Esipuhe

Melko useita muutoksia on tapahtunut tämän *C++*-kirjan toisen ja kolmannen painoksen välillä. Merkittävin on se, että C++ on käynyt läpi kansainvälisen standardoinnin, joka ei ole ainoastaan lisännyt uusia piirteitä kieleen kuten poikkeusten käsittelyn, ajonaikaisen tyyppitunnistuksen, nimiavaruudet, sisäisen Boolean-tietotyypin ja tyyppimuunnoksen uuden merkintätavan, vaan myös merkittävästi muokannut ja laajentanut olemassaolevia piirteitä, kuten malleja, luokkamekanismia sekä oliokeskeisessä että oliopohjaisessa ohjelmoinnissa, sisäkkäisiä tyyppejä ja ylikuormitetun funktion ratkaisua. Ehkä vieläkin merkittävämpää on se, että vakio C++ sisältää nyt laajan kirjaston, johon aikaisemmin viitattiin nimellä Standard Template Library eli STL. Uusi merkkijonotyyppi string, jono- ja assosiatiiviset säiliötyypit — kuten vektori, lista, kartta ja joukko — ja laaja kokoelma geneerisiä algoritmeja, joilla voidaan käsitellä näitä tyyppejä, ovat kaikki tämän uuden vakiokirjaston piirteitä. Käsiteltävänä ei ole ainoastaan paljon uutta materiaalia, vaan myös uusia tapoja ajatella, kuinka ohjelmoimme C++-kielellä. Lyhyesti sanottuna: itse asiassa ei vain C++-kieltä ole keksitty vastikään uudelleen, vaan myös tämä *C++*-kirja sitä varten; sen kolmas painos.

Tässä kolmannessa painoksessa ei vain kielen käsittely ole muuttunut perusteellisesti, vaan niin on myös kirjoittaminen: ensiksikin olemme tuplanneet itsemme! Lisäksi olemme kansainvälistäneet prosessia, vaikka olemme lujasti juurtuneet Pohjois-Amerikkaan: Stan on amerikkalainen ja Josée on kanadalainen. Lopuksi, kaksi kirjoittajaa kuvastaa C++-yhteisön kahta päätyötä: Stan osallistuu parhaillaan 3D-tietokonegrafiikkaa ja -animaatiota sisältävän C++-sovelluksen kehittämiseen Walt Disney Feature Animationilla, kun taas Josée osallistuu C++-kielen määrittelyyn ja toteutukseen. Hän on myös Core Language -alikomitean puheenjohtaja ja C++-kääntäjätiimin jäsen IBM Canada Laboratoryssa.

Stan on eräs Bellin laboratoriotiimin alkuperäisistä jäsenistä, jotka työskentelivät Bjarne Stroustrupin kanssa — joka on C++-kielen keksijä — ja on ollut mukana vuodesta 1984 lähtien. Stan työskenteli monissa cfront-toteutuksissa: alkuperäisessä C++-toteutuksessa, versiosta 1.1 versioon 3.0 vuonna 1986 sekä johti kehitystiimiä versioissa 2.1 ja 3.0. Sen jälkeen hän työskenteli Stroustrupin johdolla projektissa, joka tunnetaan nimellä *Foundation Research Project on the Object Model component* ja joka koski ohjelmointiympäristön kehittämistä.

Josée on ollut C++-kääntäjätiimin jäsen IBM Canada Laboratoryssa kahdeksan vuotta. Hän on ollut standardikomitean jäsen vuodesta 1990 lähtien. Hän oli komitean varapuheenjohtaja kolme vuotta ja hän on ollut Core Language -alikomitean puheenjohtaja neljä vuotta.

Tämä *C++*-kirjan kolmas painos edustaa mittavaa tekstin uudistusta, joka ei vain kuvasta kielen muutoksia ja laajennuksia, vaan myös muutoksia kirjoittajien näkemyksissä ja kokemuksissa.

Tämän kirjan rakenne

Tämä *C++*-kirja tarjoaa perusteellisen johdannon C++-kielen kansainväliseen standardiin. Se on perusteos siinä mielessä, että se tarjoaa tietoisesti opastavan lähestymistavan C++-kielen kuvaamiseen. (Se *ei* ole perusteos siinä mielessä, että se tarjoaisi liian yksinkertaisia tai "keveitä" kuvauksia kielestä.) Kielen ohjelmointinäkökohdat kuten poikkeusten käsittely, säiliötyypit, oliokeskeinen ohjelmointi jne. on esitelty tietyn ohjelmointitehtävän ratkaisun yhteydessä. Kielen säännöt, kuten ylikuormitetun funktiokutsun ratkaisu tai oliokeskeisen ohjelmoinnin tuetut tyyppikonversiot, joiden ei aluksi luulisi kuuluvan perusteokseen, käsitellään laajasti. Mutta me uskomme, että käsittely on välttämätöntä kielen käytännön ymmärtämiseksi ja näemme, että tähän materiaaliin palataan aina uudelleen sen sijaan, että se luettaisiin yhdeltä istumalta. Jos pidät lukemaasi aluksi liian raskaana tai yksinkertaisesti liian kuivana, jätä se materiaali syrjään myöhempää ajankohtaa varten — merkitsemme sellaiset luvut seuraavasti: 

C-kielen tietämystä ei edellytetä, vaikka jonkin nykyaikaisen, lohkorakenteisen kielen tunteminen saa läpikäynnin helpommaksi. Kirja on tarkoitettu ensimmäiseksi C++-kirjaksi; sitä ei ole tarkoitettu ensimmäiseksi ohjelmointikirjaksi! Varmistuaksemme onnistumisesta aloitamme aina yleissanastolla. Kuitenkin alkupään kappaleissa käsitellään joitakin peruskäsitteitä, kuten silmukointilauseita ja muuttujia, jotka saattavat joistakin lukijoista tuntua liian alkeismaisilta. Mutta ei kannata hermoilla: käsittelyn laajuus kasvaa nopeasti.

Suuri osa C++-kielen tehokkuudesta katsotaan tulevan sen tuesta uusille ohjelmointitavoille ja ohjelmointiongelmien ajattelutavoille. Sen vuoksi C++-kielen opiskelu tehokkaaseen käyttöön vaatii enemmän kuin vain uusien syntaksi- ja semantiikkajoukkojen opettelun. Tämän laajemman oppimisen helpottamiseksi kirja on järjestetty sarjaksi laajempia esimerkkejä. Näitä esimerkkejä käytetään sekä monien kielen yksityiskohtaisten piirteiden esittelyyn että niiden käytön motivoimiseen. Kun opimme kielen piirteiden yhteydessä täydellisen esimerkin, tulee selväksi, miksi sellaiset piirteet ovat hyödyllisiä sekä myös ajatus siitä, milloin ja miten käyttäisimme niitä tosielämän ongelmien ratkaisuissa. Lisäksi tällainen keskittyminen esimerkkeihin antaa mahdollisuuden käyttää varhain käsitteitä, jotka selitetään täydellisemmin sitten, kun lukijan tietämyspohja on vankemmalla pohjalla. Ensimmäiset esimerkit sisältävät C++-kielen yksinkertaisia peruskäsitteiden käyttötilanteita antaen makua ohjelmoinnista, jota C++-kielellä voidaan tehdä vaatimatta suunnittelun ja toteutuksen täydellistä ymmärtämistä.

Luvut 1 ja 2 muodostavat itsenäisen johdannon ja yleiskatsauksen koko C++-kieleen. Osa I on tarkoitettu vauhtiin pääsemiseksi C++-kielen tukemien käsitteiden ja piirteiden opetteluun — ja ohjelman kirjoittamisen ja suorittamisen periaatteisiin. Tämän tekstin päätyttyä sinulla tulisi olla tunne C++-kielen tuesta *ja aavistus, että et todellisuudessa ymmärtänyt sitä ollenkaan*. Mutta se on ok: sitä varten on olemassa kirjan kaikki muu teksti loppuun saakka!

Luvussa 1 esitellään kielen peruselementit: sisäiset tietotyypit, muuttujat, lausekkeet, lauseet ja funktiot. Se käy läpi pienimmän mahdollisen C++-ohjelman, käsittelee lyhyesti ohjelmiemme käännösprosessin, esittelee pintapuolisesti käsitteen, jota kutsutaan *esikääntäjäksi* ja ottaa esille ensim-

mäisen kerran tuen syötölle ja tulostukselle. Luvussa esitellään lukuisa joukko yksinkertaisia, mutta täydellisiä C++-ohjelmia, jotka rohkaisevat lukijaa kääntämään ja suorittamaan niitä. Luvussa 2 esitellään C++-kielen tuki luokkamekanismin kautta oliopohjaiselle ja oliokeskeiselle ohjelmoinnille ja kuinka ne molemmat ovat käyneet läpi taulukkokäsitteen kehityksen. Lisäksi luvussa esitellään mallit, nimiavaruudet, poikkeusten käsittely ja vakiokirjaston tuki yleisille säiliötyypeille ja geneeriselle ohjelmoinnille. Tämä luku on melko nopeatahtinen ja jotkut lukijat voivat pitää sitä liian raskaana luettavana. Jos tunnet näin, ehdotamme, että vilkuilet sen läpi ja palaat siihen myöhemmin.

C++-kielelle on tunnusomaista lukuisat piirteet, jotka sallivat käyttäjän laajentaa kieltä itseään määrittelemällä uusia tietotyyppejä, joita voidaan sitten käyttää joustavasti ja yksinkertaisesti sisäisten tietotyyppien kanssa. Ensimmäinen askel osaamiseen on ymmärtää itse peruskieli. Luvut 3 — 6 (Osa II) esittelevät kieltä tällä tasolla.

Luvussa 3 esitellään sisäiset ja yhdistetyt tietotyypit, jotka ovat kielen esimäärittelemiä sekä myös string-merkkijono-, kompleksi- ja vektoriluokkien tietotyypit, jotka tulevat C++-vakiokirjaston mukana. Nämä tyypit muodostavat kaikkien ohjelmiemme rakennuspalikoiden perustan. Luvussa 4 käsitellään yksityiskohtaisesti kielen tukemat lausekkeet kuten aritmetiikka-, vertailu- ja sijoituslausekkeet. Lauseet, jotka muodostavat pienimmän itsenäisen yksikön C++-ohjelmassa, ovat luvun 5 aiheena. Luvussa 6 keskitytään C++-kirjaston säiliötyyppeihin. Sen sijaan, että olisimme antaneet yksinkertaisen listauksen käytettävissä olevista operaatioista, olemme käyneet läpi tekstinkyselyjärjestelmän toteutuksen kuvataksemme sellaisten suunnittelua ja käyttöä.

Luvut 7 — 12 (Osa III) keskittyvät C++-kielen tukemaan proseduraalipohjaiseen ohjelmointiin. Luvussa 7 tutustutaan C++-kielen funktiomekanismiin. Funktiot sulkevat sisäänsä joukon operaatioita, jotka yleensä muodostavat yksinkertaisen tehtävän kuten `print()`. (Nimen perässä olevat tyhjät sulut ilmaisevat, että se edustaa funktiota.) Luvun 8 aiheina ovat ohjelman viittausalue, muuttujien elinaika sekä nimiavaruus. Luku 9 laajentaa luvussa 7 esille tulleita funktioita esittelemällä funktion ylikuormituksen. Funktion ylikuormitus sallii useiden sellaisten funktioilmentymien esiintymisen samalla nimellä, jotka tekevät jonkin yleisen toimenpiteen (mutta vaativat erilaiset toteutukset). Voimme määritellä esimerkiksi `print()`-funktioiden kokoelman, jotka tulostavat erityyppisiä tietoja. Luvussa 10 esitellään ja kuvataan funktiomallien käyttöä. Funktiomalli on määräys funktioesiintymien automaattiselle generoinnille, joita voi olla rajattomasti ja joiden tyyppi voi vaihdella, mutta joiden tehtävät säilyvät muuttumattomina.

C++ tukee poikkeusten käsittelypiirrettä. Poikkeus edustaa odottamatonta ohjelman käyttäytymistä kuten kaiken käytettävissä olevan muistin hupenemista ohjelmalta. Se osa ohjelmasta, jossa poikkeus tapahtuu, “heittää” poikkeuksen — joka tarkoittaa, että loppuosa ohjelmasta saa sen käyttöönsä. Ohjelman jonkun funktion pitää sitten “siepata” poikkeus ja tehdä se, mikä on välttämätöntä. Poikkeusten käsittely on jaettu kahteen lukuun. Luvussa 11 esitellään poikkeusten käsittelyn perussyntaksi käyttäen yksinkertaista esimerkkiä poikkeusluokkatyypin heittämisestä ja kiinniottamisesta. Koska ohjelmissamme todellisuudessa käsitellyt poikkeukset ovat usein oliokeskeisen luokkahierarkian luokkaolioita, poikkeuksien heittäminen ja käsittely jatkuu luvussa 19 sen jälkeen, kun oliokeskeinen ohjelmointi on esitelty.

Luvussa 12 esitellään mittava kokoelma geneerisiä algoritmeja, jotka tulevat vakiokirjaston mukana ja tutkitaan, kuinka ne käyttäytyvät luvun 6 säiliötyyppien kanssa kuten myös sisäisten taulukkotyyppien kanssa. Luku alkaa käymällä läpi ohjelmasuunnittelua yleisiä algoritmeja käyttäen. Iteraattoreita, jotka esitellään luvussa 6, käsitellään lisää luvussa 12, koska ne ovat "liima", joka sitoo geneeriset algoritmit varsinaisiin säiliöihin. Funktio-olion käsite esitellään ja kuvataan myös. Funktio-olioilla voimme antaa geneerisissä algoritmeissa käytetyille operaattoreille kuten yhtäsuuruus- tai pienempi kuin -operaattorille eri merkityksiä. Itse algoritmit ovat tämän kirjan liitteessä selitettynä yksityiskohtaisesti käyttökuvauksineen .

Luvut 13 — 16 (Osa IV) keskittyvät *oliopohjaiseen* ohjelmointiin — tarkoittaa luokkapiirteiden määrittelyä ja käyttöä itsenäisten abstraktien tietotyyppien luomiseksi. Luomalla uusia tyyppejä ongelma-alueen kuvaamiseksi, C++ sallii ohjelmoijan kirjoittaa sovelluksia niin, että hänen tarvitsee vähemmän olla huolissaan monista muistettavista asioista, jotka tekevät ohjelmoinnista vaivalloista. Sovelluksen perustietotyypit voidaan toteuttaa kerran ja käyttää uudelleen, mikä mahdollistaa ohjelmoijan keskittymisen itse ongelmaan toteutuksen yksityiskohtien sijasta. Tiedon kapselointipiirteet voivat merkittävästi yksinkertaistaa myöhempää ylläpitoa ja sovellusten kehittämistä.

Luku 13 keskittyy yleiseen luokkamekanismiin: kuinka luokka määritellään, mikä on sama kuin käsite *tiedon piilottaminen* — tarkoittaa luokan julkisen rajapinnan erottamista yksityisestä toteutuksesta — ja kuinka luokkaolioiden ilmentymiä määritellään ja käsitellään. Luvussa käsitellään myös luokan viittausaluetta, sisäkkäisiä luokkia ja luokkia nimiavaruuden jäseninä.

Luvussa 14 kerrotaan yksityiskohtaisesti C++-kielen erikoistuesta luokkaolioiden alustamisessa, hajottamisessa ja sijoittamisessa käyttäen erityisiä jäsenfunktioita, jotka on nimetty vastaavasti *muodostajaksi*, *hajottajaksi* ja *kopioinnin sijoitusoperaattoriksi*.

Käsittelemme myös jäsenittäisen alustuksen ja kopioinnin, jossa yksi luokkaolio alustetaan tai sijoitetaan toiseen luokkansa olioon, sekä sille tarkoitetun *nimetyn paluuarvon* optimoinnin tuen.

Luvussa 15 katsotaan luokkakohtaisten operaattoreiden ylikuormitusta esittämällä aluksi yleiset käsitteet ja suunnittelunäkökohdat sekä sen jälkeen tietyt operaattorit kuten sijoitus-, indeksi- ja kutsuoperaattorit sekä luokkakohtaiset new- ja delete-operaattorit. Luvussa esitellään myös luokan ystävän merkintätapa erityisine käsittelyoikeuksineen, ja syy, miksi ystäviä joskus tarvitaan. Sen jälkeen käsitellään käyttäjän määrittelemät konversiot mukaan lukien niihin liittyvät käsitteet sekä mittava esimerkki konversioiden käytöstä. Tässä luvussa käsitellään myös funktion ylikuormituksen ratkaisun säännöt sangen yksityiskohtaisesti ja mittavilla koodiesimerkeillä.

Luvun 16 aiheena ovat luokkamallit. Luokkamalli on määräys sellaisen luokan luomiseksi, jossa yksi tai useampi tyyppi tai arvo on parametroitu. Esimerkiksi vektoriluokka voi parametroida elementin tyypin, jonka se sisältää. Puskuriluokka voi parametroida sisältämänsä elementin tyypin lisäksi puskurinsa koon. Mutkikkaammassa käytössä kuten levitettävissä tietokoneohjelmissa IPC-, osoitteistus- ja tahdistusliittymät voidaan kaikki parametroida. Tähän lukuun sisältyy luokkamallin määrittelyn käsittely, kuinka luodaan tietyn tyyppisiä il-

mentymiä luokkamallista, kuinka luokkamallin jäsenet määritellään (jäsenfunktiot, staattiset jäsenet ja sisäkkäiset tyypit) ja kuinka luokkamalleja käyttävät ohjelmamme järjestetään. Luku päättyy laajaan luokkamalliesimerkkiin.

Lukujen 17, 18, 19 ja 20 (Osa IV) aiheena ovat oliokeskeinen ohjelmointi ja C++-kielen piirteet, jotka tukevat sitä. Luvussa 17 esitellään C++-kielen piirteet, jotka tukevat oliokeskeisen ohjelmoinnin pääelementtejä: periytyminen ja dynaaminen sitominen. Oliokeskeisessä ohjelmoinnissa yli/ali-suhteet (myös tyyppi/alityyppi-suhteet) määritellään sellaisten luokkien välille, jotka jakavat yhteisen käyttäytymisen. Sen sijaan, että toteutettaisiin jaettuja piirteitä uudelleen, luokka perii tiedon ja toimenpiteet yliluokaltaan. Aliluokkaan eli alityyppiin ohjelmoidaan vain sen erot yliluokastaan. Voimme esimerkiksi määritellä yliluokan luokkatyypin Employee ja kaksi aliluokkaa: TemporaryEmpl ja Manager. Nämä alityypit perivät kaikki Employee-luokkatyypin käyttäytymisen. Ne toteuttavat käyttäytymistä, joka on yksilöllistä niiden jokaiselle vastaavalle tyypille.

Toinen periytymisen aihe, jota kutsutaan nimellä *polymorfismi* eli *monimuotoisuus*, on yliluokan luokkatyypin mahdollisuus viitata mihin tahansa siitä periytyneeseen alityyppiin. Esimerkiksi Employee voi osoittaa TemporaryEmpl- tai Manager-aliluokkaa. Dynaaminen sitominen on kyky ratkaista suorituksen aikana suoritettava toimenpide, joka perustuu monimuotoisen olion todelliseen tyyppiin. C++-kielessä tämä käsitellään virtuaalifunktioiden mekanismin kautta.

Luvussa 17 esitellään oliokeskeisen ohjelmoinnin peruspiirteet. Luvussa käydään läpi Query-luokkahierarkian suunnittelu ja toteutus tekstinkyselyjärjestelmän tukemiseksi, jonka toteutuksen aloitamme luvussa 6.

Luvussa 18 esitellään monimutkaisempia periytymishierarkioita, jotka on tehty mahdolliseksi moni- ja virtuaaliperiytymisen avulla. Luvussa laajennetaan luvun 16 luokkamallin esimerkkiä kolmetasoiseksi luokkamallihierarkiaksi käyttäen moni- ja virtuaaliperiytymistä.

Luvussa 19 esitellään suorituksenaikainen tyypin tunnistuspiirre (RTTI). RTTI mahdollistaa ohjelmiemme kysellä monimuotoiselta luokkaoliolta sen tyyppiä suorituksen aikana. Voimme esimerkiksi kysyä Employee-oliolta, osoittaako se todellisuudessa Manager-tyyppiin. Lisäksi luvussa 19 palataan uudelleen poikkeusten käsittelyyn, jolloin käsitellään vakiokirjaston poikkeusluokan hierarkiaa ja kuvataan omien poikkeusluokkien määrittely ja käsittely. Se sisältää myös syvälle luotaavan katsauksen ylikuormitetun funktion ratkaisun tuesta silloin, kun kyseessä on periytyminen.

Luvussa 20 kuvataan yksityiskohtaisesti, kuinka C++-kielen *iostream*:in syöttö- ja tulostuskirjastoa käytetään. Mukana on selostus ja esimerkkejä yleisestä tietojen syöttämisestä ja tulostamisesta, syöttö- ja tulostusoperaattoreiden luokkakohtaisten ilmentymien määrittelemisestä, kuinka tiedostotiloja havaitaan ja asetetaan sekä kuinka tietoa muotoillaan. iostream-kirjasto on luokkahierarkia, joka on toteutettu käyttäen sekä virtuaalista että moniperiytymistä.

Tämä *C++*-kirja päättyy liitteeseen, joka sisältää ohjelmaesimerkin selostuksineen jokaisesta geneerisestä algoritmista aakkosjärjestyksessä helppoa hakemista varten.

Lopuksi sanottakoon, että aina, kun kirjoitetaan kirja, se, mitä jätetään sen ulkopuolelle, on usein yhtä tärkeää kuin se, mitä se sisältää. Jotkin kielen näkökohdat — kuten yksityiskohtainen muodostajien toiminnan käsittely, missä tilanteissa kääntäjä luo sisäiset tilapäiset oliot tai yleinen huoli tehokkuudesta — eivät istu kovin hyvin kielen perusesittelyyn, vaikka ne ovat yleisesti tärkeitä tosielämän sovellusten ohjelmoinnissa. Ennen kuin Stan ryhtyi tämän C++-kirjan kolmannen painoksen kirjoittamiseen, hän kirjoitti kirjan *Inside the C++ Object Model* (katso tämän esipuheen lopussa olevasta lähdeluettelosta [LIPPMAN96a]), jonka toivi toimivan mahdollisimman paljon tämän kirjan tukena. Tekstissä viitataan usein Object Model -käsittelyyn, kun lukijat saattavat tarvita yksityiskohtaisempaa läpikäyntiä, etenkin oliopohjaisessa ja oliokeskeisessä ohjelmoinnissa.

Tietyt osat C++-vakiokirjastosta on jätetty tarkoituksella pois, kuten maa- ja paikkakohtaisen tiedon tuki ja numeerinen kirjasto. C++-vakiokirjasto on erittäin laaja ja sen kaikkien kohtien esittely ei kuulu tämän kirjan piiriin. Lähdeluettelon jotkut kirjat käsittelevät kirjastoa yksityiskohtaisemmin (katso [MUSSER96] ja [STROUSTRUP97]). Uskomme, että tämän kirjan julkaisua seuraa monia kirjoja, jotka käsittelevät C++-vakiokirjaston lukuisia eri puolia.

Muutokset kolmanteen painokseen

Muutoksen kolmanteen painokseen jakautuvat neljään yleiskategoriaan:

1. Kieleen lisättyjen uusien piirteiden käsittely: poikkeusten käsittely, suorituksenaikainen tyypin tunnistus, nimiavaruudet, sisäinen bool-tyyppi ja uusi tyyli tyyppikonversion merkitsemistavalle.
2. Uuden C++-vakiokirjaston käsittely mukaan lukien kompleksi- ja string-merkkijonotyypit, auto_ptr- ja parityypit, jono- ja assosiatiiviset säiliötyypit (etenkin säiliöt lista, vektori, kartta ja joukko) ja geneeriset algoritmit.
3. Aikaisemman tekstin tarkennukset koskevat vakio-C++-kielen uudistuksia, muutoksia ja laajennuksia. Esimerkki uudistuksesta on mahdollisuus esitellä uudelleen sisäkkäinen tyyppi, jota kieli ei aikaisemmin sallinut. Esimerkki kielen muutoksesta on virtuaalifunktiosta johdetun luokan ilmentymän mahdollisuus palauttaa tyyppi, joka on julkisesti johdettu kantaluokan ilmentymän palautustyypistä. Tämä muutos tukee luokkaoperaation muotoa, jotka kutsutaan *clone*- tai *factory*-metodiksi (clone()-virtuaalifunktio käsitellään kohdassa 17.5.7). Esimerkki laajennuksesta olemassaolevaan piirteeseen on mahdollisuus määritellä eksplisiittisesti yksi tai useampi funktiomallin malliargumenteista. (Itse asiassa malleja on laajennettu melkein siihen pisteeseen, että ne vaikuttavat uudelta piirteeltä!)
4. Parannukset kielen lisäpiirteiden pääosien käsittelyssä ja järjestyksessä — erityisesti mallit, luokat ja oliokeskeinen ohjelmointi. Sivuvaikutus, jonka Stan sai aikaan siirryttyään suhteellisen pienestä C++-tuottajayhteisöstä yleiseen C++-käyttäjäyhteisöön, on hänen uskomuksensa mukaan antanut syvemmälle luotaavan käsityksen ongelmista,

joita muutoin älykkäillä ohjelmoijilla on käyttäessään C++-kieltä älykkäästi. Vastaavasti tässä kolmannessa painoksessa olemme siirtäneet painopistettä monissa tapauksissa kuvataksemme paremmin piirteen taustalla olevia käsitteitä ja kuinka niitä parhaiten käytetään sekä tuodaan esille mahdolliset kompastuskivet aina, kun se on mahdollista.

C++-kielen tulevaisuus

Tämän kirjan julkaisemisen aikaan on ISO/ANSI C++ -standardikomitea saanut valmiiksi teknisen työnsä ensimmäiselle C++-kielen kansainväliselle standardille. ISO on julkaissut standardin kesällä 1998.

C++-toteutukset, jotka tukevat C++-standardia, ovat olleet käytettävissä standardin julkaisun jälkeen. C++-kielen kehityksen oletetaan vakiintuvan. Tämä vakiintuminen mahdollistaa edistyneiden, C++-standardin mukaisesti kirjoitettujen kirjastojen kehittämisen ja keskittymisen teollisuuden erityisongelmiin. Täten C++-maailman kasvun oletetaan tapahtuvan pääosin kirjastojen alueella.

Vaikka standardi on julkaistu, standardikomitea jatkaa työtään siitä huolimatta, vaikkakin hitaampaan tahtiin keskittyen standardin käyttäjien tulkintapyyntöihin. Tämä johtaa C++-standardin vähäisiin selvennyksiin ja korjauksiin. Jos tarvetta on, kansainvälistä standardia uudistetaan joka viides vuosi, jolloin otetaan huomioon teknologiassa tapahtuneet muutokset ja teollisuudessa esille tulleet tarpeet.

Se, mitä tehdään C++-standardin julkaisun jälkeen viiden vuoden kuluttua, on yhä arvoitus. On mahdollista, että uudet kirjastokomponentit, jotka ovat laajasti käytössä teollisuuden piirissä, lisätään C++-vakiokirjaston komponenttijoukkoon. Mutta siihen saakka C++-standardikomitean työn tulos ja C++-kielen kohtalo lepäävät puhtaasti käyttäjien käsissä.

Kiitokset

Erityiskiitokset kuten tavallista menevät Bjarne Stroustrupille sekä hienosta kielestä, jonka hän on antanut meille, että avuliaisuudesta, jota hän on osoittanut vuosien aikana. Erityiskiitokset menevät myös C++-standardikomitean jäsenille heidän sitoutumisestaan ja kovasta työstä (usein täysin hyväntekeväisyydellä) ja tärkeästä panoksestaan C++-standardin hyväksi.

Seuraavat henkilöt antoivat monia hyödyllisiä kommentteja käsikirjoituksen monissa eri vaiheissa: Paul Abrahams, Michael Ball, Stephen Edwards, Cay Horstmann, Brian Kernighan, Tom Lyons, Robert Murray, Ed Scheibel, Roy Turner ja Jon Wada. Haluaisimme erityisesti kiittää Michael Ballia syvällisistä kommenteistaan ja rohkaisustaan. Olemme erityisen kiitollisia Clouis Tondolle ja Bruce Leungille tekstin tarkasta läpikäynnistä.

Stan haluaisi lisätä lämpimät kiitokset Shyh-Chyuan Huangille ja Jinko Gotohille heidän avustaan ja tuestaan Firebirdille, Jon Wadalle ja tietenkin Joséelle.

Josée haluaisi kiittää Gabby Silbermania, Karen Bennetiä ja tiimiä Centre for Advanced

Studies -keskuksessa heidän tuestaan tätä kirjaa kirjoitettaessa. Ja suuri kiitos menee Stanille, joka otti hänet mukaansa tähän suureen seikkailuun.

Lopuksi haluaisimme molemmat kiittää loistavaa toimitusjoukkoa heidän kovasta työstään ja kärsivällisyydestään: Debbie Laffertya, joka on ollut näiden C++-kirjojen toimituksessa mukana ihan alusta asti sekä Mike Hendricksonia ja John Fulleria. The Big Purple Company -yhtiö teki hienoa työtä kirjasinasettelussa. Kappaleen 6.1 kuvituksen teki Elena Driskill. Monet kiitokset Elenalle, kun annoit meidän julkaista sen uudelleen.

Kiitokset toiseen painokseen

Tämä kirja on monien sellaisten näkymättömien käsien tulosta, jotka auttavat pitämään kirjoittajansa oikeassa kurssissa. Sydämellisimmät kiitokseni menevät Barbara Moolle. Hänen rohkaisunsa, neuvonsa ja käsikirjoituksen monien eri versioiden tarkka lukeminen on ollut korvaamatonta. Erityiset kiitokset menevät Bjarne Stroustrupille hänen jatkuvasta avustaan ja rohkaisusta sekä hienosta kielestä, jonka hän on meille antanut; Stephen Dewhurstille, joka antoi paljon tukea aikaisessa vaiheessa, kun vasta opettelin C++-kieltä; ja Nancy Wilkinsonille, joka on uusi pesunkestävä cfront-koodari ja Kummikarhujen fani.

Dag Brück, Martin Carroll, William Hopkins, Brian Kernighan, Andrew Koenig, Alexis Layton ja Barbara Moo antoivat erityisen yksityiskohtaisia ja näkemyksellisiä kommentteja. Heidän oikolukunsa ovat parantaneet tätä kirjaa huomattavasti. Andy Baily, Phil Brown, James Coplien, Elizabeth Flanagan, David Jordan, Don Kretsch, Craig Rubin, Jonathan Shopiro, Judy Ward, Nancy Wilkinson ja Clay Wilson oikolukivat käsikirjoituksen useita eri vedoksia ja antoivat monia hyödyllisiä kommentteja. David Prosser selvensi monia ANSI C -kysymyksiä. Jerry Schwarz, joka toteutti iostream-pakkauksen, antoi alkuperäisen dokumentaation, johon liite A perustuu [joka on nyt luku 20 kolmannessa painoksessa!]. Hänen tuolle liitteelle tarkkaa ja ajatuksen kanssa tehtyä oikolukuaan arvostetaan suuresti. Suuret kiitokset menevät myös muulle 3.0-julkaisuversion kehitystiimille: Laura Eaves, George Logothetis, Judy Ward ja Nancy Wilkinson.

Seuraavat suorittivat käsikirjoituksen oikolukuja Addison-Wesleylle: James Adcock, Steven Bellovin, Jon Forrest, Maurice Herlihy, Norman Kerth, Darrell Long, Victor Milenkovic ja Justin Smith.

Seuraavat ovat osoittaneet virheitä monista ensimmäisen painoksen listauksista: David Beckedorff, Dag Brück, John Eldridge, Jim Humelsine, Dave Jordan, Ami Kleinman, Andrew Koenig, Tim O'Konski, Clovis Tondo ja Steve Vinoski.

Olen syvästi kiitollinen Brian Kernighanille ja Andrew Koenigille saadessani käytettäväkseni lukuisia eri kirjoitustyökaluja.

Lähdeluettelo

Seuraavat rivit joko esittelevät materiaalia, joka liittyy tämän kirjan kirjoittamiseen tai edustavat mittavaa materiaalia C++-kielestä, jota suositellaan.

[BOOCH94] Booch, Grady, *Object-Oriented Analysis and Design*, Benjamin/Cummings, Redwood City, CA (1994) ISBN 0-8053-5340-2.

[GAMMA95] Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides, *Design Patterns*, Addison Wesley Longman, Inc., Reading, MA (1995) ISBN 0-201-63361-2.

[GHEZZI97] Ghezzi, Carlo, and Mehdi Jazayeri, *Programming Language Concepts, 3rd Edition*, John Wiley and Sons, New York, NY (1997) ISBN 0-471-10426-4.

[HARBISON88] Samuel P. Harbison and Guy L. Steele, Jr, *C: A Reference Manual, 3rd Edition*, Prentice-Hall, Englewood Cliffs, NJ (1988) ISBN 0-13-110933-2.

[ISO-C++97] Draft Proposed International Standard for Information Systems — Programming Language C++ - Final Draft (FDIS) 14882.

[KERNIGHAN88] Kernighan, Brian W., and Dennis M. Ritchie, *The C Programming Language*, Prentice-Hall, Englewood Cliffs, NJ (1988) ISBN 0-13-110362-8.

[KOENIG97] Koenig, Andrew, and Barbara Moo, *Ruminations on C++*, Addison Wesley Longman, Inc., Reading, MA (1997) ISBN 0-201-42339-1.

[LIPPMAN91] Lippman, Stanley, *C++ Primer, 2nd Edition*, Addison Wesley Longman, Inc., Reading, MA (1991) ISBN 0-201-54848-8.

[LIPPMAN96a] Lippman, Stanley, *Inside the C++ Object Model*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-83454-5.

[LIPPMAN96b] Lippman, Stanley, Editor, *C++ Gems*, a SIGS Books imprint, Cambridge University Press, Cambridge, England (1996) ISBN 0-13570581-9.

[MEYERS98] Meyers, Scott, *Effective C++, 2nd Edition*, Addison Wesley Longman, Inc., Reading, MA (1998) ISBN 0-201-92488-9.

[MEYERS96] Meyers, Scott, *More Effective C++*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-63371-X.

[MURRAY93] Murray, Robert B., *C++ Strategies and Tactics*, Addison Wesley Longman, Inc., Reading, MA (1993) ISBN 0-201-56382-7.

[MUSSER96] Musser, David R., and Atul Saini, *STL Tutorial and Reference Guide*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-63398-1.

[NACKMAN94] Barton, John J., and Lee R. Nackman, *Scientific and Engineering C++, An Introduction with Advanced Techniques and Examples*, Addison Wesley Longman, Inc., Reading, MA (1994) ISBN 0-201-53393-6.

[NEIDER93] Neider, Jackie, Tom Davis, and Mason Woo, *OpenGL Programming Guide*, Addison Wesley, Inc., Reading, MA (1993) ISBN 0-201-63274-8.

[PERSON68] Person, Russell V., *Essentials of Mathematics, 2nd Edition*, John Wiley & Sons, Inc., New York, NY (1968) ISBN 0-132-84191-6.

[PLAUGER92] Plauger, P.J., *The Standard C Library*, Prentice-Hall, Englewood Cliffs, NJ (1992) ISBN 0-13-131509-9.

[SEDGEWICK88] Sedgewick, Robert, *Algorithms, 2nd Edition*, Addison Wesley Longman, Inc., Reading, MA (1988) ISBN 0-201-06673-4.

[SHAMPINE97] Shampine, L.F., R.C. Allen, Jr., and S. Pruess, *Fundamentals of Numerical Computing*, John Wiley & Sons, Inc., New York, NY (1997) ISBN 0-471-16363-5.

[STROUSTRUP94] Stroustrup, Bjarne, *The Design and Evolution of C++*, Addison Wesley Longman, Inc., Reading, MA (1994) ISBN 0-201-54330-3

[STROUSTRUP97] Stroustrup, Bjarne, *The C++ Programming Language, 3rd Edition*, Addison Wesley Longman, Inc., Reading, MA (1997) ISBN 0-201-88954-4.

[UPSTILL90] Upstill, Steve, *The RenderMan Companion*, Addison Wesley Longman, Inc., Reading, MA (1990) ISBN 0-201-50868-0.

[WERNECKE94] Wernecke, Josie, *The Inventor Mentor*, Addison Wesley Longman, Inc., Reading, MA (1994) ISBN 0-201-62495-8.

[YOUNG95] Young, Douglas A., *Object-Oriented Programming with C++ and OSF/Motif, 2nd Edition*, Prentice-Hall, Englewood Cliffs, NJ (1995) ISBN 0-132-09255-7.