Opinnäytetyö (AMK)

Tietojenkäsittely

2020

Ville-Pekka Pietarinen

PROSESSIKAAVION ESITTÄMINEN WEB-KÄYTTÖLIITTYMÄSSÄ

– Case KyberVALIOT-yhteistyöhanke

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietojenkäsittely

2020 | 28 sivua, 12 liitesivua

Ville-Pekka Pietarinen

PROSESSIKAAVION ESITTÄMINEN WEB-KÄYTTÖLIITTYMÄSSÄ

– Case KyberVALIOT-yhteistyöhanke

Tutkimus toteutettiin toimeksiantona Turun ammattikorkeakoululle osana KyberVALIOT-yhteistyöhanketta. Työn tavoitteena oli tutkia Turun ammattikorkeakoulun älylaitteen testausprosessikaaviota ja toteuttaa kaavion sisältävä interaktiivinen web-käyttöliittymä. Teoreettisessa osassa tarkasteltiin prosessikaavion määritelmää ja prosessikuvaamisen tasoja, joilla hahmotetaan kuvauksien yksityiskohtaisuutta. Lisäksi selvitettiin sovellettavan kaavion tietorakennetta ja sen määritelmää niiltä osin kuin käytännön toteutuksessa nähtiin tarpeelliseksi.

Työ tehtiin tapaustutkimuksena, jossa käytettiin konstruktiivista tutkimusmenetelmää. Työn keskeistä tulosta edustaa web-käyttöliittymätoteutus, joka keskittyi kaavion visualisoinnin ympärille. Tunnistettua tietorakennetta soveltaen kaavio mallinnettiin uudelleen CSV-muotoon tietorakenteen helpomman hallittavuuden takia. Toteutuksen aikana vertailtiin valitun JavaScript-visualisointikirjaston asettelualgoritmeja ja niiden piirtämiä kaavioita, joiden selkeydessä havaittiin eroja. Kaavion selkeää visualisointia tukevaa algoritmia hyödyntäen verkkosivu ohjelmoitiin esittämään kaavio, ja luotiin visualisoinnin sisältävä React-komponentti, joka liitettiin osaksi web-käyttöliittymäkokonaisuutta.

Verkkosivu toteutettiin tyyliteltyjä React-komponentteja tarjoavalla Material-UI-kirjastolla, jota ohjelmoitiin JavaScript-kieltä laajentavalla TypeScriptillä. Käyttöliittymälle suoritettiin kolme testitapausta, joissa käyttöliittymä toimi määritetyissä tapauksissa odotetusti. Lopuksi tarkasteltiin tuloksien merkitystä ja pohdittiin jatkokehitysehdotuksia, jotka sisältävät muun muassa käyttöliittymässä valittujen prosessivaiheiden tallennuksen.

Työn tuloksista oli merkittävää hyötyä laitetestausautomaation suunnittelussa ja jatkokehityksessä.

ASIASANAT:

prosessi, kaavio, verkko, käyttöliittymä, visualisointi

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology

2020 | 28 pages, 12 pages in appendices

Ville-Pekka Pietarinen

PRESENTATION OF PROCESS DIAGRAM WITHIN WEB USER INTERFACE

- Case CyberELITE collaborative project

The thesis was commissioned by Turku University of Applied Sciences as part of the collaborative project CyberELITE. The goal of the thesis was to analyse a process diagram developed by Turku University of Applied Sciences and implement a web interface which would include the diagram visualised. In the theoretical part, the definition of a process diagram was studied with additional attention to process modelling levels. Also, the data structure of the applied diagram was identified, and its theoretical fundamentals were presented. The thesis is a case study performed with a constructive research method.

The implementation of the web interface was built around diagram visualisation. The diagram's identified data structure was applied in the remodelling of the diagram from PDF file format to CSV format. After loading the diagram data into a JavaScript library capable of the visualisation, different layout algorithms were compared. Differences in readability were noticed between algorithm-produced layout drawings. Using the most natural layout producing algorithm, a website was programmed to present the process diagram. The visualisation was wrapped into a React component which was later integrated into the web interface. Utilising a Material-UI library offering styled React components, the interface was realised using TypeScript programming language which provides static typing to the otherwise dynamically typed JavaScript. The implemented interface was tested with three use cases in which the interface functioned as expected.

Finally, the results were analysed and proposals for further development were considered. Save functionality for selected process phases was suggested, among other things.

The implemented web interface provided considerable benefit to the planning and further development of smart device testing within the project.

KEYWORDS:

process, diagram, graph, interface, visualisation

SISÄLTÖ

LIITTEET

KUVAT

KUVIOT

SANASTO

CSV	Tiedostomuoto, jossa arvot listataan taulukkomaisesti pilkuilla erotettuna; Comma Separated Values (Shafranovich 2005)
JSON	Jäsennetyn tiedon tiedostomuoto, joka noudattaa JavaScript-ohjelmointikielen objektimerkintää; JavaScript Object Notation (Bray 2017)
konekieli	Tietokoneen suorittamat operatiiviset käskyt, kuten yhteenlasku, käsitellyille arvoille (Hemmendinger 2016)
kuvaustaso	Prosessikuvauksen tarkkuutta määrittelevä periaate, joka riippuu yleensä kuvauksen käyttötarkoituksesta (Julkishallinnon suositus 2012)
osaprosessi	Prosessihierarkian pienempi yksikkö, kun prosessi on jakautunut (Julkishallinnon suositus 2012)
PDF	Sähköisten dokumenttien tiedostomuoto, jossa dokumentin sisältö esitetään samanlaisena alustasta riippumatta; Portable Document Format (Adobe 2006, 23–24)
prosessi	Sarja toistuvia toimintoja, joissa syötteet muutetaan tuotoksiksi (Julkishallinnon suositus 2012)
prosessikaavio	Tapa esittää prosessin toiminnot visuaalisesti (Julkishallinnon suositus 2012)
tietovirta	Tiedon siirtyminen prosessissa lähteestä kohteeseen (Tietotermit 2018)
toiminto	Tietyn tuloksen tuottava sarja tehtäviä (Julkishallinnon suositus 2012)

1 JOHDANTO

Tietoverkkoja käytettävien laitteiden kasvava määrä on nostanut huolen laitteiden tietoturvallisuudesta. Yhä useampi elektroninen laite kuluttaja- ja teollisuussektorilla käyttää hyväkseen tietoja, joiden suojaton säilytys ja liikenne tuottaa huomattavia riskejä yhteiskunnan turvallisuudelle ja yksityisyydelle. Näitä uhkakuvia voidaan ehkäistä priorisoimalla älylaitteiden ja niihin liittyvien palveluiden tietoturvallisuus kehitysvaiheessa. (ENISA 2017, 7) Suomalaisten korkeakoulujen ja alueellisten liittojen yhteistyöhanke KyberVALIOT vastaa tähän tarpeeseen tarjoamalla valmistavalle teollisuudelle tietoturvapalveluita, jotka sisältävät muun muassa konsultointia ja laitetestausta (Turun ammattikorkeakoulu ei pvm.). Tämä tutkimus liittyy KyberVALIOT-hankkeen laitetestausprosessin kehittämistarpeeseen. Turun ammattikorkeakoulu hankkeen osapuolena esitti tarpeensa käyttöliittymästä, joka sisältäisi organisaation laitetestausprosessia kuvaavan kaavion. Toteutusta käytettäisiin esimerkiksi asiakastilaisuuksissa tarvittavien testausvaiheiden valitsemisessa.

Työssä tutkitaan sovellettavaa prosessikaaviota, sen tietorakennetta ja kaavion visualisointia verkkosivulla. Tavoitteena on tunnistaa kaavion yleispiirteitä prosessikuvausmäärittelyiden avulla, sillä prosessikaavioiden monipuolisuus ja kuvauksen tarkkuus vaihtelee käyttötarkoituksen mukaan. Toisekseen pyritään selvittämään kaavion tietorakenne ja soveltamaan tunnistettua tietorakennetta käyttöliittymätoteutuksessa. Kaavio on tallennettu PDF-tiedostomuotoon, jonka hyödyntäminen verkkosivulla ei vastaa tiedostomuodon käyttötarkoitusta ja jonka koneluettavuus on heikko. Prosessikaavio olisi tallennettava verkkosivulla käytettäville tekniikoille luettavampaan muotoon, jossa kaavion tietorakenne määrittelisi tietojärjestyksen. Lopullisena tavoitteena on luoda toimeksiantajan tarpeita vastaava käyttöliittymä, jonka visualisoidusta kaaviosta käyttäjä voisi valita haluamiaan prosessivaiheita.

Prosessikaavion visuaalisuus heijastuu työn painopisteisiin: keskeisenä tutkimuskysymyksenä on, miten kaavio voidaan esittää graafisesti web-käyttöliittymässä. Kysymys voidaan jakaa kolmeen osaan: mitä testausprosessikaavio esittää, mitä edellytyksiä kaavion visualisointi verkkosivulla vaatii ja miten kaavion sisältävä käyttöliittymä toteutetaan. Tutkimus keskittyy vastaamaan kahteen ensimmäiseen kysymykseen painottuen vähemmän verkkosivun kehitysteknologioihin. Työn tulokset raportoidaan aihepiireittäin pääsääntöisesti teorian esittelyn jälkeen, jotta lukijan olisi vaivattomampi seurata

selvitetyn tiedon soveltamista. Tutkimuksen tulosten ja teorian jäsentely seuraa konstruktiivista tutkimusotetta, joka tarkoittaa ennalta tutkitun tiedon soveltamista uusien realiteettien luomisessa. Lopussa esitetään tulosten johtopäätöksiä ja jatkokehitysehdotuksia.

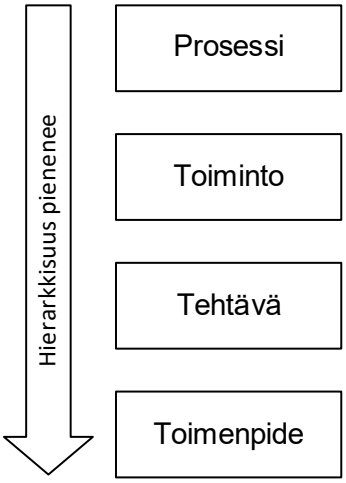
Työn toimeksiantajana toimi Turun ammattikorkeakoulu.

2 PROSESSIKAAVIO

2.1 Määritelmä

Prosessikaavio on tapa esittää prosessi visuaalisesti. Siinä prosessin toiminnot, tietovirrat ja tuotteet kuvataan niille ominaisin symbolein. Yleisesti käytetty BPMN-määritys (Business Process Model and Notation) on prosessien mallintamiseen tarkoitetuista spesifikaatioista, joka määrittelee prosessikaavioissa käytetyt symbolit ja muun graafisen ilmeen. (Julkishallinnon suositus 2012)

Kuvattavan prosessin yksityiskohtaisuus nousee esille prosessin jakautuessa toimintoihin sekä edelleen tehtäviksi ja toimenpiteiksi. Toiminto on tietyn tuloksen tuottava joukko tehtäviä. Tehtävä tarkoittaa käsittelyvaihetta, joka sisältää vastuuhenkilön tai -henkilöiden suorittamia käytännön toimenpiteitä. Kuviossa 1 on esitetty edellä mainittu hierarkia, jossa prosessi on kuvauksen suurin yksikkö ja toimenpide pienin. (Julkishallinnon suositus 2012)



Kuvio 1. Prosessi jakautuu toimintoihin, tehtäviin ja toimenpiteisiin.

Jotta prosessikaavio välittäisi kaiken tarpeellisen tiedon prosessista, on prosessi kuvattava kaavion käyttötarkoitukseen sopivalla kuvaustasolla. (Julkishallinnon suositus 2012)

2.2 Kuvaustasot

Kuvaustasoilla tarkoitetaan, kuinka yksityiskohtaisesti prosessi mallinnetaan. Julkishallinnon suositus (2012) nro. 152 luettelee neljä eri kuvaustasoa: prosessikartta, toimintamalli, prosessin kulku ja työn kulku. Näistä työn kulku on tarkin kuvaustaso. Riippuen kuvauksen käyttötarpeesta kuvausten tasot voivat olla päällekkäisiä.

Suosituksen mukaan kuvauksista on tehtävä sitä muodollisempia, mitä tarkemmalla tasolla prosessi mallinnetaan. Muodollisuus tarkoittaa esimerkiksi BPMN-määrityksen noudattamista prosessia kuvatessa.

2.2.1 Prosessikartta

Pelkistetyimmällä ja ylimmällä kuvaustasolla prosessikartta hahmottaa organisaation toiminnan kokonaisvaltaisesti. Prosessikartassa esitetään organisaation ydinprosessit ja toimintaan vaikuttavat ulkopuoliset tekijät. Tällä tasolla prosessien yhteisiä riippuvuuksia ei mallinneta. (Julkishallinnon suositus 2012)

2.2.2 Toimintamalli

Toimintamallitasolla esitetään, kuinka ydinprosessit jakautuvat osaprosesseiksi, ketkä ovat prosessien omistajat, mitkä ovat prosessien tavoitearvot ja -mittarit sekä minkälainen on prosesseihin vaikuttava ympäristö. Kuvaustasolla kaaviota täydentävät kuvausta tarkentavat tekstidokumentit. (Julkishallinnon suositus 2012)

2.2.3 Prosessin kulku

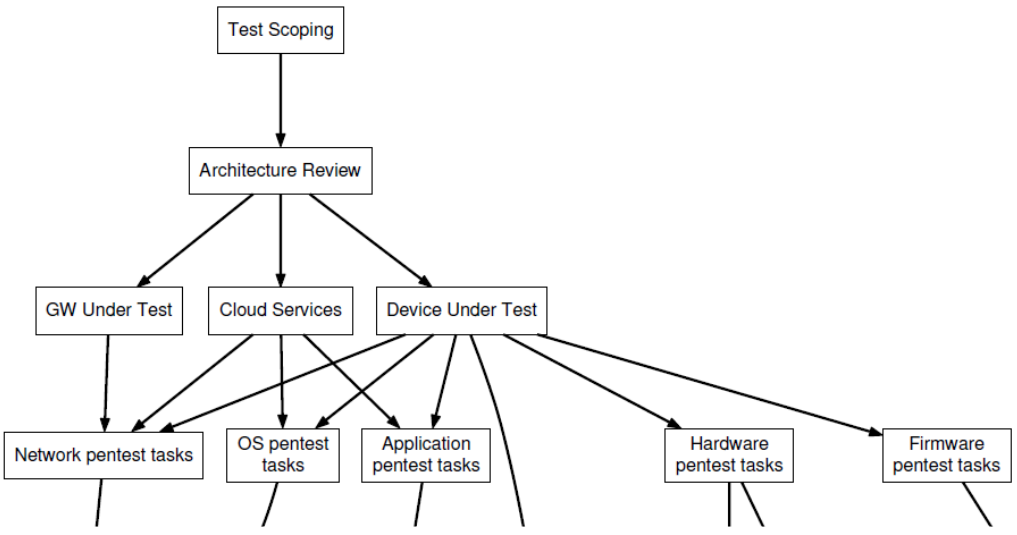
Prosessin kulun tasolla kuvataan samat asiat kuin toimintamallitasolla mutta yksityiskohtaisemmin. Tasolla yksittäiseen kuvaukseen valitaan prosessi, jonka osaprosessit, toiminnot, tehtävät ja resurssit mallinnetaan. Osaprosessit jakautuvat toiminnoiksi, tehtäviksi ja toimenpiteiksi, ja näille määritellään hierarkkinen suoritusjärjestys. (Julkishallinnon suositus 2012)

2.2.4 Työn kulku

Tarkimmin prosessia kuvataan työn kulun tasolla, jolla erityisesti määritellään eri toimintojen välillä liikkuvan tiedon muoto eli toisin sanoen tietotyyppi. Kuvaustason tarkkuus vaatii prosessin toimintojen ja toimenpiteiden numerointia, toimenpiteiden syötteiden ja tuotoksien merkintää sekä edellä mainittujen resurssien pituuden ja muodon määrittelyä. Tason mukaisesti luodun kuvauksen pohjalta voidaan toteuttaa esimerkiksi työohjeet tai sähköinen palvelu. (Julkishallinnon suositus 2012)

2.3 KyberVALIOT-hankkeen prosessikaavio

Turun ammattikorkeakoulun mallintamaan KyberVALIOT-hankkeen prosessikaavioon (kuva 1 ja liite 1) on kuvattu älylaitteen tietoturvatestausprosessi, jossa älylaitteelle suoritetaan erilaisia tietoturvaan liittyviä toimenpiteitä, kuten tietomurtotestausta. Kaaviota on tarkoitus käyttää prosessien hallintaan, asiakkaiden tarpeiden arvioimiseen ja työvaiheiden seuraamiseen. Tässä opinnäytetyössä kaaviota käytetään myös interaktiivisen web-käyttöliittymän toteuttamiseen.



Kuva 1. Kuvakaappaus KyberVALIOT-hankkeen prosessikaaviosta (liite 1).

Kuvan 1 kaavio koostuu prosessivaiheista, niiden välisistä nuolista sekä vaiheiden automatisointitilan merkinnöistä. Kaavion prosessivaiheet on järjestetty hierarkkisesti: yksi tai

useampi prosessin vaihe edeltää seuraavaa vaihetta. Hierarkkisuutta kuvaa prosessivaiheiden väliset nuolet. Esimerkiksi testauksen laajuuden määrittämisvaiheesta (Test Scoping) lähtevä nuoli osoittaa arkkitehtuurin arviointivaiheeseen (Architecture Review), mikä tarkoittaa ensimmäisen vaiheen suorittamista prosessissa ennen jälkimmäistä. Kaaviossa ei kuitenkaan ole määritelty tarkempia edellytyksiä prosessin suorittamiselle tilanteessa, jossa yhtä prosessivaihetta edeltää useampi muu vaihe.

Julkishallinnon suosituksen nro. 152 mukaan prosessikuvauksen kuvaustason on vastattava kuvauksen käyttötarkoitusta. Kuvaustasot voivat joissain tapauksissa olla kaikesta huolimatta päällekkäisiä. Turun ammattikorkeakoulun laitetestausprosessikaavion kuvaustaso voidaan arvioida toimintamallin ja prosessin kulun välille. Toimintamallitasolla kuvataan, kuinka ydinprosessi jakautuu osaprosesseiksi. Laitetestausprosessi on tässä tapauksessa kuvattu ydinprosessi, joka koostuu tietoturvatestaukseen liittyvistä toiminnoista. Kaavion suorakulmioilla merkityt prosessivaiheet esittävät toimintoja, jotka sisältävät joukon kyseiseen toimintoon liittyviä tehtäviä ja toimenpiteitä. Esimerkiksi käyttöjärjestelmän tietomurtotehtävät -prosessivaihe (OS pentest tasks) kuvaa toimintoa, joka koostuu ennalta sovituista tietohyökkäyksistä älylaitteen käyttöjärjestelmää vastaan.

Kaaviossa kuvataan prosessia osin tarkemmin kuin toimintamallitasolla. Se toteuttaa myös prosessin kulku -kuvaustason vaatimuksia, kuten työvaiheiden järjestyksen kuvaamisen. Toimintojen hierarkkinen kuvaus nuolimerkinnöillä esittää toimintojen suoritusjärjestyksen. Prosessin kulku -tasolla voidaan kuvata myös prosessin toimijat, prosessin jakautuminen pienempiin työyksiköihin ja prosessiin liittyvät resurssit, mutta näitä kyseisessä prosessikaaviossa ei ole kuvattu sen käyttötarkoituksen takia.

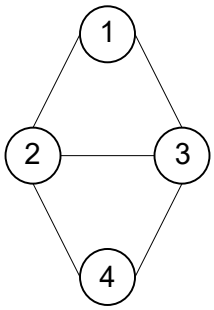
Kaavio on tallennettu PDF-tiedostomuotoon, jota ei ole tarkoitettu koneellisesti luettavaksi (Adobe 2006, 23–24). Jotta KyberVALIOT-hankkeen prosessikuvausta voidaan soveltaa interaktiivisen web-käyttöliittymän toteuttamisessa, on sen tietorakenne tulkittava ja mallinnettava uudelleen tietojärjestelmälle luettavammaksi. Eräs tapa parantaa koneluettavuutta on kuvailla kaavio tekstimuodossa, mikä vaatii kaavion tietorakenteen tunnistamisen ja sen hyödyntämistä kaavion tietoja syöttäessä.

3 VERKKOTIETORAKENNE

3.1 Määritelmä

Tietorakenteet ovat tietojoukon järjestettyjä esityksiä, joissa kuvataan yksittäisten tietoalkioiden väliset loogiset suhteet. Tietoalkiot edustavat esimerkiksi (joukkoa) numeroita tai kirjaimia osana suurempaa kokonaisuutta. Alkioiden järjestäminen malleihin nopeuttaa tiedonkäsittelyä, auttaa organisoimaan tietoa ja mahdollistaa tehokkaiden ohjelmistojen toteutuksen. Tietorakenteet jakautuvat primitiivisiin ja epäprimitiivisiin. Primitiivisiä tietorakenteita ovat esimerkiksi kokonaisluku, totuusarvo ja kirjainmerkki, joita voidaan käsitellä konekielen käskyillä suoraan. Epäprimitiiviset tietorakenteet sisältävät primitiivisiä tietorakenteita monimutkaisemmassa järjestelyssä, joiden käsittelylle ei löydy konekielestä valmiita käskyjä, vaan niiden käsittelyssä käytetään erilaisia käskyjoukkoja. Epäprimitiiviset tietorakenteet jakautuvat edelleen lineaarisiin ja epälineaarisiin rakenteisiin, jotka erotetaan siten, mikä on peräkkäisten tietoalkioiden suhde ja kuinka monta kertaa tietorakenne on käytävä läpi ennen kuin jokaisessa alkiossa on vierailtu. Lineaarisissa tietorakenteissa, kuten taulukoissa, alkiot järjestetään peräkkäin ja läpikäyntejä tehdään vain kerran. Epälineaarisissa tietorakenteissa, kuten verkossa, alkioilla on mahdollista olla yhteydessä useampaan kuin yhteen muuhun tietoalkioon ja läpikäyntejä on tehtävä useampia. (Debdutta 2018, 19–21)

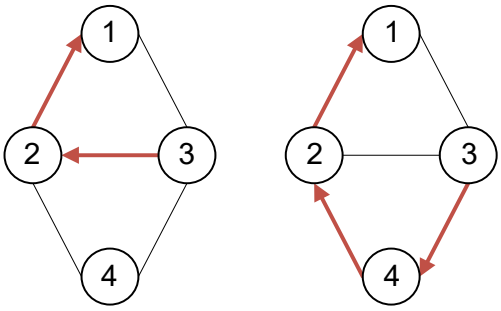
Verkko koostuu solmuista ja kaarista. Solmut ovat verkkotietorakenteen tietoalkioita, joiden väliset yhteydet merkitään kaarilla. Esimerkiksi kuvion 2 verkossa on neljä solmua ja viisi kaarta. Kahden solmun sanotaan olevan vierekkäisiä, mikäli niitä yhdistää kaari. Solmun naapureiksi kutsutaan verkon muita solmuja, joihin solmu on yhteydessä kaarella. Kuvion 2 solmun 3 naapurit ovat solmut 1, 2 ja 4. Solmulle voidaan määrittää aste, joka on solmun naapureiden määrä. Edellä mainitun solmun 3 aste on tällöin kolme. (Laaksonen 2020)



Kuvio 2. Verkko, jossa on neljä solmua.

3.2 Polku

Solmujen osajoukon läpikäynnille voidaan määrittää polkuja, jotka kulkevat kaaria pitkin lähtösolmusta kohdesolmuun (Laaksonen 2020). Kuviossa 3 on esitetty kahden identtisen verkon polut lähtösolmusta 3 kohdesolmuun 1. Vasemman verkon polku kulkee solmun 2 kautta. Oikean verkon polku kulkee solmun 4 ja 2 kautta.



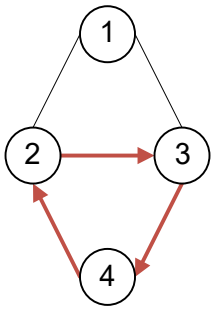
Kuvio 3. Kaksi verkon polkua punaisilla nuolilla merkittynä.

3.3 Syklisyys

Verkon polku on sykli, jos se täyttää seuraavat vaatimukset:

- polun lähtö- ja kohdesolmu on sama
- se kulkee vähintään yhtä kaarta pitkin
- se ei kulje yhtä useampaa kertaa saman kaaren tai solmun kautta (Laaksonen 2020).

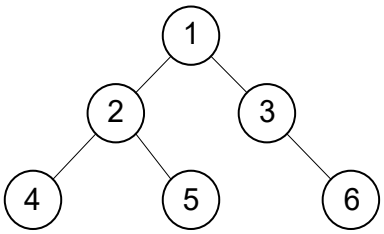
Kuvion 4 syklillisen polun lähtö- ja kohdesolmu on solmu 3. Lisäksi polku kulkee kolmea kaarta pitkin eikä etene saman solmun tai kaaren kautta useampaa kertaa. Verkko on syklitön, jos verkon mikään polku ei täytä kaikkia edellä mainittuja vaatimuksia (Laaksonen 2020).



Kuvio 4. Syklillinen polku.

3.4 Puu

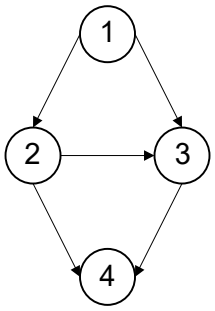
Puu on syklitön verkko, jonka solmujen määrä on yhtä pienempi kuin kaarien määrä. Lisäksi puurakenteessa voidaan kulkea solmusta toiseen vain yhtä tiettyä polkua pitkin (Laaksonen 2020). Esimerkiksi kuvion 5 puussa solmujen 3 ja 5 välillä kulkee yksi polku solmujen 2 ja 1 kautta.



Kuvio 5. Kuuden solmun ja neljän kaaren puu.

3.5 Suunnattu verkko

Suunnatussa verkossa kaaret merkitään nuolimerkein tarkoittaen yksiselitteistä suuntaa, jota polun on noudatettava kulkiessaan verkon kaaria pitkin (Laaksonen 2020). Kuvion 6 suunnatussa verkossa on kolme polkua lähtösolmusta 1 kohdesolmuun 4: solmun 2 kautta, järjestyksessä solmujen 2 ja 3 kautta sekä solmun 3 kautta. Kulkien ensin solmun 3 ja sitten solmun 2 kautta ei ole mahdollista, sillä kaaren suunta on vastakkainen.



Kuvio 6. Suunnattu verkko.

3.6 KyberVALIOT-hankkeen prosessikaavion verkko

Visuaalisesti tarkasteltuna KyberVALIOT-hankkeen prosessikaavion (liite 1) tietorakenne vastaa syklitöntä suunnattua verkkoa. Kaaviossa suorakulmioilla merkityt prosessin toiminnot esittävät verkon solmuja ja toimintojen väliset nuolet verkon kaaria. Yhteensä kaavion verkossa on 53 solmua ja 66 kaaria. Kaikki kaaret on merkitty nuolilla, mikä tarkoittaa verkon olevan suunnattu. Syklittömyys havaitaan tarkastelemalla verkkoa: ei ole olemassa polkua, jossa voidaan palata lähtösolmuun.

4 WEB-KÄYTTÖLIITTYMÄN TOTEUTUS

4.1 PDF-tiedoston soveltaminen

Jotta prosessikaaviota voidaan hyödyntää web-käyttöliittymässä, on kaavio esitettävä web-selaimelle ymmärrettävässä muodossa. Prosessikaavio on tallennettu PDF-tiedostoon, joka voidaan asettaa verkkosivulle tiedostolataukseksi tai sivulle upotettavaksi erilliseksi kehykseksi. (W3Docs 2020) Molemmissa tapauksessa käyttäjältä vaaditaan PDF-muotoa ymmärtävä selainlisäosa tai erillinen ohjelma, mikä saattaa käyttäjälle olla mahdollinen käytön este. Lisäksi suunniteltua käyttöliittymän prosessivaiheiden valintatoimintoa ei olisi mahdollista toteuttaa, sillä kaavion esittäminen olisi toteutuksen hallinnan ulkopuolella käyttäjän PDF-ohjelmiston tulkittavana.

PDF-tiedoston latauksen ja upotuksen sijaan tiedoston sisältö voidaan tulkita ja esittää verkkosivuna JavaScript-ohjelmakirjastolla kuten PDF.js-kirjastolla. Tällöin dokumentin elementit ja tyylit luetaan ja tulkitaan verkkosivulla selaimen ymmärtämäksi HTML5-merkkikieleksi. Tulkitsevaa ohjelmakirjastoa käytettäessä verkkosivun kehittäjällä on mahdollisuus esimerkiksi muokata elementtejä ja lukea tekstejä, mikä mahdollistaa esimerkiksi edellä mainitun valintatoiminnon toteuttamisen. (Mozilla and individual contributors 2020)

Pääasiallinen käyttötarkoitus PDF-tiedostomuodolle on esittää dokumentteja kuin ne on tarkoitettu näyttävän alustasta riippumatta (Adobe 2006, 23–24). Prosessikuvauksen hallinnan – esimerkiksi muutoksien käsittelyn – kannalta PDF-tiedostojen muuttaminen ei palvele tätä käyttötarkoitusta. Yleensä PDF-tiedostoja edeltävät muut tiedostomuodot, ja PDF-tuloste esittää viimeistä, vain tarkasteltavaksi tarkoitettua dokumenttia. Laitetestausprosessikaavion PDF-tiedoston visuaaliset objektit on kuvattu DOT-kielellä, jonka avulla voidaan esittää suunnattuja verkkotietorakenteita tekstimuodossa. Tekstitiedostoja tulkitsee dot-ohjelma, joka piirtää käyttäjän antamien asetusten mukaisen graafisen esityksen, kuten PDF-tulosteen. (Gansner ym. 2015, 2) Käyttöliittymätoteutus tehdään saatavilla olevan PDF-dokumentin pohjalta, joka mallinnetaan uudelleen tekstimuotoon.

4.2 Prosessikaavion listaus CSV-muotoon

Eräs tiedonvaihtoon ja muuntamiseen käytetty tiedostomuoto on CSV (Comma Separated Values), jossa tietoalkiot tallennetaan tekstimuodossa pilkuilla erotettuna. Jokainen tiedoston rivi tietoalkioineen edustaa yhtä kohdetta tai objektia, jossa tietoalkiot ovat järjestyksessä sovitussa järjestyksessä kenttien mukaan. Ylimmällä CSV-tiedoston rivillä on mahdollista nimetä kentät eli mitä tietoa kukin tietoalkio edustaa. (Shafranovich 2005)

Kaavion verkon esittäminen CSV-muodossa voidaan tehdä listaamalla solmut ja kaaret erikseen. Tekniikka soveltaa ohjelmoinnissa käytettyä kaarilistausesitystä (Laaksonen 2020). Ensiksi jokaiselle solmulle ja kaarelle määrätään oma uniikki tunniste omissa joukoissaan, jotta yksittäiset solmut ja kaaret pystytään erottamaan toisistaan rajatapauksissa. Liitteessä 2 on käytetty juoksevaa numerointia siten, että punaiset numerot edustavat solmun tunnistetta ja vihreät numerot kaaren tunnistetta. Numerointi alkaa solmun ja kaaren kohdalla ykkösestä, ja järjestys on mielivaltainen.

Tunnisteen lisäksi solmuilla ja kaarilla on muita ominaisuuksia kuten prosessin toimintojen – tai verkon tapauksessa solmujen – nimet. CSV-tiedostoissa ominaisuudet toimivat kenttinä. Solmuille merkitään kentät *id* eli tunniste, *group* eli ryhmä ja *label* eli nimike. CSV-muodossa kenttärivi on tällöin muodossa *id,group,label*. Kaarien ominaisuudet *id*, *source_id*, *target_id* ja *label* kertovat, mikä on kaaren tunniste, mitkä ovat kaaren lähtö- ja kohdesolmut sekä minkälainen solmujen välinen suhde on. *label*-kentässä käytetään *HAS_PARENT*-merkintää ilmaisemaan kohdesolmun (*target_id*) olevan isäntäsolmu lähtösolmulle (*source_id*). Lähtösolmua kutsutaan tällöin lapsisolmuksi.

Solmulistauksen sisältävän CSV-tiedoston neljä ensimmäistä riviä ovat seuraavat:

```
id,group,label

1,CyberELITE,Test Scoping

2,CyberELITE,Architecture Review

3,CyberELITE,GW Under Test
```

Solmulistauksesta huomataan, että *Test Scoping* -solmulla on tunniste 1 ja solmu kuuluu ryhmään *CyberELITE*. Ryhmän merkitseminen mahdollistaa muiden verkkojen liittämisen listaukseen niin, että eri ryhmien solmut voidaan erotella omiksi joukoiksi.

Kaarilistauksessa ylimmällä rivillä sijaitsevat kentät ovat erilaiset:

```
id,source_id,target_id,label

1,2,1,HAS_PARENT

2,3,2,HAS_PARENT

3,4,3,HAS_PARENT
```

Listauksessa solmu 1 esiintyy toisella rivillä kaaren 1 kohdesolmuna, joka on solmun 2 isäntäsolmu. Lähtö- ja kohdesolmujen välisestä suhteesta kertovan *label*-kentän arvona on isäntäsolmuominaisuutta kuvaava *HAS_PARENT*-teksti, mutta kenttä sallii myös muiden ominaisuuksien merkitsemisen. Laitetestausprosessin vaiheille voidaan esimerkiksi kartoittaa vastaavat muiden organisaatioiden prosessivaiheet siten, että muiden verkkojen solmut listataan omiin ryhmiinsä ja eri ryhmiin kuuluvien solmujen suhdetta kuvataan ominaisuudella kuten *CORRESPONDS* (vastaa).

Solmujen ja kaarien syöttäminen CSV-tiedostoihin osoittautui nopeaksi tavaksi merkitä prosessikaavion verkko muistiin. Kahden tiedoston tarkastelu auttoi hahmottamaan verkon komponenttien merkityksen, kun ne olivat listattuna erillisinä joukkoina. Toisaalta yksittäisiä solmuja pelkällä numerotunnisteella oli hankala tunnistaa, sillä kaarilistauksessa oli merkitty vain solmun tunniste eikä nimeä.

4.3 Komentosarja CSV–JSON-muunnokselle

Käyttäjän selaimen on tulkittava verkkotietorakenne JavaScript-objektiksi, jonka visualisointiohjelmakirjasto lukee. Jotta prosessikaavion verkko voitaisiin ladata JavaScript-kirjastoon, täytyy verkko muuntaa sen käyttämään muotoon. Komentosarja – kuten tässä työssä kaikki muu ohjelmointi paitsi itse visualisointikomponentti kielituen puuttuessa – toteutetaan JavaScriptiä syntaktisesti laajentavalla TypeScript-kielellä. Se tarjoaa JavaScriptiin vahvan tyypityksen, jolla JavaScriptin heikon tyypityksen tuomat virheet voidaan ohjelmoinnin aikana välttää. (Microsoft 2020)

JSON (JavaScript Object Notation) on JavaScript-objektien esittämiseen tarkoitettu syntaksi, jota JavaScript-kielellä tulkitaan siihen sisäänrakennetulla funktiolla (Mozilla and individual contributors 2020). Edellä mainittu komentosarja muuttaa CSV-tiedostot (solmu- ja kaarilistaukset sisältävät *nodes.csv* ja *edges.csv*) kahdeksi JSON-tiedostoksi: *graph.json* ja *graph.meta.json*. Ensimmäinen tiedosto (liite 4) sisältää verkon kokonaisuudessaan noudattaen liitteessä 3 määriteltyä JSON-tietomallia (schema). Solmut ja

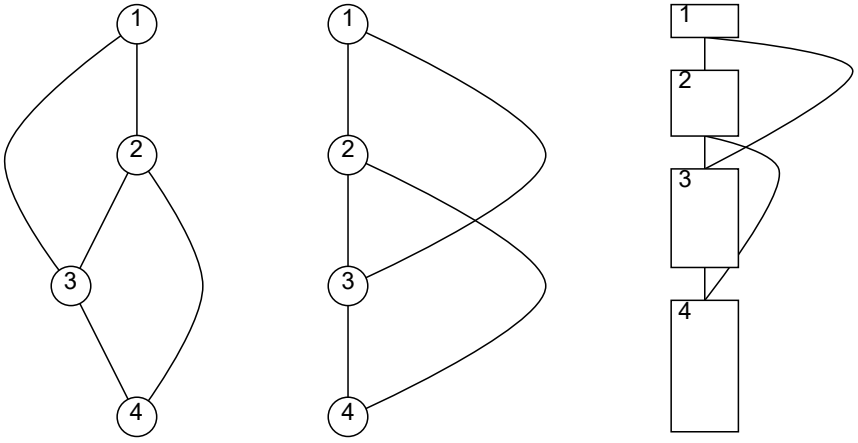
kaaret listataan taulukoihin kahden eri objektin ominaisuuden alle: *nodes* eli solmut ja *edges* eli kaaret. Taulukoissa jokaista solmua ja kaarta vastaa objekti, jonka ominaisuuksina on CSV-tiedostoissa määritetyt kentät.

Komentosarja tulostaa toiseen JSON-tiedostoon (liite 6) muodonmuunnoksen aikaleiman ja SHA256-tiivisteen. Aikaleimasta selviää, milloin koordinoidussa yleisajassa (UTC) komentosarja on suoritettu. SHA256-tiiviste edustaa verkon sisältävän tiedoston *graph.json* identiteettiä, joka vaihtuu aina, kun tiedostoa muokataan. Edellä mainitut metatiedot tarjoavat vaihtoehtoisen versioinnin verkon sisällölle, jonka muuttumisen seuraamisesta saatu tieto voi paljastua hyödylliseksi prosessin tarkistelijalle ja ylläpitäjälle. Metatietoja sisältävä tiedosto seuraa liitteessä 5 kuvailtua JSON-tietomallia.

4.4 Prosessikaavion visualisointi

Luodut JSON-tiedostot esitetään käyttöliittymän verkkosivulla toimivalla visualisointikirjastolla. Eräs tällainen kirjasto on d3-dag, joka täydentää suosittua datan visualisointikirjastoa D3 tarjoamalla asettelualgoritmeja suunnatuille syklittömille verkoille (directed acyclic graph). Algoritmit tuottavat solmujen ja kaariviivojen sijaintia koskevia tietoja, jotta verkko voidaan järjestää visuaalisesti erilaisia käyttötarkoituksia varten. (Brinkman 2020)

Kirjasto sisältää kolme asettelualgoritmia: Sugiyama, Zherebko ja Arquint (Brinkman 2020). Sugiyama-algoritmi perustuu Kozo Sugiyaman ynnä muiden vuonna 1979 esiteltyihin ideoihin solmujen asettelusta hierarkkisiin tasoihin. d3-dag-kirjaston toteutus algoritmista jakautuu kolmeen vaiheeseen. Ensiksi verkon solmut jaetaan tasoiksi niin, että jokaiselle solmulle määrätään tasoa kuvaava kokonaisluku. Lapsisolmuilla numero on suurempi kuin isäntäsolmuilla. Toiseksi jokaisen tason solmut asetetaan järjestykseen, jossa toisensa ylittävät kaaret minimoidaan. Viimeiseksi algoritmi määrää solmuille x- ja y-koordinaatit riippuen tasosta ja solmujen järjestyksestä. (Brinkman 2020) Tuloksena saadaan selkeä, puumainen rakenne. Kuvion 7 Sugiyama-kaaviossa nähdään, kuinka solmut sijaitsevat neljällä eri tasolla ja miten päällekkäistä piirtämistä ehkäistään, kun solmut ja kaaret välttävät toisiaan.



Kuvio 7. d3-dag-kirjaston piirtämät kaaviot kuvion 5 verkosta. Asettelualgoritmit vasemmalta oikealle: Sugiyama, Zherebko ja Arquint.

Toinen algoritmi, Zherebko, perustuu solmujen topologiseen esitykseen, jossa solmuilla on sama x- tai y-koordinaatti. Solmut asettuvat samalle tasolle, ja kaaret piirretään tason ulkopuolelle omille tasoilleen. (Brinkman 2020) Kuviossa 7 Zherebko-algoritmin tuottamassa asettelussa solmut asettuvat vertikaaliselle tasolle, jonka solmujen väliset kaaret kiertävät.

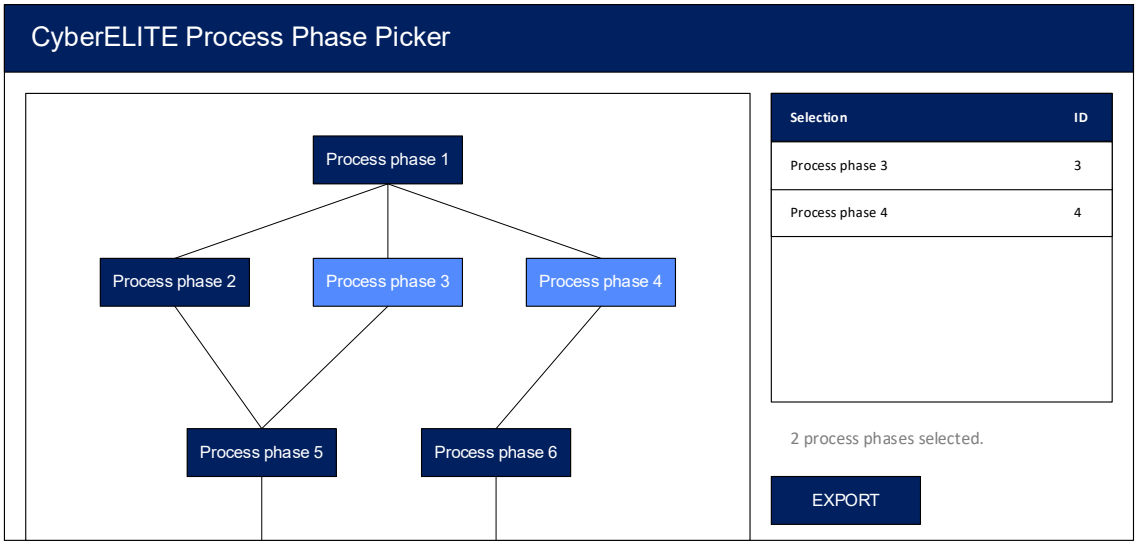
Arquint-algoritmi soveltuu korkeustietoja sisältävien solmujen, kuten suorakulmioiden, piirtämiseen. Algoritmille syötetään solmujen välinen korkeussuhdefunktio, joka määrittää kunkin solmun suorakulmion korkeuden. (Brinkman 2020) Kuvion 7 viimeisessä kaaviossa suorakulmioiden korkeus on riippuvainen kunkin solmun tunnistenumerosta.

Käyttöliittymässä tavoitteena oli seurata prosessikaavion PDF-tulosteessa noudatettua asettelua, mikä ohjasi sopivan asettelualgoritmin valintaa. Liitteissä 7–9 on piirretty prosessikaavion verkko käyttäen kaikkia kolmea visualisointikirjaston tarjoamaa algoritmia oletusasetuksilla. Sugiyama-asettelu (liite 7) tuotti selkeän, hierarkkisiin tasoihin jakautuvan esityksen. Pitkiä kaaria piirtävän Zherebko-algoritmin kaavio (liite 8) on hankalasti tulkittava, sillä kaaviota tarkastellessa on vaikeaa seurata pitkälle ulottuvia viivoja toiseen solmuun. Lisäksi solmujen sijainti yhdellä tasolla aiheuttaa kaavion tarpeettoman venymisen. Solmujen korkeustietoja hyväksikäyttävä Arquint-asettelu loi Sugiyama-asettelua muistuttavan piirroksen (liite 9), mutta käytetty prosessikuvaus ei sisällä esimerkiksi kvantitatiivisia ominaisuuksia, kuten älylaitteen testauskertoja prosessitoimintoa kohden, joita tässä tapauksessa voitaisiin soveltaa.

Lopullisen toteutuksen algoritmiksi valittiin Sugiyama, jonka asettelu selkeydellään ja puumaisella rakenteellaan muistuttaa alkuperäistä prosessikaaviota. Toteutus tehtiin d3-dag-kirjaston versiolla 0.3.5, joka ei tue TypeScriptiä, vaan kirjastoa oli käytettävä dynaamisesti tyypitetyllä JavaScriptillä. Verkkosivulle ensimmäiseksi piirrettiin solmut ympyröinä ja kaaret viivoilla vektorigrafiikkaa toteuttavana SVG-elementtinä (Scalable Vector Graphics). Tämän jälkeen solmukohtiin asetettiin prosessivaiheen nimi. Teksti kaarevan elementin sisällä vei liikaa tilaa kaaviosta, joten solmujen ympyrät muutettiin suorakulmioiksi. Tekstin keskittäminen suorakulmioon osoittautui haasteelliseksi, sillä selaimet tulkitsivat tekstiasettelun tyylisääntöjä eri tavalla. Lopuksi visualisoinnista luotiin sovelluskehyksen komponentti, joka liitettiin osaksi käyttöliittymää.

4.5 Käyttöliittymä

Toimeksiantajan ensisijaisina vaatimuksina käyttöliittymälle oli prosessivaiheiden valintatoiminto ja valittujen vaiheiden nimien sekä tunnisteiden listaaminen. Suunnittelun aikana lisävaatimuksiksi asetettiin johdonmukaisuus ja helppokäyttöisyys, jotta verkkosivun käyttökokemuksesta tulisi mahdollisimman suoraviivainen. Tavoitteet suunniteltiin saavutettavaksi käyttöliittymän pääkomponenttien yhtenäisellä ilmeellä ja loogisella sijoittelulla. Myös prosessikaavion keskeinen rooli otettiin huomioon kokonaisuutta suunniteltaessa. Kuvan 2 suunnittelumallissa kaaviolle on varattu suurin osa näyttöalasta.

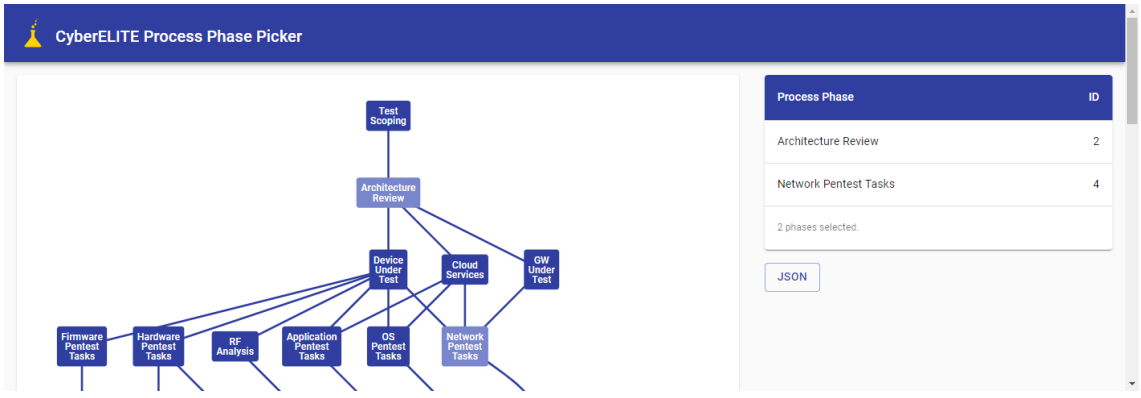


Kuva 2. Käyttöliittymän suunnittelumalli.

Käyttöliittymä suunniteltiin sisältävän kolme pääosaa: otsikkokomponentti sekä vasen ja oikea paneelikomponentti. Otsikossa tulisi lukemaan sivua kuvaava nimi suurella fontilla. Vasen paneeli koostuisi prosessikaaviosta, joka leveydellään hallitsisi käyttäjäkokemusta. Oikeassa paneelissa listattaisiin käyttäjän valitsemat prosessivaiheet sekä ilmoitettaisiin, kuinka monta vaihetta on valittu. Lisäksi sivulle lisättäisiin nappi valintatietojen viemistä varten esimerkiksi leikepöydälle.

Toteutus ohjelmoitiin TypeScript-kielellä ja tyyliteltyjä React-komponentteja tarjoavalla Material-UI-kirjastolla (Material-UI 2020). React-ohjelmistokehys mahdollistaa interaktiivisten verkkosivujen luonnin synkronoitujen komponenttien avulla, jotka auttavat jäsentämään ohjelmakoodia eristämällä sivun elementit omiksi tiedostoiksi (Facebook Inc. 2020). Material-UI-kirjaston sisältämät komponentit seuraavat Googlen luomaa Material Design -muotoilua, joka tähtää intuitiivisen käyttökokemuksen luomiseen (Google 2020).

Kuvan 3 lopullinen toteutus seurasi pääosin suunnitelmaa ja täytti asetetut vaatimukset. Material-UI-ohjelmakirjaston komponentit loivat sivulle ammattimaisen näkymän. Väriteemaksi valittiin luottamusta herättävä tummansininen, joka on yksi Material-UI-kirjaston oletusväreistä. Prosessikaavion väri ja muotoilu muutettiin noudattamaan sivun vastaavia ominaisuuksia eksplisiittisesti, sillä kaavion sisältämä SVG-elementti ei peri ominaisuuksia suoraan Material-UI-kirjastolta. Myös tulostukseen liittyviä tyylejä oli muokattava manuaalisesti, koska tulostuneessa PDF-dokumentissa prosessivaiheiden listaus oli katkaistu. Tämä mahdollisesti johtui yhteensopimattomasta komponenttirakenteesta, mutta ongelman syy jäi selvittämättä.



Kuva 3. Kuvakaappaus toteutuneesta käyttöliittymästä.

Tyylimuokkausten jälkeen eri selainten tulosteet saatiin lukukelpoisiksi, mutta komponenttien epäsopivat koot aiheuttivat tulosteeseen liikaa tyhjää tilaa. Tulostuksen lisäksi

käyttäjä voi viedä tietoja valitsemistaan prosessivaiheista JSON-napin avulla, joka avaa sivun sisäisen ikkunan (modal). Ikkunassa näytetään valitut vaiheet sekä niiden väliset yhteydet JSON-muodossa ja napit, joista käyttäjä voi valita tiedot ja kopioida ne leikepöydälle.

Suunniteltujen komponenttien lisäksi käyttöliittymään tehtiin prosessikaavioon liittyviä metatietoja sisältävä komponentti eli alatunniste (footer), joka asetettiin sivun alimmaiseksi. Alatunnisteeseen sijoitettiin verkon aikaleima, SHA256-tiiviste, linkit JSON-tiedostoihin, kuluva vuosinumero ja tekijätiedot.

4.6 Testitapaukset

Toimeksiantaja suoritti käyttöliittymälle kolme testitapausta, joissa kussakin valittiin laitteelle tehtäviä toimenpiteitä vastaavat prosessivaiheet. Testitapauksissa valittujen prosessivaiheiden lista kopioitiin ja tallennettiin suunniteltua laitetestausta varten. Listaus vastaa tietoturvatestauksen suoritusvaiheita, joita testauksen tekijä tai automaatiojärjestelmä seuraa prosessin edetessä. Ensimmäisen testitapauksen kohteena oli teollisuuden ohjainyksikkövalmistajan laite, jolle kartoitettiin teollisuusverkossa käytettävälle laitteelle tunnuksenomaiset haavoittuvuustestit (liite 10). Toisessa tapauksessa valittiin prosessivaiheet turvalaitteiden ohjainyksikkövalmistajan laitteelle, jolle suunniteltiin suoritetuksi junien ovilaitteiston haavoittuvuus- ja tietomurtotestit ISO/IEC 62443-4-1 -standardin mukaisesti (liite 11). Viimeisenä tietoturvatestattavana oli terveyden ja hyvinvointialan mobiilisovellus ja sen pilvirajapinnat, joille valittiin asianmukaisia haavoittuvuustestejä vastaavat prosessivaiheet (liite 12).

Kaikissa testitapauksissa käyttöliittymä toimi onnistuneesti. Toimeksiantajan mukaan toteutuksesta oli testausautomaation suunnittelussa ja jatkokehittämisessä merkittävää hyötyä.

5 JOHTOPÄÄTÖKSET JA POHDINTA

Tutkimuksen tavoitteena oli tarkastella Turun ammattikorkeakoulun mallintamaa prosessikaaviota ja selvittää sen soveltamista web-käyttöliittymässä. Prosessikaavioiden vaihtelevien muotojen ja käyttötarkoituksien vuoksi pyrittiin selvittämään tarvittava tietopohja työssä sovellettavan kaavion tulkitsemiseksi. Kaavion tietorakenne oli tarkoitus tunnistaa, koska tietorakenteen avulla prosessikuvauksen tiedot olisi mahdollista tallentaa verkkosivutekniikoille luettavaan muotoon. PDF-tiedostomuoto, johon prosessikaavio oli tallennettu, ei soveltunut tähän käyttötarkoitukseen, sillä tiedostomuodon tarkoitus on esittää dokumentin viimeistä muotoa. Lopuksi työn tavoitteena oli kehittää web-käyttöliittymä, joka sisältäisi kaavion visualisoituna siten kuin se on esitetty PDF-tulosteessa.

Työn merkittävimmät tulokset liittyivät sovelletun prosessikaavion määrittelyn tarkasteluun, kaavion visualisointiin ja sen edellytysten selvittämiseen. Prosessikaavio tunnistettiin kuvaavan laitetestauksen ydinprosessia, joka koostuu kaaviossa suorakulmioina merkityistä toiminnoista. Lisäksi kaavion kuvaustaso arvioitiin olevan toimintamallin ja prosessin kulun välillä, jossa ydinprosessit jakautuivat osaprosesseiksi ja edelleen toiminnoiksi. Prosessikaavion elementtien tarkastelu ja niiden määrittely edesauttoi kaavion visualisoinnin edellytyksien kartoittamisessa.

Visualisoinnin suunnittelun aikana oli selvää, ettei kaavion sisältävää PDF-dokumenttia sovelleta alkuperäisessä tiedostomuodossa verkkosivulla, vaan tiedot olisi tallennettava koneluettavampaan muotoon. Käytännössä tämä tarkoitti tietojen tallennusta tekstimuotoon. Tietojen tallennuksen edellytyksenä oli kaavion tietorakenteen tunnistaminen, jotta tiedot voidaan syöttää tekstimuodossa järjestetysti. Prosessikaavion tietorakenteeksi havaittiin syklitön suunnattu verkko, jonka visualisointiin käytettiin JavaScript-ohjelmakirjastoa. Kirjaston asettelualgoritmien piirtämiä kaavioita vertailtiin, ja algoritmien todettiin palvelevan erilaisia käyttötarkoituksia. Visualisointia tehdessä huomattiin piirretyn kaavion joustava muokattavuus: asettelualgoritmit tuottivat verkon solmuille ja kaarille pelkästään koordinaattitietoja. Muu visuaalinen ilme oli täysin tekijän hallinnassa. Visualisointi liitettiin lopulta osaksi käyttöliittymää, joka edusti työn lopputuotetta.

Työlle asetetut tavoitteet saavutettiin pääosin kokonaisuudessaan. Käyttöliittymä läpäisi toimeksiantajan suorittamat testitapaukset, mikä tuloksena merkitsee toteutuksen vastaavan toimeksiantajan tarpeita. Toimeksiantajan mukaan käyttöliittymästä oli merkittävää hyötyä testausautomaation kehittämisessä ja jatkokehittämisessä. Työn aikana

selvisi myös, että prosessikaavion tekstimuotoisesta esityksestä olisi laitetestauksen automaatiossa hyötyä, kun automaatioastetta tulevaisuudessa nostetaan. Tietorakenteen selvittäminen toi siis tutkimukselle lisäarvoa.

Tästä huolimatta toteutus jäi kaipaamaan parannuksia. Käyttöliittymän tulostusnäkymästä tuli liian väljä, minkä takia tulosteet eivät sovellu kovin hyvin esimerkiksi työvaiheiden yksinkertaiseen seuraamiseen. Tämän lisäksi voidaan pohtia, oliko CSV-muoto verkon tallennuksessa tarpeellinen, sillä käyttöliittymässä hyödynnettävä tiedostomuoto oli JSON. Saman tiedon erilaiset esitykset aiheuttavat hämmennystä yksittäisessä toteutuksessa. Tutkimuksessa tarkasteltiin myös kovin rajallisesti verkon visualisointia, jonka rooli työssä oli keskeinen. Lisäksi selvitettiin vain yhtä verkkoa visualisoivaa JavaScript-ohjelmakirjastoa, joka olisi kaivannut vertailua muihin saatavilla oleviin kirjastoihin ja niiden tuottamiin kaavion asetteluihin.

Yllä mainitut epäkohdat olisi otettava huomioon käyttöliittymän jatkokehityksessä, jossa toteutusta jalostettaisiin edelleen ja jossa siihen liitettäisiin uusia ominaisuuksia. Toteutusta voisi myös kehittää palvelinkeskeisellä ratkaisulla, jossa käyttöliittymä tarjoillaan käyttäjälle web-palvelimen kautta ja prosessikaavion tiedot säilytettäisiin tietokannassa. Ratkaisu mahdollistaisi esimerkiksi valittujen prosessivaiheiden tallennuksen ja integroinnin muiden järjestelmien kanssa. Työn suunnitteluvaiheessa edellä mainittua ratkaisua harkittiin, mutta ajatuksesta luovuttiin, kun tutkimuksen laajuutta rajoitettiin.

Myös hankkeen prosessikuvausta ja sen hallintaa olisi hyödyllistä arvioida uudelleen ennalta tehtyjen määrittelyiden pohjalta. Kuvaustason selventäminen ja BPMN-määrittelyn noudattaminen mahdollisesti lisäisi kaavion luettavuutta ja käsittelyä. Toisaalta prosessikaavion nykyisellä muodolla on luultavasti haettu prosessiin korkeamman tason näkymää, jonka kuvaamisessa BPMN-määrittelyn käyttö ei välttämättä ole tarpeen. Joka tapauksessa on muistettava, että prosessikuvauksen olisi palveltava sen käyttötarkoitusta.

Työ tehtiin tapaustutkimuksena, jossa selvitettiin toimeksiantajan tarpeita vastaavaa ratkaisua. Tutkimus ei sellaisenaan tuottanut uutta tietoa esimerkiksi verkkotietorakenteiden visualisointiin liittyen, vaan työn tarkoituksena oli esitellä lähestymistapa aihealueeseen. Samalla työ toimi tieteellisenä oppimisprosessina sen tekijälle, joka kehittyi tutkimuksen tekijänä lukuisilla osa-alueilla, kuten erityisesti tutkimuksen suunnittelussa, tieteellisen lähdemateriaalin tulkinnassa sekä tulosten raportoinnissa.

LÄHTEET

Adobe 2006. PDF Reference, sixth edition: Adobe Portable Document Format version 1.7, 23–24.

Bray, T. 2017. The JavaScript Object Notation (JSON) Data Interchange Format. Viitattu 11. marraskuuta 2020 https://tools.ietf.org/html/rfc8259.

Brinkman, E. 2020. d3-dag. Viitattu 15. lokakuuta 2020 https://github.com/erikbrinkman/d3-dag.

Brinkman, E. 2020. Module "arquint/index". Viitattu 19. lokakuuta 2020 https://erikbrinkman.github.io/d3-dag/modules/_arquint_index_.html.

Brinkman, E. 2020. Module "sugiyama/index". Viitattu 19. lokakuuta 2020 https://erikbrinkman.github.io/d3-dag/modules/_sugiyama_index_.html.

Brinkman, E. 2020. Module "zherebko/index". Viitattu 19. lokakuuta 2020 https://erikbrinkman.github.io/d3-dag/modules/_zherebko_index_.html.

Debdutta, P. 2018. Data Structure and Algorithm with C. New Delhi: Alpha Science International, 19–21.

ENISA 2017. Baseline Security Recommendations for IoT, 7.

Facebook Inc. 2020. React - A JavaScript library for building user interfaces. Viitattu 2. marraskuuta 2020 https://reactjs.org/.

Gansner, E.R.; Koutsofios, E. & North, S. 2015. Drawing graphs with dot, 2.

Google 2020. Design - Material Design. Viitattu 2. marraskuuta 2020 https://material.io/design.

Hemmendinger, D. 2016. Machine language. Viitattu 2. marraskuuta 2020 https://www.britannica.com/technology/machine-language.

Julkishallinnon suositus 2012. JHS 152 Prosessien kuvaaminen. Versio 5.10.2012. Viitattu 31. elokuuta 2020 http://docs.jhs-suositukset.fi/jhs-suositukset/JHS152/JHS152.html.

Laaksonen, A. 2020. Tietorakenteet ja algoritmit. Viitattu 31. elokuuta 2020 https://github.com/pllk/tirakirja/raw/master/tirakirja.pdf.

Material-UI 2020. Material-UI: A popular React UI framework. Viitattu 11. marraskuuta 2020 https://material-ui.com/.

Microsoft 2020. TypeScript: Handbook - TypeScript for the New Programmer. Viitattu 16. lokakuuta 2020 https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html.

Mozilla and individual contributors 2020. JSON. Viitattu 16. lokakuuta 2020 https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/JSON.

Mozilla and individual contributors 2020. PDF.js. Viitattu 30. syyskuuta 2020 https://mozilla.github.io/pdf.js/.

Shafranovich, Y. 2005. Common Format and MIME Type for Comma-Separated Values (CSV) Files. Viitattu 8. lokakuuta 2020 https://tools.ietf.org/html/rfc4180.

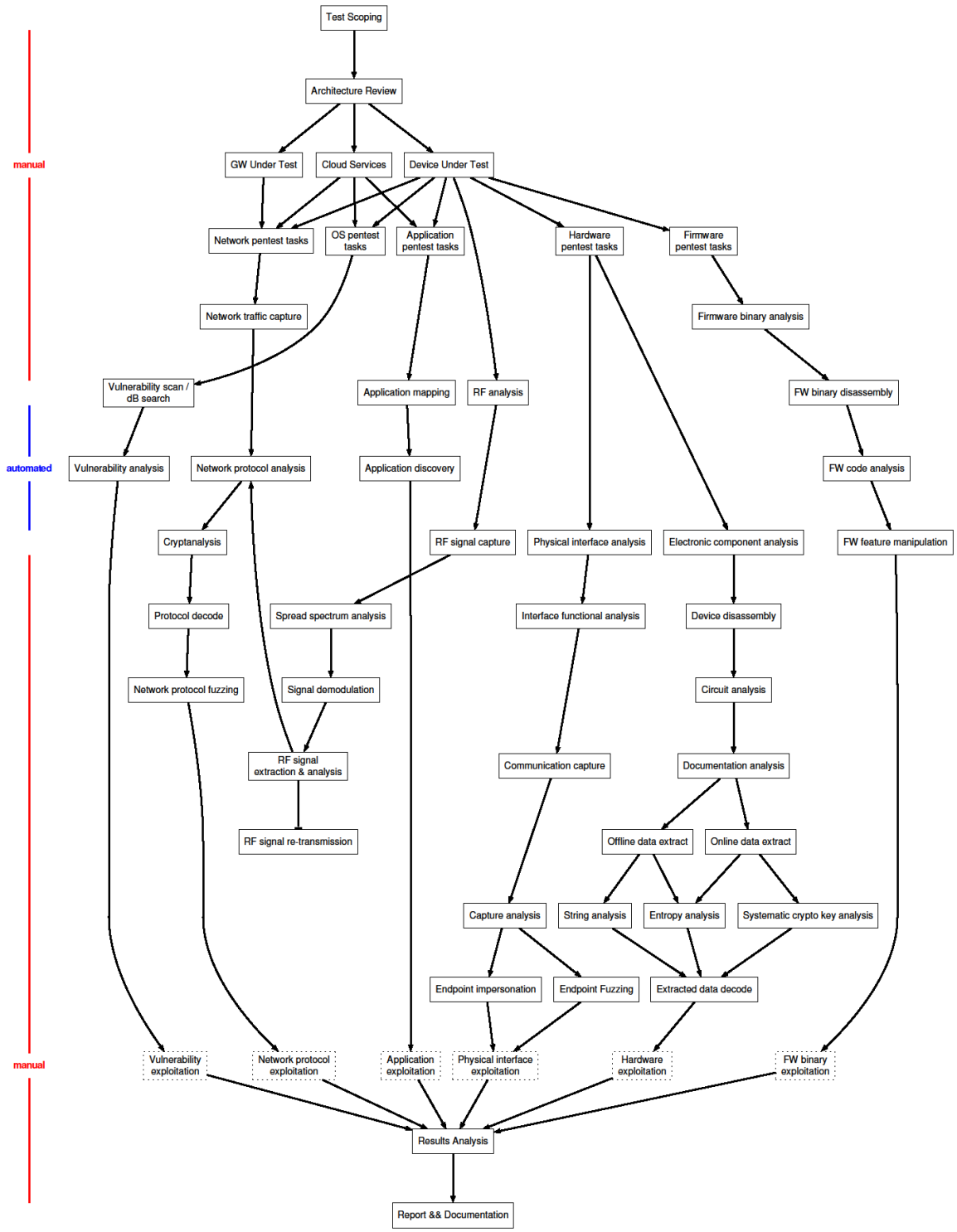
Tietotermit 2018. Finto: TT: tietovirta. Viitattu 2. marraskuuta 2020 https://finto.fi/tt/fi/page/t106.

Turun ammattikorkeakoulu ei pvm. CYBERSECURITY SERVICES. Viitattu 13. marraskuuta 2020 https://kybervaliot.turkuamk.fi/cybersecurity-services/.

W3Docs 2020. How to Embed PDF in HTML. Viitattu 30. syyskuuta 2020 https://www.w3docs.com/snippets/html/how-to-embed-pdf-in-html.html.

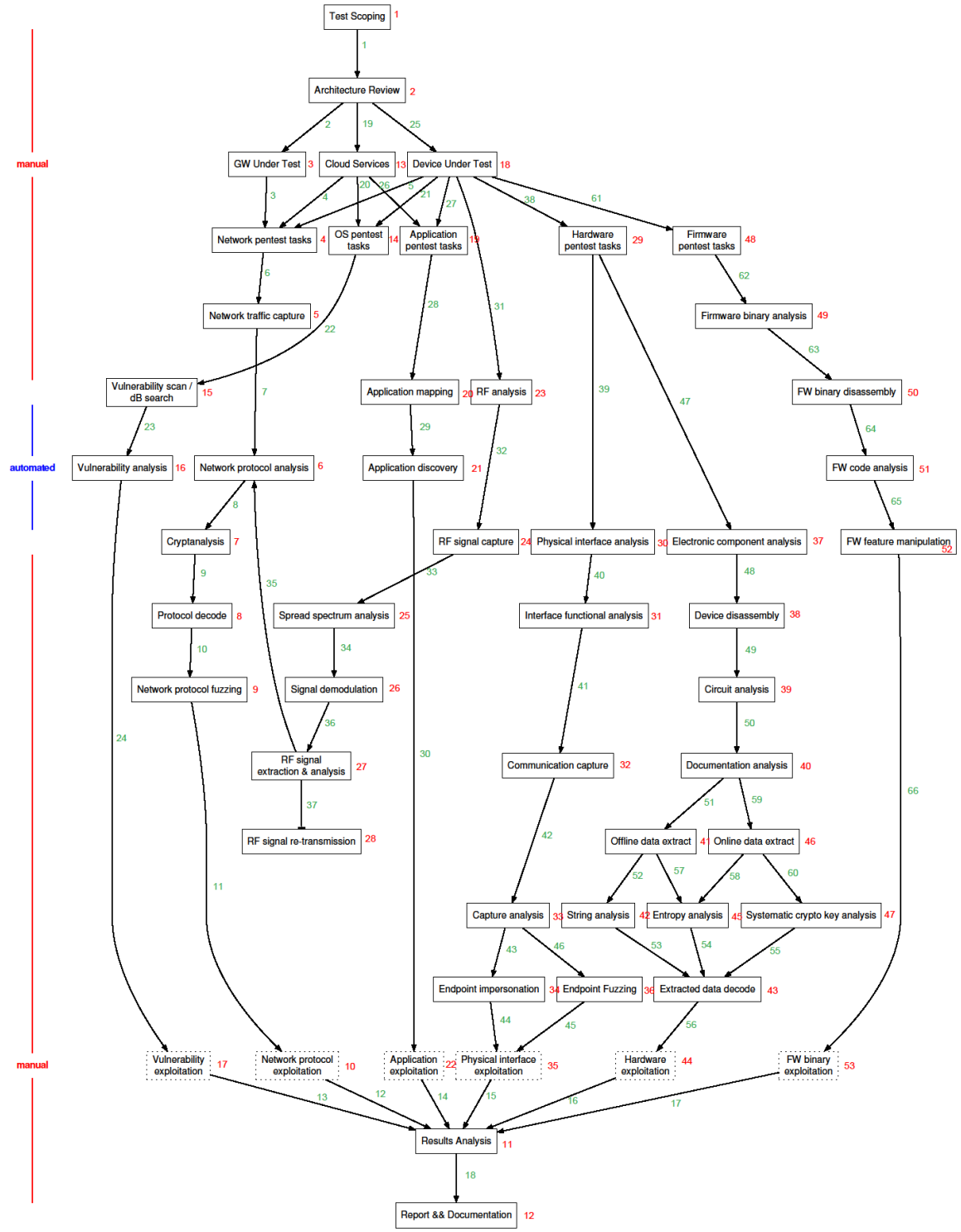
Tekijä: Jani Vanharanta 2019

KyberVALIOT-prosessikaavio



Tekijä: Jani Vanharanta 2019
Muokannut: Ville-Pekka Pietarinen 2020

KyberVALIOT-prosessikaavio verkkona



Verkon JSON-skeema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "title": "Graph used in CyberELITE web interface",
    "type": "object",
    "definitions": {
        "node": {
            "type": "object",
            "properties": {
                "id": { "type": "integer", "minimum": 1 },
                "group": { "type": "string" },
                "label": { "type": "string" }
            }
        },
        "edge": {
            "type": "object",
            "properties": {
                "id": { "type": "integer", "minimum": 1 },
                "source_id": { "type": "integer", "minimum": 1 },
                "target_id": { "type": "integer", "minimum": 1 },
                "label": { "type": "string" }
            }
        }
    },
    "properties": {
        "nodes": {
            "type": "array",
            "items": { "$ref": "#/definitions/node" }
        },
        "edges": {
            "type": "array",
            "items": { "$ref": "#/definitions/edge" }
        }
    }
}
```

Verkko JSON-muodossa


```
{
    "nodes": [
      {
        "id": 1,
        "group": "CyberELITE",
        "label": "Test Scoping"
      },
      {
        "id": 2,
        "group": "CyberELITE",
        "label": "Architecture Review"
      },
      (solmut 3-52 poistettu)
      {
        "id": 53,
        "group": "CyberELITE",
        "label": "FW Binary Exploitation"
      }
    ],
    "edges": [
      {
        "id": 1,
        "source_id": 2,
        "target_id": 1,
        "label": "HAS_PARENT"
      },
      {
        "id": 2,
        "source_id": 3,
        "target_id": 2,
        "label": "HAS_PARENT"
      },
      (kaaret 3-65 poistettu)
      {
        "id": 66,
        "source_id": 53,
        "target_id": 52,
        "label": "HAS_PARENT"
      }
    ]
  }
}
```

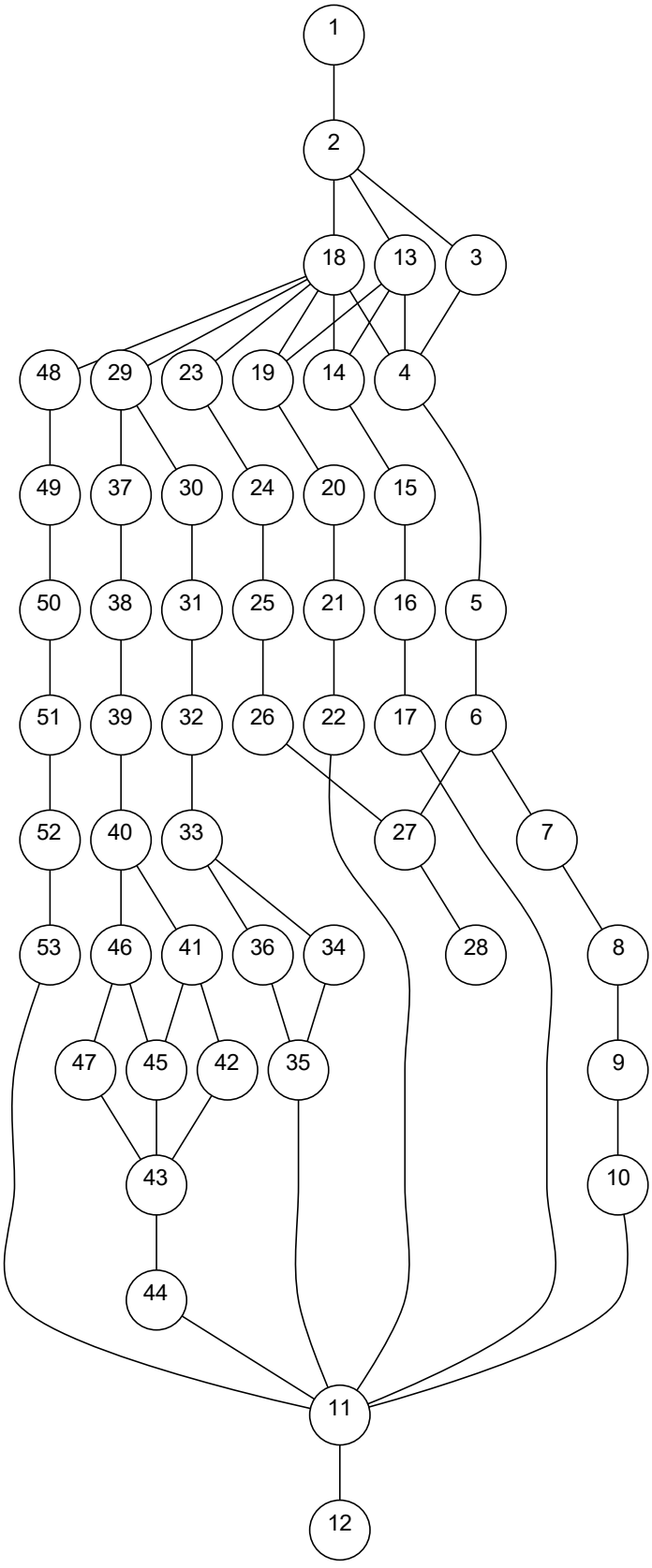
Verkon metatietojen JSON-skeema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "title": "Meta information of a graph used in CyberELITE web interface",
    "type": "object",
    "properties": {
        "timestamp": { "type": "string" },
        "hash": { "type": "string" }
    }
}
```

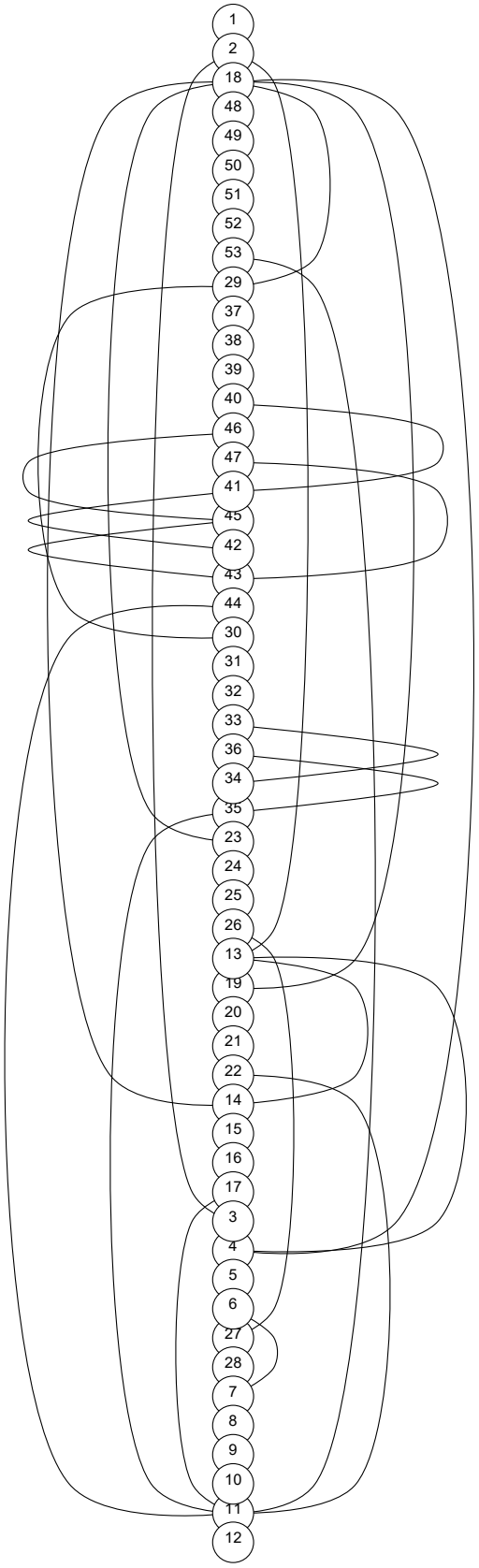
Verkon metatiedot JSON-muodossa


```
{
  "timestamp": "2020-06-12T16:48:40.757Z",
  "hash": "f5c8bc65f188c9bf0c4c9bf8c1691928ca7ae7c4285113466903f26dab21bcf5"
}
```

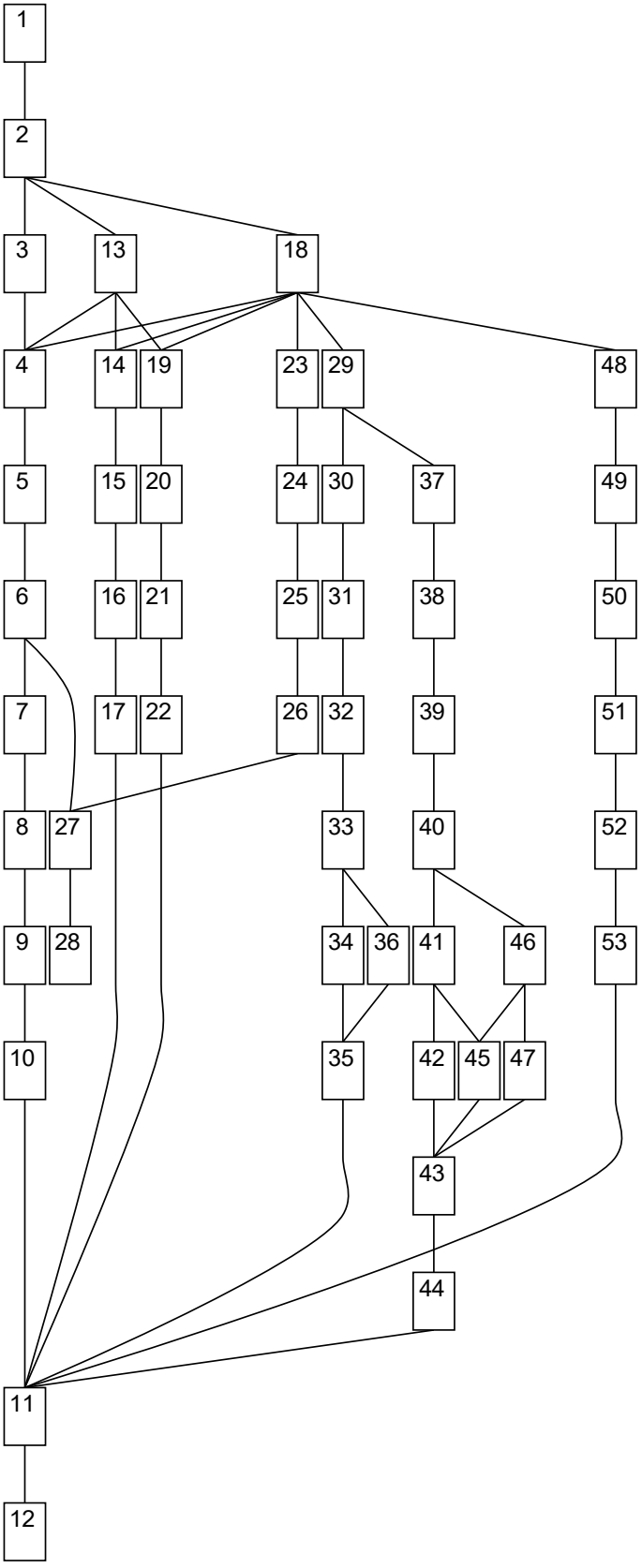
Verkko visualisoituna Sugiyama-algoritmilla



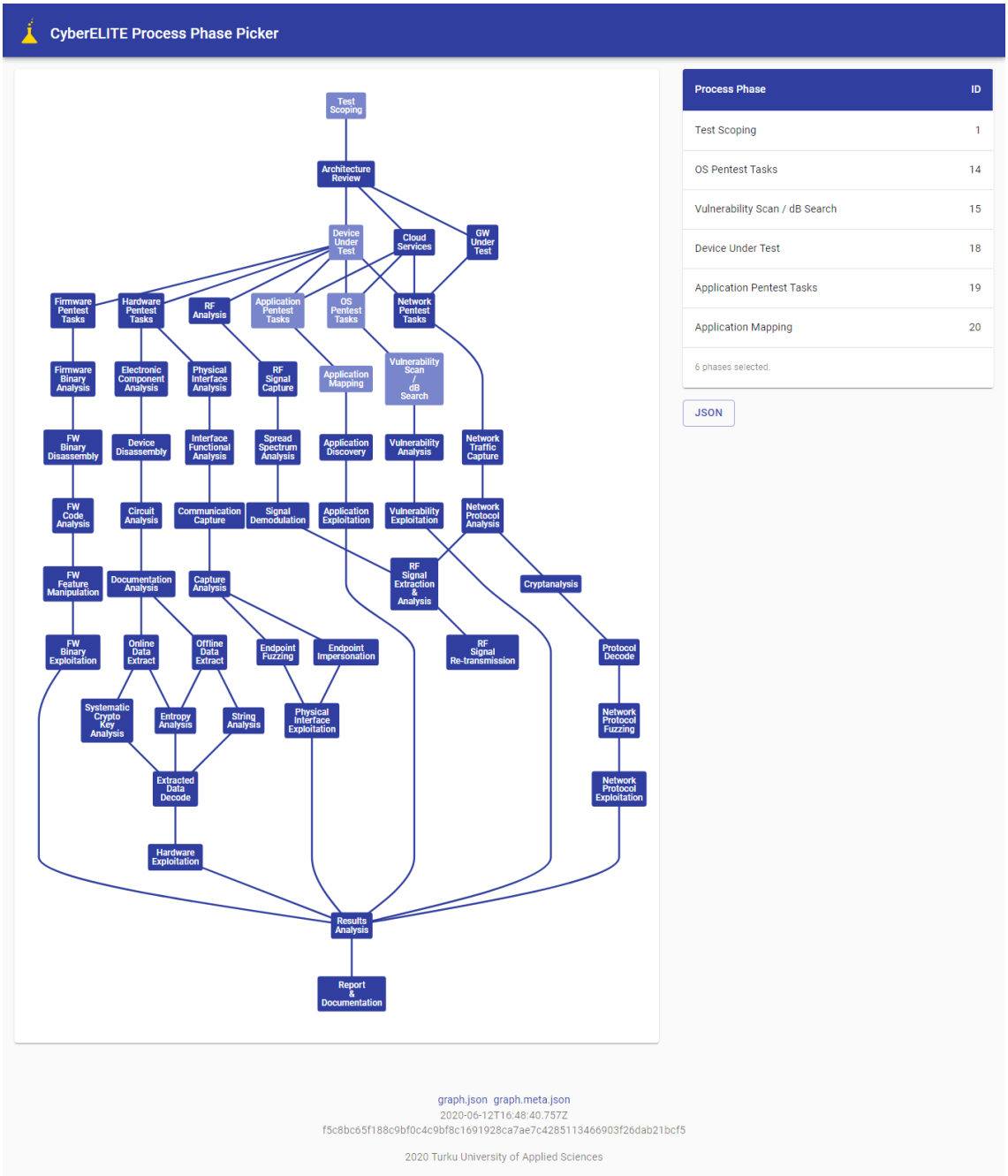
Verkko visualisoituna Zherebko-algoritmilla



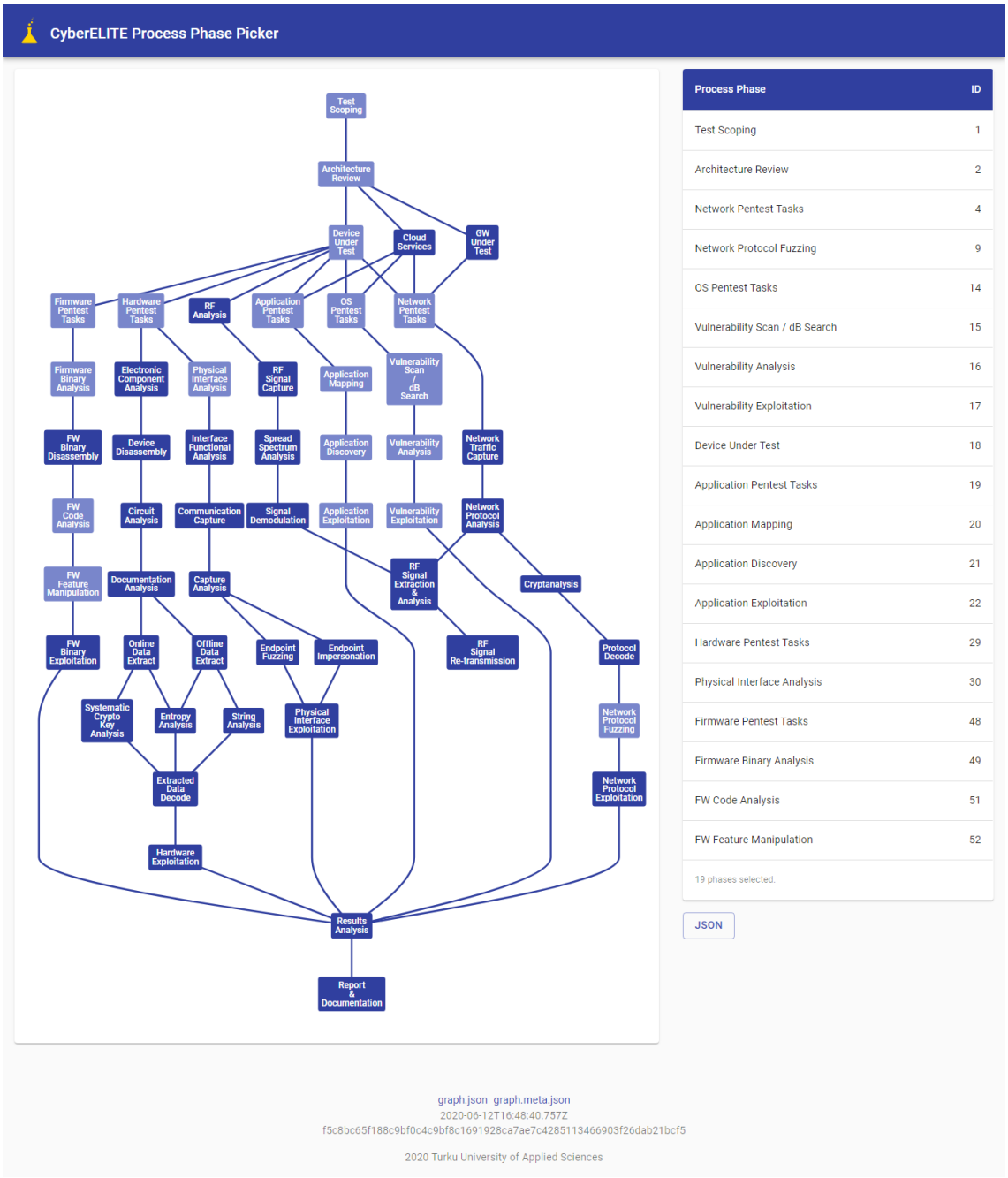
Verkko visualisoituna Arquint-algoritmilla



Testitapaus 1: Teollisuuden ohjainyksikkövalmistajan laite



Testitapaus 2: Turvalaitteiden ohjainyksikkövalmistajan laite



Testitapaus 3: Terveyden ja hyvinvointialan mobiilisovellus ja sen pilvirajapintojen testaaminen