Raportointityökalun kehittäminen Sovelia® PLM-järjestelmään

Turun yliopisto
Tietotekniikan laitos
TkK-tutkielma
Tietotekniikka
Tammikuu 2024
Elias Peltonen

TURUN YLIOPISTO
Tietotekniikan laitos

Elias Peltonen: Raportointityökalun kehittäminen Sovelia® PLM-järjestelmään

TkK-tutkielma, 35 s., 4 liites.
Tietotekniikka
Tammikuu 2024

Raportoinnin tarkoituksena on kerätä, tallentaa ja esittää tietoa muodossa, joka on helposti ymmärrettävissä ja jaettavissa. Tehokkaan ja käyttäjäystävällisen raportoinnin mahdollistamiseksi on kehitetty tietoteknisiä raportointityökaluja, joiden avulla voidaan tuottaa raportteja annetun lähtödatan pohjalta.
Tässä tutkielmassa tarkastellaan raportointityökalun kehittämistä osaksi tuotteen elinkaaren hallintajärjestelemää. Tutkielmassa käsiteltävä elinkaaren hallintajärjestelmä on kaupallinen Sovelia PLM. Raportointityökalun kehittäminen koostuu raporttidokumenttien tuottamisesta ohjelmallisesti sekä raportointityökalun käyttöliittymän suunnittelusta ja kehittämisestä. Tarkastelemme PLM-järjestelmien ja raportointityökalujen yhteistoimintaa vedoten aiempaan kirjallisuuteen aiheesta, sekä tutkimme nykyisiä olemassa olevia raportointityökaluja. Aiemman kirjallisuuden pohjalta pyrimme ymmärtämään, millaista tuotteen elinkaaren hallintaohjelmistojen tuottama data on ja miten sen erityispiirteet vaikuttavat raportointiin. Tarkastelemalla olemassa olevia raportointityökaluja pyrimme muodostamaan raportointityökalujen tyypillisimmistä ominaisuuksista ja erityispiirteistä kokonaiskuvan, jonka perusteella pyrimme ymmärtämään, mitkä lähestymistavat toimivat Sovelia PLM:n tapauksessa ja mitkä eivät.
Tarkastelemalla aiempaa kirjallisuutta aiheesta havaitsimme, että PLM-data on massadatan kaltaista hierarkkista alati muuttuvaa dataa, mikä asettaa haasteita laadukkaalle raportoinnille. Olemassa olevia raportointityökaluja tarkastelemalla huomaisimme, että osa nykyisten raportointityökalujen lähestymistavoista olivat hyödyllisiä myös Sovelia PLM:n uuden raportointityökalun tapauksessa, mutta päädyimme osittain nykyaikaisempiin teknologiavalintoihin.

Asiasanat: raportointi, PLM, raportointityökalu, ohjelmisto

Sisällys

Kuvat

Taulukot

Termistö

API ohjelmointirajapinta, engl. Application Programming Interface

BI engl. Business Intelligence, liiketoimintatiedon hyödyntäminen

Big data suom. massadata, erittäin suuret ja järjestämättömät jatkuvasti lisääntyvät tietomassat

BOM Bill of Materials, osaluettelo, tuoterakenne

CAD engl. Computer Aided Design, tietokoneen käyttö suunnittelun apuvälineenä

CSS engl. *Cascacing Style Sheets*, suom. *porrastetut tyyliarkit*, tyylisivu, jolla voidaan määritellä tyyliohjeita miten esimerkiksi HTML dokumentti voidaan esittää

HTML engl. *HyperText Markup Language*, suom. *hypertekstin merkintäkieli*, merkintäkieli, jolla internetsivut ovat kirjoitettu

JSON engl. JavaScript Object Notation, yksinkertainen tiedostomuoto tiedon välitykseen ja tallennukseen

PLM engl. Product Lifecycle Management, tuotteen elinkaaren hallinta

SQL engl. *Structured Query Language*, standardoitu kyselykieli, jolla relaatiotietokantaan voi tehdä erilaisia hakuja, muutoksia ja lisäyksiä

XML merkintäkielien standardi ja tiedostomuoto, engl. Extensible Markup Language

XSL XML-kieliperhe, joka mahdollistaa XML-pohjaisten tiedostojen ulkoasun ja rakennemuutoksen määrittelyn

ZIP tiedon pakkaukseen käytetty tiedostomuoto, joka voi sisältää useita tiedostoja ja kansioita pakattuna

1 Johdanto

Tietotekniikan avulla voidaan tehostaa ja helpottaa työnteon tuottavuutta, kun samaan tehtävään käytetty aika vähenee [1]. Tämän tutkielman kontekstissa raportoinnilla tarkoitetaan yrityksen tapaa ja toimintoja kerätä, prosessoida, tallentaa ja esittää tietoa. Raportoinnin ydinajatuksena on tuottaa tietoa muodossa, joka on helposti ymmärrettävissä ja jaettavissa [2]. Raportit voivat olla tyypiltään hyvinkin erilaisia, mutta tässä tutkielmassa keskitytään erityisesti taulukko- ja kuvaajapohjaisiin raporttidokumentteihin.

Tietotekniikan käyttö raportoinnissa on luontevaa, mikäli raportit ovat digitaalisia dokumentteja ja niiden luominen vaatii laskentatoimintoja. Manuaalinen raporttidatan kerääminen ja jäsentäminen on hankalaa ja hidasta, minkä vuoksi monet tietojärjestelmät tarjoavat raportointityökaluja automatisoimaan nämä vaiheet. Raportointityökalujen avulla olemassa olevasta suuresta määrästä dataa voidaan tuottaa selkeä ja jäsennelty esitys, joka kokoaa lähdedatan tärkeimmät seikat helposti yhdellä silmäyksellä omaksuttavaan muotoon.[3]

Tämän työn tarkoituksena oli toteuttaa raportointityökalu osaksi Sovelia PLM -järjestelmää. Sovelia PLM on kaupallinen tuotteen elinkaaren hallintajärjestelmä (engl. *Product Lifecycle Management*, PLM)[4]. PLM-järjestelmän pääasiallisena tarkoituksena on koota tietoa yrityksen tuotteiden koko elinkaaren vaiheista keskitettyyn tietojärjestelmään [5]. Tämä keskitetty tietojärjestelmä on käytettävissä yrityksen eri työryhmien ja liiketoimintajärjestelmien välillä, minkä tarkoituksena

on vähentää virheellisten tuotetietojen aiheuttamia turhia kustannuksia sekä viivästyksiä ja siten nopeuttaa yrityksen prosessia saada kehitettävä tuote markkinoille. [5]

Raportointityökalu voidaan nähdä yhtenä PLM järjestelmälle lisäarvoa tuottavana ominaisuutena.[6] Luotettavan, tehokkaan ja mukautuvan raportointityökalun avulla PLM-järjestelmä voi tuottaa enemmän lisäarvoa sen käyttäjille tarjoamalla mahdollisuuden jakaa, tallentaa ja analysoida tuotedataa eri tiedostoformaateissa sekä yrityksen sisäisten työryhmien että ulkoisten toimijoiden välillä.[6] PLM-järjestelmässä raporttien tuottaminen osaluetteloihin (engl. *Bill of Materials, BOM*) tallennetun datan pohjalta on erityisen tärkeää.[6] PLM-järjestelmiä käyttävillä yrityksillä on tyypillisesti suuria määriä tuotetietoja ja laajoista tuoterakenteista koostuvia osaluetteloita [7], jolloin tehokkaan raportoinnin suorituskykyvaatimukset korostuvat.

PLM-järjestelmien tietomallit voidaan jakaa dokumentti- ja relaatiodatapohjaisiin tietorakenteisiin. [8] Koska Sovelia PLM -järjestelmä perustuu relaatiodatapohjaiseen tietomalliin, myös tässä tutkielmassa käsitellään PLM-järjestelmän raportointia nimenomaan relaatiodatan pohjalta. Kehitettävä uusi raportointityökalu integroituu osaksi Sovelian nykyistä lähdekoodia ja sen palvelinkomponentteja. Ohjelmakokonaisuus koostu palvelinkomponentista, raportointimoottorista, joka tuottaa raporttitiedoston raportoinnin kohteena olevasta objektista, sekä konfigurointityökalusta, jonka avulla järjestelmän pääkäyttäjä voi muokata raporttien ulkonäköä ja rakennetta.

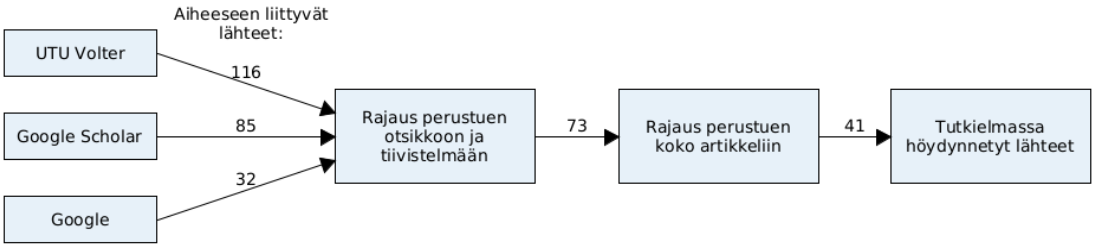
Tämän työn tutkimuskysymykset ovat:

TK1 Miten PLM-järjestelmät ja raportointityökalut toimivat yhdessä?

TK2 Millaisia ovat nykyiset raportointityökalut?

TK3 Mitä tulee ottaa huomioon raportointityökalua kehittäessä osaksi PLM-järjestelmää?

Jotta kirjallisuusanalyysi tutkimuskysymysten perusteella voitaisiin toteuttaa, tulee ensin tarkastella näiden hakumenetelmiä, joilla tämän tutkimuksen lähteet ovat kerätty. Yleisellä tasolla käytetyt lähteet voidaan jakaa akateemisiin tutkimuksiin ja artikkelihin sekä yritysten tuottamiin julkaisuihin. Tässä tutkielmassa lähteinä käytetyt akateemiset julkaisut löydettiin pääasiallisesti hyödyntämällä Turun Yliopiston kirjaston *UTU Volter*-järjestelmää, joka kerää Turun Yliopiston kirjaston aineistot yhteen tietokantaan. Osa tutkielman akateemisista lähteistä löydettiin myös hyödyntämällä *Google Scholaria*. *Googlea* hyödynnettiin pääasiassa yrityslähteiden löytämiseen. Yrityslähteitä tässä tutkielmassa ovat esimerkiksi PLM-järjestelmiä tarjoavien yritysten kirjoittamat artikkelit PLM:n ja raportoinnin yhteydestä. Tiedonhaussa käytettiin hakusanoja ”PLM” tai ”Product Lifecycle Management” ja "Business Intelligence"tai "reporting". Myös näiden yhdistelmiä ja muita sanamuotoja hyödynnettiin, esimerkiksi "report engine"tai "reporting tool". Kuvassa 1.1 on eritelty käytettyjä lähteitä ja niiden lukumääriä tarkemmin.



Kuva 1.1: Tiedonhakuprosessi

Tutkielman luvussa 2 hyödynnetään pääasiallisesti akateemisia lähteitä, jotta voidaan luoda ymmärrys PLM-strategiasta, -järjestelmistä sekä näiden yhteydestä raportointiin. Perehdymme ensin alaluvussa 2.1 PLM:n merkitykseen ja sen ominaispiirteisiin. Käytämme aiempaa kirjallisuutta perustelemaan PLM:n merkityksellisyyttä ja sen yhteyttä liiketoimintatiedon hyödyntämiseen, minkä tehokas raportointi mahdollistaa.

Alaluvussa 2.3 määritellään akateemisen kirjallisuuden avulla raportointimoottorin ja -työkalun käsitteet ja rakenne. Tämän lisäksi perehdymme raportoinnin ja PLM-järjestelmän yhteyteen. Selvitämme myös, millainen PLM-järjestelmä on toimintaympäristönä ja millaisia niiden käyttäjät ovat.

Tutkielman luvussa 3 esittelemme tapaustutkimuksen toimintaympäristön, Sovelia PLM:n, ominaisuuksia ja erityispiirteitä pohjautuen ohjelmiston dokumentaatioon. Tarkastelemme Sovelia PLM:n vanhaa raportointityökalua ja reflektoimme sen asettamia haasteita uuden raportointityökalun kehittämiselle. Tämän jälkeen perehdymme pääasiallisesti PLM-ohjelmistoja tarjoavien yritysten julkaisuihin PLM:n ja raportoinnin yhteydestä, josta päädymme tarkastelemaan kuutta olemassa olevaa raportointityökalua ja niiden ominaisuuksia. Nämä kuusi raportointityökalua valittiin, sillä niiden ominaisuudet ja toimintaympäristöt vastasivat uutta kehitettävää raportointityökalua.

Lopulta vertaamme havaintojamme kuudesta valitusta kehitettävän uuden raportointityökalun ominaisuuksiin ja erityispiirteisiin. Tämän perusteella pyrimme toteamaan, mitkä lähestymistavat soveltuivat Sovelia PLM:n tapauksessa ja mitkä eivät.

2 Raportointi ja PLM-järjestelmät

2.1 PLM-strategia ja PLM-järjestelmät

Tuotteen elinkaaren hallinta eli PLM voidaan nähdä yrityksen strategiana hallita tuotetietoja. PLM strategiana koostuu tuotteista, organisaatioista, työmenetelmistä, prosesseista, ihmisistä ja lopulta usein myös tietoteknisestä elinkaaren hallintajärjestelmästä. Tietoteknisen elinkaaren hallintajärjestelmän, eli PLM-järjestelmän, tarkoituksena on mahdollistaa ja helpottaa PLM-strategian käyttöönottoa ja hyödyntämistä käytännössä. PLM-järjestelmä sisältää siis erilaisia toiminnallisuuksia, joiden avulla sen käyttäjät voivat hallita tuotetietoja keskitetyssä järjestelmässä tuotteen koko elinkaaren ajan.

Tuotteen elinkaari voidaan jakaa alku-, keski- ja loppuvaiheeseen. Tuotteen elinkaaren pääpiirteet on hyvä ymmärtää, jotta PLM-käsitettä voidaan tarkastella syvällisemmin. Bouhaddou [9] *"PLM Model for Supply Chain Optimization"* määrittelee tuotteen elinkaaren vaiheet ja niiden piirteet seuraavasti:

- Alkuvaiheessa tuotteen vaatimuksia määritellään ja tuote on luonnosvaiheessa. Luonnosvaiheessa tuotetta voidaan kutsua prototyypiksi (engl. *prototype*) tai mallinnukseksi (engl. *mockup*).

- Keskivaiheessa tuote siirtyy tuotantoon ja valmistukseen. Tässä vaiheessa toteutetaan laadunvalvontaa ja kasaamista ja voidaan puhua jo varsinaisesta tuotteesta. Valmis tuote siirtyy jakeluverkoston kautta itse asiakkaalle. Kun

tuote on asiakkaalla, korostuu tuotteen käyttö sekä mahdollinen huolto ja asiakastuki.

- Loppuvaiheessa tuotteen elinkaari päättyy. Tuotteen valmistusta ei koeta enää tarpeelliseksi, joten tässä vaiheessa huomio keskittyy tuotannon lopettamiseen ja tuotteen kierrätykseen. [9]

Alemanni ym. [5] esittävät artikkelissaan *"Key performance indicators for PLM benefits evaluation: The Alcatel Alenia Space case study"*, että PLM-strategian keskittyessä olennaisesti tuotedatan hallintaan on PLM-ohjelmisto eli PLM-järjestelmä olennainen osa strategian hyödyntämistä käytännössä. Alemanni korostaa, että PLM-järjestelmän kehittäjän tulee kuitenkin toimia yhteistyössä asiakkaiden kanssa, jotta tuotteiden elinkaaren eri vaiheet ja prosessit voidaan sisällyttää osaksi ohjelmiston toimintoja asiakaskohtaisesti. PLM-käsitteeseen liittyvien määritelmien lisäksi on tärkeää ymmärtää PLM-strategian ja -ohjelmistojen hyötyjä, jotta niiden hyödyntämisen motiivit voidaan ymmärtää. Tarkoituksena on vastata siihen, miksi ylipäätään PLM-järjestelmiä käytetään ja kehitetään.

2.2 PLM-strategian hyödyt ja merkitys

Lyhyellä aikavälillä PLM-strategian ja PLM-järjestelmän käyttöönotto voivat vähentää aikaa, jota käytetään työntekijöiden jokapäiväisten työtehtävien suorittamiseen. Strategian avulla yrityksen tuotetiedot ovat keskitetysti saatavilla eikä ajantasaista tietoa tarvitse kysellä eri osastojen välillä. Tämä johtaa siihen, että työntekijät voivat käyttää enemmän aikaa tehtäviin, jotka tuottavat yritykselle lisäarvoa. Lisäksi tuotteiden rakenteiden ymmärtäminen ja visualisointi helpottuu PLM-järjestelmän käyttöönoton myötä. Tuoterakenteiden ymmärrystä ja jaettavuutta eri osastojen välillä voidaan parantaa entisestään myös PLM-järjestemän raportointiominaisuuksilla. [5]

Pitkällä aikavälillä hyödyt alkavat näkyä konkreettisemmin PLM-strategiaa hyödyntävien yritysten tunnusluvuissa, erityisesti myyntikatteessa. PLM-järjestelmien keskeinen hyöty on prosessien suoraviivaistaminen, mikä johtaa usein tuotteiden saamiseen markkinoille nopeammin ja useammin. Kun tuotteet pääsevät nopeammin suunnittelusta markkinoille, niiden suunnitteluun ja kehittämiseen käytetyt kustannukset luonnollisesti laskevat. [9] [5]

2.2.1 Osaluettelo PLM-järjestelmässä

Yksi PLM-järjestelmän tärkeimmistä toiminnallisuuksista on tuotteen osaluettelon (engl. *Bill of Materials, BOM*) esittäminen organisoidusti.[8] Yksinkertaisuudessaan osaluettelo on lista kaikista osista, joita tarvitaan tuotteet valmistamiseen. Osaluettelossa jokaiseen yksittäiseen osaan voidaan liittää useita tietokenttiä, kuten tuotteen valmistaja, versio, materiaali ja määrä. Tuoterakenne on osaluettelo, joka koostuu hierarkkisesti osakokoonpanoista, välikokoonpanoista, osakomponenteista ja yksittäisistä osista, eli se kerää dataa siitä, kuinka eri tuotteen komponentit ovat riippuvaisia toisistaan. Osaluetteloa voidaan käyttää viestintään, esimerkiksi valmistuskumppanien välillä, tai se voidaan rajoittaa yhteen tuotantoyksikköön. [10]

Koska osaluettelot ovat hyvin olennainen osa PLM-järjestelmää, ovat ne tärkeä kohde myös raportoinnille. [6] Osaluetteloista tuotetut raportit voivat analysoida rakennetta pintaa syvemmältä sekä luoda helposti ymmärrettävän yleiskatsauksen massiivisen osaluettelon omaavaan tuotteeseen tarjoamalla samalla esimerkiksi graafeja ja statistiikkaa tuotteesta. Koska raportit voidaan tuottaa erillisinä sähköisinä dokumentteina, voidaan laskentaa jatkaa esimerkiksi Excel-raporttien tapauksessa, tai erityisesti PDF-raportit ovat omiaan arkistoinnille myöhempää käyttöä varten.

2.2.2 Raportointi ja Business Intelligence

Liiketoimintatiedon hyödyntämisellä , (engl. *Business Intelligence*, BI) tarkoitetaan yrityksen kykyä hyödyntää dataa merkityksellisellä tavalla. PLM:n kontekstissa BI korostuu etenkin tuotetiedon hyödyntämisessä. Tätä PLM:n ja BI:n yhteyttä on tutkinut Bosch-Mauchand ym. [11] *"Preliminary Requirements and Architecture Definition for Integration of PLM and Business Intelligence Systems"*. Bosch-Mauchand totetaa, että PLM järjestelmä kulkee käsikädessä BI:n kanssa ja PLM-järjestelmän tuotedatan integraatio ja sen jaettavuus eri järjestelmien välillä on hyvin tärkeää tuotetiedon merkityksellisen hyödyntämisen kannalta. Bosch-Mauchand erittelee, että jotkin PLM-järjestelmät tarjoavat erillisiä moduuleja raporttien tuottamiseen, mutta harva raportointityökalu tai -moduuli hyödyntää BI:n periaatteita. Bosch-Mauchandin mukaan varsinkin kahden tyyppisillä raporteilla voidaan tuottaa lisäarvoa [11]:

- Dokumenttien ja objektien määrällinen analysointi. Esimerkiksi näiden summien tai tyyppien laskenta.
- PLM-järjestelmien ominaisuuksien käyttö. Esimerkiksi tuotteen osien uudelleenkäyttö ja tietokantakyselyt. [11]

Näiden lisäksi raportointia voidaan hyödyntää IT-hallinnon osa-alueilla, kuten esimerkiksi järjestelmän suorituskyvyn monitoroinnissa.

2.3 Raportointimoottorit ja -työkalut

Raportointimoottorit ja -työkalut ovat molemmat tietojen käsittelyyn ja analysointiin tarkoitettuja ohjelmistoja. Tämän tutkielman kontekstissa nämä käsitteet määritellään seuraavanlaisesti: Raportointimoottorit ovat ohjelmistoja, jotka automatisoivat raporttien luomisen ohjelmallisesti. Tyypillisesti raportointimoottori saa syötedatan, jonkinlaisen ohjeistuksen tai raporttipohjan (engl. *report template*) ja kaikki

logiikka näiden yhdistämiseen sekä lopullisen raportin muodostamiseen jää raportointimoottorin vastuulle. [12] Raportointityökalu on taas laajempi termi, ja sillä tarkoitetaan ohjelmistoa, joka on suunniteltu auttamaan raporttien luomisessa, hallinnassa ja jakelussa. Yksinkertaistettuna raportointityökalut ovat suunniteltu auttamaan käyttäjiä luomaan raportteja, kun taas raportointimoottorit ovat suunniteltu automatisoimaan itse raporttien luominen jonkin annetun lähdedatan perusteella.

Raportointimoottori on tyypillisesti taustalla toimiva, esimerkiksi palvelinsovellus, joka keskittyy raporttien prosessointiin ja renderöimiseen. Se huolehtii esimerkiksi tietojen hakemisesta, yhdistämisestä ja muotoilusta ennalta määritettyjen raporttipohjien ja ohjeiden perusteella. Se ei sisällä itse käyttöliittymää ja sen toimintoja käytetään ohjelmallisesti. Raportointityökalu sisältää taas usein jonkinlaisen käyttöliittymän, jonka avulla käyttäjä voi luoda uusia tai muokata olemassa olevia raportteja. Raportointityökalut voivat olla sisäänrakennettuja johonkin järjestelmään, esimerkiksi PLM-järjestelmään, mutta ne voivat olla myös ulkoisia ohjelmistoja, jotka pääsevät käsiksi analysoitavaan dataan esimerkiksi jonkin rajapinnan kautta. Koska raportointimoottori voi olla konfiguroitavissa loppukäyttäjien toimesta raportointityökalun avulla, tulee molempia komponentteja kehittäessä huomioida loppukäyttäjien yksilölliset ja monipuoliset tarpeet. [3]

Bambang Prasetya Adhi ym. [3] vertailevat tutkimusartikkelissa *"Performance comparison of reporting engine birt, jasper report, and crystal report on the process business intelligence"* suosituimpia kaupallisia (SAP Crystal Reports) ja avoimen lähdekoodin (BIRT ja Jasper Report) raportointityökaluja ja niiden alla toimivia raportointimoottoreita. Ahdi ym. käyttävät kokeellisia menetelmiä mitatakseen kolmea raportointityökalun osa-aluetta [3]:

- Soveltuvuutta, esimerkiksi kuinka hyvin raportointimoottori tukee erilaisen lähdedatan käyttöä
- Käytettävyyttä, joka ilmenee oppimisen helppoutena sekä toiminnallisuuden

loogisuutena ja tehokkaana käyttönä

- Tehokkuutta, joka mittaa itse järjestelmän tehokkuutta, esimerkiksi suoritusaikaa [3]

Näitä osa-alueita arvioimalla voidaan tehdä perusteltuja päätöksiä raportointityökalun ja -moottorin valinnasta, joten nämä ovat tärkeitä seikkoja ottaa huomioon näiden suunnittelussa ja kehityksessä.

2.3.1 Raportointimoottorin rakenne ja prosessi

Prosessina raportointimoottori toimii kolmella tasolla: **data**-, **logiikka**- ja **esitystasolla**. [12]

Datatasolla raportointimoottori voi hakea dataa suoraan tietotokannasta tai esimerkiksi API:n eli ohjelmointirajapinnan (engl. *Application Programming Interface*) kautta. Raportointimoottorin tapauksessa ohjelmointirajapinta voi olla esimerkiksi hakurajapinta, jonka taustalla toimivan hakumoottorin avulla raportointimoottori voi hakea tarvitsemaansa dataa täsmällisemmin. Varsinkin PLM-järjestelmän kontekstissa hakumoottori on hyvin keskeinen osa PLM-järjestelmää. Datataso määrittelee siis mistä ja miten raportointimoottori hankkii lähtödataa sekä millaiset lähtötiedot sillä on koostaa raportti. [12] Näihin lähtötietoihin lukeutuu esimerkiksi mahdolliset raporttipohjat ja muut käyttäjän määrittämät asetukset.

Logiikkatasolla raportointimoottori jäsentelee lähtödataa ja suorittaa laskentaa toisella tasolla. Lähtödatan formaatti on usein historiallisesti ollut XML (engl. *Extensible Markup Language*) sen ollessa yksi internetin yleisimmin käytetyistä dataformaateista, mutta JavaScriptin yleistyttyä myös JSON (engl. *JavaScript Object Notation*) noussut suosituksi tiedostomuodoksi. JSONin etuna on sen suora integraatio JavaScriptin yhteyteen sekä sen nopeus verrattuna vanhempaan XML-standardiin. [13] Logiikkatason toteuttaman laskennan avulla lähtödatasta voidaan

luoda esimerkiksi graafeja ja sekä taulukoida dataa, mikä mahdollistaa lasketut sarakkeet ja summat. Tämä voidaan nähdä raportointimoottorin ytimenä, sillä se tuottaa merkityksellistä dataa usein vaikeaselkoisesta lähdedatasta. [12] Lisäksi logiikkatason tulee jäsentää data siten, että se on mahdollista kirjoittaa tiedostoon esitystasolla.

Esitystasolla data kirjoitetaan tiedostoon, jolloin se voidaan tallentaa käyttäjäystävällisessä tiedostoformaatissa. [12] Suosituimpia tiedostoformaatteja ovat HTML-, Excel- ja PDF-tiedostoformaatit. [12] Esitystasolla raportin merkitys ilmenee käyttäjälle: lähtödata on esitetty helposti omaksuttavassa ja ymmärrettävässä muodossa, sekä raportti on luettavissa ja jaettavissa helposti yksittäisenä tiedostona.

2.3.2 Raportointi PLM-järjestelmässä

Kuten luvussa 2.2 todettiin, yksi PLM-järjestelmän hyödyistä on yksittäisten työntekijöiden työmäärän vähentäminen ja prosessien suoraviivaistaminen. Tietoteknisten järjestelmien etuna on varsinkin automoitu laskenta, jota hyödyntämällä voidaan vähentää inhimillisiä virheitä. [14] [1] Automoitua laskentaa hyödynnetään erityisesti erilaisten raporttien muodostamisessa.

Koska PLM:n tarkoituksena on mahdollistaa koko tuotantoketjun yhteistyö asiakkaiden, kehittäjien, toimittajien ja valmistajien välillä tuotteen eri elinkaaren vaiheissa, [9] on tärkeää, että tuotetieto elinkaaren eri vaiheissa on dokumentoitavissa, analysoitavissa ja helposti jaettavissa. Koska PLM-järjestelmien tietomallit ovat usein ohjelmistokohtaisia ja harvemmin standardinomaisia[15] on tärkeää, että tuotedataa on mahdollista viedä myös itse järjestelmän ulkopuolelle tallennettavaksi ja jaettavaksi.

PLM:n kontekstissa raporteilla tarkoitetaan tuotedataa kokoavia ja analysoivia kokonaisuuksia. Raportit voivat olla esimerkiksi digitaalisia PDF- tai Exceldokumentteja, jotka kokoavat tuotetietoja ja suorittavat laskentaa visualisoimalla

dataa esimerkiksi kuvaajin. Toinen PLM-järjestelmille tyypillinen raporttimuoto on interaktiiviset "kojelautoja"(engl. *dashboards*), jotka kokoavat useita kuvaajia ja laskettuja arvoja yksittäiseen käyttäjäystävälliseen näkymään, tarjoamalla esimerkiksi jonkin Web-käyttöliittymän. Tässä tutkielmassa keskitytään enemmän raporttidokumenttien tuottamiseen ohjelmallisesti, mutta usein näihin digitaalisiin dokumentteihin on myös mahdollista upottaa kojelautamaisia ominaisuuksia, kuten kuvaajia ja tilastoja.

Raporttidokumenttien tuottamista varten monet PLM-järjestelmät tarjoavat PLM-ohjelmiston osana raportointityökalun, jonka tarkoituksena on kerätä ja analysoida dataa kokoamalla sitä dokumenttitiedostoformaatteihin ennalta määriteltyjen sääntöjen ja mallien mukaisesti. Lisäksi raportointityökalu voi tarjota jonkinlaisen käyttöliittymän raporttien muokkaamiseen ja konfigurointiin. Alan standardina dokumenttiformaateista raporttien tapauksessa lienee PDF-, Excel- ja HTML-pohjaiset raportit, sillä useimmat raportointityökalut tarjoavat raportteja näissä tiedostomuodoissa ja ne ovat myös tuttuja suurimmalle osalle ohjelmiston käyttäjistä.

2.3.3 PLM-järjestelmä toimintaympäristönä

Rohleder ym. [7] käsittelee tutkimuksessaan *"Requirements Engineering in Business Analytics for Innovation and Product Lifecycle Management"* vaatimusten määrittelyä liiketoimintatiedon hyödyntämisessä PLM-järjestelmissä. Rohleder ym. korostaa, että PLM-järjestelmissä toimitaan usein massadatan (engl. *Big data*) parissa, sillä heidän tutkimuksen mukaan yksittäisen auton tuoterakenne voi koostua noin 120 tuhannesta yksittäisestä osasta, joilla jokaisella on tyypillisesti omia CAD-malleja (tietokoneavusteisen suunnitteluohjelman luomia tiedostoja), piirustuksia ja metadataa. Lisäksi tuotteen useat versiot ja variantit kasvattavat lopullista tietomäärää eksponentiaalisesti. Lisäksi PLM-järjestelmiä käyttää tyypillisesti useita työntekijöitä yrityksen eri osastoissa, jolloin näiden työnkulkujen erilaisuus lisää PLM-

järjelmien datan kompleksisuutta entisestään. Kompleksisuus johtaa usein siihen, että valmiit raportointimoottorit, varsinkin esimerkiksi taloudelliseen raportointiin erikoistuneet, eivät välttämättä sovi sellaisenaan käytettäväksi PLM:n kontekstissa. [7]

Kuten kappaleessa 2.2.1 todettiin, osaluettelot ovat keskeinen osa PLM-järjestelmään tallennetun datan esittämistä. Koska tuoteobjektit koostuvat osaluetteloista, myös PLM-järjestelmässä tuotteista koostettavat raportit perustuvat osaluetteloista kerättyyn lähtödataan. PLM-järjestelmän raportointimoottorit ovat siten erikoistuneita jäsentelemään ja kokoamaan hierarkkista dataa. [7] PLM-järjestelmän tarjoamille raporteille on olennaista tuotteeseen ja sen kehitykseen liittyvät seikat, kuten esimerkiksi tuotteen osien toimittajien jakauma tai tuotteen muokkaushistoria. Lisäksi osaluettelon perusteella voidaan laskea yksittäisten osien summia rakenteessa tai esimerkiksi luoda raportteja tietyistä tuotteen osista, jotka täyttävät annetut kriteerit.

Mahdollisen raportointimoottorin ohella PLM-järjestelmät sisältävät usein hakumoottorin, jonka avulla voidaan etsiä tehokkaasti ja tarkasti tietokannasta annettujen kriteerien mukaisesti. Tiedon haku on yksi PLM-järjestelmän keskeisimmistä ominaisuuksista. [16] Raporttien muodostamisessa ulkoisen hakumoottorin hyödyntäminen vähentää itse raportointimoottorin kuormaa, jolloin raportointimoottorin toiminallisuuden kehittämisessä voidaan keskittyä enemmän itse laskentaan ja lisäarvon tuottamiseen. Täten PLM-järjestelmän tapauksessa lähtödatan hakeminen voi tapahtua esimerkiksi jonkin hakurajapinnan välityksellä, jolloin raportteja voidaan muodostaa tuoterakenteiden lisäksi esimerkiksi jonkin tietyn hakulausekkeen perusteella.

PLM-järjestelmien käyttäjät ovat tyypillisesti suhteellisen suuren mittakaavan teollisuusyrityksiä. Useissa tapauksissa myös tuoterakenteet ovat valtavia [7], joten raportointimoottorin tulee olla tarpeeksi tehokas ja optimoitu, jotta myös suurista

tietorakenteista on mahdollista koostaa raportteja siedettävässä suoritusajassa. Raportointimoottorin logiikkatason lisäksi PLM-järjestelmien käyttäjillä on myös tarpeita muokata raporttien ulkoasua raportointimoottorin esitystasolla. Esimerkiksi yrityksen logojen ja raportin visuaalisen ilmeen muokkaaminen on olennainen osa taas raportointityökalun toiminnallisuutta. Koska raportointityökalun tulee olla sisällytetty saumattomasti muihin PLM-järjestelmän ominaisuuksiin hyvän käyttäjäkokemuksen varmistamiseksi, useat PLM-järjestelmiä tarjoavat yritykset käyttävät PLM-järjestelmissään tätä järjestelmää varta vasten kehitettyjä raportointityökaluja. Tarvetta varta vasten kehitetylle raportointityökalulle lisää myös mainittu PLM-datan kompleksisuus, koska olemassa olevat raportointityökaluratkaisut eivät välttämättä sovellu sellaisenaan toimimaan kompleksisen PLM-datan kanssa.

3 Tapaus: Sovelia PLM:n raportointityökalu

3.1 Sovelia PLM

Kehitettävän raportointityökalun toimintaympäristönä toimii kaupallinen PLM-järjestelmä, Sovelia PLM. Sovelia PLM:n kehitys on alkanut yli 30 vuotta sitten ja sen kehitys jatkuu aktiivisesti edelleen. Kuten muutkin PLM-järjestelmät, Sovelia PLM pyrkii ratkaisemaan valmistusalan yritysten haasteita liittyen tuotteen datan hallintaan sen koko elinkaaren ajan. [4] Sovelia PLM:n erityispiirteenä on sisältämät valmiiksi konfiguroidut *"templatet"* eli valmiit mallit objekteille ja prosesseille, jotka ovat muokattavissa asiakkaan tarpeiden mukaan. Lisäksi malleihin kuuluu muita valmiiksi konfiguroituja työkaluja että alalla hyväksi todettuja prosesseja. [17]

Sovelia PLM koostuu objekteista ja objektilinkeistä. Objekteja voidaan määritellä niiden attribuuttien avulla. Objektit voivat olla osia, piirustuksia, dokumenttilinkkejä tai linkkejä muihin toimintoihin. [17] Objektilinkkien avulla voidaan muodostaa osaluetteloita, jotka ovat tärkeitä raportoinnin kohteita. Näiden lisäksi toinen merkittävä konsepti Soveliassa on käyttäjä ja käyttäjäryhmät. Yksinkertaisuudessaan järjestelmällä voi olla luonnollisesti useita käyttäjiä ja käyttäjät voivat kutsua eri käyttäjäryhmiin. Käyttäjäryhmiä voi olla useita, mutta tärkein niistä on ymmärtää *"admin"*-ryhmä (engl. *administrator*), johon kuuluu pääkäyttäjät eli järjestel-

mänvalvojat. Järjestelmänvalvojalla on luonnollisesti oikeuden muuttaa järjestelmän asetuksia. Pääkäyttäjän konsepti on tärkeä ymmärtää, sillä osa kehitettävän raportointityökalun ominaisuuksista on saatavilla vain pääkäyttäjälle. Käyttäjäryhmien lisäksi jokaisella käyttäjällä on lisenssi, joka määrittelee osaltaan käyttäjän oikeuksia rajoittamalla esimerkiksi objektien ja objektilinkkien muokkausoikeuksia. [18]

Raportointi Sovelia PLM-järjestelmässä

Sovelia PLM ja muut PLM-järjestelmät poikkeavat valmiiden raportointityökaluratkaisujen tyypillisistä toimintaympäristöistä, sillä kuten luvussa 2.3.3 huomattiin, PLM-data on usein kompleksista ja monimuotoista. Siemensin PLM-järjestelmän verkkosivuilla [6] julkaistussa artikkelissa *"The Challenge of Getting High Quality Reports out of PLM"* esitellään haasteita, joita liittyy korkealaatuisten raporttien tuottamiseen PLM-järjestelmistä. Artikkeli nostaa esille osaluetteloiden (BOM) merkityksen PLM-järjestelmän raportoinnissa: on tärkeää että raporttien saama ja tuottama data on luotettavaa ja laadukasta, jotta päätöksenteko raporttien pohjalta olisi mahdollista. Osaluetteloihin voi tulla jopa satoja muutoksia päivittäin, joten raporttien tapauksessa on tärkeää, että niiden käyttäjät tietävät työskentelevänsä oikean datan kanssa. Täten raporttien ajantasaisuus ja selkeät aikaleimat ovat hyvin tärkeitä PLM-järjestelmän ja myös Sovelian tuottamille raporteille. Sovelia PLM:n raportointimoottorin etuna on hakurajapinnan käyttö, joka palauttaa aina hakupyynnön mukaisesti ajankohtaista dataa, mikä vähentää mahdollisten virheellisten tietojen määrää. [6]

PLM-datan kompleksisuus tekee raportointityökalun kehittämisestä haasteellista, sillä myös Sovelia PLM vaatii erityisesti tarkoitusta varten kehitetyn raportointityökaluratkaisun. Tarkoitusta varten kehitetty ratkaisu on ohjelmistokehittäjälle työläs rakentaa alusta asti itse, mutta sen etuna on sen täysi muovautuvuus tarvetta varten. On kuitenkin todettava, että myös valmiin raportointimoottorin ja -

työkalun implementoiminen Sovelia PLM:n toimintaympäristöön olisi todennäköistä työlästä juuri PLM-datan kompleksisuuden vuoksi.

Sovelia PLM:n vanha raportointityökalu

Sovelia PLM -järjestelmässä on tuotannossa ja asiakkailla käytössä vanha raportointiyökalu, joka tarjoaa raportteja PDF, Excel ja ZIP-tiedostoformaateissa. Näistä poikkeavimpia ovat ZIP-raportit eli ZIP-arkistot, jotka sisältävät useita PDF- tai Excel-raportteja pakattuna yhteen tiedostoon. Täten ZIP-raporttiin voidaan sisällyttää esimerkiksi raportti tuoterakenteen jokaiselle yksittäiselle objektille, jolloin ZIP-paketin kansiorakenne vastaa itsessään tuotteen rakennetta. Raporttidokumenttein data kerätään *"rakenneagentin"* avulla, joka on nimensä mukaisesti osaa kulkea objekti-linkki -suhteita pitkin ja täten kerätä tarvittavan lähdedatan raportin koostamista varten. Itse raporttidokumentteihin kirjoittaminen tapahtuu Java-ohjelmointikielen Apache FOP-ohjelmakirjaston [19] avulla XSL- (engl. *Extensible Stylesheet Language*, kieliperhe, joka mahdollistaa XML-pohjaisten tiedostojen ulkoasun ja rakennemuutoksen määrittelyn) ja XML-tietorakenteisiin tallennetun datan antaman ohjeistuksen avulla.

Vaikka tämä ratkaisu on toimiva ja edelleen kelvollinen tekniikka dokumenttien muotoiluun, ja siten raporttien generointiin ohjelmallisesti ennalta määritellyn datan mukaisesti, se kohtaa ongelmia datamäärien kasvaessa. Näihin ongelmiin kuuluu esimerkiksi liiallinen muistinkäyttö palvelimella, prosessin hitaus varsinkin suurempien tuoterakenteiden tapauksissa ja raporttien konfiguroinnin haasteellisuus sen ollessa täysin mahdotonta PLM-järjestelmän loppukäyttäjille. Lisäksi toiminnon ylläpito alati muuttuvassa toimintaympäristössä muuttuu todennäköisesti haasteellisemmaksi tulevaisuudessa, kun uusi teknologia syrjäyttää vanhaa yhä useammilla osa-alueilla, samalla kun vanhalle pohjalle rakennettu raportointimoottori ei pysy samalla tavalla muutoksen tahdissa.

Uuden raportointityökalun haasteet

Koska vanhan raportointityökalun modernisointi ja päivittäminen nykypäivään vaatisi perustavanlaatuisia muutoksia koko järjestelmään, päädyttiin valitsemaan kehityskohteeksi kokonaan uuden raportointityökalun kehittämisen. Uuden työkalun tarkoituksena on integroitua paremmin nykyiseen Web-pohjaiseen käyttöliittymään ja tarjota käyttäjäystävällisemmän kokemuksen raporttien luomiseen. Raporttien luominen tuoterakenteista tapahtuisi käyttäjän näkökulmasta paremmin integroituna Web-käyttöliittymään tarjoten mahdollisuuden seurata meneillään olevien raporttien edistymistä sekä perua raporttien koostamisprosesseja käyttöliittymästä käsin. Lisäksi pääkäyttäjällä tulisi olla mahdollisuus muokata ja luoda uusia raporttityyppejä muokkaamalla esimerkiksi raporteissa esitettäviä sarakkeita ja tuoterakenteiden suodattamiseen liittyviä sääntöjä. Myös raporttien ulkonäön tulee olla muokattavissa pääkäyttäjän toimesta, mikä tarkoittaa niiden visuaalisen ilmeen muokkausta lisäämällä asiakasyrityksen logoja, muokkaamalla väriteemaa ja asettelua asiakkaan tarpeiden mukaisesti.

Uuden raportointityökalun tulee ensimmäisessä vaiheessa sisältää samat toiminnallisuudet kuin vanha raportointityökalu, mutta tarkoituksena on tuottaa raportteja huomattavasti nopeammin aiheuttamalla samanaikaisesti vähemmän kuormaa palvelimelle. Raporttien sisällön ja yleisen rakenteen tulee vastata vanhempia raportteja identtisesti, mutta raporttien visuaalinen ilme tulee päivittää nykypäivän standardien mukaiseksi. Lisäksi vaaditaan syvällisiä teknologiatutkimusta ja -analyysiä olemassa olevien raportointityökalujen ja -moottorien ratkaisuista, Sovelia PLM:n ohjelmakannassa jo olemassa olevista komponenteista ja muista mahdollisesti hyödyllisistä teknologioista.

3.2 Nykyiset erilliset raportointityökalut

Ennen käytettävien teknologioiden ja tekniikoiden valintaa on hyvä perehtyä jo olemassa oleviin raportointityökaluihin ja -moottoreihin, jotta voidaan muodostaa kokonaiskuva yleisesti käytössä olevista ratkaisuista. Hyvän kokonaiskuvan avulla voidaan tehdä valistuneita päätöksiä siitä, miten raportointityökalua kannattaa lähteä kehittämään sekä ymmärtää käytössä olevien tekniikoiden hyötyjä ja rajoituksia PLM-järjestelmän kontekstissa. Analysoimme kuutta suosittua itsenäistä (engl. *standalone*) raportointityökalua, joista neljä ovat kaupallisia ja kaksi ovat avoimen lähdekoodin ohjelmistoja. Juuri nämä kuusi raportointityökalua valittiin tarkasteltavaksi, sillä ne ovat kaikki paikallisesti asennettavia ohjelmistoja (engl. *On-premise software*) kuten myös Sovelia PLM. Täten tässä tutkimuksessa ei keskitytä esimerkiksi ulkoisten pilvipalveluiden raportointityökalu verkko-ohjelmointirajapintoihin (engl. *Web API*). Yksi esimerkki tällaisesta verkkopohjaisesta API:sta on *Bold Reports* [20], joka tarjoaa monipuolisen verkkorajapinnan kautta toimivan raportointityökalun. Toinen kriteeri tarkasteluun valinnalle oli käytettävä teknologia, sillä Sovelia PLM:n nykyinen arkkitehtuuri nojaa vahvasti Java- ja JavaScript-ohjelmointikieliin, joten täysin uuden ohjelmointikielen lisäämistä järjestelmäarkkitehtuurin pinoon ylläpidettäväksi ei pidetty mielekkäänä vaihtoehtona. Lisäksi esimerkkejä sekä Javaettä JavaScript-pohjaisia raportointityökaluista löytyi runsaasti, joten uuden teknologian valitseminen ei vaikuttanut välttämättömyydeltä. Kolmas tärkeä kriteeri oli ulostulodataformaattien monipuolisuus: koska Sovelia PLM -järjestelmään kehitettävän raportointityökalun tulee tuottaa raportteja eri tiedostoformaateissa, tulee myös tarkasteltavien työkalujenkin.

Puhtaasti kaupallisia ohjelmistoja ovat JavaScript-pohjainen *ActiveReportsJS*, SAP:in, maailman suurimman yritysohjelmistoja tarjoavan yrityksen [21], kaksi raportointityökalua *SAP Business Objects* ja *SAP Crystal Reports* ja Oraclen, talouslehti Forbesin mukaan vuoden 2023 80. suurimman julkisen yrityksen [22], rapor-

tointityökalu *Orancle BI Publisher*. Vahva esimerkki taas avoimen lähdekoodin vaihtoehdosta on *JasperReports*, joka kutsuu itseään suosituimmaksi avoimen lähdekoodin Java-raportointikirjastoksi. [23]. *JasperReports* sisältää myös visuaalisen suunnittelutyökalun, *iReport Designerin*, joten *JasperReports* on kokonainen raportointityökaluohjelmisto. *JasperReports* sisältää myös vaihtoehtoisen maksullisen lisenssin yrityskäyttöön tarjoten yritysasiakkaille parempaa tukea ja ratkaisua suuremman skaalan raportointiin. [24] Toinen avoimen lähdekoodin raportointityökalu on myös Java-pohjainen *BIRT* (lyhenne engl. *"Business Intelligence Reporting Tool"*), joka tarjoaa monipuolisia raportointimahdollisuuksia täysin avoimen lähdekoodin ja Eurooppalaisen *Eclipse Foundationin* ylläpitämä ohjelmisto. [25] *BIRT* on keskittynyt raportoinnin lisäksi myös Business Intelligencen hyödyntämiseen asiakas- ja web-applikaatioissa, joten se sisältää monia erilaisia visualisointimahdollisuuksia ja raportointimoottorin lisäksi myös graafisen raporttikonfigurointityökalun.

Nykyisten raportointityökalujen yleiskuva

Raportointityökalua kehittääkseen osaksi PLM-järjestelmää varten tulee lisenssien lisäksi ottaa huomioon myös muita ominaisuuksia. Liitteenä olevassa taulukossa A.1 luokitellaan raportointiyökalujen yleisiä ominaisuuksia. Taulukoiden data on kerätty taulukossa mainituista lähteistä, kuten ohjelmiston verkkosivuilta, ohjelmakirjaston dokumentaatiosta tai myyntiesitteistä. Jo mainitut lisenssit ovat eritelty kaupallisiin ja avoimen lähdekoodin ohjelmistoihin. Avoimen lähdekoodin ohjelmistot tarjoavat asiantuntijoille lähdekoodin perehtymisen avulla syvän katsauksen raportointityökalun sisäiseen toiminnallisuuteen, mutta tässä tutkimuksessa keskitymme enemmän pintapuolisempiin ominaisuuksiin ja teknologiavalintoihin tarkemman lähdekoodianalyysin sijasta. Yksi tällaisista ominaisuuksista on lähdedatan formaatti ja mahdolliset datalähteet. Kaikki tutkitut raportointityökalut tukivat jotakin rakenteellisen datan tallentamiseen suunniteltua tiedostoformaattia eli tässä tapauksessa XML:ää

tai JSON:ia. Ainoastaan *(*ActiveReportsJS) ei tukenut XML:ää, mutta se tukee kuitenkin JavaScript-pohjaisena luonnollisesti JSON:ia. Rakenteellisen datan tukeminen tärkeää varsinkin PLM:n datan tapauksessa, jossa objektien ja siten tuotteiden väliset suhteet ovat avainasemassa. Näiden tiedostoformaattien lisäksi noin puolet tarkasteltavista raportointityökaluista tuki myös SQL-kyselyitä tietolähteenä (engl. *Structured Query Language*). Näiden lisäksi myös tavallisten tekstitiedostojen käyttö tietolähteenä on tyypillistä XML- ja JSON-tiedostojen lisäksi, kuten myös Excel-taulukkolaskentaohjelman tuottamat XLSX-tiedostot. Mielenkiintoista on, että myös Excelin tuottamat XLSX-tiedostot koostuvat itseasiasiassa vain ZIP-pakatuista XML-tiedostoista. [26]

Raportointityökalujen ohjelmointikieliä tarkastellessa suosituimmaksi ilmeni Java, sillä kaikki paitsi *(*ActiveReportsJS) olivat Java-pohjaisia. Yksi syy tähän voi olla Javan suuri ja vakaa suosio 1990-lopussa ja 2000-luvun alussa ohjelmointikielten suosiota mittaavan *TIOBE indexin* mukaan[27], jolloin suurinta osaa näistä raportointityökaluista alettu kehittää. Esimerkiksi *JasperReportsia* on alettu kehittää vuonna 2001 [28], *BIRTiä* vuonna 2004 [29] ja *BusinessObjectsia* jo vuonna 1995 ennen kuin se kuului SAP-tuoteperheeseen [30]. Kuten TIOBE indexistä voidaan havaita, on Java edelleen suhteellisen suosittu kieli käyttää vuonna 2023, mutta sen suosio on kuitenkin jo hieman laskussa. Raporttipohjien konfigurointi tapahtuu raportointityökaluissa pääsääntöisesti jonkin käyttöliittymän avulla. Käyttöliittymän lisäksi *ActiveReportsJS* tarjoaa mahdollisuuden konfiguroida raporttipohjia JSON-datan avulla, *BIRT* ja *JasperReports* XML-datan avulla ja *Oracle BI Publisher* RTF (engl. *Rich Text Format*, eroaa tavallisesta tekstistä tallentamalla tiedostoon myös fontit, fonttikoot ja yleisen muotoilun, mm. kursivoinnin) ja XSL-pohjaisesti. Tästä voidaan havaita, että visuaalinen käyttöliittymä raporttipohjien luomiselle on suosituin tapa toimia, mutta tiedostopohjainen raporttipohjien muokkaus on käypä vaihtoehto.

Nykyisten raportointityökalujen ulostuloformaatit

Raportointityökalun tärkein tehtävä on tuottaa raportteja, jolloin raporttien tulee olla myös helposti avattavissa ja esitettävissä eri laitteilla. Liitteenä olevassa taulukossa B.1 tarkastellaan samojen kuuden raportointityökalun tarjoamia ulostuloformaatteja, eli sitä millaisissa tiedostomuodoissa työkalut tarjoavat raportteja. Ulostuloformaattien valinta on erityisen tärkeää raportointityökalulle, sillä sen käyttäjät suosivat heille jo entuudestaan tuttuja tiedostomuotoja, sekä ohjelmistoteknisestä näkökulmasta, sillä raporttien tuottaminen erilaisissa tiedostomuodoissa vaatii usein tiedostomuotokohtaisia työkaluja.

Tiedostomuodot voidaan jaotella avoimiin ja suljettuihin tiedostomuotoihin. Avoimessa tiedostomuodossa olevan tiedoston avaaminen on mahdollista ilman kaupallista ohjelmaa, jolloin se on kenen tahansa avattavissa ja sillä on avoimesti saatavilla oleva spesifikaatio. [31] Suljettu tiedostomuoto on luonnollisesti avoimen tiedostomuodon vastakohta, jolloin spesifikaatio ei ole avoimesti saatavilla. Suljettu tiedostomuoto voi olla hyödyllinen joissain tapauksissa, esimerkiksi silloin jos tiedostoon tulee tallentaa tai sen avulla tulee esittää dataa siten että muut esimerkiksi avoimet tiedostomuodot eivät siihen pysty. Avoimien tiedostomuotojen suosiminen on kuitenkin kannattavaa, sillä se vähentää riippuvuutta kolmannen osapuolen ohjelmistoista tai tiedostomuotojen lisenssinhaltijoista. Pahimmassa tapauksessa tietoja voidaan lukea vain tietyillä ohjelmistopaketeilla, jotka voivat olla kohtuuttoman kalliita tai jotka voivat vanhentua. [31]

Ulostuloformaateista erityisen suosittu on Adoben kehittämä PDF (engl. *Portable Document Format*), jota käytetään erityisesti sähköiseen julkaisemiseen. PDF on ollut ISO-standardi vuodesta 2007 alkaen ja ISO (engl. *International Organization for Standardization*, kansainvälinen standardisoimisjärjestö) kuvailee sitä maailman suosituimmaksi dokumenttitiedostoformaatiksi [32]. Vaikka PDF on alun perin Adoben kehittämä kaupallinen ja suljettu tiedostomuoto, on sen spesifikaatio ISO-

standardoinnin jälkeen avoimesti saatavilla. Suosionsa, standardoinnin ja avoimen spesifikaation vuoksi PDF onkin luonnollinen valinta raportointityökalun ulostuloformaatille ja siksi myös kaikki kuusi tarkastelevaa raportointityökalua tarjoavat raportteja PDF-muodossa.

Raporttidokumenteille selkeästi suosittuja alustoja ovat erityisesti yritysmaailmassa suositun Microsoft Officen tuottamat tiedostomuodot, erityisesti taulukkolaskentaohjelma Excelin *.xlsx* ja *.xls* ja tekstinkäsittelyohjelma Wordin tuottamat *.docx* ja *.doc* tiedostot. Microsoft Officen tiedostomuotojen laaja tuki raportointityökalujen saralle ei ole varsinaisesti yllättävää ottaen huomioon, että tilastopalvelu *Statistan* mukaan Microsoftilla on noin 45 prosentin markkinaosuus toimisto-ohjelmistojen alalla Googlella sen ollessa noin 50. [33] On kuitenkin otettava huomioon, että myös Google tarjoaa omissa toimisto-ohjelmistoissaan mahdollisuuden avata ja muokata myös Microsoft Officen tuottamissa tiedostomuodoissa olevia tiedostoja. Sekä Excelin että Wordin tuottamat tiedostomuodot *.xlsx* ja *.docx* ovat suljettuja tiedostomuotoja, mutta ne ovat kuitenkin Googlen tarjoamien työkalujen lisäksi avattavissa esimerkiksi avoimen lähdekoodin ohjelmistolla LibreOfficella [34]. Täten vaikka Word ja Excel-tiedostot ovat suljetussa tiedostomuodossa, on ne mahdollista avata ilman kaupallisia ohjelmistoja. Microsoft Officen tiedostomuodoista erityisesti taulukkomuotoisten raporttien tapauksessa Excelin *.xlsx*-tiedostot ovat hyödyllisiä, sillä ne tarjoavat raportointityökalun käyttäjälle mahdollisuuden tutkia raportin sisältöä vielä pintaa syvemmältä suorittamalla itse laskentaa ja kokoamalla kuvaajia ohjelman avulla. Raportteja Excel- ja Word-tiedostoformaateissa tarjosivat kaikki tarkasteltavat raportointityökalut paitsi ActiveReportsJS, eli nämä formaatit ovat PDF-raporttien ohella erityisen suosittuja.

Kolmas suosittu tiedostomuoto on HTML (engl. *Hypertext Markup Language*, suom. *hypertekstin merkintäkieli*) , joka on erityisesti verkkosivujen kirjoittamisessa käytetty merkintäkieli. HTML ei ole ohjelmointikieli vaan merkintäkieli, joka määrit-

telee miten sisältö tulisi muotoilla. Myös HTML on avoin ja ISO-standardoitu tiedostomuoto.[35] HTML:ää käytetään usein yhteydessä CSS:n kanssa (engl. *Cascacing Style Sheets*, suom. *porrastetut tyyliarkit*) . CSS on eräänlainen tyylisivu, jolla voidaan määritellä tyyliohjeita siitä, miten esimerkiksi HTML dokumentti voidaan esittää. Kuten HTML myöskään CSS ei ole ohjelmointikieli, mutta se tarjoaa valtavasti mahdollisuuksia muotoilla HTML-dokumentteja. HTML:n ja CSS:n ohella verkkosivut hyödyntävät tyypillisesti myös JavaScriptia. HTML-raportit tarkoitettu esitettäväksi verkkoselaimessa, mikä tekee niistä hyvin alustariippumattomia. Mikäli HTML-raportit sisältävät CSS-tyyliarkin ja JavaScriptia, näitä hyödyntämällä HTML raportteihin voidaan lisätä animaatioita ja muita toiminnallisuuksia, ne voivat olla huomattavasti interaktiivisempia käyttäjälle verrattuna staattisempiin PDF-, Excel ja Word-raporttidokumentteihin. HTML-pohjaisia raportteja tarjosivat poikkeuksetta kaikki tarkasteltavat kuusi raportointityökalua.

Viimeisenä ulostuloformaattina vain *SAP BusinessObjects* ja *SAP Crystal Reports* mahdollistavat raporttien tuottamisen näiden raportointityökalujen omassa suljetussa tiedostomuodossa nimeltä *RPT*, joka lyhennelmä englannin kielen sanasta *report*. Koska *RPT* tiedostoformaatti on suljettu ja se ei ole yleisesti laajalti käytössä oleva tiedostoformaatti, *SAP BusinessObjectsin* ja *SAP Crystal Reportsin* ulkopuolella, sen avaaminen on käytännössä mahdollista vain tätä varta vasten kehitetyllä ohjelmalla toisin kuin esimerkiksi Word- ja Excel-tiedostojen tapauksessa.

3.3 Uuden raportointityökalun kehitys

Ensimmäinen haaste raportointityökalun kehittämiselle on käytettävien tekniikoiden ja teknologioiden valinta. Koska tarkoituksena on kehittää täysin uusi toiminnallisuus alusta alkaen, on tärkeää ymmärtää nykyiset Sovelia PLM:ssä käytössä olevat komponentit palvelinarkkitehtuurien näkökulmasta. Itse raportointimoottorin täytyy toimia jollakin palvelimella ja koostaa valmiit raportit palvelinpäässä ennen nii-

den lataamista käyttäjien saataville. Tällä menettelyllä voidaan vähentää asiakaspään kuormaa siirtämällä raskaampi laskenta palvelimen vastuulle. Palvelinpäässä kuormaa voidaan tasata asettamalla tehtävät jonoon odottamaan suoritusvuoroaan ja toteuttamalla raporttien suorittaminen rinnakkaisesti *monisäikeisyyttä* hyödyntämällä.

Uuden raportointimoottorin palvelinympäristö

Tarkastelemalla olemassa olevia raportointityökaluja, Java osoittautui suosituimmaksi ohjelmointikieleksi raportointityökalun ohjelmointirajapinnalle. Luvussa 3.2 kuitenkin todettiin, että Java on suosioltaan laskussa *TIOBE Indexin* mukaan vuonna 2023. Sovelia PLM hyödyntää ohjelmistopinossaan (engl. *software stack*) Java-palvelinta, mutta se on elinkaareltaan enemmänkin ylläpitovaiheessa, eikä sen alaisuuteen ole suunnitelmissa kehittää kokonaan uusia toiminnallisuuksia. Uuden raportointityökalun tarkoituksena on olla mahdollisimman tulevaisuudenkestävä (engl. *future-proof*), joten on olennaista tarkastella myös vaihtoehtoisia palvelinympäristöjä raportointityökalulle.

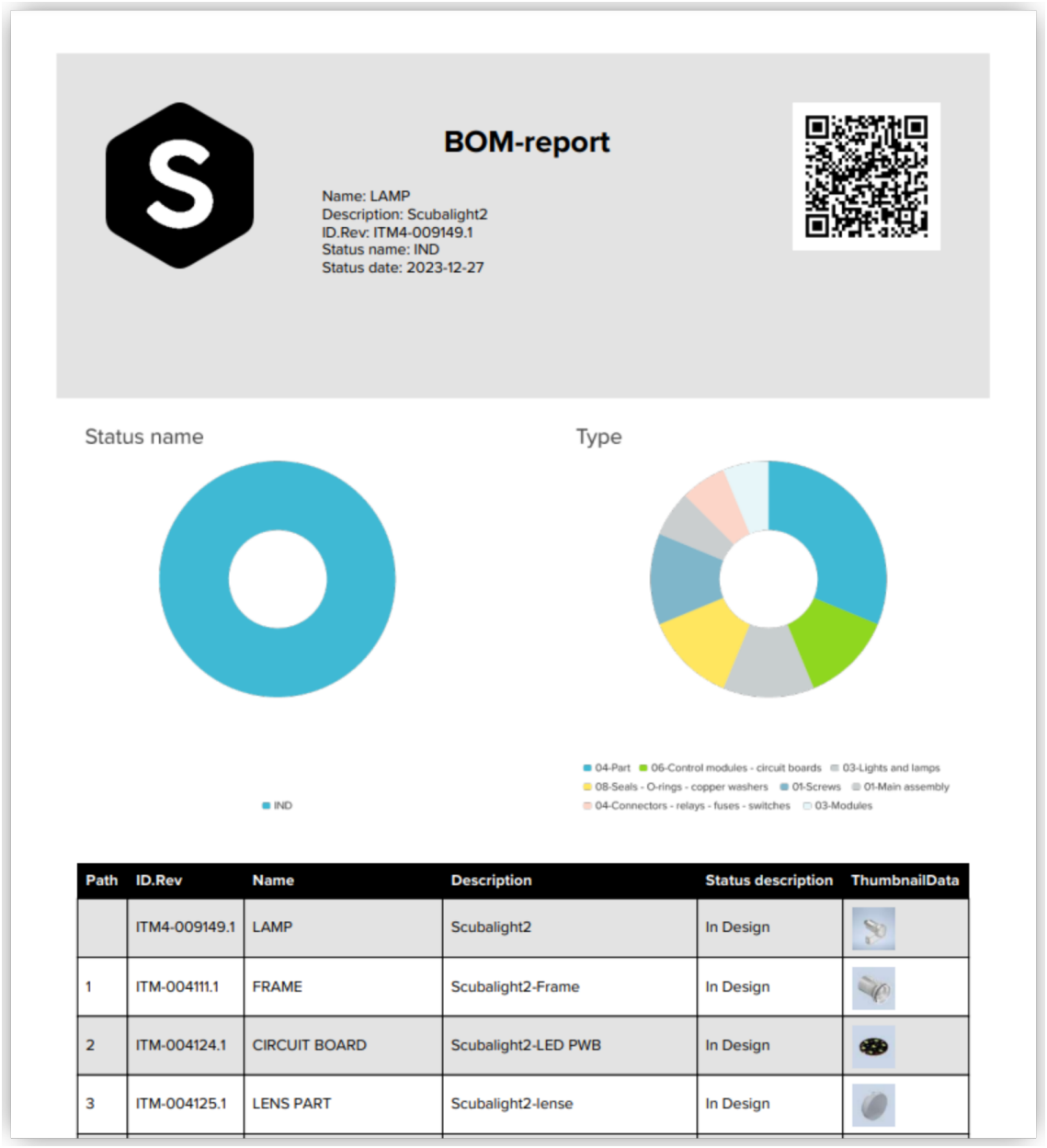
Raportointimoottorin palvelinympäristöksi valiutui lopulta Sovelia PLM:ssä jo käytössä oleva Node.js, joka on avoimen lähdekoodin alustariippumaton ajoympäristö JavaScript-koodin suorittamiseen erityisesti palvelinympäristössä. Node.js oli vuoden 2023 Stack Overflown teettämän kyselyn mukaan yleisin käytetty Web-teknologia ammattikehittäjien keskuudessa React.js:n ohella noin 42 prosentin käyttöasteella. [36] Kyselyn mukaan Node.js on kuitenkin vuoden 2023 johtava teknologia palvelinpuolella, sillä React.js on tarkoitettu enemmänkin käyttöliittymien kehittämiseen asiakaspäässä. Node.js tarjoaa ohjelmistokehittäjälle oletuksena paketinhallintajärjestelmän nimeltään NPM (engl. *Node Package Manager, NPM*, tyyliteltynä npm), joka koostuu komentorivityökalusta ja JavaScript -ohjelmapaketteja ylläpitävästä tietokannasta. NPM:n avulla erillisten ohjelmakirjastojen sisällyttäminen ja

käyttöönotto on yksinkertaista ja nopeaa.[37] Node.js:n suosion vuoksi käytettävissä on laaja kattaus erilaisia hyödyllisiä ohjelmakirjastoja. NPM helpottaa erityisesti tarvittavien ohjelmakirjastojen käyttöönottoa ja asennusta, joita raportointimoottori tarvitsee erityisesti raporttidokumenttien kokoamiseen ohjelmallisesti. Sovelia PLM:n Node.js-palvelimen tarkoituksen oli erilaisia palveluita ja rajapintoja Sovelia PLM:n Web-käyttöliittymälle. Näihin palveluihin kuuluu esimerkiksi 3D-näkymien ja -tiedostojen tarjoaminen Web-käyttöliittymälle, joten myös raportointimoottorin sisällyttäminen samalle palvelimelle oli luonteva ratkaisu, sillä myös raporttien konfigurointityökalu sekä toiminnot raporttien suorittamiseen tullaan kehittämään Sovelia PLM:n web-käyttöliittymään.

Uuden raportointityökalun ominaisuuksia

Datalähteenä Sovelia PLM:n uusi raportointimoottori käyttää Node.js pohjaisena JSONia, joka oli käytössä myös muissa tarkastelluissa raportointityökaluissa. Luvussa 3.2 todettiin JSONin olevan hyvä valinta hierarkkisen PLM-datan tallentamiseen. Luvun 2.3.1 alaluvussa koskien logiikkatasoa huomattiin JSONin olevan nopeampi verrattuna vanhempaan XML-standardiin [13]. Uusi raportointityökalu hyödyntää lähdedatan hankkimisessa Sovelia PLM-järjestelmän olemassa olevaa hakumoottoria, joka palauttaa itsessään JSON-muotoista dataa, joten muuntamisvaihetta erilliseen dataformaattiin ei tarvita.

Sovelia PLM:n uusi raportointityökalu tuottaa raportteja PDF-, Excel ja ZIP-muodoissa eli se tukee tarkasteltujen raportointityökalujen suosituimpia formaatteja. PDF-raportit tarjoavat taulukoidun tuotedatan lisäksi myös piirakkakuvaajia tuoterakenteen eri tietokenttien arvojen jakaumasta. Kuvassa 3.1 on esimerkki uuden raportointimoottorin tuottamasta raportista PDF-muodossa. Excel-raportit esittävät PDF-raporttien lailla taulukoitua tuotedataa, mutta ne mahdollistavat myös tuoterakenteiden esittämisen tasoittain ja tarjoavat erilaisia laskentamahdollisuuksia.

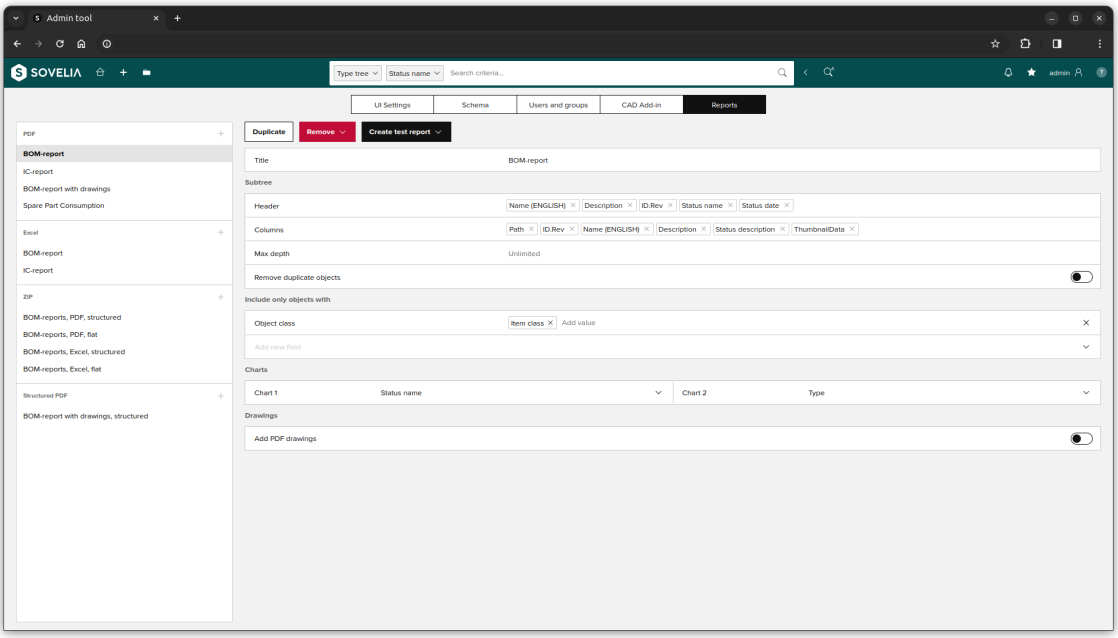


Path	ID.Rev	Name	Description	Status description	ThumbnailData
	ITM4-009149.1	LAMP	Scubalight2	In Design	
1	ITM-004111.1	FRAME	Scubalight2-Frame	In Design	
2	ITM-004124.1	CIRCUIT BOARD	Scubalight2-LED PWB	In Design	
3	ITM-004125.1	LENS PART	Scubalight2-lense	In Design	

Kuva 3.1: Kuvakaappaus Sovelia PLM:n raportointimoottorin tuottamasta BOM-raportista PDF-muodossa

ZIP-raportit kokoavat tuoterakenteesta useita PDF- tai Excel-raportteja ja lisäävät arkistoon myös tuoteobjektien liitetiedostoja, kuten piirustuksia. Uusi raportointimoottori tukee myös HTML-raportteja, mutta ne eivät kuulu ensimmäisen version vaatimusmäärittelyyn, eivätkä siten ole osa ensimmäistä julkaistavaa versiota.

Raporttipohjien konfigurointi tapahtuu uudella raportointityökalulla tapahtuu



Kuva 3.2: Kuvakaappaus Sovelia PLM:n raportointityökalun Web-käyttöliittymän kehitysversiosta

Sovelia PLM:n web-käyttöliittymässä pääkäyttäjän toimesta. Kuvassa 3.2 olevassa kuvakaappauksessa on raportointityökalun käyttöliittymä, jonka avulla voidaan luoda, muokata ja poistaa raportteja. Työkalun interaktiivisten valikoiden avulla voidaan konfiguroida myös esimerkiksi halutut tietokentät, joista muodostetaan raportin kuvaajat ja taulukot. Raportointimoottori osaa myös laskea lisäkenttiä, kuten esimerkiksi osien kokonaissummia tuoterakenteessa. Tuoterakenteista voidaan myös suodattaa pois määritellyn ominaisuuden mukaisesti tuoteobjekteja, tai raportissa esitettävä tuoterakenne voidaan asettaa koostumaan vain tietyn ominaisuuden omaavista tuoteobjekteista. ZIP-raporttipaketteihin voidaan valita halutut tuoteobjektien liitetiedostojen tiedostomuodot. Myös PDF-raportteihin on mahdollista liittää tuotteiden ja osien piirustuksia PDF-muodossa. ZIP-paketin "aliraportiksi"voidaan valita mikä tahansa järjestelmään konfiguroitu Excel- tai PDF-raportti. Aliraportilla tarkoitetaan tässä raporttia, joka ajetaan käyttöliittymässä annettujen kriteerien mukaisesti tuoterakenteen eri alirakenteille tai osille.

3.4 Reflektiota kehitysprosessista

Aiemman kirjallisuuden tarkastelu avasi monia näkökulmia PLM-järjestelmien ja raportoinnin yhteistoimintaan, joita päästiin soveltamaan käytännössä raportointityökalun kehitysprosessissa. Käymällä läpi alan tutkimuksia ja asiantuntija-artikkeleita, oli mahdollista hahmottaa selkeämpiä yhteyksiä siitä, miten tuotteen elinkaaren hallintaohjelmistot ja raportointityökalut liittyvät toisiinsa. Uuden raportointityökalun kehittämisessä korostui erityisesti tarve hierarkkisen PLM-datan perusteelliselle ymmärrykselle ja sen vaikutuksiin raporttien luomisessa. Hierarkkisen rakenteen ymmärtäminen oli tärkeää, kun pyritään vastaamaan kysymyksiin siitä, miten eri tuotteen tasojen tiedot liittyvät toisiinsa ja miten raportit voivat tarjota kokonaisvaltaisen näkymän tuotteesta. Hierakkinen data korostui alati kirjallisuutta tutkiessa ja sen merkitys myös Sovelia PLM:n tapauksessa oli ilmeinen. Tämän perusteella on suositeltavaa harkita räätälöityjä ratkaisuja PLM-järjestelmän raportointityökalun hierarkkisen datan käsittelyyn, esimerkiksi ohjelmiston sisäisenä tietomallina, jotta erityislaatuinen rakenteellinen data voidaan käsitellä skaalautuvasti tuoterakenteiden kasvaessa.

Tutkielman tulokset ovat linjassa aiemman kirjallisuuden kanssa, mutta vertailu olemassa oleviin raportointimoottoreihin ei ollut yhtä suoraviivaista. Kuten luvussa 3.2 kerrottiin, tarkasteltujen raportointityökalujen kehittäminen oli aloitettu lähes kaikkien kohdalla 2000-luvun alkupuolella, joten näiden työkalujen teknologiaratkaisut olivat ajan mukaisia. Kokonaan uutta työkalua kehittäessä on kuitenkin huomioitava nykyaikaiset trendit ja saatavilla olevat teknologiat, joten on tärkeää, että raportointityökalun kehittäjä ottaa huomioon myös nykyaikaiset teknologiavaihtoehdot. Vertailemalla raportointityökaluja saatiin kuitenkin selkeä yleiskuva niiden ominaisuuksista, joka ehdottomasti suoraviivaisti Sovelia PLM:n raportointityökalun kehitysprosessia.

Palvelinympäristö asettaa kehyksen sille, mitä työkaluja voidaan hyödyntää ke-

hittäessä raportointityökalua. Täten palvelinympäristöä voidaan pitää teknologiavalinnoista merkityksellisimpänä. Kehitysaikana huomattiin, että Node.js osoittautui hyväksi valinnaksi erityisesti sen helppokäyttöisyyden, laajan dokumentaation ja yhteisön sekä saatavilla olevien ohjelmistokirjastojen vuoksi. Lisäksi Node.js:n vakiintunut status osana Sovelia PLM-järjestelmää helpotti ominaisuuden integraatiota huomattavasti.

Uuden raportointityökalun kehityksen haasteita

Uuden raportointityökalun kehittäminen ei sujunut kuitenkaan ilman haasteita. Luvussa 3.1 mainittu rakenteellisen massadatan merkitys on olennainen osa PLM-järjestelmää ja siten myös Sovelia PLM:ää, mikä aiheutti haasteita erityisesti suorituskyvyn näkökulmasta. Laajojen datamäärien kirjoittaminen siististi formatoituun PDF-raporttiin ei ole yksiselitteistä, mikä johti raportointipohjien suunnittelun haasteellisuuteen. Tietokenttien kokoa on hankala arvioida etukäteen, joten raportointimoottorin tulee osata mukautua vaihtelevaan dataan huolehtimalla, että lopulliset raporttidokumentit ovat mahdollisimman helppolukuisia lähdedatan monimutkaisuudesta huolimatta.

Skaalautuvuus asetti haasteita erityisesti uuden raportointityökalun palvelinkomponentille. Koska raporttien saama lähtödata voi olla kooltaan tuhansia rivejä sekä Sovelia PLM:n tuottamat ZIP-raporttipaketit voivat koostua useista yksittäisistä raporteista, on ymmärrettävää, että raporttien generoiminen voi viedä useita sekunteja. Kuorman tasaamiseksi päädyttiin hyödyntämään suoritusjonoa ja rinnakkaista suorittamista palvelinpäässä. Tehokkaan ja virheenkestävän suoritusjonon kehittäminen vaati huolellista suunnittelua ja harkintaa. Lisäksi liian suuret raportit eivät saa aiheuttaa palvelimen jumiutumista tai kaatumista, joten raportin koostaminen tulee olla keskeytettävissä, jotta palvelimen resurssit voivat vapautua taas seuraavien raporttien koostamiseen. Palvelinkomponentin tulee kestää Sove-

lia PLM:n tapauksessa myös useita uudelleenkäynnistymisiä menettämättä mitään oleellista tietoa.

Hyvän käyttökokemuksen kannalta ilmoitukset käyttäjille raportin koostamisen vaiheista koettiin tärkeiksi. Tämän toteuttaminen osoittautui haasteelliseksi, sillä oli tarpeen luoda selkeä ja informatiivinen käyttöliittymä ilmoitusten näyttämiseksi reaaliaikaisesti käyttäjille. Reaaliaikaisesti esitettävä prosessin eteneminen on tärkeää erityisesti suuria raportteja generoitaessa, jotta käyttäjä voi seurata aktiivisesti prosessien etenemistä. Haasteena oli pitää käyttäjät ajan tasalla ilman, että se vaikuttaisi negatiivisesti järjestelmän suorituskykyyn ja käytettävyyteen.

Uuden raportointityökalun tulevaisuus

Uusi raportointimoottori Node.js -palvelimella on osoittautunut testausvaiheessa tehokkaaksi ja nopeaksi keinoksi muodostaa raportteja. Suoritusajat ovat huomattavasti vanhaa raportointityökalua nopeampia ja muistinkäyttö palvelimella on vähäisempää. Konfigurointia on pyritty tekemään loppukäyttäjille helpommaksi tarjoamalla graafinen Web-käyttöliittymä raporttien konfigurointiin ja raportteja voidaan luoda suoraan Sovelia PLM:n valikoista. Tulevaisuuden ominaisuuksina on tarkoitus kehittää graafinen suunnittelutyökalu PDF-raporteille, lähettää monitorointidataa raportointimoottorin suorituskyvystä sekä tarjota loppukäyttäjille enemmän mahdollisuuksia vaikuttaa raporttien ulkoasuun.

4 Yhteenveto

Tämän tutkielman ensimmäisenä tutkimuskysymyksenä oli selvittää, miten PLM-järjestelmät ja raportointityökalut toimivat yhdessä. Kysymystä taustoitettiin luvussa 2 tarkastelemalla PLM-järjestelmiä ja raportointia erikseen sekä niiden yhteyttä. Tutkielman motiiveja taustoitettiin perehtymällä PLM-strategian hyötyihin ja PLM:n yhteyteen liiketoimintatiedon hyödyntämisessä. Aiempaan kirjallisuuteen vedoten todettiin, että PLM-järjestelmä olennainen osa PLM-strategian hyödyntämistä käytännössä.[5] Osaluettelon (engl. *Bill of Materials, BOM*) merkitystä korostettiin yhtenä PLM-järjestelmän tärkeimmistä toiminnallisuuksista[8] luvussa 2.2.1. Samassa luvussa perusteltiin, että osaluettelot ovat tärkeä osa PLM-järjestelmän raportointia [6]. Aiemman tutkimuksen perusteella voidaan myös todeta, että PLM-strategiassa liiketoimintatiedon hyödyntäminen korostuu etenkin osaluetteloista koostuvan tuotetiedon hyödyntämisenä. [11] PLM-strategian ja -järjestelmän taustoittamisen jälkeen perehdyttiin raportointimoottoreihin ja työkaluihin luvussa 2.3. Aluksi määriteltiin raportointityökalun ja -moottorin käsitteet, jonka jälkeen tarkasteltiin raportointimoottorin rakennetta ja prosessia. Luvussa 2.3.1 raportointimoottori jaettiin aiemman tutkimuksen tavan mukaisesti kolmelle tasolle: data, -logiikka ja esitystasolle. [12].

Raportointimoottorin tasojen tarkastelun jälkeen pyrittiin vastaamaan ensimmäiseen tutkimuskysymykseen PLM-järjestelmien ja raportointityökalujen yhteydestä. Tarkastelemalla ensin raportointia PLM-järjestelmässä todettiin, että tässä

tutkielmassa keskitytään erityisesti digitaalisten raporttidokumenttien tuottamiseen ohjelmallisesti. Nämä raporttidokumentit ovat esimerkiksi PDF- tai Excel-muotoisia tiedostoja. PLM-järjestelmän todettiin olevan haasteellinen ja yksilöllinen toimintaympäristö aiemman kirjallisuuden perusteella, sillä valmiit raportointimoottorit eivät välttämättä sovi sellaisenaan käytettäväksi usein massadatan kaltaisen PLM-datan kanssa. [7] Lisäksi huomattiin, että PLM-järjestelmän raportointimoottorit ovat erikoistuneita toimimaan hierarkkisen datan parissa. [7] Kehittääkseen raportointityökalua PLM-järjestelmään tulee ymmärtää PLM-järjestelmä toimintaympäristönä, johon liittyy erityisesti hierarkkinen massadata, raporttidokumenttien tiedostomuodot ja muiden PLM-järjestelmän ominaisuuksien hyödyntäminen. Muiden ominaisuuksien hyödyntämisellä tarkoitetaan erityisesti hakurajapinnan käyttöä raportointimoottorin datalähteenä, sillä hakutoiminto on yksi PLM-järjestelmän keskeisimmistä toiminnallisuuksista. [16] Merkittävänä haasteena raporttien tuottamiselle PLM-järjestelmässä pidettiin raporttien tuottaman tiedon ajankohtaisuutta ja luotettavuutta alati muuttuvassa sekä laajassa PLM-datavirrassa. [6]

Luvussa 3 syvennyttiin valmiiden raportointityökaluohjelmistojen ominaisuuksien tarkastelemiseen. Tarkastelemalla nykyisiä raportointityökaluja pyrittiin vastaamaan siihen, millaisia nykyiset raportointityökalut ovat ja millaisia ominaisuuksia ne tarjoavat. Sovelia PLM esiteltiin toimintaympäristönä ja sen erityispiirteitä sekä käsitteitä avattiin, jotta kehitettävän raportointityökalun toimintaympäristöä voitaisiin ymmärtää paremmin. Tutustuimme Sovelia PLM:n vanhaan raportointimoottoriin esittelemällä sen käyttämiä teknologioita ja niihin liittyviä haasteita ja ongelmia.

Vanhan Sovelia PLM:n raportointityökalun esittelyn jälkeen tarkasteltiin luvussa 3.2 nykyisten olemassa olevien raportointityökalujen ominaisuuksia erottelemalla niiden käyttämiä lisenssejä, datalähteitä ja ohjelmointikieliä. Tutustuimme lyhyesti myös näiden raportointityökalujen konfigurointimahdollisuuksiin sekä asennusta-

paan. Jotta ymmärtäisimme näiden työkalujen tuottamia raportteja, analysoitiin myös raporttien tiedostomuotoja ja niiden valintojen motiiveja. Nykyisten raportointityökalujen tarkastelun jälkeen vertasimme näistä saatuja havaintoja uuden raportointityökalun kehitysprosessiin ja teknologiavalintoihin. Huomasimme, että osa nykyisten raportointityökalujen lähestymistavoista olivat hyödyllisiä myös Sovelia PLM:n uuden raportointityökalun tapauksessa, mutta päädyimme osittain nykyaikaisempiin ratkaisuihin. Esimerkiksi tarkasteltujen raportointityökalujen keskuudessa Java oli ylivoimaisesti suosituin ohjelmointikieli, mutta päädyimme modernimman Node.js palvelinympäristön valintaan. Ohjelmistokehittäjälle Node.js on mielestäni käyttäjäystävällinen ja erittäin runsaasti dokumentoitu alusta palvelinympäristölle ja skaalautuvuutensa sekä runsaasti saatavilla olevien hyödyllisten ohjelmakirjastojen vuoksi se hyvä valinta myös raportointimoottorille. Nykyisten raportointityökalujen suosimaa JSONia käytettiin taas myös uuden raportointityökalun lähtödatana, sekä uuden raportointityökalun tarjoamat ulostuloformaatit vastasivat läheisesti nykyisten tarkasteltujen raportointityökalujen suosimia tiedostomuotoja.

Tässä tutkielmassa ei otettu kuitenkaan huomioon sitä, että markkinoilla on useita raportointityökaluja, joita tässä ei tarkasteltu. Tarkasteltavat raportointityökalut valittiin niiden yleisen suosion sekä vertailukelpoisuuden perustella kehitettävään raportointimoottorin verrattuna, joten tutkielman tuloksia ei voida käyttää yleiskuvana kaikista saatavilla olevista raportointityökaluista. Lisäksi tarkastelimme valituista raportointityökaluista vain muutamaa ominaisuutta, emmekä syventyneet näiden työkalujen toteutusten teknisiin seikkoihin. Kehitettävä raportointityökalu on myös erillinen yksittäistapaus, sillä se on kehitetty osaksi yhtä kaupallista PLM-järjestelmää. Täten muut PLM-järjestelmät voivat poiketa siitä suuresti eikä tässä tutkielmassa kuvattua implementaatiota voida välttämättä käyttää sellaisenaan.

Jatkotutkimuksena aiheesta voitaisiin perehtyä PLM-järjestelmän ja siten raportoinnin sisäisiin tietomalleihin, tarkastella vaihtoehtoja raporttien sisällön ja ul-

koasun muokkaamiselle sekä syventyä raportointityökalun elinkaareen. Käytännössä tutkielman tuloksia voidaan hyödyntää dokumentoituna esimerkkinä raportointityökalun kehittämisestä PLM-järjestelmään, jolloin tämän tutkielman havaintoja voidaan hyödyntää vastaavanlaisia järjestelmiä kehittäessä esimerkiksi teknologiavalintojen ja haasteiden ymmärtämiseksi. PLM-järjestelmien kehittäjät ja niiden käyttäjät voivat tutkielman pohjalta parantaa ymmärrystään raportoinnin ja PLM-strategian yhteydestä, sillä aiempaa akateemista tutkimusta juurikin PLM-järjestelmien raportointityökaluista ei ole avoimesti saatavilla. Tutkielman perustella voidaankin pohjimmiltaan todeta, että raportointityökalun kehittäminen PLM-järjestelmään vaatii hierarkkisen massadatan kaltaisen PLM-datan tuntemista sekä ymmärrystä raportointityökalujen ominaisuuksien heikkouksista ja vahvuuksista.

Lähdeluettelo

[1] L. Raković, M. Sakal ja P. Matković, ”Digital workplace: Advantages and challenges”, en, *Anali Ekonomskog fakulteta u Subotici*, nro 47, s. 65–78, 2022, ISSN: 0350-2120, 2683-4162. DOI: 10.5937/AnEkSub2247065R. url: https://scindeks.ceon.rs/Article.aspx?artid=0350-21202247065R (viitattu 22. 10. 2023).

[2] S. Glöckner, *Reports & Reporting | Portal Systems Wiki*, en-US, helmikuu 2022. url: https://www.portalsystems.de/en/wiki/reporting/ (viitattu 13. 12. 2023).

[3] B. P. Adhi, D. N. Prasetya ja Widodo, ”Performance comparison of reporting engine birt, jasper report, and crystal report on the process business intelligence”, English, *IOP Conference Series. Materials Science and Engineering*, vol. 508, nro 1, huhtikuu 2019, Place: Bristol, United Kingdom Publisher: IOP Publishing, ISSN: 17578981. DOI: 10.1088/1757-899X/508/1/012129. url: https://www.proquest.com/docview/2560973452/abstract/E835BABAB8FE44D6PQ/1 (viitattu 09. 10. 2023).

[4] *About Sovelia and one digital platform | sovelia.com — sovelia.com*, https://sovelia.com/about-sovelia. (viitattu 17. 11. 2023).

[5] M. Alemanni, G. Alessia, S. Tornincasa ja E. Vezzetti, ”Key performance indicators for PLM benefits evaluation: The Alcatel Alenia Space case study”, *Computers in Industry*, vol. 59, nro 8, s. 833–841, lokakuu 2008, ISSN:

0166-3615. DOI: 10.1016/j.compind.2008.06.003. url: https://www.sciencedirect.com/science/article/pii/S0166361508000663 (viitattu 17. 10. 2023).

[6] K. German, *The Challenge of Getting High Quality Reports out of PLM - Teamcenter*, en-US, Section: News, tammikuu 2016. url: https://blogs.sw.siemens.com/teamcenter/the-challenge-of-getting-high-quality-reports-out-of-plm/ (viitattu 17. 10. 2023).

[7] C. Rohleder, J. Lin, I. Kusumah ja G. Özkan, "Requirements Engineering in Business Analytics for Innovation and Product Lifecycle Management", en, teoksessa *Advances in Conceptual Modeling*, J. Parsons ja D. Chiu, toim., sarja Lecture Notes in Computer Science, Cham: Springer International Publishing, 2014, s. 51–58, ISBN: 978-3-319-14139-8. DOI: 10.1007/978-3-319-14139-8_7.

[8] M. David ja F. Rowe, "What does PLMS (product lifecycle management systems) manage: Data or documents? Complementarity and contingency for SMEs", *Computers in Industry*, vol. 75, s. 140–150, tammikuu 2016, ISSN: 0166-3615. DOI: 10.1016/j.compind.2015.05.005. url: https://www.sciencedirect.com/science/article/pii/S0166361515300051 (viitattu 17. 09. 2023).

[9] I. Bouhaddou, A. Benabdelhafid, L. Ouzizi ja Y. Benghabrit, "PLM (Product Lifecycle Management) Model for Supply Chain Optimization", en, teoksessa *Product Lifecycle Management. Towards Knowledge-Rich Enterprises*, L. Rivest, A. Bouras ja B. Louhichi, toim., sarja IFIP Advances in Information and Communication Technology, Berlin, Heidelberg: Springer, 2012, s. 134–146, ISBN: 978-3-642-35758-9. DOI: 10.1007/978-3-642-35758-9_12.

[10] R. Jones ja L. Tate, *Visualizing Comparisons of Bills of Materials*, arXiv:2309.11620 [cs], syyskuu 2023. DOI: 10.48550/arXiv.2309.11620. url: http://arxiv.org/abs/2309.11620 (viitattu 17. 10. 2023).

[11] M. Bosch-Mauchand, M. Bricogne, B. Eynard ja J.-P. Gitto, "Preliminary Requirements and Architecture Definition for Integration of PLM and Business Intelligence Systems", en, teoksessa *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, E. Bayro-Corrochano ja E. Hancock, toim., vol. 8827, Series Title: Lecture Notes in Computer Science, Cham: Springer International Publishing, 2014, s. 265–272, ISBN: 978-3-319-12567-1 978-3-319-12568-8. DOI: 10.1007/978-3-662-44739-0_33. url: http://link.springer.com/10.1007/978-3-662-44739-0_33 (viitattu 22. 10. 2023).

[12] Y. He ja F. Gong, "Design and Implementation of the Large Enterprise Reporting Engine", teoksessa *2010 International Conference on Web Information Systems and Mining*, vol. 2, lokakuu 2010, s. 235–238. DOI: 10.1109/WISM.2010.96. url: https://ieeexplore.ieee.org/document/5662268 (viitattu 09. 10. 2023).

[13] N. Nurseitov, M. Paulson, R. Reynolds ja C. Izurieta, "Comparison of JSON and XML Data Interchange Formats: A Case Study", en,

[14] Y. Niu, L. Ying, J. Yang, M. Bao ja C. B. Sivaparthipan, "Organizational business intelligence and decision making using big data analytics", *Information Processing & Management*, vol. 58, nro 6, s. 102 725, marraskuu 2021, ISSN: 0306-4573. DOI: 10.1016/j.ipm.2021.102725. url: https://www.sciencedirect.com/science/article/pii/S0306457321002090 (viitattu 22. 10. 2023).

[15] M.-F. Sriti ja P. Boutinaud, ”PLMXQuery: Towards a Standard PLM Querying Approach”, eng, teoksessa *IFIP Advances in Information and Communication Technology*, vol. AICT-388, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, s. 379–388, ISBN: 9783642357572.

[16] J. G. Enríquez, J. M. Sánchez-Begines, F. J. Domínguez-Mayo, J. A. García-García ja M. J. Escalona, ”An approach to characterize and evaluate the quality of Product Lifecycle Management Software Systems”, *Computer Standards & Interfaces*, vol. 61, s. 77–88, tammikuu 2019, ISSN: 0920-5489. DOI: 10.1016/j.csi.2018.05.003. url: https://www.sciencedirect.com/science/article/pii/S0920548917303239 (viitattu 17. 11. 2023).

[17] *Sovelia PLM getting started - How to use Sovelia PLM — help.sovelia.com*, https://help.sovelia.com/docs/getting-started. (viitattu 17. 11. 2023).

[18] *User management — help.sovelia.com*, https://help.sovelia.com/docs/user-management. (viitattu 17. 11. 2023).

[19] *Apache(tm) FOP - a print formatter driven by XSL formatting objects (XSL-FO) and an output independent formatter.* en. url: https://xmlgraphics.apache.org/fop/ (viitattu 19. 11. 2023).

[20] *BI Report Management System & Reporting Tools | Bold Reports*, en-US, syyskuu 2022. url: https://www.boldreports.com (viitattu 03. 12. 2023).

[21] *SAP North America at a Glance: The World's Largest Provider of Enterprise Application Software*, English. url: https://www.sap.com/documents/2014/12/e4198023-0a7c-0010-82c7-eda71af511fa.html (viitattu 28. 11. 2023).

[22] ". M. TUCKER" " HANK, *The Global 2000 2023*, en. url: https://www.forbes.com/lists/global2000/ (viitattu 28. 11. 2023).

[23] *JasperReports Library Datasheet*, en. url: https://www.jaspersoft.com/resources/whitepaper/jasperreports-library-datasheet (viitattu 28. 11. 2023).

[24] *Reporting and embedded business intelligence software*, en. url: https://www.jaspersoft.com/ (viitattu 03. 12. 2023).

[25] *BIRT - Business Intelligence Reporting Tool — eclipse-birt.github.io*, https://eclipse-birt.github.io/birt-website/. (viitattu 23. 11. 2023).

[26] M. Miner, *Under the Hood: Excel Files as Zip Folders - Miner Curiosity*, English, Blog, heinäkuu 2022. url: https://minerupset.com/2022/Excel-Files-As-Zip-Folders/ (viitattu 06. 12. 2023).

[27] *TIOBE Index for November 2023*, en-US. url: https://www.tiobe.com/tiobe-index/ (viitattu 06. 12. 2023).

[28] *Origin date for JasperReports?*, en-US, syyskuu 2006. url: https://community.jaspersoft.com/forums/topic/13224-origin-date-for-jasperreports/ (viitattu 06. 12. 2023).

[29] *Eclipse Foundation and Actuate Announce Approval of Business Intelligence and Reporting Tools Project*, English, kesäkuu 2004. url: https://www.eclipse.org/org/press-release/oct62004birtpr.html (viitattu 06. 12. 2023).

[30] T. M. Mahmoud, *Creating universes with SAP BusinessObjects : create and maintain powerful SAP BusinessObjects Universes with the SAP Information Design Tool* (Professional expertise distilled), eng, 1st edition. Birmingham, [England: Packt Publishing, 2014, ISBN: 1-78217-091-X.

[31] *File Formats*. url: https://opendatahandbook.org/guide/en/appendices/file-formats/ (viitattu 10. 12. 2023).

[32] C. Naden, *The standard for PDF is revised*, en, tammikuu 2021. url: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/news/2021/01/Ref2608.html (viitattu 06. 12. 2023).

[33] *Office productivity software global market share 2022*, en. url: https://www.statista.com/statistics/983299/worldwide-market-share-of-office-productivity-software/ (viitattu 09. 12. 2023).

[34] *LibreOffice - Free Office Suite - Based on OpenOffice - Compatible with Microsoft.* url: https://www.libreoffice.org/ (viitattu 10. 12. 2023).

[35] *ISO/IEC 15445:2000*, en. url: https://www.iso.org/standard/27688.html (viitattu 10. 12. 2023).

[36] *Stack Overflow Developer Survey 2023 — survey.stackoverflow.co*, https://survey.stackoverflow.co/2023/. (viitattu 23. 11. 2023).

[37] *npm About — npmjs.com*, https://www.npmjs.com/about. (viitattu 23. 11. 2023).

[38] *ActiveReportsJS: Introduction — developer.mescius.com*, https://developer.mescius.com/activereportsjs/docs/. (viitattu 23. 11. 2023).

[39] "SAP BusinessObjects RESTful Web Service SDK User Guide for Web Intelligence and the BI Semantic Layer", (viitattu 23. 11. 2023).

[40] "SAP Crystal Reports RESTful Web Services Developer Guide", en, (viitattu 23. 11. 2023).

[41] P. Sharma ja K. McDermott, "Oracle Business Intelligence Publisher Overview & Best Practices", en, 2014.

Liite A Yleistä raportointityökaluista

Raportointi-työkalu	**Lisenssi**	**Datalähde**	**Ohjelmointi-kieli**	**Raporttipohjien konfigurointi**	**Asennus**
ActiveReportsJS [38]	kaupallinen	JSON	JavaScript	UI /JSON	paikallinen
SAP BusinessObjects [39]	kaupallinen	XML ja JSON	Java	UI	paikallinen tai pilvipalvelu
SAP Crystal Reports [40]	kaupallinen	tiedostot mm. XLSX, TXT, XML ja tietokannat (SQL)	Java	UI	paikallinen tai pilvipalvelu
Oracle BI Publisher [41]	kaupallinen	SQL, XML, XLSX ja Oracle BI:n omat formaatit	Java	UI, RTF ja XSL	paikallinen
JasperReports [23]	avoin lähdekoodi, mahdollisuus kaupalliseen lisenssiin	JavaBeans, Java object, XML ja mukautetut tietolähteet	Java	UI / XML tai .jasper	paikallinen
BIRT Project [25]	avoin lähdekoodi	XML, TXT ja tietokannat (SQL)	Java	UI / XML	paikallinen

Taulukko A.1: Yleistä raportointityökaluista.

Liite B Raportointityökalujen ulostuloformaatit

Raportointityökalu	**PDF (.pdf)**	**XLSX (.xlsx, .xls)**	**HTML (.html)**	**Word (.docx, .doc)**	**RPT (.rpt)**
ActiveReportsJS [38]	x		x		
SAP BusinessObjects [39]	x	x	x	x	x
SAP Crystal Reports [40]	x	x	x	x	x
Oracle BI Publisher [41]	x	x	x	x	
JasperReports [23]	x	x	x	x	
BIRT Project [25]	x	x	x	x	

Taulukko B.1: Raportointityökalujen ulostuloformaatit.