Pekka Sintonen

# INTERNETIN KOODIESIMERKKIEN KÄYTTÖ OHJELMISTOKEHITYKSESSÄ

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2019

# TIIVISTELMÄ

Sintonen, Pekka
Internetin koodiesimerkkien käyttö ohjelmistokehityksessä
Jyväskylä: Jyväskylän yliopisto, 2019, 30 s.
Tietojärjestelmätiede, kandidaatintutkielma
Ohjaaja: Kollanus, Sami

Tässä kandidaatintutkielmassa selvitetään kirjallisuuskatsauksella, minkälaisia etuja ja haittoja internetin koodiesimerkkien käytöstä on ohjelmistotuotannossa. Tutkielmassa käydään läpi koodiesimerkkien merkitys ohjelmointirajapintojen dokumentaatiossa, ohjelmoijille suunnatut internetin sosiaalisen median palvelut, sekä koodiesimerkkien käytön hyödyt ja haitat. Koodiesimerkit ovat lyhyitä lähdekoodilistauksia yhden tai useamman ohjelmointirajapinnan piirteen käytöstä. Ohjelmoijat hakevat ja käyttävät internetistä koodiesimerkkejä usein, ja niiden käyttö voi nopeuttaa ohjelmistokehitystä. Koodiesimerkin integrointi omaan ohjelmaan voi olla kuitenkin työlästä. Lisäksi vääränlainen koodiesimerkki voi johtaa harhaan: ohjelmoija saattaa jättää tutkimatta koodiesimerkin yksityiskohtia, ja käytetty esimerkki voi olla puutteellinen toiminnallisuuden, tietoturvan tai yksityisyyden suhteen. Koodiesimerkin käyttäminen internetistä voi myös johtaa tekijänoikeuslakien rikkomiseen.

Asiasanat: API-rajapinta, dokumentaatio, koodiesimerkki, Stack Overflow

# ABSTRACT

Sintonen, Pekka
The usage of internet code snippets in software development
Jyväskylä: University of Jyväskylä, 2019, 30 pp.
Information Systems, Bachelor's Thesis
Supervisor: Kollanus, Sami

This bachelor's thesis has been conducted as a literature review on benefits and challenges in internet code snippet reuse in software development. The thesis clarifies code snippets' importance in application programming interface documentation, internet's social media services for software engineers as well as positive and negative effects of usage of code snippets. Code snippets are small listings of software source code using one or many application programming interface features. Software developers search and use internet code snippets often, as their use may speedup the development process. Integrating the code snippet to one's own program may take effort. Also, using a wrong kind of snippet may lead astray. The developer may oversee details in the snippet and the snippet may turn out to be inadequate in functionality, security or privacy. Using a snippet from the internet may also lead to a violation of the code snippet license.

Keywords: Application programming interface, documentation, code snippet, Stack Overflow

# TAULUKOT

# SISÄLLYS

# 1 JOHDANTO

Nykyaikaisessa ohjelmistotuotannossa hyödynnetään runsaasti koodiesimerkkejä (engl. ”code example”, ”code snippets”, ”reusable code snippets”). Koodiesimerkki on lyhyt, muutaman rivin listaus lähdekoodia, joka kuvaa esimerkin voimalla jonkin toiminnallisuuden toteutusta ohjelmistoissa. Ohjelmoijat voivat käyttää koodiesimerkkejä työssään eräänlaisina nopeina ja täsmällisinä pohjina toteuttaessaan kyseisiä toiminnallisuuksia. Koodiesimerkit ovat usein hyvin käyttökelpoisia ja nopeuttavat ohjelmoijaa työssään (Acar ym., 2016). Parhaimmillaan koodiesimerkit nivoutuvat tekstimuotoisen dokumentaation osaksi (Nasehi, Sillito, Maurer & Burns, 2012).

Koodiesimerkkejä käytetään ohjelmointirajapintojen (engl. ”Application Programming Interface”, API) käytön yhteydessä. Nykyajan ohjelmistotuotanto perustuu laajalti erilaisten valmiiden toiminnallisuuksien yhdistelyyn, ja miltei jokainen tuotettu koodirivi hyödyntää vähintään yhtä API-rajapintaa (Myers & Stylos, 2016). Käytetyt API-rajapinnat voivat liittyä monenlaisiin toimintoihin, joita voivat olla vaikkapa käyttöliittymän hallinta, erilaiset maksamiseen liittyvät internetin palvelut tai ohjelmointikielen peruskirjastot. Ohjelmoijaa auttaa API-rajapinnan tuottajan julkaisema dokumentaatio, joka kuvaa API-rajapinnan toiminnot. Tärkeä osa tätä dokumentaatiota ovat koodiesimerkit. Itseasiassa ohjelmoijat pitävät koodiesimerkkejä yhtenä dokumentaation tärkeimmistä ominaisuuksista (Haenni, Lungu, Schwarz & Nierstrasz, 2014; Ko, DeLine & Venolia, 2007; Robillard & DeLine, 2011). Koodiesimerkkien käyttö on täten mielenkiintoinen ja merkittävä tietojärjestelmätieteen tutkimusalue.

Internet tarjoaa monenlaisia palveluja dokumentaation ja koodiesimerkkien suhteen. Erilaiset verkkosivustot, blogit ja sosiaalisen median palvelut auttavat ohjelmoijia. Nämä internetin palvelut ovat yleensä joukkoistettu vapaaehtoisille henkilöille, eivätkä ne näin ole osa virallista dokumentaatiota. Etenkin aloittelevat ohjelmoijat arvostavat tätä internetin vertaistukea (Fischer ym., 2017). Merkittävin ja suurin internetin ohjelmoijia palvelevista sivustoista on Stack Overflow, joka on merkittävässä osassa koodiesimerkkejä käsittelevässä kirjallisuudessa. Stack Overflown suosiosta kertoo se, että useassa tutkimuksessa ohjelmoijat ovat ilmaisseet käyttävänsä sitä vähintään yhtä merkittävässä

määrin kuin virallista API-rajapinnan dokumentaatiotakin (Acar ym., 2016; Fischer ym., 2017; Ragkhitwetsagul, Krinke & Oliveto, 2018; Sillito & Begel, 2013; Treude, Barzilay & Storey, 2011; Xia ym., 2017). Stack Overflown käytön on raportoitu nopeuttavan ohjelmoijien työtä toimivan koodin luonnissa (Acar ym., 2016).

Internetin koodiesimerkkien käyttöön liittyy myös haasteita. Vääränlaisen esimerkin käyttäjä altistaa itsensä monenlaisille vaaroille. Toiminnallisuus voi perustua vanhentuneisiin API-kutsuihin (Ragkhitwetsagul ym., 2018) tai altistaa haavoittuvuuksille tietoturvan tai yksityisyyden suhteen (Acar ym., 2016; Fischer ym., 2017). Käytetty koodiesimerkki saattaa myös johtaa tekijänoikeusrikkomukseen (An, Mlouki, Khomh & Antoniol, 2017).

Tämän tutkielman aihealueena on siis internetin koodiesimerkit ja niiden käyttö ohjelmistotuotannossa. Aihetta tutkitaan keskittyen koodiesimerkkejä internetissä tarjoavien palvelujen käytöstä toteutettuihin, pääosin empiirisiin tutkimuksiin. Aihetta lähestytään esimerkkejä käyttävän ohjelmoijan näkökulmasta. Tutkielman tutkimuskysymys on: *Mitkä ovat internetin koodiesimerkkien käytön hyödyt ja haitat?* Tutkimusmenetelmänä on kirjallisuuskatsaus. Menetelmän tarkempi kuvaus on esitetty luvun 3 alussa.

Tutkielman toisessa luvussa esitetään API-rajapinnan dokumentaation ja koodiesimerkkien teoriatausta. Tämän lisäksi esitetään internetin koodiesimerkkipalvelut sekä kuvataan tyypillinen ohjelmointiprosessi internetin koodiesimerkkien suhteen. Kolmas luku pyrkii vastaamaan tutkimuskysymykseen esittämällä kirjallisuuden perusteella koodiesimerkkien käytön hyödyt ja haitat. Tutkielman viimeinen luku kokoaa yhteen aiemmin esitetyt asiat.

# 2 API-DOKUMENTAATIO, KOODIESIMERKIT JA INTERNET

Ohjelmoijat käyttävät erilaisia API-rajapintoja päivittäisessä työssään runsaasti (Myers & Stylos, 2016). Onnistunut API-rajapintojen käyttö edellyttää niiden oikeanlaista kutsumista tavoiteltavan toiminnallisuuden aikaansaamiseksi. API-rajapinnat dokumentoidaan niiden kehittäjien toimesta, mutta nykyaikainen internet tarjoaa myös suorempia dokumentaation muotoja, kuten vaikkapa blogit, tai erinäiset kysymys & vastaus sivustot. Keskeisenä API-dokumentaation sisältönä ovat koodiesimerkit, jotka demonstroivat API kutsujen tekoa.

Tämä luku esittelee API-dokumentaation piirteitä koodiesimerkkeihin liittyen. Ensin käydään läpi API dokumentaation käytettävyyden käsite sekä koodiesimerkkien suhde siihen. Internetin tuomat sosiaalisen median palvelut esitellään tämän jälkeen. Lopuksi kuvataan internetin koodiesimerkkien hakuprosessi.

## 2.1 API-rajapinnan käytettävyys ja dokumentaatio

API-rajapinnalle voidaan esittää käytettävyysodotuksia. API käytettävyys (engl. ”API Usability”) on määritelty hyvin saman tapaisesti kuin perinteisempi tietojärjestelmän loppukäyttäjän kokema käytettävyys. API käytettävyyden keskeisiä osia ovat: opittavuus, tuottavuus, virheiden välttäminen, yksinkertaisuus, johdonmukaisuus ja eri käyttäjien ajatusmallien yhteensopivuus (Myers & Stylos, 2016).

Myersin ja Styloksen (2016) mukaan API käytettävyyden aikaansaamiseksi kehittäjillä ja tutkijoilla on käytössään pääosin samat menetelmät kuin mitä käyttöliittymän käytettävyyden yhteydessä käytettäisiin: API-rajapintaa voi analysoida esimerkiksi Nielsenin heuristisen ohjeistusten (engl. ”Nielsen’s heuristic evaluation guidelines”) mukaisesti tai sille voi tehdä kognitiivisten läpikäynnin (engl. ”cognitive walkthrough”). Erilaisilla ohjelmoijan työkaluilla

on myös merkittävä rooli API käytettävyyden aikaansaamisessa. Esimerkki tällaisesta työkalusta on vaikkapa ohjelmointiympäristön (engl. ”integrated development environment”, IDE) automaattinen täydennys (engl. ”autocomplete”) (Myers & Stylos, 2016).

Keskeisenä API käytettävyyden toteuttamisessa on dokumentaatio. API käyttäjille tehtyjen kyselytutkimusten mukaan dokumentaatio on nostettu kerta toisensa jälkeen keskeisimpien API-rajapinnan oppimisen haasteiksi (Robillard & DeLine, 2011).

Myersin ja Stylosin (2016) mukaan dokumentaation merkitys korostuu tilanteissa, joissa API kehittäjän on tehtävä tietoisia kompromisseja API käytettävyyden suhteen. Esimerkiksi jo käytössä olevaa API-rajapintaa on usein mahdotonta muuttaa rikkomatta kaikkia rajapintaa käyttäviä sovelluksia (Myers & Stylos, 2016).

## 2.2 Ohjelmoija API-rajapinnan ääressä

Ymmärtääksemme API-rajapintojen käyttöä paremmin, on aiheellista tutustua niitä käyttävien ohjelmoijien maailmaan. Ohjelmistokehitys on nopeatempoinen ala, jota kuvaa alati tapahtuva uuden oppiminen.

Ohjelmistokehityksessä tapahtuvaa oppimisprosessia voi lähestyä Knowlesin (1975) itseohjautuvan oppimisen teorian mukaan. Teoria painottaa oppijoiden oma-aloitteellisuutta, muiden ihmisten avulla tai ilman. Teorian mukaisesti oppiminen voidaan jakaa viiteen askeleeseen:

1. Oppimistavoitteiden diagnosointi
2. Oppimistavoitteiden formulointi
3. Oppimismateriaalin identifiointi
4. Parhaan oppimisstrategian valinta ja toteutus
5. Oppimistulosten evaluointi

Itseohjautuva oppiminen on perusteltua, koska oma-aloitteiset (proaktiiviset) oppijat oppivat enemmän ja nopeammin kuin muut. Oma-aloitteellisuus on myös luontainen psykologinen kehityspolku ihmisen itsenäisessä kehittymisessä (Knowles, 1975).

API-rajapinnan käyttäjät ovat siis yleensä aikuisia, jotka itseohjautuvasti optimoivat oppimisprosessejaan. Silliton ja Begelin (2013) haastattelututkimuksessa ohjelmoijat kuvailivat oppimisprosessejaan kohdehakuisiksi. Ohjelmoijien oppimistavoitteet olivat selvästi riippuvaisia niistä ominaisuuksista, joita he olivat rakentamassa. He myös tietoisesti rajoittivat oppimisen minimiin, vain juuri työn alla olevien tehtävien tarpeisiin. Ohjelmoijat eivät edes yrittäneet oppia koko API-rajapintaa. Ongelmien yhteydessä he etsivät usein sopivaa kiertotietä (Sillito & Begel, 2013).

Houn ja Lin (2011) analyysissä Java Swing forumin keskusteluista suurin osa, 82 keskustelua 172:sta, liittyi jonkin tietyn ongelman ratkaisuun. Ohjelmoijia ovat tässäkin varsin kohdehakuisia. Ainoastaan 16 keskustelua liittyi jonkin yleisemmän ominaisuuden ymmärtämiseen (Hou & Li, 2011).

Maalejin, Tiarksin, Roehmin ja Koschken (2014) tutkimuksessa ohjelmoijat ovat myös varsin tarkkoja turhan työn välttäjiä. Ohjelmointityö perustuu lähes aina jonkinlaiseen jo olemassa olevan koodin muokkaamiseen. Ohjelmoijat pyrkivät välttämään koodin ymmärtämistä, kun mahdollista. Koodin ymmärtäminen on työlästä ja ohjelmoijat yleisesti pyrkivät ymmärtämään vain juuri työn alla olevaan tehtävään tarvittavan koodin (Maalej ym., 2014).

Nämä tutkimukset antavat näkökulman ohjelmoijien maailmaan. Ulkopuolisen silmin ohjelmoijat saattavat vaikuttaa jopa laiskoilta, mutta taustalla on tärkeä itseohjautuva oppimisprosessi sekä työlään oppimisen tarkka säännöstely.

## 2.3 Koodiesimerkit

Koodiesimerkit ovat pieniä ohjelman pätkiä, joilla API-rajapinnan toiminnallisuutta dokumentoidaan. Yhdessä koodiesimerkissä voidaan viitata yhteen tai useampaan API-rajapinnan ominaisuuteen, ja miksei useampaan erilliseen API-rajapintaankin. Koodiesimerkit ovat usein tiiviissä yhteydessä perinteisemmän tekstipohjaisen dokumentaation kanssa (Nasehi ym., 2012). Tyypillisesti koodiesimerkit ovat upotettu API-rajapintaa kuvaavaan dokumentaatioon tekstin sekaan. Itsenäisesti esiintyvät koodiesimerkit ovat vähemmän hyödyllisiä.

Koodiesimerkkejä voidaan usein käyttää hyväksi leikkaa-ja-liimaa periaatteella (Fischer ym., 2017). Jos esimerkki on pitkä ja monimutkainen, sitä on kuitenkin usein lyhennetty poistamalla siitä vähemmän olennaisia osia. Tällöin esimerkki on pseudokoodia, jota ei voi sellaisenaan kopioida kohdeohjelmaan ilman muokkausta (Xia ym., 2017).

API-rajapinnan tuottajan näkökulmasta koodiesimerkit ovat hyvä tapa dokumentoida miten ja missä tilanteessa API-rajapinnan toiminnallisuutta on ajateltu kutsuttavan. Ne ovat ikään kuin luonnollinen tekstipohjaisen dokumentaation jatke. Vastaavasti API-rajapinnan käyttäjä saa koodiesimerkeistä usein tarkan kuvauksen, miten API-rajapintaa on ajateltu käyttää tai on käytetty.

Koodiesimerkit ovat erittäin kysyttyjä: Haennin ym. (2014) kyselytutkimuksessa ohjelmoijat ilmaisivat dokumentaation ja koodiesimerkkien merkityksen. 90% vastaajista piti API dokumentaatiota ja 85% vastaajista piti koodiesimerkkejä tärkeänä hyvän API:n merkkinä (Haenni ym., 2014).

Koodiesimerkit tulevat esiin myös aikaisemmassa, Kon ym. (2007) tutkimuksessa. Ohjelmointitehtävissään ohjelmoijat konsultoivat toisia ohjelmoijia, tutkivat dokumentaatiota ja käyttivät koodiesimerkkejä. Tyypillisiä ohjelmointiin liittyviä kysymyksiä ohjelmoijilla oli: Mitkä rakenteet tai funktiot voivat olla käytettävissä tämän ongelman ratkaisuun? Miten tätä rakennetta tai funktiota käytetään? Miten koordinoin muun koodin tämän rakenteen tai funktion käytön yhteydessä (Ko ym., 2007)?

Nasehi ym. (2012) tutkivat Stack Overflowta, joka on suurin ohjelmoijien kysymys & vastaus internetsivusto. He etsivät vastausta kysymykseen: minkälainen on hyvä koodiesimerkki? Vertailemalla kysymysten ja vastausten peu-

kutuksia, sekä oikeaksi valitun vastauksen ominaisuuksia, he esittävät vastauksenaan:

- Tiiviys. Hyväksi havaittu koodiesimerkki on yleensä lyhyt, maksimissaan 4 koodirivin pituinen. Koodia on saatu yksinkertaistettua, usein poistamalla siitä vähempimerkitykselliset osat, eli tekemällä siitä pseudokoodia.
- Kysymyskonteksti. On tärkeää, että esimerkki vastaa juuri siihen kysymykseen, johon vastausta ollaan etsimässä.
- Tärkeän kohdan korostaminen. Esimerkissä voi olla jokin tietty kohta, johon kulminoituu koodiesimerkin ydin.
- Askel-askeleelta ratkaisu. Koodiesimerkki voi olla useampi osainen ja kietoutunut kuvailevan tekstin sekaan, johdattaen lukijan läpi ratkaisun.
- Linkittynyt ulkoisiin lähteisiin. Koodiesimerkki, tai sitä kuvaileva teksti sisältävät linkkejä muihin lähteisiin, jossa aihe on käsitelty tai määritelty (Nasehi ym., 2012).

Koodiesimerkit ovat ilmentymä koodin uudelleenkäytöstä, joskin yleensä pieni sellainen, sillä koodiesimerkin yksi tyypillinen ominaisuus on, että se on lyhyt. Mikäli koodiesimerkki kasvaa suuremmaksi lukuisia toiminnallisuuksia omaavaksi kokonaisuudeksi, voidaan kokonaisuutta katsoa uudelleenkäytön näkökulmasta. Koodin uudelleenkäytössä lähdekoodia voidaan kopioida kolmannen osapuolen ohjelmakirjastosta, toisista avoimen lähdekoodin ohjelmista tai vaikkapa internetin kysymys & vastaus sivustoilta (Abdalkareem, Shihab & Rilling, 2017). Koodin uudelleenkäytön on laajalti hyväksytty olevan keskeinen lähestymistapa laadukkaan ohjelmisto tuottamiseksi nopeasti ja kustannustehokkaasti (Lim, 1994).

## 2.4 Sosiaalinen media dokumentaation tukena

Internetin ja sosiaalisen median mukaantulo on tuonut uuden ulottuvuuden API-rajapinnan dokumentaatioon. Internet mahdollistaa monenlaiset julkaisut: erilaiset web-sivut, blogit ja sosiaalisen median sivustot. Tässä uudessa maailmassa API-rajapinnan tuottaja ei enää vastaa dokumentaatiosta. Sosiaalisen median palveluita hyödyntäen API-dokumentaatio on joukkoistettu käyttäjien itsensä haltuun.

Keskeisin ohjelmoijien sosiaalisen median sivustoista on Stack Overflow. Tieteelliset artikkelit, ja tämä tutkielma, keskittyvät lähes yksinomaan tähän sivustoon. Stack Overflown toinen perustaja, Joel Spolsky (2018) kertoo blogikirjoituksessaan ideoista sivuston takana. Stack Overflow aloitti vuonna 2008 ja nousi nopeasti keskeiseen asemaansa ohjelmoijien keskuudessa. Stack Overflow on kysymys & vastaus sivusto, jossa sosiaalisen median menetelmin vahvistetaan vuorovaikutusta. Kysymyksiä ja vastauksia voi peukuttaa ylöspäin ja alaspäin. Kysyjät ja vastaajat ansaitsevat pisteitä toimillaan. Riittävästi pisteitä

omaavat käyttäjät voivat moderoida keskustelua ja editoida viestiketjuja (Spolsky, 2018). Kysymyksistä voi myös tehdä eräänlaisia miniblogeja vastaamalla itse omaan kysymykseensä. Sekä kysymykset että vastaukset voivat sisältää koodiesimerkkejä. Stack Overflow kokoaa sivuston kysymykset ja vastaukset julkiseen tietovarastoon. Vuoden 2017 joulukuussa tämä ”data dump” sisälsi yli 13 miljoonaa vastattua kysymystä, 8 miljoonan käyttäjän toimesta (Baltes & Diehl, 2018).

Stack Overflown suosio on sitä luokkaa, että se voidaan jopa nähdä virallisen dokumentaation korvaajana, etenkin jos virallinen dokumentaatio ei ole hyvätasoista (Treude ym., 2011). Näkemystä tukee lukuisa joukko tutkimuksia, joissa ohjelmoijat ovat ilmaisseet toimintatapansa tiedonhaun suhteen (Acar ym., 2016; Fischer ym., 2017; Ragkhitwetsagul ym., 2018; Sillito & Begel, 2013; Treude ym., 2011; Xia ym., 2017).

Stack Overflow kuuluu Stack Exchange klusteriin, johon kuuluu monia muiden alojen kysymys & vastaus sivustoja. Stack Exchange sivustoilla on samankaltaisuuksia Wikipedian kanssa.

Toinen merkittävässä määrin tutkittu sosiaalisen median alusta on Github. Github on niin ikään vuonna 2008 perustettu ohjelmistojen lähdekoodin versionhallintaan keskittynyt sivusto. Github on luonteeltaan erilainen, eikä sisällä juurikaan koodiesimerkkejä. Eräänlainen koodiesimerkkien mahdollistaja Githubissa on Gist toiminto. Se ei sisällä juurikaan sosiaalisen median toimintoja, vaan on pikemminkin julkisten ja yksityisten koodipalojen varasto (Wang, Poo-Caamaño, Wilde & German, 2015).

## 2.5 Hakuprosessi

Miten ohjelmoijat hakevat ja miten he löytävät käytettäviä internetin koodiesimerkkejä? Samoin kuin muutkin, eli hakukoneiden avulla. Internetin hakukoneet, kuten Google, Bing ja Baidu ovat nousseet tärkeimmiksi ohjelmoijien käyttämistä hakukoneista (Xia ym., 2017). Ohjelmoijat saattavat käyttää myös Stack Overflown omaa hakutoimintoa tai sitten internetissä olevia erikoistuneita koodihakukoneita.

Ohjelmoijat hakevat koodiesimerkkejä usein. Sadowskin, Stoleen ja Elbaumin (2015) Googlen ohjelmoijille tehdyssä yhdistetyssä kysely- ja käyttäjätutkimuksessa kuvattiin ohjelmoijan arkipäivää koodinhaun suhteen. Ohjelmoijat usein aloittivat käyttäen vain 1-2 hakusanaa, mutta hakutulosten perusteella tarkensivat hakuaan. Myös ohjelmoijalle tutuista aihealueista haetaan koodiesimerkkejä, joskin vähemmän kuin tuntemattomista (Sadowski ym., 2015).

Xia ym. (2017) tutkivat niin ikään yhdistetyssä kysely- ja käyttäjätutkimuksessa ohjelmoijien hakukäyttäytymistä. Koodiesimerkkien suhteen ohjelmoijat esittivät kritiikkiä: hakukoneet poistavat erikoismerkit hakulauseista, jolloin niiden perusteella ei voi hakea. Myös oikeanlaisen hakulauseen muodostaminen nähtiin vaikeana. Tutkijat esittävät tuloksenaan, että ideaalissa ohjelmoijan hakukoneessa olisi seuraavia piirteitä: (1) haku saattaa sisältää ohjel-

mointikielen erikoismerkkejä, (2) mahdollisuus määritellä, että haetaan nimenomaan koodiesimerkkejä, (3) haun integrointi ohjelmointiympäristön ja (4) hakutulosten priorisointi semanttisesti ohjelmoijan kulloiseenkin ympäristöön nähden, esimerkiksi ohjelmointikielen perusteella (Xia ym., 2017).

Shi, Zhong, Xie ja Li (2011) huomauttavat omassa tutkimuksessaan, että eri API versiot voivat olla hyvinkin erilaisia. Haussa tulisi olla mahdollisuus määritellä versio (Shi ym., 2011).

# 3 INTERNETIN KOODIESIMERKKIEN KÄYTÖN HYÖDYT JA HAITAT

Edellisessä luvussa kuvattiin ohjelmoijien toimia API-rajapinnan käyttöön liittyen. Keskeisessä osassa ovat internetin koodiesimerkit, jotka ohjaavat ohjelmoijaa tiellä onnistuneen ohjelmiston luomisessa. Tämä luku keskittyy internetin koodiesimerkkien *käytön seurauksiin*. Luvun tavoitteena on vastata tutkimuskysymykseen: Mitkä ovat internetin koodiesimerkkien käytön hyödyt ja haitat?

Tämän luvun sisältö on haettu käyttäen internetin hakukoneita, ensi sijassa Google Scholaria. Hakukriteereinä on käytetty ”code snippet” ja ”code example”, joihin on lisätty sanoja, kuten ”use”, ”risk”, ”security”, ”license” ja ”Stack Overflow”. Valtaosin haut ovat rajoitettu uusiin, vuodesta 2014 alkaen julkaistuihin töihin.

Hakutulosten analyysi ovat tiivistetty taulukkoon 1. Tässä tiivistyksessä internetin koodiesimerkkien käyttö on jaettu kolmeen pääalueeseen: Kehitysprosessi käsittelee tutkimuksia, jotka liittyvät koodiesimerkkien vaikutukseen ohjelmistokehitysprosessiin. Toiminnallisuus alueen tutkimukset käsittelevät koodiesimerkkien käytön vaikutuksia kehitettävän ohjelmiston toiminnallisuuteen. Tekijänoikeuksien hallinta on suuri koodiesimerkkien käyttöön liittyvä tutkimuskohde ja on näin tunnistettu yhdeksi pääalueeksi.

Kehitysprosessi ja Toiminnallisuus -pääalueet ovat jaettuja pienempiin ominaisuuksiin. Näiden kahden pääalueen sisältö on osin päällekkäistä: esimerkiksi heikkolaatuisten koodiesimerkkien esimerkkikeskeinen käyttö voidaan nähdä liittyvän esimerkkikeskeisyyden, integroinnin, laaturiskien ja tietoturvan ominaisuuksiin.

TAULUKKO 1 Internetin koodiesimerkkien käytön hyödyt ja haitat

| Alue | Ominaisuus | Selite |
|---|---|---|
| Kehitysprosessi | Implementaation nopeus | Internetin koodiesimerkkien käyttö voi nopeuttaa kehitysprosessia (Acar ym., 2016). |
| | Esimerkkikeskeisyys | Ohjelmoijat saattavat keskittyä liikaan sopivien koodiesimerkkien hakuun ja käyttöön (Meng, Steinhardt & Schubert, 2018). |
| | Integrointi | Oikean koodiesimerkin löytäminen ei ole aina helppoa. Esimerkin integrointi omaan ohjelmakoodiin voi olla työlästä (Ragkhitwetsagul ym., 2018; Sillito & Begel, 2013; Xia ym., 2017). |
| Toiminnallisuus | Laaturiski | Internetin koodiesimerkkien käyttö voi lisätä virheiden määrää ohjelmassa. Monet internetin koodiesimerkeistä ovat viallisia ja päivittämättömiä (Abdalkareem ym., 2017; Ragkhitwetsagul ym., 2018; Xia ym., 2017). |
| | Tietoturva ja yksityisyys | Ohjelmoijien tiedot ja taidot eivät yleensä kehity riittävästi tietoturvan ja yksityisyyden suhteen. Monet internetin koodiesimerkeistä ovat turvattomia. Näitä esimerkkejä on myös käytetty laajalti (Acar ym., 2016; Fischer ym., 2017) |
| Tekijänoikeudet | Lisenssiehdot | Internetin koodiesimerkkien käyttö voi johtaa tekijänoikeusrikkomukseen (An ym., 2017; Baltes, Kiefer & Diehl, 2017; Ragkhitwetsagul ym., 2018; Romansky, Chen, Malhotra & Hindle, 2018; Sojer & Henkel, 2011). |

## 3.1 Kehitysprosessi

Ohjelmistokehitysprosessissa tuotetaan uusia ominaisuuksia kehitettävään ohjelmistoon. Tämä luku läpikäy koodiesimerkkien käytön vaikutusta ohjelmistokehitysprosessiin. Keskeisiä kysymyksiä koodiesimerkkien käytön suhteen ovat: nopeutuuko (tai hidastuuko) kehitysprosessi ja muuttaako koodiesimerkkien käyttö tyypillistä kehitysprosessia.

Tutkimustulokset ovat tässä jaettu kolmeen ryhmään: tehokkuuteen, esimerkkikeskeisyyteen sekä koodiesimerkkien integroinnin haasteisiin. Tulosten mukaan internetin koodiesimerkkien käyttö voi nopeuttaa kehitysprosessia, mutta esimerkkikeskeisyys ja integraatio ovat ongelmallisia.

### 3.1.1 Nopea implementaatio

Acar ym. (2016) tutkivat Stack Overflown käyttöä ohjelmointitehtävien suorittamisessa Android-ympäristössä. Tutkimuksessa etupäässä aloitteleville ohjelmoijille annettiin ohjelmointitehtäviä, jotka tulee suorittaa aikarajoitetusti. Tutkimus vahvistaa monia intuitiivisesti perusteltuja näkemyksiä internetin koodiesimerkkien ja ohjelmistokehityksen luonteista:

- Ohjelmoijat käyttävät sekä virallista (tutkimuksessa Android) dokumentaatiota, sekä Stack Overflowta. Vain harva ohjelmoija tutkii vapaaehtoisesti kirjoja. Ohjelmoijat näkevät virallisen dokumentaation ja kirjat kuitenkin parempilaatuisina lähteinä. Ohjelmoijat halusivat luontaisesti vertailla eri lähteitä toisiinsa. Mikäli ohjelmoija oli rajoitettu jonkin tietyn lähteen käyttäjäksi, hän oli epävarmempi löydöstensä laadun suhteen.
- Mikäli ohjelmoija oli rajoitettu ainoastaan virallisen Android dokumentaation käyttöön, hänen tuottamansa koodi oli heikompilaatuista toiminnallisuuksien suhteen. Tämä osoittaa, että tiedonhaku Stack Overflowsta tehostaa toiminnallisuuksien nopeaa toteuttamista.
- Tehtävän aikarajan lähestyessä, ohjelmoijat usein kopioivat Stack Overflown koodiesimerkkejä suoraan omiin ohjelmiinsa (Acar ym., 2016).

Edellä mainittu Acarin ym. (2016) tutkimus käsittelee dokumentaatiota kokonaisuutena, ei ainoastaan koodiesimerkkejä. Tutkituista Stack Overflown viestiketjuista kuitenkin noin puolet sisälsivät koodiesimerkkejä, joten koodiesimerkkien voidaan nähdä olevan keskeisessä osassa tutkimustulosta.

### 3.1.2 Esimerkkikeskeisyys

Aiemmin luvussa 2 läpikäytiin Stack Overflown suosiota ohjelmoijien sosiaalisena mediana. Tämä suosio vaikuttaa selvästi ohjelmointiprosessiin. Tiedonhaun helppous johtaa eräänlaiseen lyhytjännitteisyyteen, jossa ohjelmoijat eivät edes yritä ymmärtää työn alla olevan koodin täydellistä toimintaa. Viitteitä täl-

laisesta mahdollisesta lyhytjännitteisyydestä saatiin jo Silliton ja Begelin (2013) sekä Maalejin, Tiarksin, Roehmin ja Koschken (2014) tutkimuksissa, joissa ohjelmoijat toimivat erittäin kohdehakuisesti, keskittyen tiukasti työn alla oleviin ominaisuuksiin välttäen ohjelman syvällisempää ymmärtämistä. Myös Kon, Delinen ja Venolian (2007) tutkimuksessa sosiaalinen tiedonhaku oli keskeisessä osassa ohjelmointitehtävien suorituksessa.

Mengin, Steinhardtin ja Schubertin (2018) haastattelututkimuksessa ohjelmoijat kertoivat tavoistaan opetella uuden API-rajapinnan käyttö. Tyypillinen toiminta aivan alussa oli yleiskuvan luonti valitun API-rajapinnan toiminnallisuudesta. Tämän yleiskuvan jälkeen ohjelmoijat jakautuivat kahteen leiriin: he joko haluavat saada nopeasti aikaiseksi toiminnallisuuksia tai he halusivat oppia tarkemmin API-rajapinnan toimintaa. Ensin mainitut koodiorientoituneet ohjelmoijat olivat juuri niitä, joiden fokusoituminen oli kohdehakuista. He etsivät sopivan koodiesimerkin ja heti sellaisen löydyttyä, he lopettivat API-rajapinnan opettelun, riippumatta API-rajapinnan monimutkaisuudesta (Meng ym., 2018).

Edellä kuvattu esimerkkikeskeisyys on tyypillinen ohjelmoijan piirre. Kirjallisuudessa käytetty termi sille on esimerkkikeskeinen ohjelmointi (engl. ”example oriented programming” tai ”example centric programming”). Barzilay ym. (2013) käyttivät kyseistä termiä kuvatessaan Stack Overflown toimintaa.

Mikäli esimerkkikeskeisyys ryöstäytyy käsistä, voi lopputuloksena olla karu esimerkkisillisalaatti. Bloggari Christian Heilmann (2015) käytti tällaisia ratkaisuja tarjoavista ohjelmoijista leikillistä termiä ”Full stackoverflow developer”. Kyseiset ohjelmoijat rakentavat ohjelmansa lähes kokonaan koodiesimerkkien voimin. He suuntaavat suoraan internetin sivustoille ja odottavat löytävänsä koodia, jolla ratkaista ongelmansa. He eivät halua lainkaan perehtyä taustoittavaan perustyöhön. Ja yllättävän usein he myös onnistuvat tässä. (Heilmann, 2015).

### 3.1.3 Integrointi

Internetistä löydettyjä koodiesimerkkejä joudutaan usein muokkaamaan, jotta niitä voidaan käyttää ohjelmiston kehityksessä. Anekdoottinen tieto kertoo, että internet on täynnä toimimattomia koodiesimerkkejä (Heilmann, 2015). Tätä näkemystä tukevat useat tutkimukset, joissa koodiesimerkkien integrointi kehitettävään ohjelmaan nähdään ongelmallisena prosessina.

Xian ym. (2017) haastattelututkimuksen tuloksissa ohjelmoijat kertovat, että internetin koodiesimerkit eivät ole helposti uudelleenkäytettäviä. Usein löytynyt esimerkki ei ole käyttökelpoinen: se saattaa olla heikosti toteutettu, heikosti ylläpidettävissä tai laajennettavissa (Xia ym., 2017).

Silliton ja Begelin (2013) haastattelututkimuksen tuloksissa internetin koodiesimerkkien käyttö nähtiin työläänä. Jotkin löydetyistä esimerkeistä eivät toimineet. Useamman eri esimerkin vertailu oli aikaa vievää, sen oikean löytä-

minen ei ollut helppoa. Tämä integraatioprosessi nähtiin tosin myös hyvänä askeleena APIn käytön oppimisen suhteen (Sillito & Begel, 2013).

Ragkhitwetsagul ym. (2018) kyselytutkimuksessaan raportoivat, että peräti 70% koodiesimerkkien käyttäjistä näkivät ongelmana koodiesimerkkien kohtaannon. Stack Overflowssa esitetty koodiesimerkki vastasi esitettyyn kysymykseen, mutta se ei kuitenkaan ole juuri oikea ratkaisu koodiesimerkin käyttäjän ongelmaan (Ragkhitwetsagul ym., 2018).

## 3.2 Toiminnallisuus

Kun oikea koodiesimerkki on löytynyt ja sen avulla on ratkaistu jokin ohjelmointiongelma, on myös vaarana, että käytetty koodiesimerkki johtaa ongelmiin ohjelman toiminnallisuuden suhteen. Toiminnallisuuteen liittyvät riskit ovat tässä luvussa jaettu toiminnallisuuteen sekä tietoturvaan ja yksityisyyteen liittyviin riskeihin, perustuen koodiesimerkkien käytön seurauksiin Acarin ym. (2016) tutkimuksen tulosten mukaisesti.

Koodiesimerkissä voi olla jokin vika tai puutteellisuus. Jos koodiesimerkkiä käyttävä ohjelmoija ei sitä havaitse, voi seurauksena olla tämän toiminnallisen ongelman kopioituminen työn alla olevaan ohjelmaan (Ragkhitwetsagul ym., 2018). Toiminnallinen ongelma voi myös liittyä tietoturvaan ja yksityisyyden hallintaan. Koodiesimerkit voivat myös vanhentua, jos jokin käytetty API-rajapinta muuttuu (Ragkhitwetsagul ym., 2018). Etenkin tietoturvaan ja yksityisyyteen liittyvät koodiesimerkit voivat myös vanhentua, jos jokin ympäristön oletus muuttuu. Esimerkiksi, ennen turvalliseksi luokiteltu salausalgoritmi saatetaan myöhemmin luokitella turvattomaksi, jolloin sitä käyttävät koodiesimerkit myös muuttuvat turvattomiksi (Fischer ym., 2017).

Viallinen koodiesimerkki tulisi aina korjata, jotta vältytään vian leviämiseltä uusiin ohjelmiin (Ragkhitwetsagul ym., 2018). Koodiesimerkki voidaan myös kommentoida sen sisältämän vian suhteen (Fischer ym., 2017).

Edellisessä luvussa läpikäyty ohjelmoijien tendenssi etsiä ja käyttää koodiesimerkkejä ongelmakentän moniulotteisen ymmärtämisen sijasta saattaa myös vaikuttaa kopioinnista johtuviin ongelmiin. Koodiesimerkin käyttäjät usein kopioivat koodin suoraan omaan ohjelmaansa ilman riittävää kritiikkiä (Maalej ym., 2014; Sillito & Begel, 2013). Esimerkkikeskeisen ohjelmoinnin voidaan täten olettaa vaikuttavan myös kehitettävän ohjelman toiminnallisuuteen. Viitteitä tästä antaa Acarin ym. (2016) tutkimus, jossa Stack Overflown käyttö johti funktionaalisesti toimivampaan, mutta tietoturvan suhteen heikompaan ohjelmaan.

Kirjallisuudessa esitetyt tutkimustulokset antavat viiteitä ongelmien laajuudesta. Tutkimusten perusteella koodiesimerkeistä löytyy virheitä ja tietoturvahaavoittuvuuksia. Koodiesimerkkien julkaisijat eivät useinkaan päivitä esimerkkejään. Myös koodiesimerkkien käyttäjät jättävät varoitukset usein huomiotta. Koodiesimerkkien käytöllä voidaan täten nähdä olevan mahdollisesti merkittävä ohjelmiston laatua alentava vaikutus.

### 3.2.1 Toiminnallisuusriski

Abdalkareem, Shihab ja Rilling (2017) tutkivat avoimen lähdekoodin Android sovelluksia ja niistä löytyviä Stack Overflown koodiesimerkkejä. He käyttivät sovellusten lähdekoodin versionhallintaa ja sinne kirjattuja kommittitekstejä tutkiakseen milloin koodiesimerkkiä oli käytetty ja mitä muutoksia samaan kooditiedostoon oli tehty jälkeenpäin. Jos kooditiedoston kommitteja oli kommentoitu sanoilla ”fix, ”bug”, ”defect”, ”patch” tai ”error”, katsoivat tutkijat kyseessä olevan bugikorjauksen. Tutkijat laskivat bugikorjausten yleisyyden ennen koodiesimerkin käyttöä ja sen jälkeen. Tällä tavoin määriteltyjen bugikorjausten yleisyys yli kolminkertaistui koodiesimerkin käytön jälkeen. Tutkijat esittävät alustavana tutkimuksenaan koodiesimerkkien käytön mahdollisena seurauksena bugien lisääntymisen (Abdalkareem ym., 2017).

Edellä kuvatun Abdalkareemin ym. (2017) lähdekoodiin perustuvan tutkimuksen lisäksi toiminnallisuusriskiä kuvaa Ragkhitwetsagulin ym. (2018) kyselytutkimus. Kyseinen tutkimus oli suunnattu Stack Overflown käyttäjille. Kyselyyn vastaajista 49% olivat kohdanneet viallisen koodiesimerkin (Ragkhitwetsagul ym., 2018).

Samansuuntaisesti Xian ym. (2017) tutkimuksessa ohjelmoijat valittivat internetin koodiesimerkkien heikkoa laatua. Useat koodiesimerkit eivät ole uudelleenkäytettävissä tai ovat huonosti koodattuja, vaikeasti ylläpidettäviä ja vaikeasti laajennettavissa. Koodiesimerkkeihin voi olla myös vaikeaa luottaa ja itse koodaamalla saisi helpommin (Xia ym., 2017).

Koodiesimerkkien vanhentuminen on myös suuri ongelma. Koodiesimerkin käyttämä API saattaa muuttua tai jokin muu muutos saattaa tehdä esimerkistä vanhentuneen. Ragkhitwetsagul ym. (2018) tekivät kyselytutkimuksen Stack Overflown aktiivisimpien kysymyksiin vastaavien henkilöiden näkemyksistä. Kyselyyn vastanneet ovat yleisesti tietoisia esimerkkiensä vanhentumisesta, mutta 19% vastaajista jättää päivittämättä julkaisemiaan esimerkkejä, vaikka he olisivat saaneet ilmoituksen niiden vanhentumisesta. Vastaavasti koodiesimerkkien käyttäjistä 68% kertoivat törmänneensä vanhentuneeseen koodiesimerkkiin (Ragkhitwetsagul ym., 2018).

### 3.2.2 Tietoturva ja yksityisyys

Tietoturva ja yksityisyys ovat merkittäviä tekijöitä kaikessa ohjelmistokehityksessä. Valitettavasti erilaista haavoittuvuuksista ja hakkeroinneista kertovat uutiset ovat arkipäivää.

Balebakon, Marshin, Lin, Hongin ja Cranorin (2014) pienissä ohjelmistoalan yrityksissä työskenteleville ohjelmoijille suunnattu kysely- ja haastattelututkimus osoittaa tietoturvan monimutkaisen ja ristiriitaisen prosessin. Monen vastaajan mukaan tietoturva ja yksityisyys ovat osa kehitysprosessia, mutta eivät korkealla prioriteetilla. Applikaation tuotto ja muut ohjelmointitehtävät nähdään tärkeämpinä (Balebako ym., 2014). Tämä tutkimustulos osoittaa alan

yleisen priorisoinnin: toiminnallisuus on usein tärkeämpää kuin tietoturva ja yksityisyys.

Taustoittavassa kyselytutkimuksessaan Acar ym. (2016) esittävät tuloksenaan, että Android ohjelmoijien tiedot ja taidot eivät usein kehity riittävästi tietoturvan suhteen. Tietoturva-asiat tulevat esiin ohjelmointityössä, mutta kuitenkin siinä määrin harvakseltaan, että asiantuntijuutta ei ehdi syntyä. Vastauksia tietoturvaan liittyviin kysymyksiin ohjelmoijat etsivät etupäässä Stack Overflowsta, sekä yleisemmin hakukoneita käyttämällä. Virallista (Android) dokumentaatiota käytetään tietolähteenä huomattavasti vähemmän. Ainoastaan Android spesifisten käyttäjäoikeuksien osalta ohjelmoijat käyttivät virallista dokumentaatiota ensisijaisena lähteenään. Tällaisessa ympäristössä oikeanlainen dokumentaatio, ja etenkin Stack Overflown esimerkit, ovat erityisen tärkeitä (Acar ym., 2016).

Käyttäytymistutkimuksessaan Acar ym. (2016) huomioivat, että mikäli ohjelmoijilla oli pääsy Stack Overflown lähteisiin, oli lopputuloksena funktionaalisesti toimivampi, mutta tietoturvan suhteen heikompi ohjelma. Tutkijat kävivät läpi testiryhmän käyttämät Stack Overflown viestiketjut. Puolet relevanteista viestiketjuista sisälsivät koodiesimerkkejä. Näistä esimerkeistä peräti puolet sisälsivät pelkästään turvattomiksi luokiteltavia esimerkkejä. Turvattomat esimerkit oli kommentoitu ainoastaan 30% tapauksista (Acar ym., 2016).

Acarin ym. (2016) tutkimus osoittaa koodiesimerkkien käytön tietoturvariskin reaalimaailmassa. Ohjelmoijat usein kopioivat esimerkeistä turvatonta koodia ohjelmiinsa kyseenalaistamatta lähteitään. Tutkimuksessa ohjelmoijan kokeneisuus korreloi toiminnallisuuden suhteen, kokeneemmat ohjelmoijat onnistuivat siis luomaan useammin toimivaa koodia. Tietoturvan suhteen ohjelmoijan kokeneisuudella ei kuitenkaan ollut merkitystä (Acar ym., 2016).

Fischer ym. (2017) tutkivat niin ikään Android sovelluksia ja niiden sisältämiä Stack Overflown koodiesimerkkejä. He analysoivat yli 4000 Stack Overflown Android tietoturvaan liittyviä koodiesimerkkejä käyttäen koneoppimisen menetelmiä. Koodiesimerkeistä 1161 luokiteltiin turvattomiksi. Tämän jälkeen heidän analysointijärjestelmänsä analysoi yli 1,3 miljoonaa Android applikaatiota Google Play -kaupasta. Yli 14% näistä applikaatioista sisälsi vähintään yhden turvattomaksi luokitellun Stack Overflown koodiesimerkin (Fischer ym., 2017).

Fischerin ym. (2017) tutkimus sisälsi vielä yhden mielenkiintoisen tiedon: koodiesimerkki saattoi sisältää varoituksen sen sisältävästä tietoturvariskistä. Tämä varoitus ei kuitenkaan estänyt ohjelmoijia käyttämästä kyseistä esimerkkiä. Päinvastoin, varoituksia sisältäviä esimerkkejä oli käytetty jopa muita enemmän (Fischer ym., 2017).

## 3.3 Tekijänoikeudet

Koodiesimerkkejä käytetään usein leikkaa-ja-liimaa periaatteella, kulloiseenkin kohdeohjelmaan mukauttaen. Syyllistyykö ohjelmoija tekijänoikeusrikkomuk-

seen näin tehdessään? Tutkimusten perusteella näin usein tapahtuu, koodiesimerkkien käytön yleisyydestä ja suurimman verkkopalvelun, Stack Overflown, lisensoinnista johtuen.

Internetin koodiesimerkit, kuten myös avoimen lähdekoodin ohjelmistokirjastot ovat vapaasti saatavilla ja niiden käyttäjä ei erityisesti tee eikä hyväksy minkäänlaisia lisenssiehtoja (Sojer & Henkel, 2011). Koodiesimerkit eivät kuitenkaan ole tekijänoikeudettomia. Lähtökohtaisesti niihin pätee samat tekijänoikeuslait kuin kaikkeen muihinkin ohjelmistoihin.

Toteutuuko tämä tekijänoikeus reaalimaailmassa? Useamman empiirisen tutkimuksen mukaan ainakin Stack Overflown koodiesimerkkejä käytetään runsaasti, eikä tekijänoikeuksia pääsääntöisesti huomioida. Internetin hakukoneet eivät tunnu löytävän tapauksia, jossa koodiesimerkkien käyttö olisi johtanut syytetyn penkille. Joitain viitteitä saattanee antaa Ragkhitwetsagul ym. (2018) joiden tuloksissa 9% koodiesimerkkien käyttäjistä ilmoittivat kohdanneensa laillisuusongelmia (engl. ”legal issues”) esimerkkien käytön suhteen. Koska kyseessä oli anonyymi tutkimus, eivät tutkijat pystyneet varmentamaan vastauksia mitenkään (Ragkhitwetsagul ym., 2018).

Tutkimusten yleinen tulos on, että koodiesimerkkien käyttöön liittyy potentiaalisesti tekijänoikeuteen liittyviä riskejä. Ohjelmoijien ymmärrys tekijäoikeuksien suhteen on usein vajavainen (An ym., 2017; Baltes ym., 2017; Ragkhitwetsagul ym., 2018; Romansky ym., 2018; Sojer & Henkel, 2011).

### 3.3.1 Tekijänoikeuksien piiriin kuuluminen

Koodiesimerkkien pituus vaihtelee lyhyestä, yhden tai kahden rivin esimerkistä huomattavasti pidempiin, kymmenien rivien esimerkkeihin. Arnoud Engelfriet (2016) esittää blogikirjoituksessaan näkemyksensä koodiesimerkkien tekijänoikeuksien rajoitteista. Erityisesti pienissä esimerkeissä nousee esiin kysymys, onko työ tarpeeksi luova, jolloin se voisi ylittää teoskynnyksen ja näin kuulua tekijänoikeuden piiriin. Mikäli kaksi riippumatonta ohjelmoijaa päätyisivät oleellisesti samaan ongelman ratkaisuun, ei ratkaisu ole luova, eikä näin ollen teoskynnys ylity. Koodiesimerkin pituuden kasvaessa ohjelmoijan tekemien valintojen määrä kasvaa ja työ voidaan nähdä luovana. Yleisesti alle 10-rivinen esimerkki nähdään ei-luovana, mutta maailmassa ei liene mitään lainsäädäntöä, joka asettaisi rajan mihinkään tiettyyn rivimäärään. Yleisesti, mikä tahansa esimerkki, joka on pidempi kuin 1 tai 2 riviä standardi API kutsuja voitaisiin nähdä myös luovana (Engelfriet, 2016). Tämä Engelfrietin kirjoitus on nähtävä pelkästään asiantuntijan mielipiteenä, jo pelkästään siitäkin syystä, että eri maiden lainsäädäntö määrittelee luovuuden eri lähtökohdista (”Creative Commons FAQ”, 2018a).

Erittäin lyhyiden koodiesimerkkien käyttöä voidaan myös ajatella puolustettavan kohtuullisen käytön periaatteen (engl. ”fair-use”) perusteella (Ragkhitwetsagul ym., 2018). Alkuperäinen Stack Overflown kehittäjä, Jeff Atwood (2009) ilmaisi vastaavan näkemyksensä blogikirjoituksessaan. Atwoodin mukaan “koodiesimerkki kuuluu katkelman kategoriaan ja tulisi täten olla va-

paasti käytettävissä kohtuullisen käytön periaatteiden mukaisesti". ("Stack Exchange Meta", 2009). Ainakaan yhdysvaltalaisen kohtuullisen käytön periaatteen ei kuitenkaan yleisesti katsota pätevän ohjelmistojen lähdekoodiin (Engelfriet, 2016).

Baltesin ja Diehlin (2018) tutkimuksessa keskimääräinen Stack Overflown koodiesimerkin pituus oli 12 riviä, tai 455 merkkiä joulukuussa 2017. Valtaosa koodiesimerkeistä voidaan siis katsoa kuuluvan tekijänoikeuksien piiriin.

### 3.3.2 Stack Overflown lisensointi

Suurimman julkisen koodiesimerkkejä tarjoavan verkkopalvelun, Stack Overflown, koodiesimerkkien lisensointi luo merkittävän ongelmakentän niiden käyttäjille. Stack Overflown käyttöehdoissa todetaan, että koodiesimerkkien kirjoittaminen viestiin johtaa niiden julkaisemiseen Creative-Commons ShareAlike 3.0 Unported (CC BY-SA 3.0) -lisenssillä ("Stack Overflow Terms of Service", 2018). Kyseinen lisenssi on ensisijaisesti tarkoitettu perinteisemmille julkaisumuodoille, kuten kuville ja tekstille, ei niinkään ohjelmien lähdekoodille. Itse Creative-Commons organisaatio ei suosittele lisenssiä käytettävän ohjelmistoille ("Creative Commons FAQ", 2018b). Koodiesimerkkiä käyttävän ohjelmiston tulee CC BY-SA 3.0 lisenssin (2018) mukaisesti noudattaa seuraavanlaisia lisensointiehtoja:

- "Nimeä — Sinun on mainittava lähde asianmukaisesti, tarjottava linkki lisenssiin sekä merkittävä, mikäli olet tehnyt muutoksia. Voit tehdä yllä olevan millä tahansa kohtuullisella tavalla, mutta et siten, että annat ymmärtää lisenssinantajan suosittelevan sinua tai teoksen käyttöäsi."
- "JaaSamoin — Jos remiksaat tai muokkaat aineistoa taikka luot sen pohjalta uusia aineistoja, sinun on jaettava muutoksiasi samalla lisenssillä kuin alkuperäistä aineistoa."

An ym. (2017) tulkitsevat näiden ehtojen tarkoittavan koodiesimerkkien käyttäjälle seuraavanlaisia käytännön vaateita:

- Käytettyyn koodiesimerkkiin on viitattava
- Koodiesimerkin CC BY-SA 3.0 lisenssiin on viitattava
- Koodiesimerkkiin tehdystä muutoksesta on ilmoitettava
- Se työ, johon koodiesimerkkiä on käytetty, on myös julkaistava CC BY-SA 3.0 (tai myöhemmällä) lisenssillä.

Etenkin viimeinen ehto voidaan nähdä mahdollisesti hyvinkin rajoittavana tekijänä koodiesimerkkien käytölle Stack Overflowsta. Creative Commons ShareAlike lisenssi on vahva *copyleft* lisenssi, joka velvoittaa lisensoidun materiaalin käyttäjän julkaisemaan johdetun teoksen samalla lisenssillä (Baltes ym., 2017). CC BY-SA:n myöhäisempi versio 4.0 onkin yhteensopiva GNU General Public License (GPL) version 3 kanssa (Stallman, 2018).

Stack Overflown lisenssin lisäksi voi myös olla niin, että koodiesimerkillä on myös jokin muu lisenssi jo julkaisuvaiheessa. Esimerkki voi perustua vaikkapa johonkin avoimen lähdekoodin ohjelmakirjastoon. Koodiesimerkin yhteydessä tulee tällöin mainita myös tämä toinen lisenssi. Kyseinen koodiesimerkki

on tällöin kaksoislisensoitu (engl. ”dual licensed”). Koodiesimerkin käyttäjän on tällöin huomioitava myös tämä toinen lisenssi ja sen yhteensopivuus kehitettävän ohjelmiston kanssa (Baltes ym., 2017).

Suljetun lähdekoodin ohjelmistossa ei voida käyttää Stack Overflowsta kopioitua ei-triviaalia koodiesimerkkiä, ellei sitä ole kaksoislisensoitu jollain sallivammalla lisenssillä (Baltes ym., 2017).

### 3.3.3 Stack Overflown lisenssin noudattaminen reaalimaailmassa

An ym. (2017) tutkivat lähdekoodin liikkumista Stack Overflowsta sovelluksiin ja takaisin Stack Overflowhun. Yksikään havaituista koodiesimerkkejä käyttäneestä sovelluksesta ei noudattanut edellä kuvattuja Stack Overflown lisensointiehtoja. Vastaavasti sovellusten lähdekoodin pohjalta luodut koodiesimerkit olivat muutamaa poikkeusta lukuun ottamatta julkaistu siten, että ohjelmiston alkuperäinen lisenssi ei seurannut koodiesimerkin mukana. Tutkijat käyttivät termiä ”code laundering” tällaisesta toiminnasta, jossa koodin alkuperäinen lisenssi vaihtuu toiseksi (An ym., 2017).

An ym. (2017) huomauttavat myös, että Stack Overflown käyttäjä käyttää yleensä pseudonimeä, jolloin koodiesimerkin tekijänoikeutta on mahdotonta tarkalleen jäljittää. Kyseessä voi siis olla oman koodin kopioiminen. Lisäksi sellaisissa tapauksissa, jossa koodiesimerkin nähtiin vaeltaneen sovelluksesta toiseen (esimerkiksi sovelluksen A koodi päätyy ensin Stack Overflowhun ja sitten sieltä sovellukseen B), on mahdotonta aukottomasti määritellä Stack Overflown osuutta. Voihan olla niinkin, että koodia kopioidaan projektista toiseen ilman sosiaalisen median tukea. (An ym., 2017).

Koodin matkaamista Stack Overflown kautta ovat tutkineet myös Romansky, Chen, Malhotra ja Hindle (2018). Heidän tuloksissaan suuri joukko Python kielisistä Stack Overflown koodiesimerkeistä oli kopioita Python dokumentaatiosta tai pip-modulien lähdekoodista. Tekijänoikeuksien mukainen attribuutio lähteeseen oli lähestulkoon olematonta (Romansky ym., 2018).

Ragkhitwetsagul ym. (2018) tutkivat Stack Overflown käyttäjiä kyselytutkimuksella aktiivisimmille käyttäjille. Valtaosa julkaistuista koodiesimerkeistä oli itsekirjoitettuja, siis tutkimukseen vastaajien mukaan, mutta myös muista lähteistä tulleita esimerkkejä käytettiin. Yli 98% vastaajista eivät kopioi lähdekoodin lisenssiä julkaisemiinsa koodiesimerkkeihin ja 69% eivät koskaan tarkasta lähdekoodin lisenssin yhteensopivuutta CC BY-SA 3.0 lisenssiin. Koodiesimerkkien käyttäjät eivät ole juurikaan sen parempia: 85% vastanneista koodiesimerkkien käyttäjistä eivät ole tietoisia esimerkkien CC BY-SA 3.0 lisenssistä ja 62% eivät viittaa käytettyyn koodiesimerkkiin omassa ohjelmassaan. 66% käyttäjistä eivät myöskään tarkasta oman ohjelmansa ja Stack Overflown lisenssien yhteensopivuutta (Ragkhitwetsagul ym., 2018).

Baltes ym. (2017) tutkivat Java koodin kulkeutumista Stack Overflowsta Githubiin. Heidän alustavassa kyselytutkimuksessa puolet ohjelmoijista ilmoittivat, etteivät liitä vaadittavaa viittausta käyttämäänsä koodiesimerkkiin. Koodianalyysin perusteella tutkijat kuitenkin osoittivat, että vähintään ¾ koodiesi-

merkkien käytöstä ei sisällä viitettä. Niistä projekteista, joihin koodiesimerkkejä kopioitiin, maksimissaan 1,8% on julkaistu CC BY-SA tai yhteensopivalla lisenssillä (Baltes ym., 2017).

# 4 YHTEENVETO JA POHDINTA

Tämän tutkielman tavoite oli kirjauskatsauksen menetelmällä käydä läpi internetin koodiesimerkkien käyttöä. Tutkimuskysymyksenä oli: *Mitkä ovat internetin koodiesimerkkien käytön hyödyt ja haitat?*

Kirjallisuuden perusteella tutkimustuloksina on esitetty tiivistys internetin koodiesimerkkien vaikutuksista ohjelmistokehityksessä:

- Koodiesimerkkien käyttö internetistä voi **nopeuttaa** kehitystyötä (Acar ym., 2016).
- Liiallinen **esimerkkikeskeisyys** voi vaikuttaa kielteisesti ohjelmoijan kykyyn ymmärtää tekemänsä koodin merkitystä (Meng ym., 2018).
- Koodiesimerkin **integrointi** omaan koodiin voi olla vaikeaa ja työlästä (Ragkhitwetsagul ym., 2018; Sillito & Begel, 2013; Xia ym., 2017).
- Koodiesimerkit voivat olla vanhentuneita tai muuten viallisia, jolloin syntyy **toiminnallisuusriski** (Abdalkareem ym., 2017; Ragkhitwetsagul ym., 2018; Xia ym., 2017).
- Käytetyt koodiesimerkit voivat myös sisältää vikoja ja puutteita, jotka johtavat **tietoturva- ja yksityisyysriskeihin** (Acar ym., 2016; Fischer ym., 2017).
- Internetin koodiesimerkit ovat usein tekijänoikeuslainsäädännön mukaisesti suojattuja teoksia. **Tekijänoikeudet** voivat muodostua ongelmaksi (An ym., 2017; Baltes ym., 2017; Ragkhitwetsagul ym., 2018; Romansky ym., 2018; Sojer & Henkel, 2011).

Tutkimusaiheena internetin koodiesimerkkien käyttö on vielä monella tapaa alkuvaiheessa. Kaikkia havaittuja pääalueita tulisi tutkia tarkemmin. Itse koodiesimerkkien käyttö on tunnettu ja tutkittu alue jo menneisyydestä, mutta internetin kysymys & vastaus sivustojen osuus on uudempi alue.

Kirjallisuudessa esitetyt tutkimustulokset kuvaavat aika negatiivisen tilannekuvan internetin koodiesimerkkien maailmasta: vanhentuneista ja yksinkertaisesti vääristä esimerkeistä lähtien ohjelmoijiin, jotka kopioi-ja-liimaa peri-

aatteella kopioivat kaiken mitä sattuvat löytämään. Lopputuloksena saadaan buginen ja tietoturvaton ohjelma, joka saattaa vielä rikkoa tekijänoikeuksiakin.

Miten tilannetta voisi parantaa? Koodiesimerkkien laatua tulisi parantaa. Ketkä ottaisivat aloitteekseen tämän? Tai edes minimissään vialliset ja tietoturvattomat esimerkit tulisi merkitä jotenkin. Osaltaan tätä tapahtuukin Stack Overflowssa, mutta kuten Fischerin ym. (2017) tutkimuksessa käy ilmi, ei varoituksia välttämättä huomioida. Vastuuta ongelmista tällä alueella voi vierittää myös osittain API-rajapintojen tuottajille. API-rajapinnan tulisi toimia niin, että virheellisen tai tietoturvattoman kutsun teko ei tapahdu vahingossa.

Tekijänoikeuksien suhteen tilanne on ongelmallinen. Stack Overflown alkuperäinen tavoite lienee ollut tarjota koodiesimerkkejä vapaasti käytettäväksi (”Stack Exchange Meta”, 2009). Käyttöehdot ja käytetty Creative Commons CC BY-SA lisenssi eivät kuitenkaan puhu samaa kieltä. Stack Overflown koodiesimerkkien käyttäjä altistaa itsensä ja organisaationsa mahdollisille tekijänoikeusrikkomuksille. Todellinen tilanne ei liene kuitenkaan niin paha, koodiesimerkkejä käytetään runsaasti, eikä oikeustapauksia ole.

Acarin ym. (2016) tutkimus paljastaa myös yhden merkittävän ongelman: API-rajapintojen dokumentaatio ei ole riittävän hyvällä tasolla. Tiedonhaku Stack Overflowsta on helpompaa kuin API-rajapinnan virallisesta dokumentaatiosta. Tätä voi toki selittää osin ohjelmoijien esimerkkikeskeisyydellä, jota on tässä tutkielmassa kuvattu muutamaan otteeseen. Voisiko dokumentaatiota kuitenkin parantaa? Ja lisäksi, kannattaisiko API-rajapinnan tuottajien olla aktiivisempia sosiaalisen median palveluissa, kuten Stack Overflowssa, tuottaen koodiesimerkkejä ja kommentoiden muiden esittämiä koodiesimerkkejä?

Lopuksi, monissa tämän tutkielman viittaamissa tutkimuksissa tunnistetaan koodiesimerkkien käyttö koneellisesti. Tämä on kehittyvä ala, esimerkiksi Romanskyn ym. (2018) tutkimuksessa tutkijat raportoivat rakentaneensa koodiduplikaatteja löytävän palvelun, jonka suorituskyky on monikymmenkertainen aikaisempiin vastaaviin ohjelmistoihin. Voisiko tulevaisuuden kuva olla jokin koodiplagioinnin paljastava palvelu, hieman tieteellisen tekstin URKUND -palvelun tapaisesti? Tällainen palvelu soveltuisi luonnollisesti vain avoimen lähdekoodin ohjelmistoille. Sen avulla myös koodiesimerkkien käytön tekijänoikeuksiin liittyvä keskustelu voisi saada uuden käänteen.

Jatkotutkimusaiheita voisivat olla vaikkapa:

- Esimerkkikeskeisyyden vaikutus kehitettävän ohjelmiston laatuun. Miten ja missä määrin toiminnalliset ja tietoturvaan ja yksityisyyteen liittyvien vikojen väärä riippuu ohjelmoijien esimerkkikeskeisyydestä? Entä esimerkkikeskeisyyden vaikutukset kehityksen nopeuteen?
- Miten Stack Overflown esimerkkejä voitaisiin luokitella (turvallinen / turvaton, hyvä / huono) automaattisesti ja voisiko ohjelmoijan käyttämä IDE varoittaa huonon esimerkin käytöstä?
- API-rajapintojen tuottajien rooli Stack Overflowssa.

# LÄHTEET

Abdalkareem, R., Shihab, E. & Rilling, J. (2017). On code reuse from StackOverflow: An exploratory study on android apps. *Information and Software Technology, 88*, 148-158. doi:10.1016/j.infsof.2017.04.005

Acar, Y., Backes, M., Fahl, S., Doowon Kim, Mazurek, M. L. & Stransky, C. (2016). You get where you're looking for: The impact of information sources on code security. *IEEE Symposium on Security and Privacy*. (s. 289-305). doi:10.1109/SP.2016.25

An, L., Mlouki, O., Khomh, F. & Antoniol, G. (2017). Stack overflow: A code laundering platform? *IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER).* doi:10.1109/SANER.2017.7884629

Balebako, R., Marsh, A., Lin, J., Hong, J. & Cranor, L. (2014). The privacy and security behaviors of smartphone app developers. Proceedings of *Workshop on Usable Security (USEC).*

Baltes, S. & Diehl, S. (2018). Usage and attribution of stack overflow code snippets in GitHub projects. *Empirical Software Engineering,* doi:10.1007/s10664-018-9650-5

Baltes, S., Kiefer, R. & Diehl, S. (2017). Attribution required. Proceedings of the *39th International Conference on Software Engineering Companion*, 161-163. doi:10.1109/ICSE-C.2017.99

Creative Commons — attribution-ShareAlike 3.0 unported — CC BY-SA 3.0. (26.12.2018). Haettu osoitteesta https://creativecommons.org/licenses/by-sa/3.0/deed.fi

Creative Commons FAQ. (26.12.2018a). Haettu osoitteesta https://creativecommons.org/faq/#what-is-an-adaptation

Creative Commons FAQ. (26.12.2018b). Haettu osoitteesta https://creativecommons.org/faq/#can-i-apply-a-creative-commons-license-to-software

Engelfriet, A. (7.1.2016). What is the license status of StackOverflow code snippets? [blogikirjoitus]. Haettu osoitteesta https://ictrecht.nl/2016/01/07/what-is-the-license-status-of-stackoverflow-code-snippets/

Fischer, F., Böttinger, K., Xiao, H., Stransky, C., Acar, Y., Backes, M. & Fahl, S. (2017). Stack overflow considered harmful? the impact of Copy&Paste on android application security. *2017 IEEE Symposium on Security and Privacy (SP).* doi:10.1109/SP.2017.31

Haenni, N., Lungu, M., Schwarz, N. & Nierstrasz, O. (2014). A quantitative analysis of developer information needs in software ecosystems. *Proceedings of the 2014 European Conference on Software Architecture Workshops.* (s. 12:1–12:6). doi:10.1145/2642803.2642815

Heilmann, C. (17.7.2015). The full stackoverflow developer [Blogikirjoitus]. Haettu osoitteesta https://christianheilmann.com/2015/07/17/the-full-stackoverflow-developer/

Hou, D. & Li, L. (2011). Obstacles in using frameworks and APIs: An exploratory study of programmers' newsgroup discussions. *IEEE 19th International Conference on Program Comprehension.* (s. 91-100). doi:10.1109/ICPC.2011.21

Knowles, M. S. (1975). *Self-directed learning: A guide for learners and teachers*. Englewood Cliffs (N.J.): Cambridge.

Ko, A. J., DeLine, R. & Venolia, G. (2007). Information needs in collocated software development teams. Proceedings of the *29th international conference on Software Engineering.* (s. 344–353). doi:10.1109/ICSE.2007.45

Lim, W. C. (1994). Effects of reuse on quality, productivity, and economics. *IEEE Software, 11*, 23-30. doi:10.1109/52.311048

Maalej, W., Tiarks, R., Roehm, T. & Koschke, R. (2014). On the comprehension of program comprehension. *ACM Transactions on Software Engineering and Methodology (TOSEM), 23*(4), 1-37. doi:10.1145/2622669

Meng, M., Steinhardt, S. & Schubert, A. (2018). Application programming interface documentation: What do software developers want? *Journal of Technical Writing and Communication; London, 48*(3), 295-330. doi:10.1177/0047281617721853

Myers, B. A. & Stylos, J. (2016). Improving API usability. *Commun. ACM, 59*(6), 62–69. doi:10.1145/2896587

Nasehi, S. M., Sillito, J., Maurer, F. & Burns, C. (2012). What makes a good code example?: A study of programming Q&A in StackOverflow. *28th IEEE International Conference on Software Maintenance (ICSM).* (s. 25-34) IEEE. doi:10.1109/ICSM.2012.6405249

Ragkhitwetsagul, C., Krinke, J. & Oliveto, R. (21.6.2018). Awareness and Experience of Developers to Outdated and License-Violating Code on Stack Overflow: An Online Survey. Haettu osoitteesta https://arxiv.org/abs/1806.08149

Robillard, M. & DeLine, R. (2011). A field study of API learning obstacles. *Empirical Software Engineering, 16*(6), 703-732. doi:10.1007/s10664-010-9150-8

Romansky, S., Chen, C., Malhotra, B. & Hindle, A. (31.7.2018). Sourcerer's apprentice and the study of code snippet migration. Haettu osoitteesta https://arxiv.org/abs/1808.00106

Sadowski, C., Stolee, K. T. & Elbaum, S. (2015). How developers search for code: A case study. Proceedings of the *2015 10th Joint Meeting on Foundations of Software Engineering*. (s. 191–201). doi:10.1145/2786805.2786855

Shi, L., Zhong, H., Xie, T. & Li, M. (2011). An empirical study on evolution of API documentation. Proceedings of the *14th international conference on Fundamental approaches to software engineering: part of the joint European conferences on theory and practice of software.* (s. 416–431). Haettu osoitteesta http://dl.acm.org/citation.cfm?id=1987434.1987473

Sillito, J. & Begel, A. (2013). App-directed learning: An exploratory study. *6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE).* (s. 81-84). doi:10.1109/CHASE.2013.6614736

Sojer, M. & Henkel, J. (2011, Dec 1,). License risks from ad hoc reuse of code from the internet. *Communications of the ACM, 54*, 74-81. doi:10.1145/2043174.2043193

Spolsky, J. (6.4.2018). The Stack Overflow Age [Blogikirjoitus]. Haettu osoitteesta https://www.joelonsoftware.com/2018/04/06/the-stack-overflow-age/

Stack Exchange Meta. (15.10.2009). What is up with the source code license on Stack Overflow? Haettu osoitteesta https://meta.stackexchange.com/questions/25956/what-is-up-with-the-source-code-license-on-stack-overflow

Stack Overflow Terms of Service. (21.5.2018). Haettu osoitteesta https://stackoverflow.com/legal/terms-of-service/public

Stallman, R. (15.12.2018). GNU license compatibility and relicensing. Haettu osoitteesta https://www.gnu.org/licenses/license-compatibility.html

Treude, C., Barzilay, O. & Storey, M. (2011). How do programmers ask and answer questions on the web?: Proceedings of the *33rd International Conference on Software Engineering*. (s. 804-807). doi:10.1145/1985793.1985907

Wang, W., Poo-Caamaño, G., Wilde, E. & German, D. (2015). What is the gist? *IEEE/ACM 12th Working Conference on Mining Software Repositories*. (s. 314-323). doi:10.1109/MSR.2015.36

Xia, X., Bao, L., Lo, D., Kochhar, P., Hassan, A. & Xing, Z. (2017). What do developers search for on the web? *Empirical Software Engineering, 22*(6), 3149-3185. doi:10.1007/s10664-017-9514-4