Automaatiosta vauhtia ohjelmointityöhön

Mallipohjainen kehittäminen vie ohjelmointialaa kohti ohjelmistoteollisuutta.

Panu Räty

Tarvitsemme yhä monimutkaisempia tietojärjestelmiä kaikilla elämänalueilla, ja paine sovellusten nopeaan tuottamiseen kasvaa. Osaavien ohjelmoijien määrä ei riitä paikkaamaan alan kasvutarpeita.

Siksi ohjelmistontuotannossa esitään apua savuliipputeollisuuden hiotuista prosesseista. Yksi ratkaisu tuotannon nopeuttamiseksi ja laadun parantamiseksi on mallipohjainen ohjelmistokehitys, jossa osa tarvittavasta koodista generoidaan automaattisesti.

Automaatiosta hakee tehokkuutta myös Jyväskylässä toimiva Soulcore. Vuonna 2014 perustettu neljäntoista työntekijän softatalo on ehtinyt rakentaa liiketoimintaratkaisuja monelle tunnetulle yritykselle ja yhteisölle Kotipizzasta Espoon kaupunkiin, Opetusalan ammattijärjestöön OAJ:hin ja VR Groupiin. Yksittäisten hankkeiden hinnat ovat vaihdelleet välillä 20 000–350 000 euroa.

Erityistä ratkaisuissa on niiden tuotantotapa.

"Kaikki sovelluksemme perustuvat mallipohjaiseen sovelluskehitykseen", sanoo Soulcoren toimitusjohtaja **Tiina Vestman**.

"Yleensä mallin pohjalta generoidaan vain jokin osa koodista. Esimerkiksi lomakegeneraattorella luodaan html-lomake. Meillä on alusta lähtien ollut tavoitteena generoida toimivia sovelluksia suoraan mallista."

MALLINTAMISEN AJATUS ei ole uusi. Malleja on hyödynnetty pitkään esimerkiksi ohjelmistojen rakenteen hallitsemisessa ja dokumentaatiossa. Mallipohjaisuuden käyttöaluet ovat kuitenkin jatkuvasti laajentuneet.

Yhteistä mallipohjaisille menetelmille on pyrkimys korkeaan abstraktiotasoon. Koodia siis kirjoitetaan vähän, mutta sitäkin tiivimmässä muodossa. Esimerkiksi niin sanotut low code -työkalut tarjoavat mahdollisuuden synnyttää yksinkertaisia sovelluksia käytännössä liki täysin ilman perinteistä ohjelmointia.

Mallipohjaisessa sovelluskehittämisessä koodi generoidaan järjestelmälle ennalta luotujen sääntöjen ja järjestelmämallin pohjalta.

Vestman selittää, että järjestelmän pohjatiedot kaivetaan suoraan asiakkaan liiketoiminnan arkea kuvaavista Excel-taulukoista, asiakkaan vanhoista järjestelmistä ja raporteista. Tietoja täydennetään keskusteluilla asiakkaan kanssa.

"Kun järjestelmä on saatu mallinnettua, pystymme periaatteessa muutamassa päivässä tekemään ihan mehevän kokoisen sovelluksen, esimerkiksi asiakasrekisterin", Vestman sanoo.

Yleensä Soulcore generoi järjestelmän ensimmäisen version asiakkaalle jo tarjousvaiheessa.

"Meillä ei ole pelkästään tarjousta, vaan työnäyte, josta päästään jatkamaan heti eteenpäin."

PROTOTYYPIN TOIMINTOJA tarkemmetaan yhdessä asiakasyrityksen ammattilaisten kanssa.

"Asiakkaan ei tarvitse kertoa kuin omasta bisneksestään, mitä tietoja hän käsittelee ja miten hän niitä käsittelee", Vestman sanoo.

Määrittelyssä edetään kohti eri käyttäjäryhmien tarpeita työpöytänäkymien rakentamiseksi. Yrityksen työprosessien selvittämistä tarvitaan puolestaan työnkulkujen ja hälytysten saamiseksi kohdalleen. Viimeisenä kokonaisuuteen lisätään raportointityökalut.

Ohjelmoija mallintaa generoitavan järjestelmän täsmäkielellä eli niin sanotulla toimialakohtaisella mallinnuskielellä (domain specific language, dsl), johon mallipohjainen kehittäminen yleensä perustuu.

Vestman muistuttaa koodin generoimisen hyvistä puolista. Koodigeneraattori ei ole esimerkiksi vasta

MALLI-POHJAINEN SOVELLUS-KEHITYS

MIKÄ Menetelmä, joissa koodi generoidaan ennalta luotujen sääntöjen ja järjestelmämallin pohjalta. Englanniksi *model-driven development*.

MIKSI Koodia pystytään generoimaan automaattisesti.

MIKSI EI Mallinnuskielten ja generaattorien saatavuus rajoittaa käytettävyyttä, koska ne ovat aina toimialakohtaisia.

tullut koulunpenkiltä, eikä se kärsi tehottomista päivistä. Se tekee tasaista lautaa joka päivä.

"Konepellin alustaa voidaan ylpeydellä esittää. Kädenjälki on samanlaista kauttaaltaan, kun siinä on ollut automatiikka apuna."

SOULCOREN PÄÄARKKITEHTI Jarno Leikas toteaa, että automatisoinnin tarkoitus on yksinkertaisesti helpottaa ohjelmoijien työtä.

"Emme yritä korvata ohjelmistokehittäjää, vaan tehdä ohjelmistokehittämisestä tehokkaampaa", Leikas sanoo.

Automatiikalla tuotettu ohjelmisto ei sisällä tavallisia pikkubugeja siinä missä käsin naputeltu koodi. Generoidun koodin ohjelmointivirheet ovat useimmiten mallinnusvaiheessa syntyneitä logiikkaongelmia tai väärinkäsityksiä.

"Meillä on oma sovelluskehikko, joka hoitaa esimerkiksi navigointia ja autentikointia. Sitä meidän ei tarvitse joka kerta generoida uudaksi, vaan laitamme ikään kuin tietosisällön tai hyötykuorman ajettavaksi sovelluskehikkoomme", Leikas sanoo.

Koodin generoimiseen käytetyt pohjat saavat järjestelmät muistuttamaan rakenteellisesti toisiaan. Jokainen sovellus on silti ainutkertainen, sillä asiakkaiden tarpeisiin perustuvat järjestelmämallit ovat keskenään erilaisia.

Vaikka generointi sopii jopa järjestelmäpäivityksiin, kaikkeen mallinnustyökalut eivät sentään venny. Etenkin monimutkaiset liiketoimintasäännöt, käyttäjyyden hienosäätö ja järjestelmien väliset integraatiot vaativat edelleen perinteisen ohjelmoinnin ilmaisuvoimaa.

Leikas toteaa, että automaation avulla kehittäjät voivat keskittyä oikeisiin asioihin.

"Kehittäjälle tämä on kivaa. Kyllähän tämä poistaa sen tylsän osuuden työnteosta."

Sovellukset voidaan siis periaatteessa tuottaa alusta loppuun ilman perinteistä full stack -ohjelmoijaa. Käyttöliittymä edellyttää kuitenkin käsityötä.

"Ainoa automatisoimatta jäävä asia on se, mitä ruudussa näkyy – eli missä kohtaa asiat ovat ja mitkä ovat fontit ja värit. Pystymme automatisoimaan kaiken muun."

AITOCODE-TYÖNIMEN saanut hanke lupaa Rintasen mukaan ohjelmistojen kehitys-, testaus-, käyttöönotto- ja ylläpitokulujen merkittävää vähenemistä. Myös sovellusten tietoturva voi parantua "massivisesti".

Tällä hetkellä järjestelmä tuottaa c-kieltä, mutta tulevaisuudessa käyttäjä voi valita vapaasti muitakin ohjelmointikieliä.

Aalto-yliopistossa syntyneestä teknikasta ei ole toistaiseksi julkaistu ainoatakaan tutkimusta. Julkaisut vaikeuttaisivat algoritmien kaupallistamista. Tutkimuskohteeksi on valittu hankkeeseen liittyvä ohjelmistojen automaattinen validointi.

"Voimme automaattisesti todistaa, että järjestelmälläm me on tietyt ominaisuudet."

Viime vuodet Aitocodea on rakennettu Business Finlandin myöntämällä apurahalla, mutta rahat on nyt käytetty. Syksyllä tiimi aikoo suunnitella elävän elämän pilotsovelluksia Aalto-yliopistolle ja yrityskumppaneille riskirahoittajien vakuuttamiseksi.

Lopullinen tavoite on kunnianhimoinen.

"Meidän pidemmän tähtäimen visiomme on sellainen, että kymmenet tai sadat tuhannet ohjelmoijat käyttävät tätä maailmanlaajuisesti."

ASIAKKAALLE AUTOMAATIO TARJOAA NOPEUTTA

Yksi Soulcoren asiakkaista on kaatopaikkajyrä ja kierrätysrepijöitä maailmalle vievä konepajayritys Tana. Soulcore rakensi Tanalle järjestelmän tuotannossa havaitujen laatupoikkeamien käsittelyyn ja hallintaan.

Tanan hankintajohtaja **Suvi Kupiainen** kertoo, että aikaisemmin laatupoikkeamat oli kirjattu yksinkertaisesti Excel-taulukkoon, jonka hän esitteli Soulcoren väelle ensimmäisessä tapaamisessa.

Kun kehittäjät seuraavan kerran palasivat virallisen tarjouksen kanssa, heillä oli mukanaan tarjouksen lisäksi myös "80-prosenttisesti valmis ratkaisu". Prototyyppiä tarkennettiin työpajoissa, joissa järjestelmään lisättiin esimerkiksi uusia kenttiä. Työpajoja tarvittiin kaikkiaan kolme.

Uusi järjestelmä oli Kupiaisen mukaan parissa kuukaudessa jo tuotantokäytössä.

"Olemme ottaneet järjestelmän käyttöön myös ulkoisin, asiakkailta tulevin poikkeamin. Kaikki kentältä tulevat havainnot ja poikkeamat kirjataan nyt samaan järjestelmään."

"KYLLÄHÄN TÄMÄ POISTAA SEN TYLSÄN OSUUDEN TYÖNTEOSTA."

AUTOMAATIOLLA KOKONAISIA JÄRJESTELMIÄ

Olisiko kokonaisia tietojärjestelmiä mahdollista määritellä automaatiolla ilman sovelluskehittäjän lisälemmää koodia? Tällaista ratkaisua rakentaa Aalto-yliopiston tietotekniikan laitoksen professori **Jussi Rintanen** timeineen.

Vielä toistaiseksi korkean abstraktiotason ohjelmointityökalut tarvitsevat ihmisohjelmoijaa paikkamaan sovelluksen puutteita.

"Meidän hommassa ne monimutkaisetkin asiat automatisoidaan", Rintanen sanoo.

"Käymme suoraan käsiksi siihen keskeiseen ongelmaan, miten käyttöliittymän täytyy toimia ja miten tietokannassa olevaa tietoa luetaan ja kirjoitetaan. Automaatiomme yhdistää nämä kaksi asiaa yhteen – eli käyttäjälle tarjottavan toiminnallisuuden plus tietokantaoperaatiot."