

KUVALUETTELO