

Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

262

Alan Burns
Andrew M. Lister
Andrew J. Wellings

A Review of Ada Tasking



Springer-Verlag

Berlin Heidelberg New York London Paris Tokyo

Editorial Board

D. Barstow W. Brauer P. Brinch Hansen D. Gries D. Luckham
C. Moler A. Pnueli G. Seegmüller J. Stoer N. Wirth

Authors

Alan Burns

Postgraduate School of Computing, University of Bradford
Bradford, United Kingdom

Andrew M. Lister

Department of Computer Science, University of Queensland
Queensland, Australia

Andrew J. Wellings

Department of Computer Science, University of York
York, United Kingdom

The analysis reported in this book was undertaken while A.M. Lister was on sabbatical leave at the University of York, UK.

CR Subject Classification (1987): C.2.4, D.1.3, D.3.1-4

ISBN 3-540-18008-7 Springer-Verlag Berlin Heidelberg New York

ISBN 0-387-18008-7 Springer-Verlag New York Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in other ways, and storage in data banks. Duplication of this publication or parts thereof is only permitted under the provisions of the German Copyright Law of September 9, 1965, in its version of June 24, 1985, and a copyright fee must always be paid. Violations fall under the prosecution act of the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1987

Printed in Germany

Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
2145/3140-543210

ABSTRACT

The Ada programming language can now be said to have come of age. Validated compilers are available; and there is a growing number of organisations committed to using the language for sizeable and significant applications. Development of the language has been accompanied by extensive analysis and comment, much of which has focussed on the tasking facilities. This book draws on the available literature to present a comprehensive review of the Ada tasking model. The model is examined in terms of its treatment of concurrency, synchronisation and communication; its application to embedded and distributed systems; its formal semantics; and its implementation. Reported weaknesses are discussed, and the implications of proposed changes are evaluated.

TABLE OF CONTENTS

1 INTRODUCTION	1
1.1 The Steelman Requirements	2
1.2 Concurrent Programming Constructs	2
1.2.1 Concurrent execution	3
1.2.2 Synchronisation and communication	4
1.3 Summary	7
2 THE ADA TASKING MODEL	9
2.1 An Overview of Ada	9
2.2 The Ada Tasking Model	17
2.2.1 Parallelism	17
2.2.2 The rendezvous	17
2.2.3 Shared variables	23
2.2.4 Task termination	24
2.2.5 Exceptions and tasks	24
2.2.6 Transactions	25
2.3 Task Attributes	25
2.4 Summary	26
3 FORMAL ASPECTS	27
3.1 Formal Semantics for Ada Tasking	27
3.1.1 Semantic problems with Ada	28
3.1.2 The SMoLCS approach	28
3.2 Specification and Verification	29
3.2.1 Place- and predicate-transition nets	29
3.2.2 Temporal logic	29
3.2.3 An axiomatic proof system	30
3.2.4 Path expressions	31
3.2.5 Other techniques	31

4 CONCURRENT PROGRAMMING	33
4.1 Concurrent Execution	33
4.1.1 Task initialisation	33
4.1.2 Task termination	34
4.2 Communication via the Rendezvous	34
4.2.1 Asymmetric naming and security	34
4.2.2 Asynchronous communication	35
4.2.3 Communication failure	35
4.2.4 Transactions	36
4.2.5 Nested rendezvous	39
4.2.6 The client-server paradigm	40
4.2.6.1 Modularity	40
4.2.6.2 Expressive power	42
4.2.6.3 The expressive power of the select statement	43
4.2.6.4 Ease of use	50
4.2.6.5 Summary of client-server transactions	50
4.2.7 The asymmetry of the select statement: polling	51
4.2.8 Task abortion	52
4.3 Communication via Shared Variables	52
4.4 The Role of Tasks in Program Design	53
4.5 Summary	54
5 EMBEDDED SYSTEMS	55
5.1 Real-Time Control	55
5.1.1 Task priority	55
5.1.2 Timing facilities	56
5.2 Low Level Device Handling	56
5.3 Summary	60
6 DISTRIBUTED SYSTEMS	63
6.1 The Distributed System Model	63
6.2 Designing a Distributed System as a Single Ada Program	64
6.2.1 Post-partitioning	65
6.2.2 Pre-partitioning	66
6.3 Communicating Ada Programs	68
6.3.1 Model of concurrency	68
6.3.2 Inter-program communication	69
6.3.3 Configuration management	71
6.4 Processor Failure and Network Partitioning	71

6.5 Summary	72
7 IMPLEMENTATION ISSUES	73
7.1 Concurrent Execution	73
7.1.1 Task dispatching	73
7.1.2 Task creation	74
7.1.3 Task termination	75
7.1.4 Heap management	75
7.2 Synchronisation and Communication	75
7.3 Exceptions and Tasks	76
7.4 State Transition Diagrams for Ada Tasks	76
7.5 Performance	82
7.6 Optimisation Techniques	82
7.7 Implementation Requirements for Embedded Computer Systems	85
7.8 Implementation of Tasking in a Distributed Environment	86
7.8.1 Task creation and migration	86
7.8.2 Task termination	86
7.8.3 Synchronisation and communication	87
7.8.3.1 The rendezvous	87
7.8.3.2 Timed and conditional entry calls	90
7.8.3.3 Shared variables	91
7.9 Summary	91
8 LANGUAGE CHANGES AND EXTENSIONS	93
8.1 Task Initialisation	93
8.2 Task Termination	93
8.2.1 Library task termination	93
8.2.2 Storage deallocation	93
8.3 Implementing Transactions	94
8.3.1 The two-rendezvous solution	94
8.3.2 Increasing the expressive power	95
8.3.3 Conditional wait synchronisation	96
8.4 Entry Families and the Select Statement	97
8.5 Asymmetry of the Select Statement	100
8.6 Priority	101
8.7 Distributed Systems	101
8.7.1 Virtual nodes	101
8.7.2 Fault-tolerance	102
8.8 Optimisation Pragmas	102

9 CONCLUSION	103
ACKNOWLEDGEMENT	105
APPENDIX A: Task Initialisation in Highly Parallel Systems	107
APPENDIX B: Asynchronous Agents	111
APPENDIX C: Implementation of a Reliable Transaction	115
APPENDIX D: Using Families For Handling Priority Requests	121
REFERENCES	125