

# Luento 11

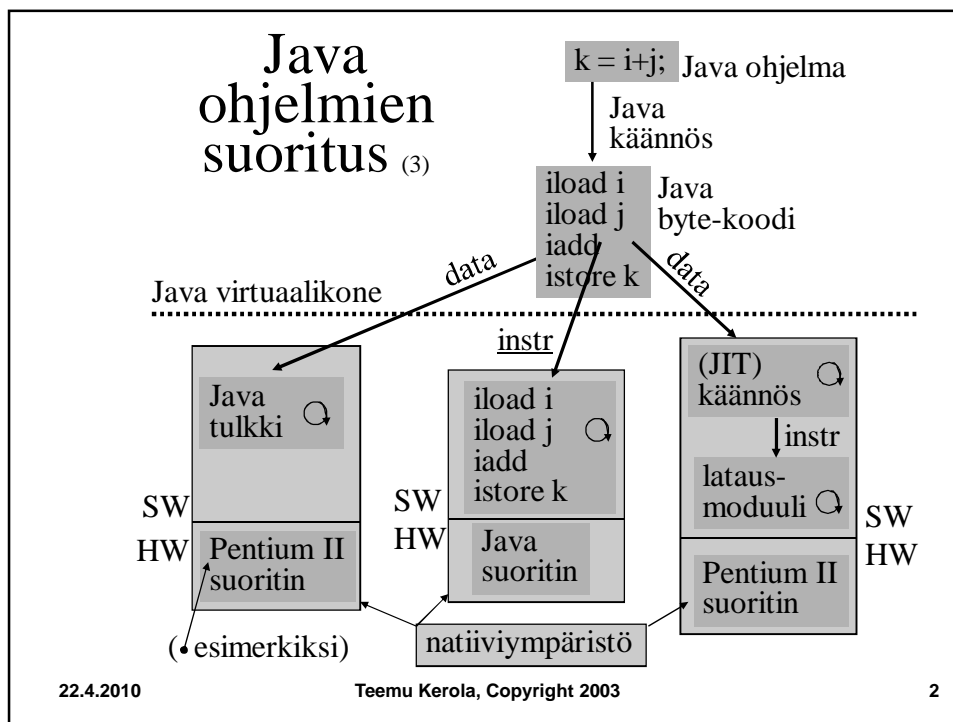
## Tulkinta ja emulointi

Tulkinta ja emulointi  
 Java ohjelman suoritus,  
 tulkinta ja kääntäminen  
 Suorittimen emulointi  
 C#, ttk-91, Crusoe

22.4.2010

Teemu Kerola, Copyright 2003

1



## Java virtuaalikone (JVM) <sup>(5)</sup>

- Hypoteettinen suoritin, toteutus eri tavoilla
- Geneerinen, sitä on ”helppo” simuloida kaikilla todellisilla suorittimilla
  - käännökseen tai tulkitsemiseen perustuva suoritus
- Useita säikeitä (thread) voi olla ”samanaikaisesti” suorituksessa
  - suorittimen mikroaikaskaalassa vain yksi kerrallaan
- Tietorakenteet
  - mm. virtuaalikoneen suorittimen ”rekisterit”
  - luodaan JVM:n käynnistämisen yhteydessä
- Käskyt
  - virtuaalikoneen suorittimen konekäskyt
  - 226 käskyä á 32 bittiä

22.4.2010

Teemu Kerola, Copyright 2003

3

## JVM:n tietorakenteet

- JVM pino ks. Fig. 4-10 [Tane99]
  - kuten tavallinen AT-pino
  - koostuu useista *kehyksistä* (frames) (vrt. aktivointitietue) ja operandipinosta
  - käyttö: kehyksille ainoastaan push/pop operaatiot, operandipinon alkioille myös push/pop
  - ei tarvita yhtenäistä muistialuetta
  - allokoidaan keosta (heap)
  - toteutuksesta riippuen rajallinen tai dynaamisesti laajennettavissa
  - tila loppu ⇒ StackOverflowError, OutOfMemoryError

<http://java.sun.com/docs/books/vmspec/2nd-edition/html/VMSpecTOC.doc.html>

22.4.2010

Teemu Kerola, Copyright 2003

4

## JVM:n tietorakenteet (jatkuu)

- JVM keko (JVM heap)
  - yhteinen kaikille saman virtuaalikoneen säikeille
  - automaattinen roskienkeruu (garbage collector)
    - ei-käytössä (implisiittisesti ”vapautettu”) oleva muistialue palautetaan uusiokäyttöön (vapaaksi)
    - ei tarvita erikseen *free* operaatiota Java ohjelmassa
    - voi hidastaa suoritusta milloin vain (ongelma?)
  - toteutuksesta riippuen kiinteän kokoinen tai dynaamisesti laajennettavissa
  - ei tarvitse muodostaa yhtenäistä muistialuetta natiivijärjestelmän keossa
  - tila loppu ⇒ OutOfMemoryError

22.4.2010

Teemu Kerola, Copyright 2003

5

## JVM:n tietorakenteet (jatkuu)

ks. Fig. 4-10 [Tane99]

- JVM metodialue (JVM Method Area)
  - yhteinen kaikille JVM säikeille
  - vastaa tavallista kääntäjän tuottamaa koodisegmenttiä
  - loogisesti osa JVM kekoa
  - toteutuksesta riippuen kiinteän kokoinen tai dynaamisesti laajennettavissa
  - tila loppu ⇒ OutOfMemoryError

22.4.2010

Teemu Kerola, Copyright 2003

6

## JVM:n tietorakenteet (jatkuu)

ks. Fig. 4-10 [Tane99]

- Javan suoritusaikainen vakioallas (runtime constant pool)
  - joka luokalle (class) ja liittymälle (interface)
  - suoritusaikainen esitystapa tiedoston *class constant\_pool* -taulukolle
  - vastaa vähän tavallista symbolitaulua
  - useita erilaisia vakioita (käännösaikaiset literaalit, suor. aikana ratkottavat attribuutit, ...)
  - talletetaan JVM metodialueelle
  - tila loppu ⇒ `OutOfMemoryError`

22.4.2010

Teemu Kerola, Copyright 2003

7

## JVM:n tietorakenteet (jatkuu)

- Natiivimetodien pinot (Native Method Stacks)
  - toteutus voi käyttää tavallisia pinoja ("C stacks") sellaisten natiivimetodien tukena, jota ei ole kirjoitettu Javalla
  - käytetään myös Java tulkin toteutuksessa
  - ei JVM toteutuksissa, joissa ei natiivimetoodeja
  - toteutuksesta riippuen kiinteän kokoinen tai dynaamisesti laajennettavissa
  - tila loppu ⇒ `StackOverflowError`,  
`OutOfMemoryError`

22.4.2010

Teemu Kerola, Copyright 2003

8

## JVM:n tietorakenteet (jatkuu)

ks. Fig. 4-10 [Tane99]

- JVM rekisterit
  - PC osoittaa johonkin JVM metodialueelle
  - CPP osoittaa vakioaltaaseen
  - LV on paikallisten muuttujien kantaosoite (vähän kuten FP ttk-91:ssä)
  - SP osoittaa JVM operandipinon huipulle
  - kaikki rekisterit implisiittisiä, niitä ei erikseen nimetä JVM konekäskyissä

22.4.2010

Teemu Kerola, Copyright 2003

9

## JVM:n tietorakenteet (jatkuu)

ks. Figs 4-12, 4-13 [Tane99]

- JVM kehys (frame, raami)
  - talletetaan JVM pinoon, luodaan metodin kutsun yhteydessä, vapautetaan metodista poistuttaessa
  - paikalliset muuttujat
  - parametrit, paluuarvo ja välitulokset
  - dynaamisen linkityksen toteutusväline
  - keskeytysten toteutusväline

22.4.2010

Teemu Kerola, Copyright 2003

10

## JVM kehyksen data <sup>(1)</sup>

- Paikalliset muuttujat sisältävä taulukko ks. Fig. 4-13 [Tane99]
  - viittaukset indeksoituna (0, 1, 2, ...) rekisterin LV suhteen
  - indeksit sanoina
  - kaksi sanaa vaativa muuttuja (long, double) sijoitetaan kahteen peräkkäiseen (32 bittiseen) sanaan
  - big-endian talletus
- Parametrit, paluarvon ja välitulokset sisältävä operandipino
  - SP osoittaa pinon huipulle
  - pinoarkkitehtuuri (vs. rekisteriarkkitehtuuri)

22.4.2010

Teemu Kerola, Copyright 2003

11

## JVM:n tiedon osoitusmoodit <sup>(4)</sup>

- Välitön operandi iINC 2 (34) Java: xLoc += 34;
- Indeksoitu iINC (2)34 tehollinen muistiosoite (LV) + 2
- Pino-osoitus iADD Java: a1 = a2+a3;
  - ks. Fig. 4-9 [Tane99]
  - korvaa pinon kaksi päällä olevaa kokonaislukua niiden summalla
- Taulukko-osoitus pinon kautta
  - aLOAD 1
  - iLOAD 2
  - iALOAD
  - iSTORE 3
  - Korvaa pinon pinnalla olevat taulukon alkuosoite ja indeksi k.o. taulukon alkiolla
  - Java: a = T[i];

22.4.2010

Teemu Kerola, Copyright 2003

12

## JVM käskyt

- Peruslaskutoimitukset ks. Fig. 5-36 [Tane99]
  - add, sub, mul, div, rem, neg
- Boolean
  - and, or, xor, shl, shr, ushr
- Pinon hallinta
  - dup, pop, swap, tauluk. luonti, esitystavan muutokset
- Load/Store
  - load, aload, store, astore, push-käskyt
- Vertailut
- Kontrollinsiirrot
- Muut

22.4.2010

Teemu Kerola, Copyright 2003

13

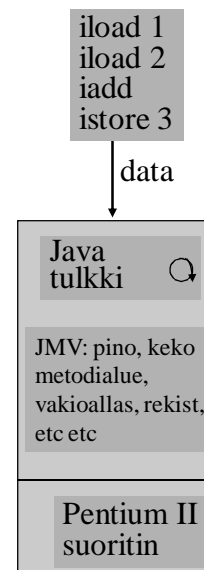
22.4.2010

Teemu Kerola, Copyright 2003

14

## Java tulkki

- Emuloi JVM konekielen käskyjä (byte-koodia)
- Yksi (byte-koodi) käsky kerrallaan
- JVM rekisterit ja muistialueet emuloitu tulkin tietorakenteina muistissa
  - vrt. KOKSI ja TTK-91
- Hidasta, mutta joustavaa



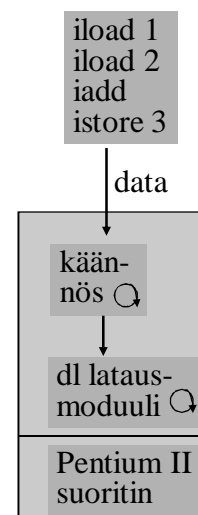
22.4.2010

Teemu Kerola, Copyright 2003

15

## Käännös natiivikoneelle

- (a) Käännetään tavukoodi suoraan natiivikoneen konekielelle ja suoritetaan se normaalin ohjelman tapaan
- (b) Käännetään tavukoodi ensin korkean tason kielelle (esim C), joka sitten käännetään natiivikoneen konekielelle
  - alkuosa riittää tehdä kerran
  - loppuosa on jo valmiina yleensä
- Ongelma: dynaaminen linkitys



22.4.2010

Teemu Kerola, Copyright 2003

16



## Java JIT käänнос

- JIT = Just-in-Time
- Emulointi ja/tai käänнос tilanteesta riippuen
- Käännä luokka natiivikonekielelle dynaamisesti linkitettäväksi moduuliksi, mutta vasta juuri ennen luokan metodin kutsua
- Tarvitsee paljon muistia
- Voi hidastaa suoritusta, jos käännökseen menee enemmän aikaa kuin tulkitsemiseen
  - käänнос vasta 2. kutsukerralla?
- JVM rekisterit ja muistialueet emuloitu tulkin tietorakenteina, joita natiivikoodi myös käyttää

22.4.2010 Teemu Kerola, Copyright 2003 17

## Java suoritin: Sun PicoJAVA II

- Suorittimen määrittely, jonka mukaisessa koneessa byte-koodi -muodossa olevia ohjelmia voidaan sellaisenaan suorittaa
- Valinnainen välimuisti ja liukulukusuoritin
- Kaikki 226 JVM konekäskyä
  - jotkut käskyt toteutettu aliohjelmilla, jotka aktivoidaan keskeytyskäsittelemekanismin avulla
- Myös 115 muuta konekäskyä käyttöjärjestelmän ja muiden ohjelmointikielten toteuttamiseksi

22.4.2010 Teemu Kerola, Copyright 2003 18

## PicoJAVA II pino

- 64 (välimuisti-) laiterekisteriä JVM pinon huipun talletukseen
  - loput JVM pinosta muistissa

Shawn Lauzon,  
Survey of the JavaChip

22.4.2010 Teemu Kerola, Copyright 2003 19

## PicoJAVA II rekisterit

- 25 rekisteriä á 32 bittiä
  - PC, LV, CPP, SP (pino kasvaa alaspäin)
  - OPLIM alaraja SP:lle; alitus aiheuttaa keskeytyksen
  - FRAME osoittaa paikallisten muuttujataulukon jälkeen talletettuun metodin paluusoitteeseen
  - PSW (tilarekisteri)
  - rekisteri, joka kertoo pinon välimuistirekistereiden tämänhetkisen käytön
  - 4 rekisteriä keskeytysten ja break-point'ien käsittelyyn
  - 4 rekisteriä säikeiden hallintaan
  - 4 rekisteriä C ja C++ ohjelmien toteutukseen
  - 2 rajarekisteriä sallitun muistialueen rajoittamiseen
  - suorittimen version numero ja konfiguraatiorekisterit

22.4.2010 Teemu Kerola, Copyright 2003 20

## PicoJAVA ylim. käskyt

- Read/write ylimääräisille rekistereille
- Osoittimien manipulointikäskyt
  - mitä tahansa muistialuetta voidaan suoraan lukea/kirjoittaa
  - tarvitaan C/C++ varten
- C/C++ aliohjelmien kutsu ja paluukäskyt
- Natiivi HW manipulointi
  - tyhjä välimuisti (osittain? kokonaan?), ...
- Muut käskyt
  - power on/off, ...

22.4.2010

Teemu Kerola, Copyright 2003

21

## PicoJAVA toteutuksia (2)

- Sun microJAVA 701
  - valinnainen välimuisti
  - oma muistiväylä
  - PCI väylä muille laitteille
  - 16 ohjelmitavaa I/O johdinta
    - näppäimet, LEDit, ...
  - 3 ohjelmitavaa ajastinta ( $\Rightarrow$  kellolaitekeskeytykset)
  - suunnattu halpisiin kannettaviin laitteisiin (kämment mikro, PDA - Personal Digital Assistant)
- Sun ultraJAVA
  - nopeampi, parempi, kalliimpi, ...
  - suunnattu grafiikka- ja multimediasovelluksiin

22.4.2010

Teemu Kerola, Copyright 2003

22

## Muita Java-suorittimia

- JEM (Rockwell Collins)
- PSC1000 (Patriot Scientific)
  - dSys (Saksa), lääketieteellisiä laitteita
- MJ501 (LG Semicon)
  - TV, älykortit
- JSR-001, Real-Time Specification for Java (Java Community Process, "Sun Microsystems")
  - aJile: aJ-80, aJ-100, älykkäät liikkuvat laitteet



ks. aJ100-WRP kuva

22.4.2010

Teemu Kerola, Copyright 2003

23

## Sun MAJC

- MAJC - Microprocessor Architecture for Java Computing
  - suoritinarkkitehtuurin määrittely
  - tavoitteena suuri nopeus Java, C ja C++ sovelluksille
  - suunnattu multimediasovelluksiin verkossa
  - tukee hyvin JIT-käännöstä

22.4.2010

Teemu Kerola, Copyright 2003

24

## MAJC toteutus: MAJC 5200

- 1-4 suoritinta (2 suorittimen lastu, v. 1999)
- Useiden (peräkkäin kutsuttavien) metodien samanaikainen suoritus eri suorittimilla
  - ennakoivalle (speculative) suoritukselle oma kasa
  - peruutus (rollback), jos ennakoitu suoritus meni pieleen
- 4 säiettä suorituksessa per suoritin
  - säikeen vaihto nopeampaa kuin muistista luku!
  - laiterekisterit 4:lle säikeelle! (hyper-threaded processor)
  - välimuistin hudin aikana suoritetaan muita säikeitä
- VLIW arkkitehtuuri – 4 konekäskyä samanaikaisesti
- Suunnattu interaktiiviseen TV:hen, virtuaalitodellisuussovelluksiin, ...

22.4.2010

Teemu Kerola, Copyright 2003

25

22.4.2010

Teemu Kerola, Copyright 2003

26

## C# eli "C sharp" <sup>(1)</sup>

- C#
  - Javan kaltainen kieli
  - kehittäjä: Anders Hejlsberg (Turbo Pascal, Delphi, J++)
  - osa Microsoftin .Net -ympäristöä
  - Mono – open source .Net for Linux [www.go-mono.com](http://www.go-mono.com)
  - nivoutuu hyvin Windows XP:n kanssa
  - ECMA (European Computer Manufacturers Association) standardi (MS, HP ja Intel)
- MSIL – virtuaalikoneen konekieli
  - Microsoft Intermediate Language
  - sopiva "välikieli" kaikille korkean tason kielille: C, C++, Pascal, Java, C#, Visual Basic
  - suoritus ainoastaan (JIT) käänösten avulla
  - CLR virtuaalikone (Common Language Runtime)

22.4.2010

Teemu Kerola, Copyright 2003

27

22.4.2010

Teemu Kerola, Copyright 2003

28

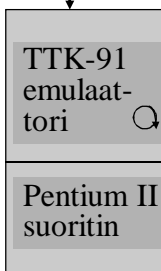
## TTK-91 Emulointi

- TTK-91 konekielen emulointi
- KOKSI simulaattorin osa
- Yksi käsky kerrallaan
- TTK-91 koneen rekisterit ja muisti emuloitu tulkin tietorakenteina

ks. simulaattorin koodi, luento 5  
(kurssikansio)

```
load R1, 234  
add R1, =5  
mul R1, R2
```

data



22.4.2010

Teemu Kerola, Copyright 2003

29

22.4.2010

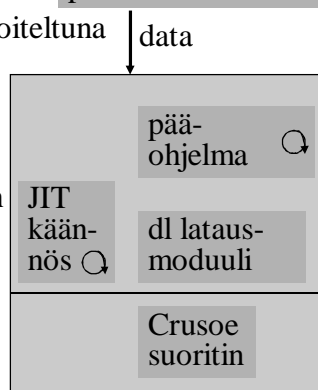
Teemu Kerola, Copyright 2003

30

## Transmetan Crusoe suoritin (8)

- x86 konekielen emulointi, JIT käänös
- Natiivi käskykanta ei ole tärkeä
- ”nopeampi, sama teknologia”?
- ”yhtä nopea, vähemmän virtaa”
- Monta x86 käskyä yhtä aikaa paloitetuna emuloinnissa, sekajärjestyksessä
- x86 rekisterit emuloitu natiivijärjestelmän laiterekistereillä
- x86 muisti emuloitu rekistereiden avulla suojattuna tietorakenteina
- Tarkat keskeytykset:
  - suorituksen peruutus
  - uusi käänös hitaalle koodille
  - uusi hidas tarkka emulointi

```
movl %esp, %ebp
subl $4, %esp
pushl %eax
```

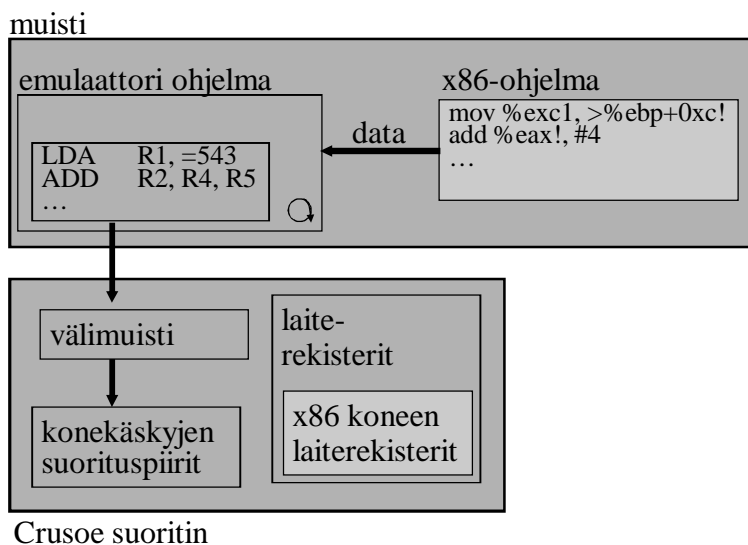


22.4.2010

Teemu Kerola, Copyright 2003

31

## Crusoe emulaattorin suoritus



22.4.2010

Teemu Kerola, Copyright 2003

32



