Vergleich der Nebenläufigkeitskonzepte von Ada, CHILL, Erlang und Java - Fallstudien 1

Peter Brömel, Diplomarbeit Friedrich-Schiller-Universität Jena, April 1999

www1.informatik.uni-jena.de/themen/pap-talk/da-pb.pdf

Ziel der Arbeit, die eine Fortsetzung und Erweiterung von [BE 99] darstellt, ist es, durch Fallstudien den Vergleich der Nebenläufigkeitsaspekte von vier Programmiersprachen - Ada, CHILL, Erlang und Java - zu erweitern und zu vertiefen. Die Sprachen benutzen unterschiedliche Ansätze für die nebenläufige Programmierung. Um die Nebenläufigkeitskonzepte der Sprachen zu vergleichen, wurden folgende Aufgabenstellungen gewählt, die in jeder der Sprachen realisiert wurden:

1. Der nebenläufige Quicksort-Algorithmus
2. Rückmeldung bei asynchroner Kommunikation
3. Das Leser-Schreiber-Problem
   - 3.1 Das Leser-Schreiber-Problem mit Priorität für Leser
   - 3.2 Das Leser-Schreiber-Problem mit Priorität für Schreiber
4. Kabinenbahn (nach [HH 94])
5. Bakery-Algorithmus

Der Vergleich erfolgte anhand der vorher definierten Kriterien Lesbarkeit, Kommunikationsaufwand, Nebenläufigkeitsgrad, Lebendigkeit, Verklemmung, Aushungern und Erfordernis einer fairen Maschine.

Aus den betrachteten Aufgaben lassen sich folgende Anforderungen an die Nebenläufigkeitskonzepte einer Sprache ableiten:

- Synchrone Kommunikation
- Asynchrone Kommunikation
- Koordinierter Zugriff auf gemeinsame Variablen
- Zeitüberwachung (Timeout)

Ein Aspekt der nebenläufigen Programmierung kommt bei den betrachteten Aufgabenstellungen nicht vor, und zwar die Einhaltung von Zeitbedingungen (Echtzeitprogrammierung). Dazu wurden in einer zweiten Diplomarbeit Untersuchungen angestellt [Eck 99].

Die o.a. Anforderungen werden von den Sprachen in sehr unterschiedlicher Art und Weise erfüllt.

Prozesskommunikation

> Ada basiert auf synchroner Kommunikation und unterstützt daher eine asynchrone Kommunikation nicht direkt.
>
> CHILL und Erlang basieren auf asynchroner Kommunikation und unterstützen daher eine synchrone Kommunikation nicht direkt.
>
> Java unterstützt keine direkte Kommunikation zwischen Prozessen.

Koordinierter Zugriff zu Objekten

> Ada enthält mit dem Protected-Object ein sehr flexibles Konzept für den koordinierten Zugriff.
>
> CHILL enthält mit dem Region das klassische Monitorkonzept inklusive Events zum Warten auf einen geeigneten Objektzustand.
>
> Erlang enthält kein Sprachkonstrukt für den koordinierten Zugriff.
>
> Java unterstützt den koordinierten Zugriff in einer einfachen Grundform (exklusiver Zigriff). Das Warten auf einen geeigneten Objektzustand kann nur ineffizient durch ein gemäßigtes aktives Warten realisiert werden.

Timeout ist in allen vier Sprachen möglich.

Insgesamt gesehen werden die o.a. Anforderungen von Ada am besten erfüllt, wobei die neuen Elemente in Ada95 daran einen deutlichen Anteil haben, und zwar speziell beim koordinierten Zugriff.

Immer dann, wenn eine Sprache die eine der o.a. Anforderungen nicht direkt unterstützt, benutzt wird, wird ein entsprechendes Programm umständlicher und unübersichtlicher als in einer Sprache, welche ein entsprechendes Konstrukt enthält.

1. Betreuer: Prof. Dr. Jürgen F.H. Winkler, FSU Jena

2. Betreuer: PD Dr. Heinz Toparkus, FSU Jena

BE 99 Brömel, Peter; Ecke, Frank: Description and Comparison of the Concurrency Concepts of Ada, Java, and CHILL. Studienarbeit, Friedrich-Schiller-Universität Jena, 13. Jan. 1999
www1.informatik.uni-jena.de/themen/pap-talk/studarb-pb-fe.htm

Eck 99 Ecke, Frank: Comparison of the Concurrency Concepts of Ada, CHILL, Erlang, and Java - Case Studies 2. Diplomarbeit Friedrich-Schiller-Universität Jena, 14. Juli 1999
www1.informatik.uni-jena.de/themen/pap-talk/da-fe.pdf

HH 94 Herrtwich, Ralf Guido; Hommel, Günter: Nebenläufige Programme. Springer, Berlin usw., 1994.
3-540-57783-1